Parsing Any Domain English text to CoNLL Dependencies

Sudheer Kolachina, Prasanth Kolachina

LTRC, IIIT-Hyderabad
sudheer.kpg08@research.iiit.ac.in, prasanth_k@research.iiit.ac.in

Abstract

It is well known that accuracies of statistical parsers trained over Penn treebank on test sets drawn from the same corpus tend to be overestimates of their actual parsing performance. This gives rise to the need for evaluation of parsing performance on corpora from different domains. Evaluating multiple parsers on test sets from different domains can give a detailed picture about the relative strengths/weaknesses of different parsing approaches. Such information is also necessary to guide choice of parser in applications such as machine translation where text from multiple domains needs to be handled. In this paper, we report a benchmarking study of different state-of-art parsers for English, both constituency and dependency. The constituency parser output is converted into CoNLL-style dependency trees so that parsing performance can be compared across formalisms. Specifically, we train rerankers for Berkeley and Stanford parsers to study the usefulness of reranking for handling texts from different domains. The results of our experiments lead to interesting insights about the out-of-domain performance of different English parsers.

Keywords: Statistical parsing, Parser benchmarking, Discriminative reranking, Constituency-to-dependency conversion

1. Introduction

Natural language parsers lie at the core of various natural language processing (NLP) applications such as machine translation (MT), question answering (QA), information extraction/retrieval (IE/IR), etc. Building accurate, wide-coverage parsers has been one of the main goals in NLP research for more than a decade now. Much of the work in this area has focused on English although in recent times, work on parsing other languages is steadily increasing. Constituency-based parsing has been the dominant paradigm for building parsers for English. However, of late, there has been a considerable interest in dependency parsing for English because the simplicity of the dependency representation makes it amenable for a variety of applications. There exist today a number of constituency parsers for English based on different statistical parsing approaches that claim respectable performance at parsing English texts (Charniak and Johnson, 2005; Klein and Manning, 2003; Petrov et al., 2006; Petrov and Klein, 2007). The standard practice in much of the parsing literature is to train the parsing models over sections 02-21 of the Penn treebank (PTB) (Marcus et al., 1994) and report parsing performance in terms of Parseval F-scores (Sekine and Collins, 1997) on sections 22 and 23 of the same corpus. However, it is well-known that the performance of any statistical model tends to be better on datasets that are similar in domain to the dataset used to train the model. Owing to this *domain-bias*, the F-scores of PTB-trained parsers reported on test sets drawn from the same corpus tend to be overestimates of the actual parsing performance of these models. This gives rise to the need for evaluation of parsing performance on corpora from different domains. Gildea (2001) is an early work that studies the variation in performance of a statistical parser on different corpora.

Comparative evaluation of parsers on different corpora can lead to interesting insights about the relative strengths/weaknesses of different parsing approaches. Moreover, such an evaluation is also necessary to inform the choice of parsers while building real world applications such as MT, QA, IE/IR etc. For example, Petrov et al. (2010) is a recent work which studies the performance of different parsers at handling question constructions, and can therefore, be used to guide choice of parser while building a QA system.

In recent times, the development of efficient constituency-to-dependency conversion procedures such as Stanford dependencies (De Marneffe et al., 2006) has been an important development as it allows for performance of parsers to be compared across formalisms. Cer et al. (2010) and Çetinoğlu et al. (2010) are two recent studies which report a comparative evaluation of a number of state-of-art English parsers, both dependency and constituency. While Cer et al. (2010) is a detailed evaluation of different parsers based on the Stanford typed dependency scheme, Çetinoğlu et al. (2010) base their comparison on a LFG-inspired dependency scheme. However, in both these studies, parsing performance is evaluated only on test sets from the PTB corpus. In this paper, we attempt a similar detailed benchmarking of a number of statistical parsers, both dependency and constituency. We compare the performance of parsers of both formalisms using a dependency-based evaluation. The constituency-to-dependency converter used in our experiments differs from the one used in these earlier works. The main point of departure in our study is the evaluation of parsers on different kinds of corpora and not just the PTB sections as in the earlier works.

The motivation for such a benchmarking study comes from the need to identify a high quality parser for English that can be used for analysis of source sentences in an English-to-Indian language machine translation system. Since it is desirable that the source analysis assign a dependency structure, we opt for a dependency-based evaluation of parsers. Another important goal of our work is to study the influence of reranking and self-training while parsing texts from different domains.

2. Treebanks

In our benchmarking, we use different treebank corpora available for English for training and testing the parsers. The most well-known and also the largest of them is the Penn treebank (Marcus et al., 1994). The Penn treebank (PTB) consists of 49207 sentences from the Wall Street Journal (WSJ) newspaper corpus manually annotated with syntactic structure using a constituency-based annotation scheme. The treebank is split into 24 sections. Of these, sections 02-21 are usually used to train statistical parsers and sections 22, 23 and 24 are treated as development and test sets. All the parsing models studied in our work are trained using a similar partitioning of the treebank.

The Brown corpus is a balanced corpus of English texts drawn from multiple domains and genres (Nelson and Kučera, 1979). A subset of this corpus, manually annotated using the PTB scheme, is distributed as part of Treebank-3. This Brown parsed corpus consists of texts from different genres of fiction such as folklore, memoirs, mystery, adventure, romance and humor. The difference in domain of these Brown parsed texts compared to the WSJ qualify them as suitable test sets to study the out-of-domain performance of parsers trained over the PTB. In addition, the diversity of genres within the Brown parsed corpus makes it possible to study the variation of parsing performance across different genres.

