A Comparison of Chinese Parsers for Stanford Dependencies

Wanxiang Che[†] car@ir.hit.edu.cn
Valentin I. Spitkovsky[‡] vals@stanford.edu
Ting Liu[†] tliu@ir.hit.edu.cn

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

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

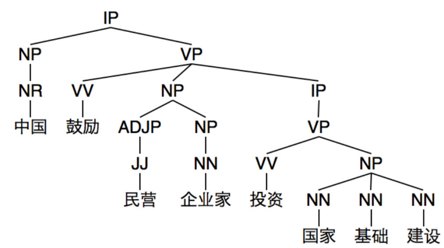
Abstract

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

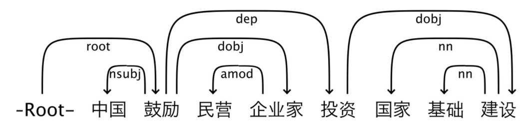
1 Introduction

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

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



(a) A constituent parse tree.



(b) Stanford dependencies.

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

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

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

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

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

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

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

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

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

2 Methodology

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

2.1 Parsers

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

2.2 Corpus

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

2.3 Settings

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

2.4 Features

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

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

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

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

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

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

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

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

3 Results

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

3.1 Chinese

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

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

3.2 English

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

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

4 Analysis

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

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

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

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

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

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

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

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

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

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

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

5 Discussion

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

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

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

6 Conclusion

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

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

Acknowledgments

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

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

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

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

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

Second author gratefully acknowledges the continued help and support of his advisor, Dan Jurafsky, and of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program, under the Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, AFRL, or the US government.

References

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

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

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

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

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

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

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

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

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

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

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

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

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

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

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado, June. Association for Computational Linguistics.

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

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

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

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1–11, Stroudsburg, PA, USA. Association for Computational Linguistics.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, June. Association for Computational Linguistics.

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

Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for Chinese POS tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1180–1191, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

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

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

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

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio, June. Association for Computational Linguistics.

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

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

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.

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

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

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

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

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

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

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

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

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 188–193, Stroudsburg, PA, USA. Association for Computational Linguistics.

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