Discriminative Parse Reranking for Chinese with Homogeneous and Heterogeneous Annotations

Weiwei Sun[†‡] and **Rui Wang**[†] and **Yi Zhang**[†‡]
[†]Department of Computational Linguistics, Saarland University
[‡]German Research Center for Artificial Intelligence (DFKI)
D-66123, Saarbrücken, Germany
{wsun,rwang,yzhang}@coli.uni-saarland.de

Abstract

Discriminative parse reranking has been shown to be an effective technique to improve the generative parsing models. In this paper, we present a series of experiments on parsing the Tsinghua Chinese Treebank with hierarchically split-merge grammars and reranked with a perceptron-based discriminative model. In addition to the homogeneous annotation on TCT, we also incorporate the PCTB-based parsing result as heterogeneous annotation into the reranking feature model. The reranking model achieved 1.12% absolute improvement on F1 over the Berkeley parser on a development set. The head labels in Task 2.1 are annotated with a sequence labeling model. The system achieved 80.32 (B+C+H F1) in CIPS-SIGHAN-2010 Task 2.1 (Open Track) and 76.11 (Overall F1) in Task 2.2 (Open Track)[1].

1 Introduction

The data-driven approach to syntactic analysis of natural language has undergone revolutionary development in the last 15 years, ever since the first few large scale syntactically annotated corpora, i.e. treebanks, became publicly available in the mid-90s of the last century. One and a half decades later, treebanks remain to be an expensive type of language resources and only available for a small number of languages. The main issue that hinders large treebank development projects is the difficulties in creating a complete and consistent annotation guideline which then constitutes the very basis for sustainable parallel annotation and quality assurance. While traditional linguistic studies typically focus on either isolated language phenomena or limited interaction among a small groups of phenomena, the annotation scheme in treebanking project requires full coverage of language use in the source media, and proper treatment with an uniformed annotation format. Such high demand from the practical application of linguistic theory has given rise to a countless number of attempts and variations in the formalization frameworks. While the harsh natural selection set the bar high and many attempts failed to even reach the actual annotation phase, a handful highly competent grammar frameworks have given birth to several large scale treebanks.

The co-existence of multiple treebanks with heterogeneous annotation presents a new challenge to the consumers of such resources. The immediately relevant task is the automated syntactic analysis, or parsing. While many state-of-the-art statistical parsing systems are not bound to specific treebank annotation (assuming the formalism is predetermined independently), almost all of them assume homogeneous annotation in the training corpus. Therefore, such treebanks can not be simply put together when training the parser. One approach would be to convert them into an uniformed representation, although such conversion is usually difficult and by its nature error-

[1]This result is achieved with a bug-fixed version of the system and does not correspond to the numbers in the original evaluation report.

prune. The differences in annotations constitute different generative stories: i.e., when the parsing models are viewed as mechanisms to produce structured sentences, each treebank model will associate its own structure with the surface string independently. On the other hand, if the discriminative view is adopted, it is possible to use annotations in different treebanks as indication of goodness of the tree in the original annotation.

In this paper, we present a series of experiments to improve the Chinese parsing accuracy on the Tsinghua Chinese Treebank. First, we use coarse-to-fine parsing with hierarchically split-merge generative grammars to obtain a list of candidate trees in TCT annotation. A discriminative parse selection model is then used to rerank the list of candidates. The reranking model is trained with both homogeneous (TCT) and heterogeneous (PCTB) data. A sequence labeling system is used to annotate the heads in Task 2-1.

The remaining part of the paper is organized as follows. Section 2 reviews the relevant previous study on generative split-merge parsing and discriminative reranking models. Section 3 describes the work flow of our system participated in the CIPS-SIGHAN-2010 bake-off Task 2. Section 4 describes the detailed settings for the evaluation and the empirical results. Section 5 concludes the paper.

2 Background

Statistical constituent-based parsing is popularized through the decade-long competition on parsing the Wall Street Journal sections of the English Penn Treebank. While the evaluation setup has for long seen its limitation (a frustratingly low of 2% overall improvement throughout a decade of research), the value of newly proposed parsing methods along the way has clearly much more profound merits than the seemly trivial increase in evaluation figures. In this section we review two effective techniques in constituent-based statistical parsing, and their potential benefits in parsing Chinese.

