Improve Chinese Parsing with Max-Ent Reranking Parser

Ling-Ya Huang
Department of Computer Science
Brown University, Providence, RI
lenorah@cs.brown.edu

Abstract

With the high-performance of statistical parsers on English (Charniak and Johnson, 2005), we are interested in adapting the parser to different languages. This report focuses on the improvement tasks for Chinese parsing using a Max-Ent reranking parser (Charniak parser). After the adaption to Chinese, the parser reached an f-score of 78.02% on *Chinese Treebank 4.0* and 82.21% on *Chinese Treebank 5.1*. We also tried the self-training techniques on Chinese parsing. The experimental setups will be described in detail along with the results and the analysis.

I. Introduction

Parsing has been a fundamental step for today's natural language understanding tasks. The state of the art parser (Charniak and Johnson, 2005) reaches 91 f-score on English Penn Treebank, using a generative model to produce 50-best parses list, and feeds these parses into a MaxEnt reranker to select the best parse based on the features extracted from the parses. While the parser uses some lexical information such as head of the tree to help parsing, the parser's mechanism of maximum-entropy model is essentially language neutral. After we adapted the parser to Chinese with few modifications of language dependent features, the parser worked well overall on Chinese as described in Lian's work (Lian, 2005). However, there was still room for improvement. In Figure1, the right tree is a non-PennTreebank sentence parsed by the parser with reranker after adaption to Chinese. The VP-NP attachment is not correct since we're expecting to see the left tree. After further digging into the particular problematic sentence with very detailed probabilities breakdowns, there were still no obvious defects in the calculations. Therefore, we switched gears to focus on the general improvements of the Chinese parsing of Max-Ent reranking parser. In the following sections this report will describe the attempts to adapt self-training on Chinese parsing and the results. Also we will talk more about the Chinese adaption process and the comparison with previous and others' work.

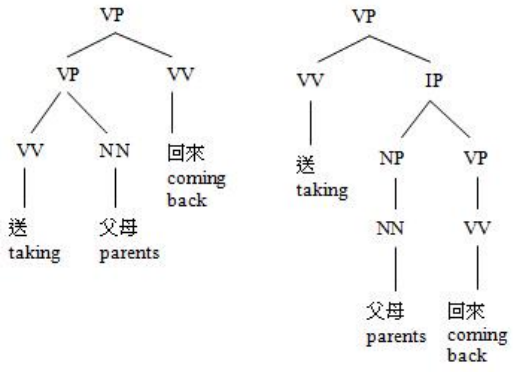

Figure 1: On the left is the tree we expect to see after parsing. On the right is the tree we get. The VP-NP attachment seem to be opposite.

II. Self-Training experiments

The self-training techniques have been proved effective on English parsing in McClosky's work (2006), and it is natural to then try on a non-English language and see if the same effects can also apply on Chinese. The experimental setups comprised two elements, the selection of the external corpus opposed to our baseline corpus *CTB4.0*, and the mixing ratio of the two corpus or the weighting of the baseline corpus.

Corpus selection

At first we tried on LDC data *English Chinese Parallel Financial News*, and soon we realized it is a bad choice as its financial news characteristics consists of lots of tables and charts of numbers, which increases the complexity of the corpus and does not help to improve the accuracy of the parsing. We then chose another LDC corpus data, *Chinese Gigaword Third Edition*. This corpus is a comprehensive archives of newswire text data, containing four sources: Agence France Presse (afp_cmn), Central News Agency, Taiwan, Xinhua News Agency, Zaobao Newspaper. These sources correspond to foreign translated news in China, local news in China, in Taiwan, and in Singapore. Comparing to the sources newswire data in CTB4.0, the Gigaword corpus overlapped at some domain of the source, but still considered a different external corpus against the Chinese Penn Treebank data.

To prepare the data as the parsing input material, some preliminary cleanup tasks needed to be done. Since the raw data format varies from newswire with html tags to conversational blog data, we created the program Dataproc to handle all the different data formats and convert the formats to the standard parsed-ready form.