The Questionbank (Judge et al., 2006) is a corpus of 4000 questions annotated using the PTB annotation scheme. In our study, we use the Questionbank corpus to benchmark the performance of various English parsers at parsing questions. We made a few necessary corrections to the original annotations following the steps mentioned here [1].

Foster and van Genabith (2008) created a parsed corpus of 1000 sentences from the British National Corpus (BNC). The sentences were assigned manually annotated constituent structures based on the PTB annotation scheme. The sentences in this set were chosen such that each sentence contains a word that appears as a verb in the BNC but not in the usual training sections of PTB. We used this test set of sentences in our benchmarking as it was designed to be a difficult set for WSJ-trained parsers.

Finally, we also consider two treebanks from the biomedical domain in our benchmarking. We expect the parsing performance on these two test sets to reflect the ability of parsers to negotiate texts from technical domains which have high incidence of unknown, out-of-domain vocabulary items. The first test set is the Brown-Genia corpus (Lease and Charniak, 2005) which contains 215 sentences from the Genia corpus (Kim et al., 2003) [2]. We also use the Genia treebank corpus (Tateisi et al., 2005) as a test set since it is sufficiently large and therefore, accuracy reported on this dataset would be a more reliable indicator of parsing performance on this domain. It must be noted there is no overlap between these two treebanks.

[1] http://nlp.stanford.edu/data/QuestionBank-Stanford.shtml

[2] Available from http://www.cs.brown.edu/~mlease/parser-treebank.tgz

3. Constituency-to-Dependency conversion

As already mentioned, the development of high quality constituency-to-dependency conversion procedures in recent years has made comparisons of parsers across formalisms possible. The output parses of a constituency parser are converted to dependency structures using a constituency-to-dependency converter. The constituency-to-dependency conversion procedure is also used to automatically convert constituency treebanks into their dependency versions. Dependency parsers are trained over such automatically converted treebanks in case hand-crafted dependency treebanks do not exist. The performance of all parsers is evaluated using the standard dependency evaluation metrics of labeled attachment score (LAS), unlabeled attachment score (UAS) and label accuracy (LA).

In our work, we study the constituency-to-dependency conversion procedure proposed by Johansson and Nugues (2007). This procedure improves upon earlier conversion procedures such as Yamada and Matsumoto (2003) by using more sophisticated head-finding rules and by making use of function tags and traces, if present, to recover long-distance dependencies and non-projective dependencies. The *pennconverter*[3] is an implementation of this conversion procedure and was used to create dependency versions of the Penn treebank from which datasets were created for the CoNLL shared tasks on dependency parsing (Nivre et al., 2007; Surdeanu et al., 2008). For this reason, dependencies extracted by this converter are also popularly known as CoNLL dependencies. Since other existing converters such as the Stanford dependencies and LFG-based dependencies were explored in previous work (Cer et al., 2010; Çetinoğlu et al., 2010), we chose to work with the CoNLL dependencies so that a comparison across these different schemes is possible.

We used the pennconverter to create dependency versions of the Penn treebank and the Brown treebank both of which are annotated with function tags. The dependency parsers used in our experiments are trained over sections 02-21 of this dependency version of the Penn treebank. For other corpora such as the Questionbank, BNC corpus and the biomedical treebanks, in the absence of function tags and traces, the pennconverter is unable to recover all dependencies. Following Foster and van Genabith (2008), we handle this issue by applying a function tagger that assigns function tags to constituents in these corpora. We use the state-of-art function tagger of Chrupala (2007) in our experiments. The same combination of function tagger and pennconverter is used to convert the output parses of constituency parsers into dependency structures.

4. Parsers

In this section, we briefly describe the various parsers considered in our benchmarking study.

4.1. Constituency parsers

4.1.1. Charniak-Johnson Parsers

The Charniak-Johnson (CJ) parser is a reranking parser that does parsing in two stages (Charniak and Johnson, 2005).

[3] Available at http://nlp.cs.lth.se/software/treebank_coverter

Parser	Description
cj0	pre-trained parsing and reranker models
cj1	pre-trained parsing model + retrained MaxEnt reranker
cj2	Self-trained Charniak + retrained MaxEnt reranker
berkeley0	pre-trained sm-6 parsing model in the Berkeley parser
berkeley1	sm-5 model trained over PTB sections 02-21
berkeley2	berkeley0 + MaxEnt reranker trained using final split of PTB
berkeley3	berkeley0 + MaxEnt reranker trained using non-final split of PTB
berkeley4	berkeley1 + MaxEnt reranker trained using non-final split of PTB
stanford0	pre-trained PCFG parsing model in the Stanford parser
stanford1	pre-trained model trained only over PTB sections 02-21
stanford2	stanford1 + MaxEnt reranker trained using non-final split of PTB
matetools	pre-trained dependency parsing model in the MateTools parser
idparser	dependency parsing model trained over PTB sections 02-21

Table 1: Brief summary of the different parsers

The first stage of parsing is done using Charniak's lexicalized history-based generative statistical parser (Charniak, 2000). In the second stage, Johnson's discriminative reranker uses a large number of non-local features defined over the entire parse tree to rerank the k-best parses produced by the first stage parser. Charniak's parser is reported to give an F-score of 89.1 on section 23 of the WSJ corpus. When combined with the Johnson reranker, the F-score on the same section significantly improved to 91.3. McClosky et al. (2006) introduced self-training in this parsing setup as a result of which, the F-score improved to 92.1. We study the performance of both the original CJ parser [4] as well as the self-trained version of (McClosky et al., 2006). [5] In our initial experiments with the CJ parser, we noticed that the pre-trained model did not give the same performance as reported in the literature. So, we retrained the MaxEnt reranker following the procedure described in (Gao et al., 2007). Thus, we have three versions of the CJ parser as shown in Table 1.