Comparing with many other languages, statistical parsing for Chinese has reached early success, due to the fact that the language has relatively fixed word order and extremely poor inflectional morphology. Both facts allow the PCFG-based statistical modeling to perform well. On the other hand, the much higher ambiguity between basic word categories like nouns and verbs makes Chinese parsing interestingly different from the situation of English.

The type of treebank annotations also affects the performance of the parsing models. Taking the Penn Chinese Treebank (PCTB; Xue et al. (2005)) and Tsinghua Chinese Treebank (TCT; Zhou (2004)) as examples, PCTB is annotated with a much more detailed set of phrase categories, while TCT uses a more fine-grained POS tagset. The asymmetry in the annotation information is partially due to the difference of linguistic treatment. But more importantly, it shows that both treebanks have the potential of being refined with more detailed classification, on either phrasal or word categories. One data-driven approach to derive more fine-grained annotation is the hierarchically split-merge parsing (Petrov et al., 2006; Petrov and Klein, 2007), which induces subcategories from coarse-grained annotations through an expectation maximization procedure. In combination with the coarse-to-fine parsing strategy, efficient inference can be done with a cascade of grammars of different granularity. Such parsing models have reached (close to) state-of-the-art performance for many languages including Chinese and English.

Another effective technique to improve parsing results is discriminative reranking (Charniak and Johnson, 2005; Collins and Koo, 2005). While the generative models compose candidate parse trees, a discriminative reranker reorders the list of candidates in favor of those trees which maximizes the properties of being a good analysis. Such extra model refines the original scores assigned by the generative model by focusing its decisions on the fine details among already "good" candidates. Due to this nature, the set of features in the reranker focus on those global (and potentially long distance) properties which are difficult to model with the generative model. Also, since it is not necessary for the reranker to generate the candidate trees, one can easily integrate additional external information to help adjust the ranking of the analysis. In the following section, we will de-

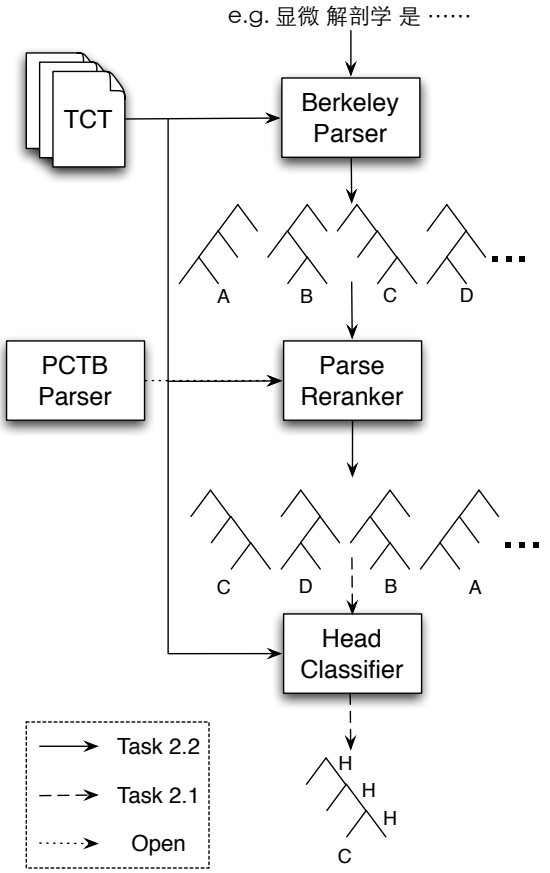

Figure 1: Workflow of the System

scribe the reranking model we developed for the CIPS-SIGHAN-2010 parsing tasks. We will also show how the heterogeneous parsing results can be integrated through the reranker to further improve the performance of the system.

3 System Description

In this section, we will present our approach in detail. The whole system consists of three main components, the Berkeley Parser, the Parse Reranker, and the Head Classifier. The workflow is shown in Figure 1. Firstly, we use the Berkeley Parser trained on the TCT to parse the input sentence and obtain a list of possible parses; then, all the parses[2] will be re-ranked by the Parse Reranker; and finally, the Head Classifer will annotate the head information for each constituent

[2]In practice, we only take the top n parses. We have different n values in the experiment settings, and n is up to 50.

