Task-oriented Evaluation of Syntactic Parsers and Their Representations

Yusuke Miyao[†] Rune Sætre[†] Kenji Sagae[†] Takuya Matsuzaki[†] Jun'ichi Tsujii^{†‡*}

[†]Department of Computer Science, University of Tokyo, Japan

[‡]School of Computer Science, University of Manchester, UK

*National Center for Text Mining, UK

{yusuke,rune.saetre,sagae,matuzaki,tsujii}@is.s.u-tokyo.ac.jp

Abstract

This paper presents a comparative evaluation of several state-of-the-art English parsers based on different frameworks. Our approach is to measure the impact of each parser when it is used as a component of an information extraction system that performs protein-protein interaction (PPI) identification in biomedical papers. We evaluate eight parsers (based on dependency parsing, phrase structure parsing, or deep parsing) using five different parse representations. We run a PPI system with several combinations of parser and parse representation, and examine their impact on PPI identification accuracy. Our experiments show that the levels of accuracy obtained with these different parsers are similar, but that accuracy improvements vary when the parsers are retrained with domain-specific data.

1 Introduction

Parsing technologies have improved considerably in the past few years, and high-performance syntactic parsers are no longer limited to PCFG-based frameworks (Charniak, 2000; Klein and Manning, 2003; Charniak and Johnson, 2005; Petrov and Klein, 2007), but also include dependency parsers (Mc-Donald and Pereira, 2006; Nivre and Nilsson, 2005; Sagae and Tsujii, 2007) and deep parsers (Kaplan et al., 2004; Clark and Curran, 2004; Miyao and Tsujii, 2008). However, efforts to perform extensive comparisons of syntactic parsers based on different frameworks have been limited. The most popular method for parser comparison involves the direct measurement of the parser output accuracy in terms of metrics such as bracketing precision and recall, or dependency accuracy. This assumes the existence of a gold-standard test corpus, such as the Penn Treebank (Marcus et al., 1994). It is difficult to apply this method to compare parsers based on different frameworks, because parse representations are often framework-specific and differ from parser to parser (Ringger et al., 2004). The lack of such comparisons is a serious obstacle for NLP researchers in choosing an appropriate parser for their purposes.

In this paper, we present a comparative evaluation of syntactic parsers and their output representations based on different frameworks: dependency parsing, phrase structure parsing, and deep parsing. Our approach to parser evaluation is to measure accuracy improvement in the task of identifying protein-protein interaction (PPI) information in biomedical papers, by incorporating the output of different parsers as statistical features in a machine learning classifier (Yakushiji et al., 2005; Katrenko and Adriaans, 2006; Erkan et al., 2007; Sætre et al., 2007). PPI identification is a reasonable task for parser evaluation, because it is a typical information extraction (IE) application, and because recent studies have shown the effectiveness of syntactic parsing in this task. Since our evaluation method is applicable to any parser output, and is grounded in a real application, it allows for a fair comparison of syntactic parsers based on different frameworks.

Parser evaluation in PPI extraction also illuminates domain portability. Most state-of-the-art parsers for English were trained with the Wall Street Journal (WSJ) portion of the Penn Treebank, and high accuracy has been reported for WSJ text; however, these parsers rely on lexical information to attain high accuracy, and it has been criticized that these parsers may overfit to WSJ text (Gildea, 2001; Klein and Manning, 2003). Another issue for discussion is the portability of training methods. When training data in the target domain is available, as is the case with the GENIA Treebank (Kim et al., 2003) for biomedical papers, a parser can be retrained to adapt to the target domain, and larger accuracy improvements are expected, if the training method is sufficiently general. We will examine these two aspects of domain portability by comparing the original parsers with the retrained parsers.

2 Syntactic Parsers and Their Representations

This paper focuses on eight representative parsers that are classified into three parsing frameworks: *dependency parsing, phrase structure parsing,* and *deep parsing.* In general, our evaluation methodology can be applied to English parsers based on any framework; however, in this paper, we chose parsers that were originally developed and trained with the Penn Treebank or its variants, since such parsers can be re-trained with GENIA, thus allowing for us to investigate the effect of domain adaptation.

