Improve Parsing Performance by Self-Learning

Yu-Ming Hsieh, Duen-Chi Yang, and Keh-Jiann Chen

Chinese Knowledge and Information Processing (CKIP)
Institute of Information Science,
Academia Sinica, Taipei
{morris, ydc, kchen}iis@sinica.edu.tw

Abstract

There are many methods to improve performances of statistical parsers. Among them, resolving structural ambiguities is a major task. In our approach, the parser produces a set of *n*-best trees based on a feature-extended PCFG grammar and then selects the best tree structure based on association strengths of dependency word-pairs. However, there is no sufficiently large Treebank producing reliable statistical distributions of all word-pairs. This paper aims to provide a self-learning method to resolve the problems. The word association strengths were automatically extracted and learned by parsing a giga-word corpus. Although the automatically learned word associations were not perfect, the built structure evaluation model improved the bracketed *f*-score from 83.09% to 86.59%. We believe that the above iterative learning processes can improve parsing performances automatically by learning word-dependence knowledge continuously from web.

1. Introduction

How to solve structural ambiguity is an important task in building a high-performance statistical parser, particularly for Chinese. Since Chinese is analytic language, words play different grammatical functions without inflections. A great deal of ambiguous structures will be produced by parsers if no structure evaluator is applied. There are three main steps in our approach aim to disambiguate the structures. The first step is to have parser produce *n*-best structures. Secondly, we extract word-to-word association from large corpora and build semantic information. The last one is to build a structural evaluator to find the best tree structure from *n*-best. Formerly, there were some approaches proposed to resolve structure ambiguities. For instances,

- ***to add on lexical dependencies.*** Collins (1999) solves structural ambiguity by extracting lexical dependencies from Penn WSJ Treebank and applying dependencies to the statistic model. Lexical dependency (or Word-to-word association, WA) is one type of semantic information. It is a current trend to add on semantic related information in traditional parsers. Some incorporated word-to-word association in their parsing models, such as the Dependency Parsing in Chen et al. (2004). They take advantage of statistic information of word dependency in the parsing process to produce dependency structures. However, word association methods suffer low coverage for lacking very large tree-annotated training corpora, while checking dependency relation between word pairs.

- ***to add on word semantic knowledge.*** CiLin and HowNet information are used in the statistic model in the experiment of Xiong et al. (2005). Their results prove to solve common parsing mistakes efficiently.
- ***to use re-annotation method in grammar rule.*** Johnson (1998) thinks that re-annotating each node with the category of its parent category in Treebank is able to improve parsing performance. Klein et al. (2003) proposes internal/external/tag-splitting annotation strategies to obtain better results.
- ***to build evaluator.*** Some people re-rank the structure values and find out the best parse (Collins, 2000; Charniak et al., 2005). At first hand, their parser produces a set of candidate parses for each sentence. Later, the reranker finds out the best tree through relevance features. The performance is better that without the reranker.

This paper is going to show a self-learning method to produce imperfect (due to errors produced by automatic parsing) but unlimited amount of word association data to evaluate the *n*-best trees produced by a feature-extended PCFG grammar. The parser with this WA evaluation is considerably superior to those without evaluation.

The organization of the paper is as follows: Section 2 describes how to generate *n*-best trees in a simple way. In Section 3, we account for building word-to-word association and a primitive semantic class as well. As to the design of evaluating model, our probability model, coordination of rule probability and word association probability are presented in section 4. In Section 5 we discuss and explain the experimental data and results. Ambiguities of PoS are to be considered in a practical system. Section 6 deals with further experiment on automatic tagging with PoS. Finally, we offer concluding remarks in section 7.

2. Feature extension of PCFG grammars for producing the *n*-best trees

It is clear that Treebanks (Chen et al., 2003) provide not only instances of phrasal structures and word dependencies but also their statistical distributions. Recently, probabilistic preferences for grammar rules and feature dependencies were incorporated to resolve structure-ambiguities and had great improvements on parsing performances. However, the automatic extracted grammars and feature-dependence pairs suffer the problem of low coverage. We proposed different approaches to solve these two different types of low coverage problems. For the low coverage of extracted grammar, a linguistically-motivated grammar generalization method is proposed in Hsieh et al. (2005). And the low coverage of word association pairs is resolved by a self-learning method of automatic parsing and extracting word dependency pairs from very large corpora.