4.1.2. Berkeley Parser and Discriminative Reranking

The Berkeley parser is an accurate constituency parser based on induction of latent PCFGs from constituency treebanks (Petrov et al., 2006). The latent non-terminal symbols in the PCFG are derived using an iterative *split-merge* technique. The probabilities of the rules in the grammar are estimated using the EM-algorithm in each iteration, followed by a smoothing step to reduce the risk of *over-fitting* to the training data. The number of split-merge cycles can be varied to learn grammars of different granularity. Petrov and Klein (2007) note that the sm-6 grammar (6 split-merge iterations) trained over the PTB could be overfitted to the WSJ corpus. In our experiments, we try to verify this by comparing the performance of the sm-5 and sm-6 grammars trained over PTB on different test sets. An important distinction between the Berkeley parser and the other parsers used in our experiments is the absence of *lexicalization* in the learnt grammars.

We also investigate the role of reranking in tackling domain differences by combining the Berkeley parser with the discriminative Johnson reranker. The reranker is trained using features reported in Johnson (2005). We train two different kinds of reranker models- one using sections 02-21 of the PTB as the training data and section 24 as the development (*final* split) and the other using sections 02-21 both for training and development (*non-final* split). The procedure to train the reranker using the non-final split of PTB is the same as described in (Johnson and Ural, 2010). Rerankers are trained for both the sm-5 and sm-6 parsing models.

4.1.3. Stanford parser

Among the different parsing models available in the Stanford parser, we consider the lexicalized PCFG parser[6] in our benchmarking study (Klein and Manning, 2003). This parser implements a factored product model, with separate PCFG phrase structure and lexical dependency experts. The PCFG parser begins with a raw n-ary treebank grammar obtained from the trees in the training section and performs a horizontal and vertical markovization in order to capture the external context to deal with sparsity arising from infrequent or unseen rule types. The probabilities over the sub-categorized grammar are estimated using maximum likelihood estimation followed by smoothing.

We also study the effect of reranking the output of the Stanford parser in comparison to the Charniak-Johnson and reranked Berkeley parsers. The reranker for the Stanford parser is trained using the non-final split of PTB and the same feature set as described in the case of Berkeley parser.

4.2. Dependency parsers

4.2.1. MateTools parser

The MateTools parser (Bohnet, 2010) is based on the Maximum-spanning tree algorithm using second-order features. It uses the MIRA algorithm combined with a hash kernel to learn the dependency structures from a treebank. The parser includes a parallel feature extraction process and a parsing algorithm to considerably improve the speed of the parser while still retaining the high accuracy of graph-based dependency parsing approaches. The MateTools parser contains a pre-trained model for parsing English text trained over sections 02-21 of the dependency version of PTB.

4.2.2. ISBN parser

The idparser (Titov and Henderson, 2007) implements a projective dependency parsing algorithm similar to a stan-

[4] Available at `https://bitbucket.org/bllip/bllip-parser/get/tip.tar.bz2`

[5] Available from `http://cs.brown.edu/~dmcc/selftraining/selftrained.tar.gz`

[6] Stanford Parser Version 1.6.4: 2010-11-30.

corpus		CJ		Berkeley		Stanford		Matetools	
		LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS
WSJ	section-22	85.85	90.14	86.59	90.82	82.95	87.35	84.58	88.87
	section-23	86.18	90.10	86.84	90.84	83.26	87.48	84.68	88.66
	section-24	85.02	89.43	86.32	90.71	82.54	87.17	83.85	88.45
Brown	Popular lore (cf)	82.11	87.38	83.42	88.56	80.18	85.63	81.13	86.81
	Letters and Memoirs (cg)	81.00	86.50	82.09	87.63	78.89	84.70	80.50	86.19
	General fiction (ck)	79.50	85.74	79.68	85.94	76.87	83.57	78.38	85.03
	Mystery fiction (cl)	79.42	85.87	79.21	85.77	76.42	83.31	78.35	85.27
	Science fiction (cm)	80.91	86.39	80.99	86.48	79.15	84.96	80.13	86.30
	Western fiction (cn)	79.77	86.44	79.92	86.53	77.09	84.19	79.10	85.83
	Romance stories (cp)	78.91	84.93	79.36	85.38	76.42	82.92	78.22	84.64
	Humor (cr)	78.56	84.60	78.88	85.05	75.91	82.49	78.05	84.14
Bio-medical	Brown-Genia	82.38	84.98	82.47	85.09	79.25	82.34	77.41	81.98
	Genia	82.10	84.29	82.30	84.57	78.68	81.62	76.64	81.38
QuestionBank		84.97	90.81	86.18	90.93	93.94	95.52	70.70	88.37
BNC		82.23	86.13	83.63	86.80	80.17	83.84	77.84	84.29

Table 2: Performance of pre-trained models of the parsers on different corpora

dard shift-reduce algorithm for context-free grammars. The underlying model is a latent-variable model for generative dependency parsing based on Incremental Sigmoid Belief Networks. In other words, the features for parsing are induced automatically using latent variables. As inference is intractable in this class of graphical models, the parser uses a variational inference method to compute the best parse for a given input sentence.

Of the two dependency parsers considered in our study, the matetools is a graph-based parser while the idparser is a transition-based parser and therefore, represent the two main kinds of dependency parsing algorithms. An additional consideration in choosing these two parsers was their availability under GPL. Table 1 contains a summary of the different parsers considered in our benchmarking study.