2.1 Dependency parsing

Because the shared tasks of CoNLL-2006 and CoNLL-2007 focused on data-driven dependency parsing, it has recently been extensively studied in parsing research. The aim of dependency parsing is to compute a tree structure of a sentence where nodes are words, and edges represent the relations among words. Figure 1 shows a dependency tree for the sentence "IL-8 recognizes and activates CXCR1." An advantage of dependency parsing is that dependency trees are a reasonable approximation of the semantics of sentences, and are readily usable in NLP applications. Furthermore, the efficiency of popular approaches to dependency parsing compare favorable with those of phrase structure parsing or deep parsing. While a number of approaches have been proposed for dependency parsing, this paper focuses on two typical methods.

MST McDonald and Pereira (2006)'s dependency parser,¹ based on the Eisner algorithm for projective dependency parsing (Eisner, 1996) with the second-order factorization.



Figure 1: CoNLL-X dependency tree



Figure 2: Penn Treebank-style phrase structure tree

KSDEP Sagae and Tsujii (2007)'s dependency parser,² based on a probabilistic shift-reduce algorithm extended by the pseudo-projective parsing technique (Nivre and Nilsson, 2005).

2.2 Phrase structure parsing

Owing largely to the Penn Treebank, the mainstream of data-driven parsing research has been dedicated to the phrase structure parsing. These parsers output Penn Treebank-style phrase structure trees, although function tags and empty categories are stripped off (Figure 2). While most of the state-of-the-art parsers are based on probabilistic CFGs, the parameterization of the probabilistic model of each parser varies. In this work, we chose the following four parsers.

NO-RERANK Charniak (2000)'s parser, based on a lexicalized PCFG model of phrase structure trees.³ The probabilities of CFG rules are parameterized on carefully hand-tuned extensive information such as lexical heads and symbols of ancestor/sibling nodes.

RERANK Charniak and Johnson (2005)'s reranking parser. The reranker of this parser receives n-best⁴ parse results from NO-RERANK, and selects the most likely result by using a maximum entropy model with manually engineered features.

BERKELEY Berkeley's parser (Petrov and Klein, 2007).⁵ The parameterization of this parser is op-

¹http://sourceforge.net/projects/mstparser

²http://www.cs.cmu.edu/~sagae/parser/

³http://bllip.cs.brown.edu/resources.shtml ⁴We set n = 50 in this paper.

⁵http://nlp.cs.berkeley.edu/Main.html#Parsing



Figure 3: Predicate argument structure

timized automatically by assigning latent variables to each nonterminal node and estimating the parameters of the latent variables by the EM algorithm (Matsuzaki et al., 2005).

STANFORD Stanford's unlexicalized parser (Klein and Manning, 2003).⁶ Unlike NO-RERANK, probabilities are not parameterized on lexical heads.

2.3 Deep parsing

Recent research developments have allowed for efficient and robust deep parsing of real-world texts (Kaplan et al., 2004; Clark and Curran, 2004; Miyao and Tsujii, 2008). While deep parsers compute theory-specific syntactic/semantic structures, predicate argument structures (PAS) are often used in parser evaluation and applications. PAS is a graph structure that represents syntactic/semantic relations among words (Figure 3). The concept is therefore similar to CoNLL dependencies, though PAS expresses deeper relations, and may include reentrant structures. In this work, we chose the two versions of the Enju parser (Miyao and Tsujii, 2008).

ENJU The HPSG parser that consists of an HPSG grammar extracted from the Penn Treebank, and a maximum entropy model trained with an HPSG treebank derived from the Penn Treebank.⁷

ENJU-GENIA The HPSG parser adapted to biomedical texts, by the method of Hara et al. (2007). Because this parser is trained with both WSJ and GENIA, we compare it parsers that are retrained with GENIA (see section 3.3).

3 Evaluation Methodology

In our approach to parser evaluation, we measure the accuracy of a PPI extraction system, in which This study demonstrates that **IL-8** recognizes and activates **CXCR1**, **CXCR2**, and the **Duffy antigen** by distinct mechanisms.

