LEARNING A LIGHTWEIGHT ROBUST DETERMINISTIC PARSER

Aboy WONG and Dekai WU

Human Language Technology Center
Department of Computer Science
University of Science and Technology
Clear Water Bay, Hong Kong
{aboy,dekai}@cs.ust.hk

ABSTRACT

We describe a method for automatically learning a parser from labeled, bracketed corpora that results in a fast, robust, lightweight parser that is suitable for real-time dialog systems and similar applications. Unlike ordinary parsers, all grammatical knowledge is captured in the learned decision trees, so no explicit phrase-structure grammar is needed. Another characteristic of the architecture is robustness, since the input need not fit pre-specified productions. Even without using specific lexical features, we have achieved respectable labeled bracket accuracies of about 81% precision and 82% recall. Processing speed is more than 500 words per CPU second. We keep the parameter space small (in comparison to other statistically learned parsers) by using only part-of-speech tags and constituent labels as features. Without any optimization, the decision trees consume only 6M of memory, making it possible to run on platforms with limited memory. The learning method is readily applicable to other languages. Preliminary experiments on a Chinese corpus (which contains about 3000 sentences from Chinese primary school text) have yielded results comparable to that for English.

1. INTRODUCTION

In the past decade, significant advances in parsing by incorporating statistical methods have been reported. Some concentrated on improving scores for ranking all possible parses [12, 8, 9, 17], while some worked on search algorithms and pruning strategies to reduce the search space [11, 4].

However, most of the parsing algorithms are still based on classical methods that are non-deterministic in nature. Thus, they may not be fast enough to give instant response as required by most on-line applications.

In fact, deterministic parsing is not impossible. Deterministic parsing is first hypothesized by Marcus in his thesis [15]: Natural language can be parsed by a mechanism that operates "strictly deterministically" in that it does not simulate a non-deterministic machine...

The ultimate goal of this work is to develop a deterministic parser. Apart from being accurate, lightweight and fast, our parser is also robust, automatically learned and readily applicable to other languages.

This paper is organized as follows. Section 2 and 3 describes the base parsing model and the improved parsing model. Section 4 gives the experimental results. Section 5 compares the performance of our parser with other parsers. Section 6 concludes this paper.

2. BASE PARSING MODEL

The basis of our parser is a shift-reduce parser [1] consisting of a stack, an input stream and a decision control mechanism. The core part of our work is to learn the decision control mechanism, for which we employ a novel Shift/Reduce decision algorithm and a novel Constituent Labeling decision algorithm.

2.1. Child-Type Tagging

In ordinary shift-reduce parsers, a Reduce action refers to a specific production, and it groups constituents in a way that matches the right-hand-side of the production. However, we have no explicit productions in our system. Instead, a Reduce action chooses a group of constituents as determined by a tagging scheme inspired to some extent by [12, 16]. Whenever a new constituent is pushed onto the stack, either by a Shift action or a Reduce action, a decision tree is used to tag it with one of four Child-Type values:

- UNARY, which means the constituent is the only child of its parent.
- LEFT, which means the constituent is the left-most child of its parent.
- RIGHT, which means the constituent is the right-most child of its parent.
- MID, which means the constituent is a middle child of its parent.

Assigning the constituent a Child-Type tag directly determines whether to Shift or Reduce. LEFT or MID signifies an incomplete constituent and therefore represents a Shift action. Conversely, RIGHT or UNARY represents a Reduce action where a single constituent is grouped in the case of a UNARY tag, or otherwise all constituents up to the top LEFT tag in the stack.

2.2. Constituent Labeling

Apart from making decisions on the constituent boundaries, we need to to assign a label to every constituent. In ordinary shift-reduce parsers, the label of the new constituent formed by a Reduce action is given by the left-hand-side of

the production. Again, since we have no explicit productions, an alternative method is required. The solution is to introduce a second decision tree to predict the constituent label.

3. IMPROVED PARSING MODEL

We improved the base parsing model by introducing a Base NP model and expanding the POS tag for prepositions.

3.1. Base NP Model

Base NP is a noun phrase that doesn't include any recursive NPs. Lots of works focusing on Base NP bracketing has been reported [16, 8, 5]. It is commonly believed that finding Base NP before parsing can improve parsing accuracy.

In our NP model, the tag set used is similar to our Child-Type Tag system. The tags includes:

- UNARY, means the word itself is a Base NP
- LEFT, means the word is the left-most child of a Base NP
- RIGHT, means the word is the right-most child of a Base NP
- MID, means the word is a middle child of a Base NP
- OUT, means the word is outside of a Base NP

A separate decision tree is learned for this task.