The linguistically-motivated generalized grammars are derived from probabilistic context-free grammars (PCFG) by right-association binarization and feature embedding (Hsieh et al., 2005). The binarized grammars have better coverage than the original grammars directly extracted from treebank. Features are embedded in the lexical and phrasal categories to improve the precision of generalized grammar. The important features adopted in our grammar are described in the following:

Head (Head feature):	The PoS of phrasal head will propagate all intermediate nodes within the constituent.
Example:	S(NP(Head:Nh:他)|S’$_{\text{-Head:VF}}$(Head:VF:叫|S’$_{\text{-Head:VF}}$(NP(Head:Nb:李四)| VP(Head:VC:撿| NP (Head:Na:球)))))
Linguistic motivations:	To constrain the sub-categorization frame.
Left (Leftmost feature):	The PoS of the leftmost constitute will propagate one–level to its intermediate mother-node only.
Example:	S(NP(Head:Nh:他)|S’$_{\text{-Head:VF}}$(Head:VF:叫|S’$_{\text{-NP}}$(NP(Head:Nb:李四)| VP(Head:VC:撿| NP(Head:Na:球)))))
Linguistic motivation:	To constrain linear order of constituents.
Head 0/1 **(Existence of phrasal head):**	If phrasal head exists in intermediate node, the nodes will be marked with feature 1; otherwise 0.
Example:	S(NP(Head:Nh:他)|S’$_{\text{-1}}$(Head:VF:叫|S’$_{\text{-0}}$(NP(Head:Nb:李四)|VP(Head:VC:撿| NP(Head:Na:球)))))
Linguistic motivation:	To enforce unique phrasal head in each phrase.

There are two functions in applying the embedded features: one is to increase the precision of the grammar and the other is to produce more candidate parse structures. With features embedded in phrasal categories, PCFG parsers are forced to produce varieties of different possible structures[1]. In order to achieve a better *n*-best oracle performance (i.e. the ceiling performance achieved by picking the best structure from n bests), we designed some different feature-embedded grammars and try to find a grammar with the better *n*-best oracle performance. For instance, “S(NP(Head:Nh:他)|Head:VF:叫| NP(Head:Nb:李四)| VP(Head:VC:撿| NP(Head:Na:球)))”. The explanations of feature sets are as follow.

<u>Rule type-1:</u>
Intermediate node: add on “Left and Head 1/0” features.
Non-intermediate node: if there is only one member in the NP, add on “Head” feature.
Example: S(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S’$_{\text{-Head:VF-1}}$(Head:VF:叫|S’$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:球)))))

<u>Rule type-2:</u>
Intermediate node: add on “Left and Head 1/0” features.
Non-intermediate node: add on “Head and Left” features, if there is only one member in the NP, add on “Head” feature.
Example: S$_{\text{-NP-Head:VF}}$(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S’$_{\text{-Head:VF-1}}$(Head:VF:叫|S’$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP$_{\text{-Head:VC}}$(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:球)))))

<u>Rule type-3:</u>
Intermediate: add on “Left, and Head 1/0” features.
Top-Level node: add on “Head and Left” features. (see example of S$_{\text{-NP-Head:VF}}$)
Non-intermediate node: if there is only one member in the NP, add on “Head” feature.
Example: S$_{\text{-NP-Head:VF}}$(NP$_{\text{-Head:Nh}}$(Head:Nh:他)|S’$_{\text{-Head:VF-1}}$(Head:VF:叫|S’$_{\text{-NP-0}}$(NP$_{\text{-Head:Nb}}$(Head:Nb:李四)|VP(Head:VC:撿| NP$_{\text{-Head:Na}}$(Head:Na:球)))))

[1] The parser adopts an Earley's Algorithm. It is a top-down left-to-right algorithm. So, in parts that have the same non-terminals, we keep only the best structure after pruning, to reduce the load of calculating and thus fasten the parsing speed. Therefore, if we add different features in the Top-Level rules, we'll get more results.