The molar ratio of serum **retinol-binding protein** (**RBP**) to **transthyretin** (**TTR**) is not useful to assess vitamin A status during infection in hospitalised children.

Figure 4: Sentences including protein names

ENTITY1(IL-8) $\xrightarrow{\text{SBJ}}$ recognizes $\xleftarrow{\text{OBJ}}$ **ENTITY2**(CXCR1)

Figure 5: Dependency path

the parser output is embedded as statistical features of a machine learning classifier. We run a classifier with features of every possible combination of a parser and a parse representation, by applying conversions between representations when necessary. We also measure the accuracy improvements obtained by parser retraining with GENIA, to examine the domain portability, and to evaluate the effectiveness of domain adaptation.

3.1 PPI extraction

PPI extraction is an NLP task to identify protein pairs that are mentioned as interacting in biomedical papers. Because the number of biomedical papers is growing rapidly, it is impossible for biomedical researchers to read all papers relevant to their research; thus, there is an emerging need for reliable IE technologies, such as PPI identification.

Figure 4 shows two sentences that include protein names: the former sentence mentions a protein interaction, while the latter does not. Given a protein pair, PPI extraction is a task of binary classification; for example, (IL-8, CXCR1) is a positive example, and $\langle RBP, TTR \rangle$ is a negative example. Recent studies on PPI extraction demonstrated that dependency relations between target proteins are effective features for machine learning classifiers (Katrenko and Adriaans, 2006; Erkan et al., 2007; Sætre et al., 2007). For the protein pair IL-8 and CXCR1 in Figure 4, a dependency parser outputs a dependency tree shown in Figure 1. From this dependency tree, we can extract a dependency path shown in Figure 5, which appears to be a strong clue in knowing that these proteins are mentioned as interacting.

⁶http://nlp.stanford.edu/software/lex-parser. shtml

⁷http://www-tsujii.is.s.u-tokyo.ac.jp/enju/

Figure 6: Tree representation of a dependency path

We follow the PPI extraction method of Sætre et al. (2007), which is based on SVMs with SubSet Tree Kernels (Collins and Duffy, 2002; Moschitti, 2006), while using different parsers and parse representations. Two types of features are incorporated in the classifier. The first is bag-of-words features, which are regarded as a strong baseline for IE systems. Lemmas of words before, between and after the pair of target proteins are included, and the linear kernel is used for these features. These features are commonly included in all of the models. Filtering by a stop-word list is not applied because this setting made the scores higher than Sætre et al. (2007)'s setting. The other type of feature is syntactic features. For dependency-based parse representations, a dependency path is encoded as a flat tree as depicted in Figure 6 (prefix "r" denotes reverse relations). Because a tree kernel measures the similarity of trees by counting common subtrees, it is expected that the system finds effective subsequences of dependency paths. For the PTB representation, we directly encode phrase structure trees.

3.2 Conversion of parse representations

It is widely believed that the choice of representation format for parser output may greatly affect the performance of applications, although this has not been extensively investigated. We should therefore evaluate the parser performance in multiple parse representations. In this paper, we create multiple parse representations by converting each parser's default output into other representations when possible. This experiment can also be considered to be a comparative evaluation of parse representations, thus providing an indication for selecting an appropriate parse representation for similar IE tasks.

Figure 7 shows our scheme for representation conversion. This paper focuses on five representations as described below.

CoNLL The dependency tree format used in the 2006 and 2007 CoNLL shared tasks on dependency parsing. This is a representation format supported by several data-driven dependency parsers. This repre-



Figure 7: Conversion of parse representations



Figure 8: Head dependencies

sentation is also obtained from Penn Treebank-style trees by applying constituent-to-dependency conversion⁸ (Johansson and Nugues, 2007). It should be noted, however, that this conversion cannot work perfectly with automatic parsing, because the conversion program relies on function tags and empty categories of the original Penn Treebank.

PTB Penn Treebank-style phrase structure trees without function tags and empty nodes. This is the default output format for phrase structure parsers. We also create this representation by converting ENJU's output by tree structure matching, although this conversion is not perfect because forms of PTB and ENJU's output are not necessarily compatible.