5. Experimental Results and Discussion

The latest distributions of most of the parsers considered in our study contain pre-trained models trained over sections 02-21 of the PTB and some additional corpora in the case of the Stanford parser. We start our benchmarking experiments with these pre-trained models. In the case of constituency parsers, we treat the accuracies obtained using these pre-trained models as the baselines. The performance of these models on different treebank corpora is shown in Table 2. From the values in the table, it seems that the pre-trained model in Berkeley parser outperforms all other pre-trained models on most of the test sets. Upon significance testing, we observed that while the Berkeley model outperforms the Stanford and the Matetools pre-trained models on most test sets, the difference in the accuracies of Berkeley and CJ parsing models is not significant on five sections of the Brown corpus (ck, cl, cm, cn, cr), the Brown-Genia corpus and the BNC test set. The Stanford pre-trained model gives an exceptionally high performance on the Questionbank compared to all other parsers. This is due to the use of a part of the Questionbank to train this pre-trained model. Another important observation that differs from conclusions of previous works is that dependency parsers for English do not necessarily perform worse than their constituency counterparts. The Matetools parser either outperforms or matches the Stanford parser on all the test sets except the Questionbank and BNC. The difference in accuracy between Stanford and Matetools models on the Brown-Genia test set is not significant. In addition, the difference between the accuracies of the Matetools parser and the CJ parser is not significant on the following genres in the Brown corpus- science fiction (cm), romance (cp) and humor (cr).

corpus	cj0		cj1		cj2	
	LAS	UAS	LAS	UAS	LAS	UAS
wsj 22	85.85	90.14	87.60	91.63	**87.63**	**91.88**
wsj 23	86.18	90.10	87.80	91.56	**88.15**	**92.04**
wsj 24	85.02	89.43	86.57	90.92	**87.81**	**92.09**
brown cf	82.11	87.38	84.13	89.21	**84.96**	**90.08**
brown cg	81.00	86.50	82.76	88.08	**83.64**	**89.04**
brown ck	79.50	85.74	81.37	87.36	**82.19**	**88.18**
brown cl	79.42	85.87	81.36	87.54	**82.20**	**88.36**
brown cm	80.91	86.39	82.30	87.61	**82.57**	**87.78**
brown cn	79.77	86.44	81.63	88.09	**82.56**	**88.96**
brown cp	78.91	84.93	80.69	86.45	**81.86**	**87.54**
brown cr	78.56	84.60	80.31	86.27	**81.06**	**87.09**
brown-genia	82.38	84.98	83.82	86.14	**84.56**	**86.94**
genia	82.10	84.29	82.08	84.25	**83.49**	**85.61**
questionbank	84.97	90.81	87.36	93.16	**88.20**	**94.14**
bnc	82.23	86.13	84.18	87.53	**85.27**	**88.45**

Table 3: Performance of Charniak-Johnson parsing models

Our next set of experiments were focused on studying different versions of each of the constituency parsers. Table 3 shows the performance of different versions of the CJ parser. The cj1 model which contains the retrained reranker significantly outperforms the baseline cj0 on all the test sets. There is a significant difference between the accuracies of these two versions on all test sets except the Brown-Genia corpus. The difference in precision is not significant for this corpus although the difference in LAS and UAS scores is nearly 1.5%. On the larger Genia corpus, the cj1 model performs slightly worse (significant) than the cj0 model. Both these observations put together suggest that retraining the reranker has a detrimental effect when it comes to biomedical domain test sets. The self-trained version of CJ parser, cj2 model, significantly outperforms the pre-trained cj0 model on all test sets. The difference between the accuracies of cj2 and cj1 models is significant on all but three datasets, the exceptions being section-22 of WSJ, Brown cm section and Brown-Genia.

The performance of different models of the Berkeley parser is shown in Table 4. Comparing the performance of the sm-6 (berkeley0) and the sm-5 (berkeley1) models, we notice that the accuracies of berkeley0 are greater than berkeley1

corpus	berkeley0		berkeley1		berkeley2		berkeley3		berkeley4	
	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS
wsj 22	86.59	90.82	85.94	90.17	**86.78**	**90.97**	86.28	90.49	85.79	89.99
wsj 23	86.84	90.84	86.41	90.27	**87.07**	**91.08**	86.30	90.26	86.28	90.14
wsj 24	**86.32**	**90.71**	85.79	90.24	85.89	90.39	85.80	90.27	85.63	90.10
brown cf	83.42	88.56	82.98	88.20	**83.94**	**89.05**	83.65	88.85	82.59	87.82
brown cg	82.09	87.63	81.27	86.94	**82.41**	**88.01**	82.22	87.86	80.88	86.54
brown ck	79.68	85.94	79.62	85.92	**79.78**	**86.17**	79.49	85.85	79.25	85.54
brown cl	79.21	85.77	78.74	85.38	**79.29**	**85.99**	78.94	85.69	78.52	85.13
brown cm	80.99	86.48	80.82	86.52	**81.04**	**86.79**	80.83	86.56	80.48	86.13
brown cn	**79.92**	**86.53**	79.85	86.49	79.82	86.49	79.57	86.28	79.52	86.17
brown cp	79.36	85.38	**79.80**	84.98	79.55	**85.67**	79.16	85.29	78.65	84.68
brown cr	78.88	85.05	78.63	84.82	**79.59**	**85.74**	79.26	85.52	78.33	84.51
brown-genia	82.47	85.09	81.49	83.96	**83.09**	**85.51**	82.71	85.00	81.32	83.65
genia	82.30	84.57	81.61	83.90	**83.16**	**85.36**	82.57	84.84	80.99	83.33
questionbank	**86.18**	**90.93**	85.24	89.85	85.00	90.75	83.72	90.76	83.70	88.87
bnc	83.63	86.80	82.96	86.13	**84.41**	**87.53**	84.17	87.29	82.58	85.79