			Gigaword sentences added							
			CTB alone	25	125	250	1250	2500	12500	25000
CTB 4.0 sentences (multiples) used	3000	(0.25x)	71.77	71.78	71.99	71.93	72.7	72.94	74.41	74.57
	4500	(0.37x)	73.12	73.08	73.15	73.24	73.9	73.91	74.74	75.19
	6000	(0.5x)	74.2	74.22	74.46	74.46	74.88	74.93	75.38	75.67
	7500	(0.62x)	75.33	75.27	75.31	75.2	75.4	75.52	75.7	76.09
	9000	(0.75x)	75.86	75.84	75.96	75.83	75.87	76.18	76.02	76.3
	10500	(0.87x)	76.31	76.31	76.18	76.21	76.24	76.2	76.2	76.23
	12334	(1x)	76.73	76.7	76.79	76.75	76.64	76.58	76.59	76.56
	123340	(10x)		76.53	76.38	76.33	76.95	77.08	76.96	77.02
CTB 5.1 sentences used	18104	(1x)	80.85	81.04	80.96	80.72	81.42	81.63	81.6	82.42
	181040	(10x)		80.58	80.76	80.39	80.8	80.81	81.21	81.54
	16676	(1x w/ 2xtest)	79.51	79.53	79.6	79.37	79.35	79.53	80.16	80.57
	17163	(1x w/ 4xtest)	78.79	78.85	78.87	78.83	78.91	79.29	79.87	80.86

Table1: The f-measure of the experiments matrix, showing performance of different models. The experiments were run on both CTB4.0 and CTB5.1. For CTB4.0, we differed the training data size to compare the results. For CTB5.1, the difference were set up on testing data size.

Mixing the corpus

Among all the parsed-ready data we have, including financial news and blog texts, we chose Gigaword as the external corpus to mix with CTB. In particular, the translated news (afp_cmn) was used. There are over 25,000 sentences in afp_cmn, and to self-train on these corpus, we first need to parse all the sentences to produce the parses as the training data for the next step. We then split the parsed results into 1000 units with 25 parses per unit, and in this way we can adjust the weightings of the self-train corpus by adding any multiples of 25 sentences. For the Chinese Treebank4.0, our training data has 12334 sentences, and for the same purpose of adjusting the weightings, we also split the CTB4 training file into different sizes starting from 3000 sentences and increased the sentence counts by 1500 for each Gigaword weighting option. We also run self-training on CTB5 but with different experimental setups. Besides of the baseline setup of 1xCTB and 10xCTB, different train/test data set split up were applied on CTB5. The complete experimental settings are shown in the matrix of Table1. Following the data selection, the data was then mixed to generate the language models. A model was trained by the concatenation file of CTB and Gigaword for each combination. To evaluate the models, the test section of CTB was parsed by the parser with different models. The entire process is illustrated in Figure 2.

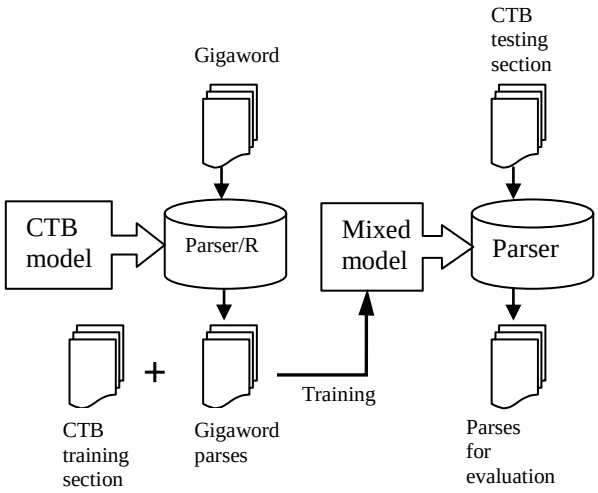

Figure2: The illustration of self-training experiment process. CTB model refers to Chinese Treebank 4.0 trained model, and Mixed Model refers to the model trained by the concatenation file of CTB+Gigaword parses. The first Parser marked as Parser/R means it's Parser+Reranker. The second Parser is the first stage parser only.

The results listed in Table 1 seem to indicate that self-training helped on small size corpus, but as the corpus size grew larger, the effects diminished and even hurt the performance by adding too many self-trained data. As we can see in Figure 3, the lines of 3000, 4500, and 6000 CTB4 sentences climb up as we added more self-trained sentences, the f-score improves up to 3% for 3000 CTB4 sentences with 25,000 Gigaword sentences added, and the lines representing CTB4 sentences over 10500 have no improvements while the Gigaword sentence counts increase. However, the small size corpus may not be statistically significant since it is trivial for machine learning experiments that the more training data we have, the higher performance we get. The ratio of the Gigaword/CTB corpus for the effective cases reach as high as 8 (25,000/3000), and the 3 small corpus lines all end up lower than the larger ones. Both observations imply the performance improvement might just simply be an effect of the