Rules and their statistical probabilities are extracted from the transformed structures. The grammars are derived and trained from Sinica Treebank. Sinica Treebank contains 38,944 tree-structures and 230,979 words. Table 1 shows the number of rule types in each grammar and Table 2 shows their 50-best oracle bracketed *f*-scores on three sets of testing data. The three sets of testing data used in our experiments represent "moderate", "difficult" and "easy" scale of Chinese language respectively. We adopt PARSEVAL measures to evaluate the bracketed *f*-score (BF)[2] as Table 2. A bracket represents the phrasal scope. The reason we don't use labeled *f*-score is that we aim to evaluate the phrasal scope, rather than the effect brought by phrasal category.

Table 1. Numbers of rules for each grammar.

	Rule Type		
	Rule-1	Rule-2	Rule-3
Rule number	9,899	26,797	13,652

Table 2. The 50-best oracle performances from the different grammars.

Testing Data	Sources	Hardness	Rule Type		
			Rule type-1	Rule type-2	Rule type-3
Sinica	Balanced corpus	Moderate	92.97	94.84	96.25
Sinorama	Magazine	Difficult	90.01	91.65	93.91
Textbook	Elementary school	Easy	93.65	95.64	96.81

From the above table, we can observe that the “Rule type-3” outperforms the “Rule type-1” and “Rule type-2”. We adopt the approach used in Charniak et al. (2005) to analyze the *n*-best parse. Table 3 shows the bracketed *f*-score values of different candidate trees. From the result, we observe that the improvement after *n*=5 is slight. Thus the number of ambiguous candidates can be dynamically adjusted according to the complexity of input sentences. For normal sentences, we may consider to take *n*=5 in order to minimize the complexity. For long sentences or sentences with auto PoS tagging should take as large as *n*=50 to raise the ceiling of the best *f*-score.

Table 3. Oracle bracketed *f*-scores as a function of number n of *n*-best parses.

Testing Data	*n*					
	1	2	5	10	25	50
Sinica	91.88	94.39	95.91	96.17	96.25	96.25
Sinorama	86.69	90.44	92.87	93.47	93.86	93.91
Textbook	92.24	95.01	96.21	96.61	96.78	96.81

[2] The harmonic mean of bracketed precision (BP) and bracketed recall (BR), i.e. $BF = \frac{2 * BP * BR}{BP + BR}$

For each candidate tree, its syntactic plausibility is obtained by rule probabilities produced by PCFG parser. Yet, we need semantic related information to help with finding the best tree structure among candidate trees. In the next section, we will see methods to get semantic related information.

3. Auto-Extracting world knowledge

In our experiments, we use a Gigaword Chinese corpus instead of texts from web to extract word dependence pairs. The Gigaword corpus contains about 1.12 billion Chinese characters, include 735 million characters from Taiwan's Central News Agency (traditional characters), and 380 million characters from Xinhua News Agency (simplified characters)[3]. Word associations are extracted from the texts of Central News Agency (CNA). First we use Chinese Autotag System (Tsai et. al., 2003), developed by Academia Sinica, to process the segmentation and PoS tagging of the texts. This system reaches a performance of 95% segmentation ability and 93% tagging ability. Then we parse each sentence[4] in the corpus and assign semantic roles to each constituent. Based on the head word information, we extract dependence word-pairs between head words and their arguments or modifiers. There are three types of the word pairs: (a) head word on the left hand side: (H_W_C, X_W_C); (b) head word on the right hand side: (X_W_C, H_W_C); (c) coordinating structure: (H_W_C, H_W_C). In the word pairs, "H" denotes Head, "W" means word, and "C" refers to PoS tag, "X" refers to any semantic role other than Head role. Figure 1 is an example of extracted word associations. The following illustrates how the automatic knowledge extraction works. We input a Chinese sentence to the parser:

他　叫　李四　撿　球
Ta　jiao　Li-si　jian　qiu
He　ask　L-isi　pick　ball
"He asked Li-si to pick up the ball."