Table 4: Performance of Berkeley parser models

corpus	stanford0		stanford1		stanford2	
	LAS	UAS	LAS	UAS	LAS	UAS
wsj 22	82.95	87.35	83.01	87.43	**85.94**	**90.27**
wsj 23	83.26	87.48	83.26	87.50	**85.93**	**90.01**
wsj 24	82.54	87.17	82.43	87.10	**84.78**	**89.42**
brown cf	80.18	85.63	80.18	85.57	**82.16**	**87.51**
brown cg	78.89	84.70	78.82	84.60	**80.58**	**86.37**
brown ck	76.87	83.57	76.72	83.47	**78.51**	**85.08**
brown cl	76.42	83.31	76.16	83.07	**78.13**	**84.96**
brown cm	79.15	84.96	78.42	84.46	**80.31**	**86.28**
brown cn	77.09	84.19	76.71	83.84	**78.49**	**85.54**
brown cp	76.42	82.92	75.99	82.51	**77.69**	**84.06**
brown cr	75.91	82.49	75.84	82.37	**77.49**	**84.04**
brown-genia	79.25	82.34	80.20	83.31	**81.98**	**85.03**
genia	78.68	81.62	78.68	81.42	**80.85**	**83.43**
questionbank	**93.94**	**95.52**	79.26	87.47	82.03	89.90
bnc	80.17	83.84	80.05	**83.74**	**82.17**	82.55

Table 5: Performance of Stanford parsing models

on all datasets except cp (Romance) section of Brown corpus. However, the difference in accuracies between these two models is not significant for most sections of the Brown corpus (ck, cl, cm, cn, cp and cr). Although these results show a trend similar to the one reported in (Petrov and Klein, 2007), this does not seem to be due to overfitting of the sm-6 model to the WSJ corpus. This is because the sm-6 model significantly outperforms the sm-5 model on other out-of-domain test sets such as the biomedical treebanks, Questionbank and the BNC test set. At this stage, it seems that the similar performance of sm-6 and sm-5 models on the Brown corpus needs further investigation and cannot be simply attributed to overfitting. Our next observation is about the effect of reranking the Berkeley parser. Comparing the accuracies of reranked versions and the 1-best models (berkeley2, berkeley3 against berkeley0, berkely4 against berkeley1), we noticed that reranking does not improve the performance as much as in the CJ parser. Comparing berkeley2 and berkeley0, significant improvements in accuracy are observed on three sections of Brown corpus (cf, cg and cr), biomedical treebanks and the BNC test set. Even in the case of the PTB test sets, only slight improvements are observed and that too, only in the case of only one reranked model- berkeley2. The berkeley3 reranked model, trained using the non-final split, in fact, lowers the accuracies on PTB and Brown test sets. A similar pattern can be observed for berkeley4, the sm-5 reranked model trained using non-final split of the PTB. These results suggest that rerankers trained using the final split with a larger training set ($39,825$ sentences) and a smaller development set (1345 sentences) are better than ones trained using the non-final split with smaller training set ($35,852$ sentences) and larger development set ($3,976$ sentences). Another striking observation is that all Berkeley reranked models perform significantly worse than their 1-best counterparts on the Questionbank.

The performance of different Stanford parsing models are shown in Table 5. The accuracies of stanford0 and stanford1 models are almost similar on most of the test sets, the only exception being the Questionbank. As mentioned earlier, the exceptionally high performance of stanford0 model on the Questionbank is due to the use of a part of this corpus to train this model. The other important observation is that reranking the Stanford parser significantly improves the accuracies on all test sets, a trend similar to the CJ parser. This suggests that reranking seems to be relatively more effective in the case of lexicalized parsers such as CJ and Stanford as opposed to Berkeley parser.

corpus	matetools		idparser	
	LAS	UAS	LAS	UAS
wsj 22	84.58	88.87	**84.88**	**89.01**
wsj 23	**84.68**	**88.66**	84.58	88.43
wsj 24	**83.85**	**88.45**	83.44	88.04
brown cf	**81.13**	**86.81**	78.57	84.14
brown cg	**80.50**	**86.19**	77.57	82.69
brown ck	**78.38**	**85.03**	76.03	82.52
brown cl	**78.35**	**85.27**	75.67	82.58
brown cm	**80.13**	**86.30**	77.88	83.82
brown cn	**79.10**	**85.83**	76.20	83.33
brown cp	**78.22**	**84.64**	75.14	80.92
brown cr	**78.05**	**84.14**	74.33	80.35
brown-genia	**77.41**	**81.98**	75.47	79.88
genia	**76.64**	**81.38**	75.08	79.47
questionbank	**70.70**	**88.37**	67.61	88.31
bnc	**77.84**	**84.29**	74.26	81.08

Table 6: Performance of dependency parsers

Our next set of experiments were aimed at comparing the graph-based dependency parsing algorithm of the Matetools parser with the shift-reduce algorithm of the idparser. In the case of the idparser, a dependency parsing model was trained over sections 02-21 of the dependency version of the PTB converted using the pennconverter. The accuracies of the dependency parsers on different test sets are shown in Table 6. The matetools parsing model significantly outperforms the idparser on all the out-of-domain test sets. In addition, the variation in performance of the idparser across different test sets suggests strong overfitting of the idparser model to the WSJ corpus.