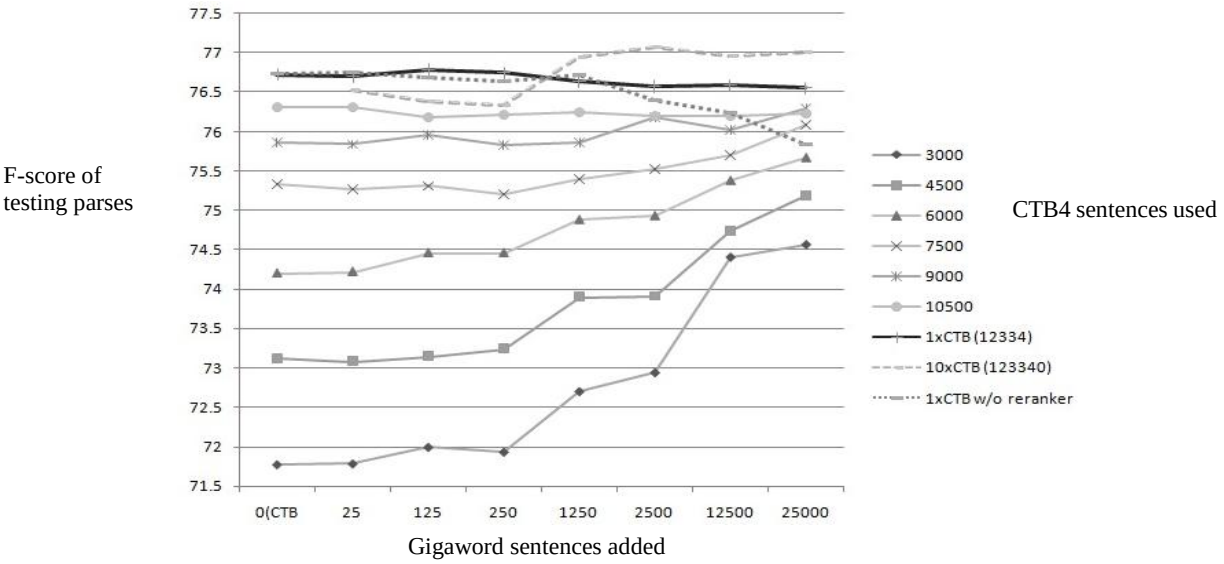

Figure 3: Every data point indicates a f-measure obtained by the testing data parsed with the model. For instance, the first data point represents the f-score 71.73 (the detail numbers are listed at Table 1) of testing data parsed with the (3000,0) model, which means the model is trained by 3000 CTB4 sentences without adding any Gigaword sentences.

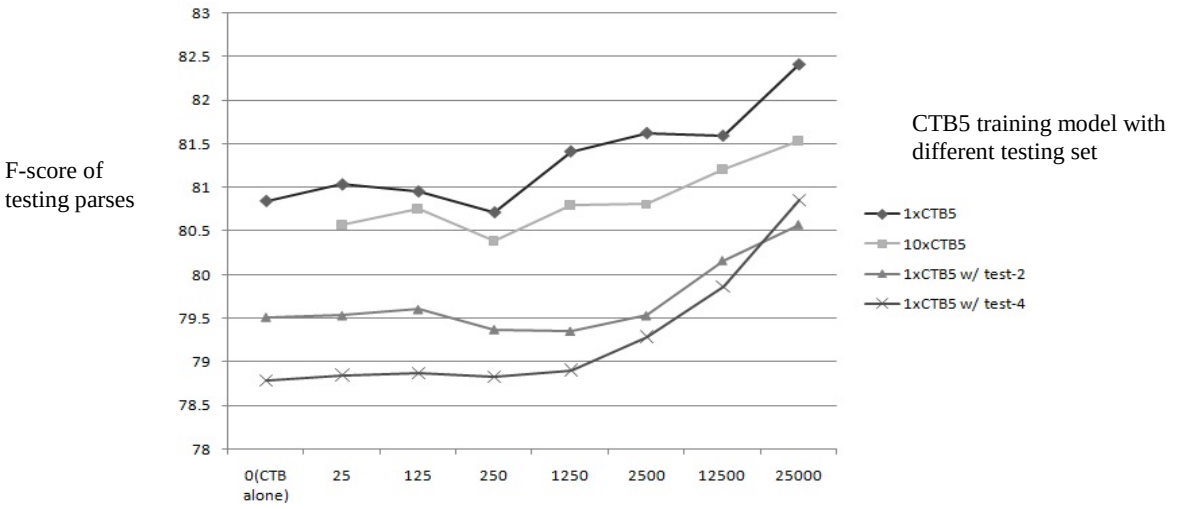

Figure 4: The f-scores of self-training on CTB5. 1xCTB5 and 10xCTB used the same test data set. The lower two lines used different train/test split up. See Table 3 for detailed setup.