HD Dependency trees of syntactic heads (Figure 8). This representation is obtained by converting PTB trees. We first determine lexical heads of nonterminal nodes by using Bikel's implementation of Collins' head detection algorithm⁹ (Bikel, 2004; Collins, 1997). We then convert lexicalized trees into dependencies between lexical heads.

SD The Stanford dependency format (Figure 9). This format was originally proposed for extracting dependency relations useful for practical applications (de Marneffe et al., 2006). A program to convert PTB is attached to the Stanford parser. Although the concept looks similar to CoNLL, this representa-

⁸http://nlp.cs.lth.se/pennconverter/

⁹http://www.cis.upenn.edu/~dbikel/software. html



Figure 9: Stanford dependencies

tion does not necessarily form a tree structure, and is designed to express more fine-grained relations such as apposition. Research groups for biomedical NLP recently adopted this representation for corpus annotation (Pyysalo et al., 2007a) and parser evaluation (Clegg and Shepherd, 2007; Pyysalo et al., 2007b).

PAS Predicate-argument structures. This is the default output format for ENJU and ENJU-GENIA.

Although only CoNLL is available for dependency parsers, we can create four representations for the phrase structure parsers, and five for the deep parsers. Dotted arrows in Figure 7 indicate imperfect conversion, in which the conversion inherently introduces errors, and may decrease the accuracy. We should therefore take caution when comparing the results obtained by imperfect conversion. We also measure the accuracy obtained by the ensemble of two parsers/representations. This experiment indicates the differences and overlaps of information conveyed by a parser or a parse representation.

3.3 Domain portability and parser retraining

Since the domain of our target text is different from WSJ, our experiments also highlight the domain portability of parsers. We run two versions of each parser in order to investigate the two types of domain portability. First, we run the original parsers trained with WSJ¹⁰ (39832 sentences). The results in this setting indicate the domain portability of the original parsers. Next, we run parsers re-trained with GE-NIA¹¹ (8127 sentences), which is a Penn Treebank-style treebank of biomedical paper abstracts. Accuracy improvements in this setting indicate the possibility of domain adaptation, and the portability of the training methods of the parsers. Since the parsers listed in Section 2 have programs for the training

with a Penn Treebank-style treebank, we use those programs as-is. Default parameter settings are used for this parser re-training.

In preliminary experiments, we found that dependency parsers attain higher dependency accuracy when trained only with GENIA. We therefore only input GENIA as the training data for the retraining of dependency parsers. For the other parsers, we input the concatenation of WSJ and GENIA for the retraining, while the reranker of RERANK was not retrained due to its cost. Since the parsers other than NO-RERANK and RERANK require an external POS tagger, a WSJ-trained POS tagger is used with WSJtrained parsers, and geniatagger (Tsuruoka et al., 2005) is used with GENIA-retrained parsers.

4 Experiments

4.1 Experiment settings

In the following experiments, we used AImed (Bunescu and Mooney, 2004), which is a popular corpus for the evaluation of PPI extraction systems. The corpus consists of 225 biomedical paper abstracts (1970 sentences), which are sentence-split, tokenized, and annotated with proteins and PPIs. We use gold protein annotations given in the corpus. Multi-word protein names are concatenated and treated as single words. The accuracy is measured by abstract-wise 10-fold cross validation and the one-answer-per-occurrence criterion (Giuliano et al., 2006). A threshold for SVMs is moved to adjust the balance of precision and recall, and the maximum f-scores are reported for each setting.

4.2 Comparison of accuracy improvements

Tables 1 and 2 show the accuracy obtained by using the output of each parser in each parse representation. The row "baseline" indicates the accuracy obtained with bag-of-words features. Table 3 shows the time for parsing the entire AImed corpus, and Table 4 shows the time required for 10-fold cross validation with GENIA-retrained parsers.

When using the original WSJ-trained parsers (Table 1), all parsers achieved almost the same level of accuracy — a significantly better result than the baseline. To the extent of our knowledge, this is the first result that proves that dependency parsing, phrase structure parsing, and deep parsing perform

¹⁰Some of the parser packages include parsing models trained with extended data, but we used the models trained with WSJ section 2-21 of the Penn Treebank.