In the remainder of this section, we briefly present the comparative patterns of performance of the parsers on each of the different types of corpora. The LAS and UAS scores of different parsers on the different test sets are shown in the stacked barplot in Figures 1, 2, 3, 4 and 5. In the case of the constituency parsers, the pre-trained model (baseline) is shown alongside the best performing model of each parser. It is interesting to note that relative ordering among parsers is similar on test sets from the same domain.

On all three test sets from the PTB, the self-trained version of CJ parser (cj2) performs best, followed by the Berkeley reranked model (berkeley2). The pre-trained model in

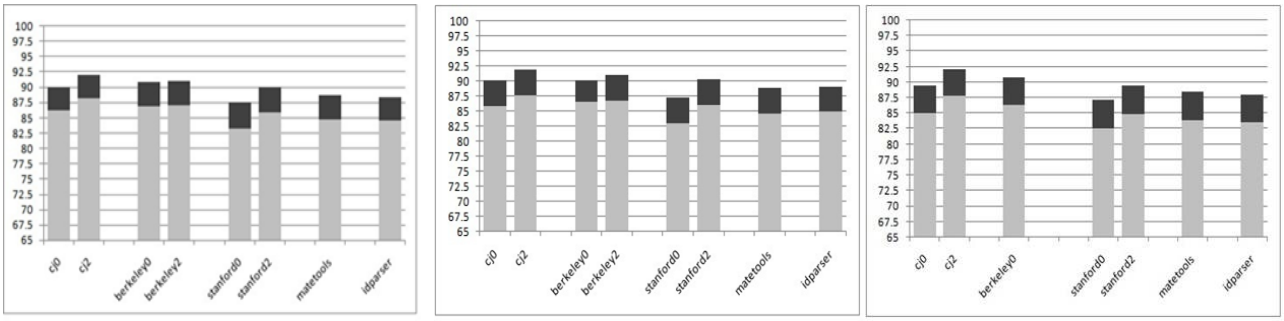

Figure 1: Comparative performance of parsers on PTB sections 22, 23 and 24.

the Stanford parser (stanford0) gives the lowest accuracies on these datasets. The two dependency parsers give higher accuracies than the Stanford pre-trained model.

On all the different sections of the Brown corpus, we observe that the self-trained version of the CJ parser gives the highest scores, followed by the Berkeley reranked and baseline models. The baseline model of the CJ parser achieves slightly lesser scores than the Berkeley baseline model, followed by the Matetools dependency parser. The scores obtained by the Stanford reranked model are comparable to those obtained from the Matetools parser. The performance of the idparser is quite low on all sections indicating overfitting to the training corpus.

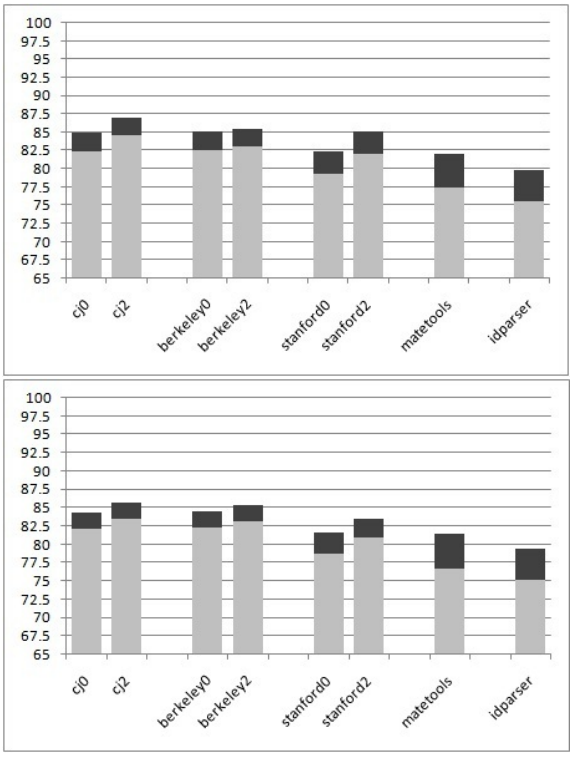

Figure 3: Bio-medical treebanks: Brown-Genia (top) and Genia (bottom)

On both the bio-medical treebanks, the self-trained CJ parser performs the best, closely followed by the Berkeley reranked and the Berkeley baseline models. While the Matetools dependency parser achieves much less LAS score in comparison to the self-trained CJ parser and the Berkeley reranked model, the UAS scores are comparable.

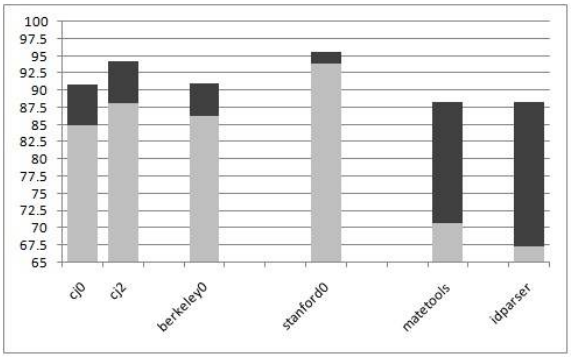

Figure 4: Parser performance comparison on Questionbank.

The variation in parsing performance across parsers is greatest in the case of the Questionbank. While the pre-trained Stanford model gives exceptionally high performance, the dependency parsers give their worst performances on this dataset. Additionally, reranking the Berkeley parser lead to a drop in the parsing performance over the baseline model on this test set.