3.2. Preposition Tag Expansion

Prepositional Phrase attachment is another area which receives lot of attentions [3, 7, 18, 20]. Lexical or semantic features are used to tackle the problem. However, in our model, we restrict ourselves to simple syntactic features in order to keep our model slim. So we anticipate a poor performance on prepositional phrase attachments.

To remedy that, we expands the current POS tag set as follows. Preposition that appears more than 100 times in the training data are extracted. There are totally 49 of them. We create a new POS tag for each of them, so at the end, we have 49 POS tags representing the 49 most frequent prepositions and a POS tag representing the rest of prepositions.

4. EXPERIMENTS AND RESULTS

We use parsed sentences from sections 2–21 (about 40000 sentences) of the Penn Wall Street Journal corpus, release 2 [14], as the training data and section 23 (about 2400 sentences) as the testing data. The PARSEVAL measures [10] are used as the evaluation criteria:

$$\textbf{Labeled Precision} = \frac{\#correct\ constituents\ proposed}{\#constituents\ proposed}$$

$$\textbf{Labeled Recall} = \frac{\#correct\ constituents\ proposed}{\#constituents\ in\ treebank\ parse}$$

Crossing Brackets(CBs) = #constituents which violate constituent boundaries with a constituent in the Treebank parse.

A constituent is correct if and only if it spans the same set of words (ignoring punctuation, i.e. all tokens tagged as commas, colons or quotes) and has the same label as a constituent in the Treebank parse.

m, n	#nodes in Decision Tree	error rate
3, 3	148k	6.274%
4, 3	160k	6.107%
4, 4	162k	6.092%
5, 3	170k	6.011%
5, 4	169k	6.005%

Table 1: Results of Child-Type tagging

m, n	#nodes in Decision Tree	error rate
3, 2	24k	2.017%
4, 3	26k	2.027%

Table 2: Results of Constituent Labeling

4.1. Base Model:Child-Type Tagging Results

Below is the features that we used in training the Child-Type tagging module.

$s_1, ..., s_m, t_1, ..., t_m, c_1, c_2, i_1, ..., i_n$, where:

- s_1 represent the label of the constituent on the top of stack,
- s_k represent the label of k-th constituent counted from the top of the stack,
- t_k is the corresponding Child-Type tag of s_k,
- c_1 and c_2 is the first and last child label of s_1,
- i_k is the POS tag of the $k-th$ word in the input

We have run a couple of experiments by varying m and n. The training examples are extracted from section 2–21, and the testing examples are extracted from section 22. Table 1 shows the error rates of the Child-Type tagging results.

4.2. Base Model:Constituent Labeling Results

In the Constituent Labeling module, we use the followings as features,

$s_2, ..., s_m, t_2, ...t_m, c_1, c_2, c_3, c_4, i_1, ...i_n$, where:

- s_k, t_k and i_k has the same meaning as in the Child-Type Tag model,
- c_1, c_2, c_3 and c_4 is the first, second, second last and last child of the top constituent in stack

Table 2 shows the labeling results with different values of m and n. Training data comes form section 2–21 and testing data comes from section 22 of the Penn Treebank.

4.3. Parsing Results

We use the best model from each of the above 2 modules to build our parser. Table 3 shows the parsing result of the Base Model(B), the Base Model with Base NP Model(N), and the Base Model with both Base NP Model and Preposition Tag Expansion(P). We also show the result of the (P) model on inputs tagged by Brill's tagger [2] in column (T).

Model	B	N	P	T
Lab. Precision	0.7926	0.7978	0.8146	0.7772
Lab. Recall	0.8039	0.8108	0.8233	0.7886
Precision	0.8284	0.8298	0.8455	0.8166
Recall	0.8402	0.8433	0.8545	0.8286
CB/Sent.	1.9030	1.87981	1.61671	1.9367
Sent. w/0CB	0.4700	0.4812	0.5104	0.4526
Sent. w/$\leq$2CBs	0.7151	0.7220	0.7522	0.7023
speed(words/s)	733.488	623.986	541.877	528.049

Table 3: Parsing results

Lab. Precision	Lab. Recall	Precision	Recall
0.7924	0.8028	0.8620	0.8734

Table 4: Parsing result of the Base Model on Chinese

The precision and recall differs from the Labeled precision and recall in that, a constituent is correct if the span is correct, the constituent label is not considered.