¹¹The domains of GENIA and AImed are not exactly the same, because they are collected independently.

	CoNLL	PTB	HD	SD	PAS
baseline	48.2/54.9/51.1				
MST	53.2/56.5/54.6	N/A	N/A	N/A	N/A
KSDEP	49.3/63.0/55.2	N/A	N/A	N/A	N/A
NO-RERANK	50.7/60.9/55.2	45.9/60.5/52.0	50.6/60.9/55.1	49.9/58.2/53.5	N/A
RERANK	53.6/59.2/56.1	47.0/58.9/52.1	48.1/65.8/55.4	50.7/62.7/55.9	N/A
BERKELEY	45.8/67.6/54.5	50.5/57.6/53.7	52.3/58.8/55.1	48.7/62.4/54.5	N/A
STANFORD	50.4/60.6/54.9	50.9/56.1/53.0	50.7/60.7/55.1	51.8/58.1/54.5	N/A
ENJU	52.6/58.0/55.0	48.7/58.8/53.1	57.2/51.9/54.2	52.2/58.1/54.8	48.9/64.1/55.3

Table 1: Accuracy on the PPI task with WSJ-trained parsers (precision/recall/f-score)

	CoNLL	PTB	HD	SD	PAS
baseline			48.2/54.9/51.1		
MST	49.1/65.6/55.9	N/A	N/A	N/A	N/A
KSDEP	51.6/67.5/58.3	N/A	N/A	N/A	N/A
NO-RERANK	53.9/60.3/56.8	51.3/54.9/52.8	53.1/60.2/56.3	54.6/58.1/56.2	N/A
RERANK	52.8/61.5/56.6	48.3/58.0/52.6	52.1/60.3/55.7	53.0/61.1/56.7	N/A
BERKELEY	52.7/60.3/56.0	48.0/59.9/53.1	54.9/54.6/54.6	50.5/63.2/55.9	N/A
STANFORD	49.3/62.8/55.1	44.5/64.7/52.5	49.0/62.0/54.5	54.6/57.5/55.8	N/A
ENJU	54.4/59.7/56.7	48.3/60.6/53.6	56.7/55.6/56.0	54.4/59.3/56.6	52.0/63.8/57.2
ENJU-GENIA	56.4/57.4/56.7	46.5/63.9/53.7	53.4/60.2/56.4	55.2/58.3/56.5	57.5/59.8/58.4

Table 2: Accuracy on the PPI task with GENIA-retrained parsers (precision/recall/f-score)

	WSJ-trained	GENIA-retrained
MST	613	425
KSDEP	136	111
NO-RERANK	2049	1372
RERANK	2806	2125
BERKELEY	1118	1198
STANFORD	1411	1645
ENJU	1447	727
ENJU-GENIA		821

Table 3: Parsing time (sec.)

equally well in a real application. Among these parsers, RERANK performed slightly better than the other parsers, although the difference in the f-score is small, while it requires much higher parsing cost.

When the parsers are retrained with GENIA (Table 2), the accuracy increases significantly, demonstrating that the WSJ-trained parsers are not sufficiently domain-independent, and that domain adaptation is effective. It is an important observation that the improvements by domain adaptation are larger than the differences among the parsers in the previous experiment. Nevertheless, not all parsers had their performance improved upon retraining. Parser

	CoNLL	PTB	HD	SD	PAS
baseline			424		
MST	809	N/A	N/A	N/A	N/A
KSDEP	864	N/A	N/A	N/A	N/A
NO-RERANK	851	4772	882	795	N/A
RERANK	849	4676	881	778	N/A
BERKELEY	869	4665	895	804	N/A
STANFORD	847	4614	886	799	N/A
ENJU	832	4611	884	789	1005
ENJU-GENIA	874	4624	895	783	1020

Table 4: Evaluation time (sec.)

retraining yielded only slight improvements for RERANK, BERKELEY, and STANFORD, while larger improvements were observed for MST, KSDEP, NO-RERANK, and ENJU. Such results indicate the differences in the portability of training methods. A large improvement from ENJU to ENJU-GENIA shows the effectiveness of the specifically designed domain adaptation method, suggesting that the other parsers might also benefit from more sophisticated approaches for domain adaptation.