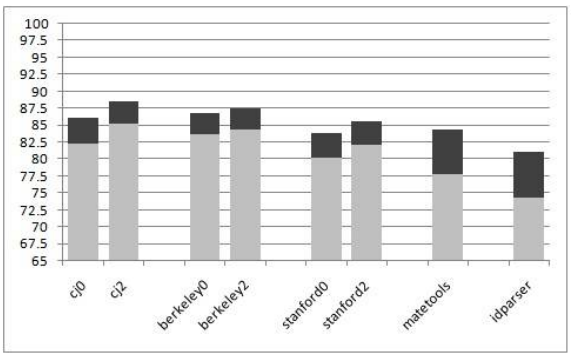

Figure 5: Comparison of parsers on BNC treebank.

In the case of BNC test set, the self-trained CJ parser gives the best performance followed by the Berkeley reranked and baseline models. The CJ baseline model and the reranked Stanford model give only slightly lower scores compared to the Berkeley pre-trained model. The LAS

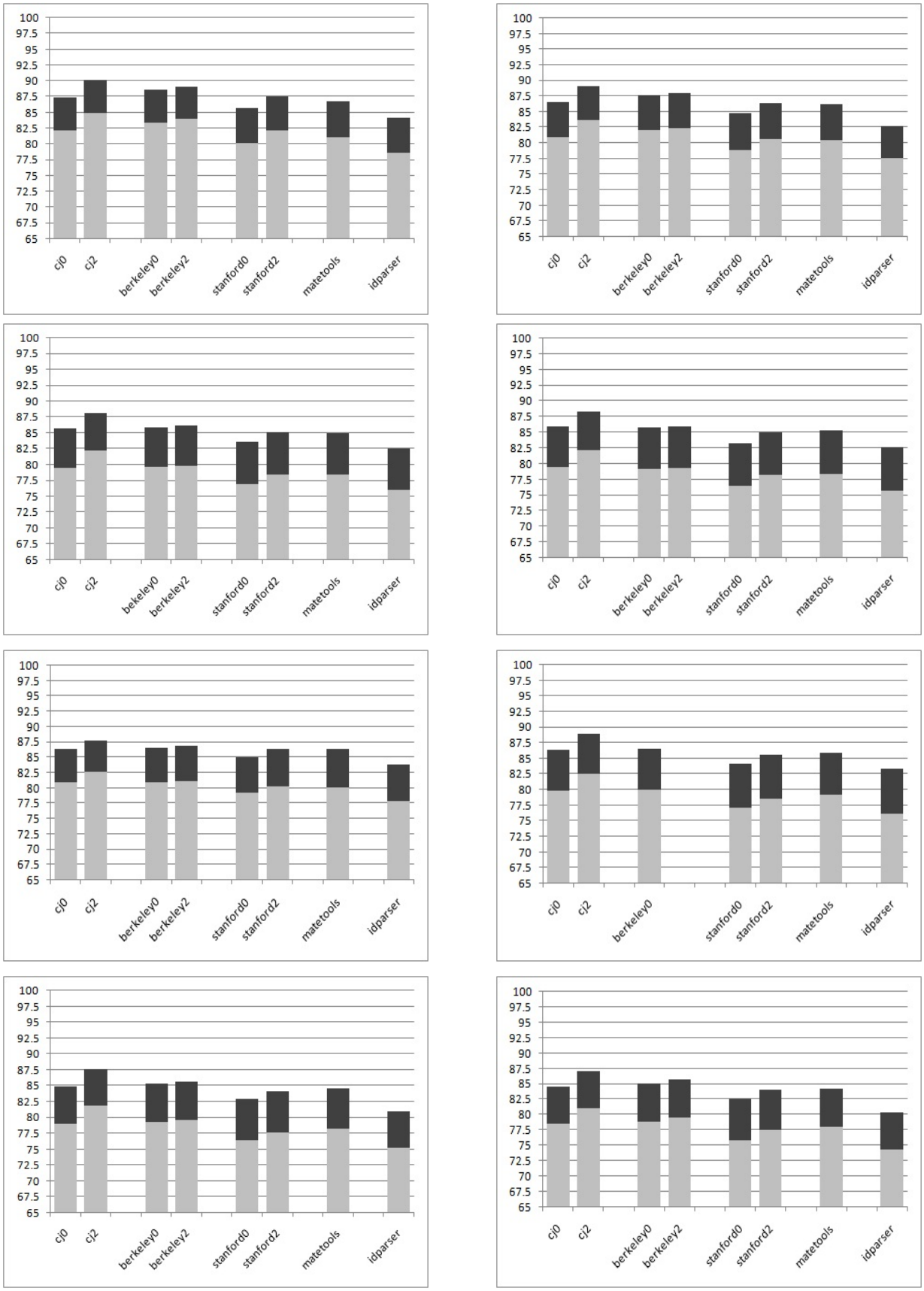

Figure 2: Comparative performance of different parsers on the Brown corpus: row 1 - cf, cg, row 2 - ck, cl, row 3 - cm, cn, row 4 - cp, cr

scores of the Matetools parser is quite low although the UAS is comparable to the Stanford pre-trained model. Both the LAS and UAS accuracies are lowest in the case of the idparser.