small corpus size, which is not relevant to the self-training techniques. On the other hand, if we look at results of CTB5 in Figure 4, the average improvement is about 1.5%. The upper 2 lines represent the base models of 1x and 10x CTB as we did on CTB4. The lower 2 lines are parsing results for different train/test split up. Because of the small size of the test data set, we wanted to make sure the improvement was independent from the test data size. We split up the training and testing data from different article sections. The detailed setup is listed in Table 3. Judging from the plot, we would conclude that self-training help on the parsing accuracy regarding to the CTB5 data, which is inconsistent with the results of CTB4. We will further discuss this issue in the later sections.

Besides of the baseline model weightings, we also tried some other experiments to get more statistical numbers. To rule out the impacts of the corpus difficulties, we take different chunks of the CTB4 training file of the same size (3000 sentences) to be mixed with Gigaword sentences. This is a simplified sanity check to verify that the improvements of the smaller corpus were not because the first half of the training data are easy sentences for parsing. The results in Table2 showed the difficulties of the sentences are quite even. Another interesting experiment was made to reverse the ratio of Gigaword/CTB, which means using heavy CTB with light weights Gigaword, to

see if the results differ. The CTB training file was multiplied 10 times and then concatenated with Gigaword parses. The results show some improvements for both CTB4 and CTB5, but again, not consistent enough to reach a conclusion. Note that the first data point at 0 (CTB only) were missing in the 10x model, and it is due to the unknown words factoring. While the training data was copied 10 times, the sparse words became more frequent since their counts also multiplied by 10, and this event would then make the parser blinded with the unknown words at the training stage and complain about it while an unknown word occurred during the parsing stage. Since the 10x model is just a scaled model of 1x, we think it is valid to drop the experiment of model 0 (CTB only) and start the 10x experiments from model 1 (adding 25 Gigaword sentences). The third experiment is to test if the reranker has any effects to the self-training process. For the first parsing process, we used parser (first-stage) only instead of using parser plus reranker to get the Gigaword parses. As the line in Figure 3 shows, to self-train with non-reranked parses actually hurt. The performance dropped as we added more Gigaword. Finally, there's one data inconsistency worth noting for 1x CTB only model. The best f-score we have reached for on CTB4 is 76.94, but here in the self-training experiments we only got 76.73. We are not sure if the 0.2% drop was due to the experimental noises or if there is an error in the self-training process to cause this fall of performance.

	0 (CTB alone)	125	2500	12500	25000
1	71.31	71.60	72.08	73.17	73.74
2	69.23	69.57	71.02	72.77	73.51
3	71.51	71.95	72.59	73.48	73.92
4	71.28	71.55	72.34	73.35	73.95

Table2: The first column indicates the chunk order of 3000 sentences in CTB training file, which corresponding to sentence 1 to 3000, sentence 3001 to 6000, and so on. The first row is the Gigaword sentences added. The f-scores inside the matrix shows the parsing difficulty is even, for the 4 chunks.

CTB	Training	Dev	Test
4.0	12,334	1,456	1,378
5.1 (Article Range)	18,104 (1-270, 400-1151)	352 (301-325)	348 (271-300)
	17,163 (1-300, 500-1151)	352 (301-325)	776 (400-454)
	16,676 (1-300, 400-799, 900-1151)	352 (301-325)	1,289 (800-885)

Table3: The experimental setup comparison between CTB4.0 (used in Bikel2004, Charniak2005) and CTB5.1 (used in Petrov2007, Chiang2002, Xue2002). The numbers are the sentences counts used in the train/dev/test set. For CTB4.0 we don't have the article sections information, and for CTB5.1, the article sections were listed below the sentence counts.

Parser	LP	LR	F
CTB 4.0			
Charniak 2005			77.00
Charniak 2009	78.6	75.3	76.94
Charniak (reranked) 2005			78.40
Charniak (reranked) 2009	80.2	75.9	78.02
Bikel 2004	79.0	74.7	76.80
CTB 5.1			
Chiang and Bikel 2002*	78.0	75.2	76.58
Petrov 2007	84.8	81.9	83.32
Charniak 2009	82.1	79.6	80.85
Charniak (reranked) 2009	83.8	80.8	82.27
CTB4 test data on CTB 5.1 model			
Petrov 2007	89.7	85.6	87.58
Charniak 2009	92.3	88.8	90.54

* Chiang and Bikel 2002 might be using CTB1.0 or 2.0, not CTB5.1. However, the training/dev/test sections split-up described in the paper are identical with other parsers using CTB5.1. Since CTB5.1 is the superset of earlier versions, the numbers are comparable.