Algorithm 1: The *Perptron* learning procedure.

input : Data $\{(\mathbf{x}_t, y_t), t = 1, 2, ..., m\}$
Initialize: $\mathbf{w} \leftarrow (0, ..., 0)$
for $i = 1, 2, ..., I$ **do**
  for $t =$SHUFFLE $(1, ..., m)$ **do**
    $y_t^* = \arg\max_{y \in \mathrm{GEN}_{\mathrm{n}}^{\mathrm{best}}(\mathbf{x}_t)} \mathbf{w}^\top \Phi(\mathbf{x}_t, y)$
    if $y_t^* \neq y_t$ **then**
      $\mathbf{w} \leftarrow \mathbf{w} + (\Phi(\mathbf{x}_t, y_t) - \Phi(\mathbf{x}_t, y_t^*))$
    end
  end
  $\mathbf{w}_i \leftarrow \mathbf{w}$
end
return aw $= \frac{1}{I} \sum_{i=1}^{I} \mathbf{w}_i$

on the best parse tree. For parse reranking, we can extract features either from TCT-style parses or together with the PCTB-style parse of the same sentence. For example, we can check whether the boundary predictions given by the TCT parser are agreed by the PCTB parser. Since the PCTB parser is trained on a different treebank from TCT, our reranking model can be seen as a method to use a heterogenous resource. The best parse tree given by the Parse Reranker will be the result for Task 2.2; and the final output of the system will be the result for Task 2.1. Since we have already mentioned the Berkeley Parser in the related work, we will focus on the other two modules in the rest of this section.

3.1 Parse Reranker

We follow Collins and Koo (2005)'s discriminative reranking model to score possible parse trees of each sentence given by the Berkeley Parser.

Previous research on English shows that structured perceptron (Collins, 2002) is one of the strongest machine learning algorithms for parse reranking (Collins and Duffy, 2002; Gao et al., 2007). In our system, we use the averaged perceptron algorithm to do parameter estimation. Algorithm 1 illustrates the learning procedure. The parameter vector $\mathbf{w}$ is initialized to $(0, ..., 0)$. The learner processes all the instances (t is from 1 to n) in each iteration (i). If current hypothesis ($\mathbf{w}$)

fails to predict $\mathbf{x}_t$, the learner update $\mathbf{w}$ through calculating the difference between $\Phi(\mathbf{x}_t, y_t^*)$ and $\Phi(\mathbf{x}_t, y_t)$. At the end of each iteration, the learner save the current model as $\mathbf{w} + i$, and finally all these models will be added up to get $\mathbf{aw}$.

3.2 Features

We use an example to show the features we extract in Figure 2.

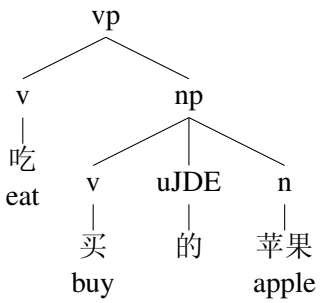

Figure 2: An Example

Rules The context-free rule itself: $\mathrm{np} \rightarrow \mathrm{v} + \mathrm{uJDE} + \mathrm{np}$.

Grandparent Rules Same as the Rules, but also including the nonterminal above the rule: $\mathrm{vp}(\mathrm{np} \rightarrow \mathrm{v} + \mathrm{uJDE} + \mathrm{np})$

Bigrams Pairs of nonterminals from the left to right of the the rule. The example rule would contribute the bigrams $\mathrm{np}(\mathrm{STOP}, \mathrm{v})$, $\mathrm{np}(\mathrm{v}, \mathrm{uJDE})$, $\mathrm{np}(\mathrm{uJDE}, \mathrm{np})$ and $\mathrm{np}(\mathrm{np}, \mathrm{STOP})$.

Grandparent Bigrams Same as Bigrams, but also including the nonterminal above the bigrams. For instance, $\mathrm{vp}(\mathrm{np}(\mathrm{STOP}, \mathrm{v}))$

Lexical Bigrams Same as Bigrams, but with the lexical heads of the two nonterminals also included. For instance, $\mathrm{np}(\mathrm{STOP}$,买$)$.