While the accuracy level of PPI extraction is the similar for the different parsers, parsing speed

		RERANK			ENJU			
		CoNLL	HD	SD	CoNLL	HD	SD	PAS
KSDEP	CoNLL	58.5 (+0.2)	57.1 (-1.2)	58.4 (+0.1)	58.5 (+0.2)	58.0 (-0.3)	59.1 (+0.8)	59.0 (+0.7)
RERANK	CoNLL		56.7 (+0.1)	57.1 (+0.4)	58.3 (+1.6)	57.3 (+0.7)	58.7 (+2.1)	59.5 (+2.3)
	HD			56.8 (+0.1)	57.2 (+0.5)	56.5 (+0.5)	56.8 (+0.2)	57.6 (+0.4)
	SD				58.3 (+1.6)	58.3 (+1.6)	56.9 (+0.2)	58.6 (+1.4)
ENJU	CoNLL					57.0 (+0.3)	57.2 (+0.5)	58.4 (+1.2)
	HD						57.1 (+0.5)	58.1 (+0.9)
	SD							58.3 (+1.1)

Table 5: Results of parser/representation ensemble (f-score)

differs significantly. The dependency parsers are much faster than the other parsers, while the phrase structure parsers are relatively slower, and the deep parsers are in between. It is noteworthy that the dependency parsers achieved comparable accuracy with the other parsers, while they are more efficient.

The experimental results also demonstrate that PTB is significantly worse than the other representations with respect to cost for training/testing and contributions to accuracy improvements. The conversion from PTB to dependency-based representations is therefore desirable for this task, although it is possible that better results might be obtained with PTB if a different feature extraction mechanism is used. Dependency-based representations are competitive, while CoNLL seems superior to HD and SD in spite of the imperfect conversion from PTB to CoNLL. This might be a reason for the high performances of the dependency parsers that directly compute CoNLL dependencies. The results for ENJU-CoNLL and ENJU-PAS show that PAS contributes to a larger accuracy improvement, although this does not necessarily mean the superiority of PAS, because two imperfect conversions, i.e., PAS-to-PTB and PTB-to-CoNLL, are applied for creating CoNLL.

4.3 Parser ensemble results

Table 5 shows the accuracy obtained with ensembles of two parsers/representations (except the PTB format). Bracketed figures denote improvements from the accuracy with a single parser/representation. The results show that the task accuracy significantly improves by parser/representation ensemble. Interestingly, the accuracy improvements are observed even for ensembles of different representations from the same parser. This indicates that a single parse representation is insufficient for expressing the true

52

Bag-of-words features	48.2/54.9/51.1
Yakushiji et al. (2005)	33.7/33.1/33.4
Mitsumori et al. (2006)	54.2/42.6/47.7
Giuliano et al. (2006)	60.9/57.2/59.0
Sætre et al. (2007)	64.3/44.1/52.0
This paper	54.9/65.5/59.5

Table 6: Comparison with previous results on PPI extrac-tion (precision/recall/f-score)

potential of a parser. Effectiveness of the parser ensemble is also attested by the fact that it resulted in larger improvements. Further investigation of the sources of these improvements will illustrate the advantages and disadvantages of these parsers and representations, leading us to better parsing models and a better design for parse representations.

4.4 Comparison with previous results on PPI extraction

PPI extraction experiments on AImed have been reported repeatedly, although the figures cannot be compared directly because of the differences in data preprocessing and the number of target protein pairs (Sætre et al., 2007). Table 6 compares our best result with previously reported accuracy figures. Giuliano et al. (2006) and Mitsumori et al. (2006) do not rely on syntactic parsing, while the former applied SVMs with kernels on surface strings and the latter is similar to our baseline method. Bunescu and Mooney (2005) applied SVMs with subsequence kernels to the same task, although they provided only a precision-recall graph, and its f-score is around 50. Since we did not run experiments on protein-pair-wise cross validation, our system cannot be compared directly to the results reported by Erkan et al. (2007) and Katrenko and Adriaans (2006), while Sætre et al. (2007) presented better results than theirs in the same evaluation criterion.