Here is the sentence after segmentation and PoS tagging:

他(Nh)　叫(VF)　李四(Nb)　撿(VC)　球(Na)

The parser analyzes the sentence structure and assigns roles to each phrase. And then word-pair knowledge of heads and their modifiers are extracted as shown in Figure 1. The processes above are repeated in new data, no matter in Gigaword or texts from the internet. Finally we obtain a great deal of knowledge on words and their relations, and the amount of knowledge is on the increase. Meanwhile the evaluator takes this knowledge is for reference as well.

[3] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T09
[4] An existing parser is used to produce 1-best tree of a sentence.

Parsing and role assignment:

S(agent:NP(Head:Nhaa:他)|Head:VF2:叫|goal:NP(Head:Nba:李四)|theme:VP(Head: VC2: 撿| goal:NP(Head:Nab:球)))

Word association extraction:

	Role1	PoS1	Word1	Role2	PoS2	Word2
a	X	Nh	他	H	VF	叫
b	H	VF	叫	X	Nb	李四
c	H	VF	叫	X	VC	撿
d	H	VC	撿	X	Na	球

Figure 1. A sample for word association extraction.

We have 37,489,408 sentences that are successfully parsed and with word association information. And the number of extracted word associations is 221,482,591. The extracted word to word associations that undergo structure analysis and head word assignment are not perfectly correct, but they are more informative than simply taking words on the left and right hand window.

3.1. Coverage rates of the word associations

Data sparseness is always a problem of statistical evaluation methods. We test our extracted word association data in five different levels of granularities. Level-1 to Level-5 represents HWC_WC, HW_W, HC_WC, HW_C, and HC_C respectively. We like to see the bi-gram coverage rates for each level of representation. We divide word association data into ten. Figure 2 shows coverage relationships between five levels and sizes of word association data for three testing data. The extracted word association data are divided into ten layers of different sizes for each level.