6. Conclusions

In this paper, we attempt a detailed benchmarking study of a number of statistical parsers, both constituency and dependency. We study the performance of different parsers at parsing text from domains that differ substantially from the newspaper domain of the WSJ corpus underlying the PTB used to train the parsers. We apply the parsers to texts from other treebank corpora such as the Brown corpus, BNC corpus, Questionbank and also biomedical texts such as the Genia corpus and the Brown-Genia corpus. In particular, we study the influence of discriminative reranking by training rerankers for the Berkeley and Stanford parsers. The following conclusions can be drawn from the results of our experiments:

- Reranking seems to be more useful when the base parser is lexicalized. While it improves the parsing performance on all test sets for the Stanford parser, it can sometimes even lead to a drop in the accuracies for the Berkeley parser (Questionbank).
- Although reranking gives significant improvements over the 1-best parsing performance, it cannot compete alone with self-training.
- Dependency parsers do not necessarily perform worse than their constituency counterparts, especially on the PTB and Brown test sets. However, the drop in performance on other out-of-domain corpora is more drastic in the case of dependency parsers. Perhaps, the introduction of reranking in the dependency parsers can help address this issue.

Acknowledgements

We would like to thank Vineet Chaitanya for his constant guidance and encouragement. We thank the following for their helpful suggestions– David McClosky, Slav Petrov, Daniel Cer, Richard Johansson, David Vadas, Ivan Titov, Jennifer Foster, Grzegorz Chrupala, Özlem Çetinoğlu, Taraka Rama and Uthej. We would also like to thank the anonymous reviewers for their insightful comments.

7. References

B. Bohnet. 2010. Very High 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.

Ö. Çetinoğlu, J. Foster, J. Nivre, D. Hogan, A. Cahill, and J. van Genabith. 2010. LFG without C-structures. In *Proceedings of the 9th International Workshop on Treebanks and Linguistic Theories*.

D. Cer, M.C. De Marneffe, D. Jurafsky, and C.D. Manning. 2010. Parsing to Stanford Dependencies: Trade-offs between speed and accuracy. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), May.

E. Charniak and M. 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.

E. Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of the 1st conference on North American Chapter of the Association for Computational Linguistics*, pages 132–139. Association for Computational Linguistics, June.

G. Chrupala. 2007. Better Training for Function Labeling. In *Proceedings of the International Conference RANLP-2007*, Borovets, Bulgaria.

M.C. De Marneffe, B. MacCartney, and C.D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of LREC*, volume 6, pages 449–454.

J. Foster and J. van Genabith. 2008. Parser Evaluation and the BNC: Evaluating 4 constituency parsers with 3 metrics. In *Proceedings of the Sixth conference on International Language Resources and Evaluation (LREC'08)*.

J. Gao, G. Andrew, M. Johnson, and K. Toutanova. 2007. A Comparative Study of Parameter Estimation Methods for Statistical Natural Language Processing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 824–831, Prague, Czech Republic, June. Association for Computational Linguistics.

D. Gildea. 2001. Corpus Variation and Parser Performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 167–202. Association for Computational Linguistics.

R. Johansson and P. Nugues. 2007. Extended Constituent-to-Dependency Conversion for English. In *Proceedings of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*. Citeseer.

M. Johnson and A.E. Ural. 2010. Reranking the Berkeley and Brown Parsers. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 665–668, Los Angeles, California, June. Association for Computational Linguistics.

J. Judge, A. Cahill, and J. van Genabith. 2006. QuestionBank: Creating a Corpus of Parse-Annotated Questions. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 497–504, Sydney, Australia, July. Association for Computational Linguistics.

J.D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. 2003. Genia corpus- A Semantically Annotated Corpus for Bio-Textmining. *Bioinformatics*, 19(suppl 1):i180.

D. Klein and C.D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, July. Association for Computational Linguistics.

M. Lease and E. Charniak. 2005. Parsing Biomedical Literature. In *Proceedings of 2nd International Joint Conference on Natural Language Processing*, pages 58–69. Asian Federation of Natural Language Processing.

M. Marcus, G. Kim, M.A. Marcinkiewicz, R. MacIntyr, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of the workshop on Human Language Technology*, HLT '94, pages 114–119, Stroudsburg, PA, USA. Association for Computational Linguistics.

D. McClosky, E. Charniak, and M. Johnson. 2006. Effective Self-training for Parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA, June. Association for Computational Linguistics.

F. W. Nelson and H. Kučera. 1979. *Manual of Information to accompany a Standard Corpus of present-day edited American English, for use with digital computers*. Brown University, Department of Lingustics.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*. Association for Computational Linguistics.

S. Petrov and D. Klein. 2007. Improved Inference for Unlexicalized Parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York, April. Association for Computational Linguistics.

S. Petrov, L. Barrett, R. Thibaux, and D. 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.

S. Petrov, P. Chang, M. Ringgaard, and H. Alshawi. 2010. Uptraining for Accurate Deterministic Question Parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 705–713, Cambridge, MA, October. Association for Computational Linguistics.

S. Sekine and MJ. Collins. 1997. Evalb bracket scoring program. web page.

M. Surdeanu, R. Johansson, A. Meyers, L. Màrquez, and J. 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.

Y. Tateisi, A. Yakushiji, T. Ohta, and J. Tsujii. 2005. Syntax Annotation for the GENIA corpus. In *Proceedings of 2nd International Joint Conference on Natural Language Processing*, pages 222–227. Asian Federation of Natural Language Processing.

I. Titov and J. Henderson. 2007. A Latent Variable Model for Generative Dependency Parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 144–155, Prague, Czech Republic, June. Association for Computational Linguistics.

H. Yamada and Y. Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of the Eighth International Conference on Parsing Technologies*. Association for Computational Linguistics.