5 Related Work

Though the evaluation of syntactic parsers has been a major concern in the parsing community, and a couple of works have recently presented the comparison of parsers based on different frameworks, their methods were based on the comparison of the parsing accuracy in terms of a certain intermediate parse representation (Ringger et al., 2004; Kaplan et al., 2004; Briscoe and Carroll, 2006; Clark and Curran, 2007; Miyao et al., 2007; Clegg and Shepherd, 2007; Pyysalo et al., 2007b; Pyysalo et al., 2007a; Sagae et al., 2008). Such evaluation requires gold standard data in an intermediate representation. However, it has been argued that the conversion of parsing results into an intermediate representation is difficult and far from perfect.

The relationship between parsing accuracy and task accuracy has been obscure for many years. Quirk and Corston-Oliver (2006) investigated the impact of parsing accuracy on statistical MT. However, this work was only concerned with a single dependency parser, and did not focus on parsers based on different frameworks.

6 Conclusion and Future Work

We have presented our attempts to evaluate syntactic parsers and their representations that are based on different frameworks; dependency parsing, phrase structure parsing, or deep parsing. The basic idea is to measure the accuracy improvements of the PPI extraction task by incorporating the parser output as statistical features of a machine learning classifier. Experiments showed that state-of-theart parsers attain accuracy levels that are on par with each other, while parsing speed differs significantly. We also found that accuracy improvements vary when parsers are retrained with domainspecific data, indicating the importance of domain adaptation and the differences in the portability of parser training methods.

Although we restricted ourselves to parsers trainable with Penn Treebank-style treebanks, our methodology can be applied to any English parsers. Candidates include RASP (Briscoe and Carroll, 2006), the C&C parser (Clark and Curran, 2004), the XLE parser (Kaplan et al., 2004), MINIPAR (Lin, 1998), and Link Parser (Sleator and Temperley, 1993; Pyysalo et al., 2006), but the domain adaptation of these parsers is not straightforward. It is also possible to evaluate unsupervised parsers, which is attractive since evaluation of such parsers with goldstandard data is extremely problematic.

A major drawback of our methodology is that the evaluation is indirect and the results depend on a selected task and its settings. This indicates that different results might be obtained with other tasks. Hence, we cannot conclude the superiority of parsers/representations only with our results. In order to obtain general ideas on parser performance, experiments on other tasks are indispensable.

Acknowledgments

This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan), Genome Network Project (MEXT, Japan), and Grant-in-Aid for Young Scientists (MEXT, Japan).

References

- D. M. Bikel. 2004. Intricacies of Collins' parsing model. Computational Linguistics, 30(4):479–511.
- T. Briscoe and J. Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *COLING/ACL 2006 Poster Session*.
- R. Bunescu and R. J. Mooney. 2004. Collective information extraction with relational markov networks. In *ACL 2004*, pages 439–446.
- R. C. Bunescu and R. J. Mooney. 2005. Subsequence kernels for relation extraction. In *NIPS 2005*.
- E. Charniak and M. Johnson. 2005. Coarse-to-fine nbest parsing and MaxEnt discriminative reranking. In ACL 2005.
- E. Charniak. 2000. A maximum-entropy-inspired parser. In NAACL-2000, pages 132–139.
- S. Clark and J. R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *42nd ACL*.
- S. Clark and J. R. Curran. 2007. Formalism-independent parser evaluation with CCG and DepBank. In *ACL* 2007.
- A. B. Clegg and A. J. Shepherd. 2007. Benchmarking natural-language parsers for biological applications using dependency graphs. *BMC Bioinformatics*, 8:24.