Trigrams All trigrams within the rule. The example rule would contribute the trigrams $\mathrm{np}(\mathrm{STOP}, \mathrm{STOP}, \mathrm{v})$, $\mathrm{np}(\mathrm{STOP}, \mathrm{v}, \mathrm{uJDE})$, $\mathrm{np}(\mathrm{v}, \mathrm{uJDE}, \mathrm{np})$, $\mathrm{np}(\mathrm{uJDE}, \mathrm{np}, \mathrm{STOP})$ and $\mathrm{np}(\mathrm{np}, \mathrm{STOP}, \mathrm{STOP})$.

Combination of Boundary Words and Rules The first word and the rule (i.e. 买$+(\mathrm{np} \rightarrow \mathrm{v} + \mathrm{uJDE} + \mathrm{np})$), the last word and the rule one word before and the rule, one word after and the rule, the first word, the last word and the rule, and the first word's POS, last word's POS and the rule.

Combination of Boundary Words and Phrasal Category : Same as combination of boundary words and rules, but substitute the rule with the category of current phrases.

Two level Rules Same as Rules, but also including the entire rule above the rule: $\mathrm{vp} \rightarrow \mathrm{v} + (\mathrm{np} \rightarrow \mathrm{v} + \mathrm{uJDE} + \mathrm{np})$

Original Rank : The logarithm of the original rank of n-best candidates.

Affixation features In order to better handle unknown words, we also extract morphological features: character n-gram prefixes and suffixes for n up to 3. For example, for word/tag pair 自然环境/n, we add the following features: (prefix1,自,n), (prefix2,自然,n), (prefix3,自然环,n), (suffix1,境,n), (suffix2,环境,n), (suffix3,然环境,n).

Apart from training the reranking model using the same dataset (i.e. the TCT), we can also use another treebank (e.g. the PCTB). Although they have quite different annotations as well as the data source, it would still be interesting to see whether a heterogenous resource is helpful with the parse reranking.

Consist Category If a phrase is also analyzed as one phrase by the PCTB parser, both the TCT and PCTB categories are used as two individual features. The combination of the two categories are also used.

Inconsist Category If a phrase is not analyzed as one phrase by the PCTB parser, the TCT category is used as a feature.

Number of Consist and Inconsist phrases The two number are used as two individual featuers. We also use the ratio of the number of consist phrases and inconsist phrase (we add 0.1 to each number for smoothing), the ratio of the number of consist/inconsist phrases and the length of the current sentence.

POS Tags For each word, the combination of TCT and PCTB POS tags (with or without word content) are used.

3.3 Head Classifier

Following (Song and Kit, 2009), we apply a sequence tagging method to find head constituents. We suggest readers to refer to the original paper for details of the method. However, since the feature set is different, we give the discription of them in this paper. To predict whether current phrase is a head phrase of its parent, we use the same example above (Figure 2) for convenience. If we consider np as our current phrase, the following features are extracted,

Rules The generative rule, $\mathrm{vp} \rightarrow \mathrm{v} + (\mathrm{np})$.

Category of the Current Phrase and its Parent np, vp, and $(\mathrm{np}, \mathrm{vp})$.

Bigrams and Trigrams $(\mathrm{v}, \mathrm{np})$, $(\mathrm{np}, \mathrm{STOP})$, $(\mathrm{STOP}, \mathrm{v}, \mathrm{np})$, and $(\mathrm{np}, \mathrm{STOP}, \mathrm{STOP})$.

Parent Bigrams and Trigrams $\mathrm{vp}(\mathrm{v}, \mathrm{np})$, $\mathrm{vp}(\mathrm{np}, \mathrm{STOP})$, $\mathrm{vp}(\mathrm{STOP}, \mathrm{v}, \mathrm{np})$, $\mathrm{vp}(\mathrm{np}, \mathrm{STOP}, \mathrm{STOP})$.

Lexical Unigram The first word 买, the last word 苹果, and together with the parent, (vp,买) and (vp,苹果)

4 Evaluation

4.1 Datasets

The dataset used in the CIPS-ParsEval-2010 evaluation is converted from the Tsinghua Chinese Treebank (TCT). There are two subtasks: (1) event description sub-sentence analysis and (2) complete sentence parsing. On the assumption that the boundaries and relations between these event description units are determined separately, the first task aims to identify the local fine-grained syntactic structures. The goal of the second task is to evaluate the performance of the automatic parsers on complete sentences in real texts. The training dataset is a mixture of several genres, including newspaper texts, encyclopedic texts and novel texts.