From the table, we observed a tiny improvement in accuracy by incorporating the Base NP model. However the parsing speed drops at the same time. By expanding the POS tag set for prepositions, the increase in accuracy is more significant. Around 2% increase in Labeled Precision and Labeled Recall is observed, but the speed drops further. When evaluated on input containing tagging errors, 3%–4% drop in the accuracy is observed.

4.4. Parsing Results on Chinese

In respect to the portability objective which we stated at the beginning of this paper, we also tested our parser on another language, Chinese. It is well known that the difference in sentence structure between Chinese and English is quite large, thus running our parser on Chinese is a good test for portability. Due the the lack of Chinese parsed sentence, the best resource we can find is [19], which contains about 3000 sentences from Chinese primary school text. Table 4 shows the parsing result of our base model. Observed from the table, the result is comparable to that of English. This, to certain degree, indicates that our learning algorithm is readily applicable to language other than English. Cautions should be taken, however, as the Chinese sentences that we used in both training and testing are simpler than the English sentences. On the other hand, we haven't incorporate the NP model and preposition tag expansion technique for the Chinese experiment.

5. COMPARISON TO OTHER PARSERS

In the literature, a lot of parsers have been proposed. We divided the previous works into 2 groups, tag based parsers and word based parsers.

5.1. Tag Based Parsers

In a tag based parser, the input is a sequence of POS tags, so no lexical feature is available. The parameter space is relatively smaller and the parsing accuracy is relatively lower.

Model:	PC	PL	O	PL	O
Sent. Len.	2–12	2–12	2–12	2–40	2-40
Ave. Sent. Len.	8.7	8.3	8.4	21.6	21.7
Lab. Precision(%)		87.1	88.9	81.9	81.5
Lab. Recall(%)		85.2	91.8	79.5	82.3
Precision(%)	88.6	89.8	91.4	83.0	84.6
Recall(%)	91.7	90.7	94.3	80.7	85.5
Ave. CBs		0.27	0.09	1.99	1.59
Sent. w/0CBs(%)		84.5	92.5	41.5	51.8

PC=PCFG, PL=PLCG, 0=Our Model
Unavailable data are leaves as blank.

Table 5: A Comparison between Tag based parsers

We compare our result with 2 parsers. One is based on probabilistic context free grammar(PCFG) [6] and the other is based on probabilistic left corner grammar(PLCG) [13]. Our work differs from the above works, in that, our parsing algorithm is deterministic. There is no redundant sub-structures generated by our parser. Moreover, there is no explicit grammar in our system.

To the best of our knowledge, the accuracy of our parser is the highest among all tag based parsers evaluated on the Penn Treebank corpus. A comparison of the results is shown in table 5.

5.2. Word Based Parsers

A word based parser take a sequence of words as input. POS tags are assigned either by an automatic POS tagger or by the parser itself. A far more rich set of features are available, including, lexical identities, morphological information, word classifications, verb sub-categorizations and semantic classes, etc. We compare our result with two word based parsers, the SPATTER and the CONTEXT.

Our parser is similar to SPATTER in the sense that both of us use extension[1]/Child-Type tag to keep track of constituent boundaries. However, the parsing algorithm of SPATTER is based on dynamic programming which is non-deterministic and the search space is huge.

Both CONTEXT and our parser are deterministic. We differs in that, CONTEXT employ a rich set of linguistic features in learning and requires a human expert to guild the training. So the learning process is not fully automated.

Table 6 shows the parsing accuracy of our parser, SPATTER and CONTEXT.

It may appears that the performance of CONTEXT is the best. However, CONTEXT is trained and tested on sentences containing only the 3000 most frequent words and the testing sentence is shorter in average. If we compare SPATTER with CONTEXT on testing sentences with similar average length, the performance is similar. As observed from the table, our parser is a few percent lower in accuracy. However, we use only simple syntactic features and the whole learning process is automatic.