- M. Collins and N. Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In ACL 2002.
- M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In 35th ACL.
- M.-C. de Marneffe, B. MacCartney, and C. D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC 2006*.
- J. M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING* 1996.
- G. Erkan, A. Ozgur, and D. R. Radev. 2007. Semisupervised classification for extracting protein interaction sentences using dependency parsing. In *EMNLP* 2007.
- D. Gildea. 2001. Corpus variation and parser performance. In *EMNLP 2001*, pages 167–202.
- C. Giuliano, A. Lavelli, and L. Romano. 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *EACL* 2006.
- T. Hara, Y. Miyao, and J. Tsujii. 2007. Evaluating impact of re-training a lexical disambiguation model on domain adaptation of an HPSG parser. In *IWPT 2007*.
- R. Johansson and P. Nugues. 2007. Extended constituent-to-dependency conversion for English. In *NODALIDA 2007*.
- R. M. Kaplan, S. Riezler, T. H. King, J. T. Maxwell, and A. Vasserman. 2004. Speed and accuracy in shallow and deep stochastic parsing. In *HLT/NAACL'04*.
- S. Katrenko and P. Adriaans. 2006. Learning relations from biomedical corpora using dependency trees. In *KDECB*, pages 61–80.
- J.-D. Kim, T. Ohta, Y. Teteisi, and J. Tsujii. 2003. GE-NIA corpus — a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19:i180–182.
- D. Klein and C. D. Manning. 2003. Accurate unlexicalized parsing. In ACL 2003.
- D. Lin. 1998. Dependency-based evaluation of MINI-PAR. In *LREC Workshop on the Evaluation of Parsing Systems*.
- M. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1994. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- T. Matsuzaki, Y. Miyao, and J. Tsujii. 2005. Probabilistic CFG with latent annotations. In *ACL 2005*.
- R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL* 2006.
- T. Mitsumori, M. Murata, Y. Fukuda, K. Doi, and H. Doi. 2006. Extracting protein-protein interaction information from biomedical text with SVM. *IEICE - Trans. Inf. Syst.*, E89-D(8):2464–2466.

- Y. Miyao and J. Tsujii. 2008. Feature forest models for probabilistic HPSG parsing. *Computational Linguistics*, 34(1):35–80.
- Y. Miyao, K. Sagae, and J. Tsujii. 2007. Towards framework-independent evaluation of deep linguistic parsers. In *Grammar Engineering across Frameworks* 2007, pages 238–258.
- A. Moschitti. 2006. Making tree kernels practical for natural language processing. In EACL 2006.
- J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In ACL 2005.
- S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL* 2007.
- S. Pyysalo, T. Salakoski, S. Aubin, and A. Nazarenko. 2006. Lexical adaptation of link grammar to the biomedical sublanguage: a comparative evaluation of three approaches. *BMC Bioinformatics*, 7(Suppl. 3).
- S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, and T. Salakoski. 2007a. BioInfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(50).
- S. Pyysalo, F. Ginter, V. Laippala, K. Haverinen, J. Heimonen, and T. Salakoski. 2007b. On the unification of syntactic annotations under the Stanford dependency scheme: A case study on BioInfer and GENIA. In *BioNLP 2007*, pages 25–32.
- C. Quirk and S. Corston-Oliver. 2006. The impact of parse quality on syntactically-informed statistical machine translation. In *EMNLP* 2006.
- E. K. Ringger, R. C. Moore, E. Charniak, L. Vanderwende, and H. Suzuki. 2004. Using the Penn Treebank to evaluate non-treebank parsers. In *LREC 2004*.
- R. Sætre, K. Sagae, and J. Tsujii. 2007. Syntactic features for protein-protein interaction extraction. In *LBM 2007 short papers*.
- K. Sagae and J. Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *EMNLP-CoNLL* 2007.
- K. Sagae, Y. Miyao, T. Matsuzaki, and J. Tsujii. 2008. Challenges in mapping of syntactic representations for framework-independent parser evaluation. In the Workshop on Automated Syntatic Annotations for Interoperable Language Resources.
- D. D. Sleator and D. Temperley. 1993. Parsing English with a Link Grammar. In 3rd IWPT.
- Y. Tsuruoka, Y. Tateishi, J.-D. Kim, T. Ohta, J. Mc-Naught, S. Ananiadou, and J. Tsujii. 2005. Developing a robust part-of-speech tagger for biomedical text. In 10th Panhellenic Conference on Informatics.
- A. Yakushiji, Y. Miyao, Y. Tateisi, and J. Tsujii. 2005. Biomedical information extraction with predicateargument structure patterns. In *First International Symposium on Semantic Mining in Biomedicine*.