The annotation in the dataset is different to the other frequently used Chinese treebank (i.e. PCTB) Whereas TCT annotation strongly reflects early descriptive linguistics, PCTB draws primarily on Government-Binding (GB) theory from 1980s. PCTB annotation differs from TCT annotation from many perspectives:

- TCT and PCTB have different segmentation standards.
- TCT is somehow branching-rich annotation, while PCTB annotation is category-rich. Specifically the topological tree structures is more detailed in TCT, and there are not many flat structures. However constituents are detailed classified, namely the number of phrasal categories is small. On the contrary, though flat structures are very common in PCTB, the categorization of phrases is fine-grained. In addition, PCTB contains functional information. Function tags appended to constituent labels are used to indicate additional syntactic or semantic information.
- TCT contains head indices, making head identification of each constituent an important goal of task 1.
- Following the GB theory, PCTB assume there are *movement*s, so there are empty category annotation. Because of different theoretical foundations, there are different explanations for a series of linguistic phenomena such as the usage of function word “的”.

In the reranking experiments, we also use a parser trained on PCTB to provide more syntactic clues.

4.2 Setting

In order to gain a representative set of training data, we use cross-validation scheme described in (Collins, 2000). The dataset is a mixture of three genres. We equally split every genre data into 10 subsets, and collect three subset of different genres as one fold of the whole data. In this way, we can divide the whole data into 10 balanced subsets. For each fold data, a complement parser is trained using all other data to produce multiple hypotheses for each sentence. This cross-validation

n	1	2	5	10	20	30	40	50
F1	79.97	81.62	83.51	84.63	85.59	86.07	86.38	86.60

Table 1: Upper bound of f-score as a function of number n of n-best parses.

scheme can prevent the initial model from being unrealistically "good" on the training sentences. We use the first 9 folds as training data and the last fold as development data for the following experiments. For the final submission of the evaluation task, we re-train a reranking model using all 10 folds data. All reranking models are trained with 30 iterations.

For parsing experiments, we use the Berkeley parser[3]. All parsers are trained with 5 iterations of split, merge, smooth. To produce PCTB-style analysis, we train the Berkeley parse with PCTB 5.0 data that contains 18804 sentences and 508764 words. For the evaluation of development experiments, we used the EVALB tool[4] for evaluation, and used labeled recall (LR), labeled precision (LP) and F1 score (which is the harmonic mean of LR and LP) to measure accuracy.

For the head classification, we use SVM$^{\text{hmm5}}$[5], an implementation of structural SVMs for sequence tagging. The main setting of learning parameter is C that trades off margin size and training error. In our experiments, the head classification is not sensitive to this parameter and we set it to 1 for all experiments reported. For the kernel function setting, we use the simplest linear kernel.

4.3 Results

4.3.1 Upper Bound of Reranking

The upper bound of n-best parse reranking is shown in Table 1. From the 1-best result we see that the base accuracy of the parser is 79.97. 2-best and 10-best show promising oracle-rate improvements. After that things start to slow down, and we achieve an oracle rate of 86.60 at 50-best.

4.3.2 Reranking Using Homogeneous Data

Table 2 summarizes the performance of the basic reranking model. It is evaluated on short sentences (less than 40 words) from the development data of the task 2. When 40 reranking candidates are used, the model gives a 0.76% absolute improvement over the basic Berkeley parser.

	POS(%)	LP(%)	LR(%)	F1
Baseline	93.59	85.60	85.36	85.48
$n = 2$	93.66	85.84	85.54	85.69
$n = 5$	93.62	86.04	85.73	85.88
$n = 10$	93.66	86.22	85.85	86.04
$n = 20$	93.70	86.19	85.87	86.03
$n = 30$	93.70	86.32	86.00	86.16
$n = 40$	93.76	86.40	86.09	86.24
$n = 50$	93.73	86.10	85.81	85.96

Table 2: Reranking performance with different number of parse candidates on the sentences that contain no more than 40 words in the development data.