Table4: The test data parsing performance compared to other and previous work.

III. Parser adaption to Chinese

As mentioned in previous sections, to adapt the Charniak parser for Chinese, the main task is to identify the language-dependent lexical information, namely the head finding rules that we need to modify for Chinese. As a continuing work of Lian's paper, we modified the heads.cc and heads.h according to the headsinfo.txt used in the first stage. Also we identified several corpus errors in CTB4 during the training process and resolved the parser failures. As we ported the codes to train and test on CTB5.1, we also added handling code in reranker for functional tags first seen in corpus. After all the miscellaneous preliminary tasks were done, the parser was ready to parse Chinese on the data sets listed in Table3. The results are shown in Table4.

There are several numbers we can discuss in Table4. First, to compare the results with Charniak parser 2005, the f-scores drop 0.04% on parser and 0.38% on reranker on CTB4.0. The performance difference in parser might just be noises, but the reranker difference was not negligible. We need to further investigation on the issue. To compare the results with other statistical constituents parsers, Berkeley parser is one of the best choices since it reports the f-score of 83.32 on Chinese parsing and claims it is the state of the art. However, the numbers was run on different corpus, which motivated us to run the experiments on CTB5.1. With the same data set split up, our work reaches 82.21, which is about 1 point lower.

To further compare our work with Berkeley parser, we also

setup the experiments to run CTB4 on Berkeley parser but failed because it currently does not support training on a single Chinese file. Therefore, instead of retraining Berkeley parser with CTB4, we used their built-in CTB5 grammar (model) and run CTB4 test data on it. The f-score got as high as 87.34, and the same test/train experiment setups even reached 90.54 on our work. Nevertheless, we can clearly see in Table3, the training set in CTB5 is about 50% larger than CTB4. Normally with a larger training data size, better parsing performances are expected to be seen. Another evidence of the benefits of a larger training data set is the feature extraction counts in reranker. While it extracts 195,578 features from CTB4 training data, the feature extractions number of CTB5 is 572,574, almost 3 times larger than CTB4.

IV. Future work and issues discussion

For the future work, we recommend starting the experiments with CTB6.0 for its convenient supporting of Unicode encoding, which will save a lot of encoding conversion works. As for the punctuations--although making the POS the same as actual punctuations instead of tagged as PU, as used in CTB, might help a bit for parsing accuracy--the tagging accuracy could be affected. Also it caused some latent bugs in the parsers due to the incompliance with the gold files. The final thing needing review is the head finding rules of Chinese. The rules seem to work fairly, but they should be checked with more solid linguistics background knowledge.

In this report, we have walked through the self-training experiments and the parser adaption work to parse Chinese. Overall self-training works well on small data size, and also makes some improvements on CTB5. The results are inconsistent for the two corpus (CTB5 and CTB4) we run on and therefore we cannot reach a conclusion of whether self-training helps on Chinese parsing or not. However, we think the result look promising and we believe with more delicate research work, we can eventually make good improvements for Chinese parsing performance.

.

REFERENCES

Eugene Charniak. A maximum-entropy-inspired parser. In Proceedings of the first conference on North American chapter of the Association for Computational Linguistics, pages 132-139, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

Eugene Charniak and Mark Johnson. 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 2005. Association for Computational Linguistics.

Heng Lian. Chinese Language Parsing with Maximum-Entropy-Inspired Parser. Master's thesis, Brown University, Providence, RI, 2005.

David McClosky, Eugene Charniak, and Mark Johnson. 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 2006. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. Reranking and Self-Training for Parser Adaptation. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL'06), pages 337-344, Sydney, Australia, July 2006. Association for Computational Linguistics**.**

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning Accurate, Compact, and Interpretable Tree Annotation. In proceedings of COLING-ACL 2006.

Slav Petrov and Dan Klein**.** Improved Inference for Unlexicalized Parsing. In proceedings of HLT-NAACL 2007.

Daniel M. Bikel, and David Chiang. 2000. Two statistical parsing models applied to the Chinese Treebank. In Martha Palmer, Mitch Marcus, Aravind Joshi, and Fei Xia, editors, Proceedings of the Second Chinese Language Processing Workshop, pages 1–6, Hong Kong.

Daniel M. Bikel, 2004. On the Parameter Space of Lexicalized Statistical Parsing Models. Ph.D Thesis, University of Pennsylvania