[1]The term extension used in Magerman's paper is equivalent to Child-Type

Model	S	S	C	O
Deterministic	No	No	Yes	Yes
Auto. learned	Yes	Yes	No	Yes
Sent. Len.	4-25	4-40	4-45	4-40
Ave. Sent. Len.	16.8	22.3	17.1	22.0
Lab. Precision(%)	88.1	84.5	89.8	77.7
Lab. Recall(%)	87.6	84.0	89.6	78.9
CBs / Sent.	0.63	1.33	1.03	1.94
Sent. w/0CBs(%)	69.8	55.4	56.3	45.3
Sent. w/$\leq$2CBs(%)	92.1	80.2	84.9	70.2

S=SPATTER, C=CONTEXT, O=Our Model

Table 6: A Comparison between Word based parsers

6. CONCLUSION

We presented a lightweight, robust, automatically learned and deterministic parser. The parser is driven by two simple decision algorithm, Child-Type Tagging and Constituent Labeling. We have incorporated the Base NP model and the preposition tag expansion technique into the base model. More than 2% improvements is observed. The accuracy of our parser is highest among tag based parsers and comparable to some state-of-the-art parsers. The speed of our parser is more than 500 words per CPU second and only 6M of memory is needed for loading the decision trees. This make our parser suitable for on-line applications with limited memory.

7. REFERENCES

[1] A. V. Aho, R. Sethi, and J. D. Ullman, *Compilers. Principles, Techniques, and Tools.* Reading, Massachusetts: Addison-Wesley, 1986.

[2] E. Brill, "Some advances in transformation-based part of speech tagging," in *Proceedings of AAAI94*, 1994.

[3] E. Brill and P. Resnik, "A rule-based approach to prepositional phrase attachment disambiguation," in *Proceedings of The 15th International Conference on Computational Linguistics*, pp. 1198–1204, 1994.

[4] S. A. Caraballo and E. Charniak, "Figures of merit for best-first probabilistic chart parsing," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 127–132, 1996.

[5] C. Cardie and D. Pierce, "Error-driven pruning of treebank grammars for base noun phrase identification," in *Proceedings of COLING-ACL'98*, pp. 218–224, 1998.

[6] E. Charniak, "Tree-bank grammars," in *Technical Report CS-96-02, Dept of Computer Science, Brown University*, 1996.

[7] M. Collins and J. Brooks, "Prepositional phrase attachment through a backed-off model," in *Proceedings of the Third Workshop on Very Large Corpora*, pp. 27–38, 1995.

[8] M. J. Collins, "A new statistical parser based on bigram lexical dependencies," in *Proceedings of 34th Annual Meeting of the Association for Computational Linguistics*, pp. 184–191, 1996.

[9] M. J. Collins, "Three generative, lexicalised models for statistical parsing," in *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics*, pp. 17–23, 1997.

[10] B. et al., "A procedure for quantitatively comparing the syntatic coverage of english grammars," in *Proceedings of the February 1991 DARPA Speech and Natural Language Workshop*, 1991.

[11] J. Goodman, "Global thresholding and multiple pass parsing," in *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 11–25, 1997.

[12] D. M. Magerman, "Natural language parsing as statistical pattern recognition," in *Doctoral dissertation*, 1994.

[13] C. D. Manning and B. Carpenter, "Probabilistic parsing using left corner language models," in *Proceedings of the Fifth International Workshop on Parsing Technologies, MIT, Boston MA, 1997*, 1997.

[14] B. S. Marcus and M. Marcinkiewics, "Building a large annotated corpus of english: the penn treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[15] M. P. Marcus, *A Theory of Syntactic Recognition for Natural Language*. Cambridge, Massachusetts: MIT Press, 1980.

[16] L. A. Ramshaw and M. P. Marcus, "Text chunking using transformation-based learning," in *ACL Third Workshop on Very Large Corpora*, pp. 82–94, 1995.

[17] A. Ratnaparkhi, "A linear observed time statistical parser based on maximum entropy models," in *EMNLP-2: Second Conference on Empirical Methods in Natural Language Processing*, pp. 1–10, 1997.

[18] A. Ratnaparkhi, "Statistical models for unsupervised prepositional phrase attachment," in *Proceedings of the 17th International Conference on Computational Linguistics (COLING-ACL'98)*, pp. 1079–1085, 1998.

[19] Y. Shiwen, Z. Qiang, Z. Wei, Z. Yunyun, Z. Weidong, C. Baobao, and S. Zhifang, "Tagged singapore chinese primary school text," *An international journal of Chinese and Oriental Languages Information Processing Society*, vol. 5, 1995.

[20] A. S. Yeh and M. B. Vilain, "Some properties of preposition and subordinate conjunction attachments," in *Proceedings of COLING-ACL '98: 26th Annual Meeting of the Association for Computational Linguistics and 17th International conference on Computational Linguistics*, pp. 1436–1442, 1998.