4.3.3 Reranking Using Heterogeneous Data

Table 3 summarizes the reranking performance using PCTB data. It is also evaluated on short sentences of the task 2. When 30 reranking candidates are used, the model gives a 1.12% absolute improvement over the Berkeley parser. Comparison of Table 2 and 3 shows an improvement by using heterogeneous data.

	POS(%)	LP(%)	LR(%)	F1
$n = 2$	93.70	85.98	85.67	85.82
$n = 5$	93.75	86.52	86.19	86.35
$n = 10$	93.77	86.64	86.29	86.47
$n = 20$	93.79	86.71	86.34	86.53
$n = 30$	93.80	86.72	86.48	86.60
$n = 40$	93.80	86.54	86.22	86.38
$n = 50$	93.89	86.73	86.41	86.57

Table 3: Reranking performance with different number of parse candidates on the sentences that contain no more than 40 words in the development data.

[3] http://code.google.com/p/berkeleyparser/

[4] http://nlp.cs.nyu.edu/evalb/

[5] http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html

Task 1	"B+C"-P	"B+C"-R	"B+C"-F1	"B+C+H"-P	"B+C+H"-R	"B+C+H"-F1	POS
Old data	82.37	83.05	82.71	79.99	80.65	80.32	81.87

Table 4: Final results of task 1.

Task 2	dj-P	dj-R	dj-F1	fj-P	fj-R	fj-F1	Avg.	POS
Old data	79.37	79.27	79.32	71.06	73.22	72.13	75.72	81.23
New data	79.60	79.13	79.36	70.01	75.94	72.85	76.11	89.05

Table 5: Final results of task 2.

4.3.4 Head Classification

The head classification performance is evaluated using gold-standard syntactic trees. For each constituent in a gold parse tree, a structured classifier is trained to predict whether it is a head constituent of its parent. Table 6 shows the overall performance of head classification. We can see that the head classification can achieve a high performance.

P(%)	R(%)	$F_{\beta=1}$
98.59%	98.20%	98.39

Table 6: Head classification performance with gold trees on the development data.

4.3.5 Final Result

Table 4 and 5 summarize the final results. Here we use the reranking model with heterogeneous data. The second line of Table 5 shows the offical final results. In this submission, we trained a model using an old version of training data. Note that, the standard of POS tags of the "old" version is different from the latest version which is also used as test data. For example, the name of some tags are changed. The third line of Table 4[6] shows the results predicted by the newest data[7]. This result is comparable to other systems.

5 Conclusion

In this paper, we described our participation of the CIPS-SIGHAN-2010 parsing task. The generative coarse-to-fine parsing model is integrated with a discriminative parse reranking model, as well as a head classifier based on sequence labeling. We use the perceptron algorithm to train the reranking models and experiment with both homogenous and heterogenous data. The results show improvements over the baseline in both cases.

[6]There are two sentences that are not parsed by the Berkeley parser. We use a simple strategy to solve this problem: We first roughly segment the sentence according to punctuation; Then the parsed sub-sentences are merged as a single *zj*.

[7]We would like to thank the organizer to re-test our new submission.

Acknowledgments

The first author is supported by the German Academic Exchange Service (DAAD). The second author is supported by the PIRE scholarship program; the third author thanks DFKI and the Cluster of Excellence on Multimodal Computing and Interaction for their support of the work.

References

Charniak, E. and M Johnson. 2005. oarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, pages 173–180.

Collins, Michael and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Collins, Michael and Terry Koo. 2005. Discriminative reranking for natural language parsing. In *Computational Linguistics*, volume 31(1), pages 25–69.

Collins, Michael. 2000. Discriminative reranking for natural language parsing. In *Computational Linguistics*, pages 175–182. Morgan Kaufmann.

Collins, Michael. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *EMNLP '02:*

Proceedings of the ACL-02 conference on Empirical methods in natural language processing, pages 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Gao, Jianfeng, Galen Andrew, Mark Johnson, and Kristina 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.

Petrov, S. and D. Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL-2007*, Rochester, NY, USA, April.

Petrov, Slav, 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.

Song, Yan and Chunyu Kit. 2009. Pcfg parsing with crf tagging for head recognition. In *Proceedings of the CIPS-ParsEval-2009*.

Xue, Nianwen, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Zhou, Qiang. 2004. Annotation scheme for chinese treebank (in chinese). *Journal of Chinese Information Processing*, 18(4):1–8.