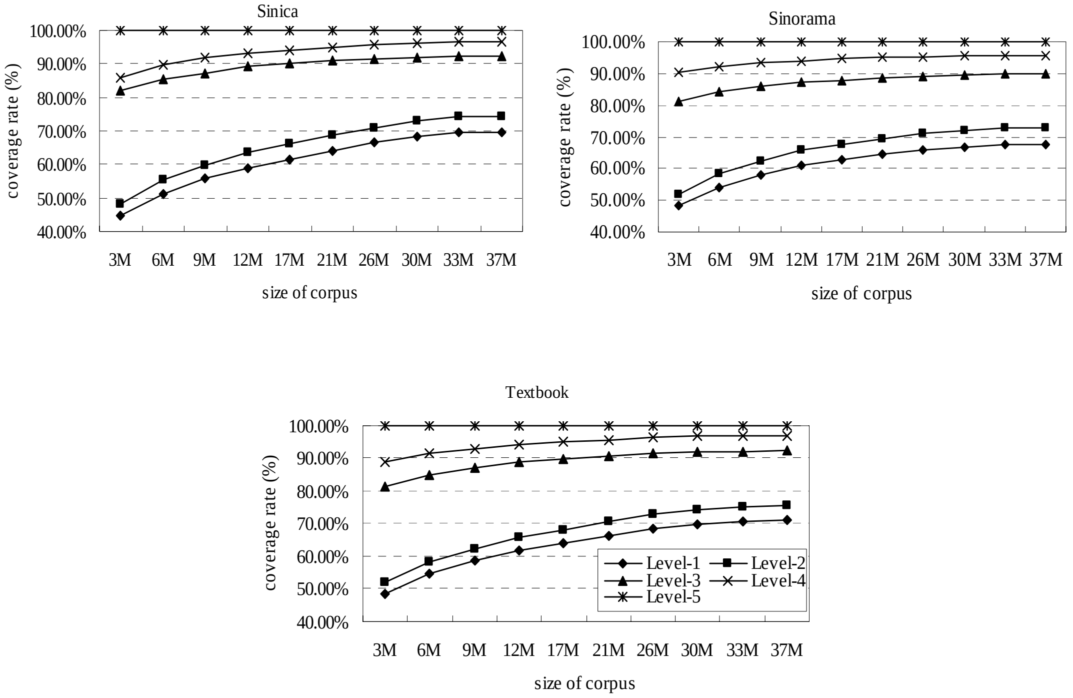

Figure 2. Coverage rates vs. size of Corpus: (a) Sinica; (b) Sinorama; (c) Textbook.

Figure 2 shows that larger data increases the coverage rates, but the coverage of the fine-grained level word associations, e.g. Level-1 (HWC_WC), is about 70%, which are far from saturation. Nonetheless the coverage rate can be improved by reading more texts from web. The coarse-grained level associations, e.g. Level-5 (HC_C), cover the most category bi-gram. But it may not be very useful, since syntactic associations which are partially embedded in the PCFG are redundant. To achieve a better evaluation model, we derived new associations between semantic classes. Criteria for semantic classification are discussed in the following section.

3.2. Incorporating semantic knowledge

In this section, we propose a simple approach to build a semantic-class-based relation for words, and that will be Level-6 (HS_S). Semantic class information is put into Level-6 in order to get high coverage and to avoid redundant syntactic associations in other levels. Besides, we hope to smooth the problem of data sparseness.

The idea is to classify words into their head morpheme. It begins with the transformation of every input "WORD, POS" in the data. We adopt affix database of high frequency verbs and nouns (Chiu et al., 2004) to setup noun and verb classes. There are 34,857 corresponding affixes. As to determinative measures (DM), we refer to the dictionary of measure words, and divide the DMs in the data into thirteen categories, according to the meanings of measure words. The thirteen categories include general, event, length, science, approximate measures, weight, square measures, container, capacity, time, currency value, classification measures, and measures of verbs. Finally we consult parts of speech analyses (CKIP, 1993) and the transformation rules of Figure 3 to build our semantic class. Take "張三, Nb" for example, its semantic class is "PersonalName" in our classification.

```
Notation:     WORD: user input Word
              POS: user input PoS of the word
              CLASS: transformation class of the word
              Affix(WORD): input WORD to find mapping affix from table
              Prefix(WORD): prefix of the WORD
              Suffix(WORD): suffix of the WORD
              DM(WORD): input Word to find DM category
Input:        WORD, POS
Output:       CLASS
Initial Step:
        CLASS=WORD;
        if WORD in affix table then CLASS=affix(WORD);
        if POS is verb or adverb then CLASS=POS+prefix(WORD);
        if POS is noun then CLASS=POS+suffix(WORD);
Mapping Step:
        if POS is non-predicative adjective then CLASS='A'+prefix(WORD);     /* e.g. A */
        if POS is preposition then CLASS='P'+suffix(WORD);    /* e.g. P */
        if POS is SHI then CLASS='SHI';     /* e.g. 是 */
        if POS is V_2 then CLASS='V_2';     /* e.g. 有 */
        if POS is DM or Measure and exist in DM table then CLASS=DM(WORD); /* e.g. DM/Nf */
        if POS is conjunction then CLASS=POS+prefix(WORD);     /* e.g. Caa/Cab/Cba/Cbb */
        if POS is determinative then CLASS=POS;     /* e.g. Nep/Neqa/Neqb/Nes/Neu */
        if POS is pronoun then CLASS=WORD;     /* e.g. Nh */
        if POS is time noun then CLASS='Time';     /* e.g. Nd */
        if POS is Postposition/Place Noun/Localizer then CLASS='Location';     /* e.g. Ng/Nc/Ncd */
        if POS is Proper Noun and is family names then CLASS='PersonalName';     /* e.g. Nb */
        if POS is aspectual adverb, CLASS=POS    /* e.g. Di */
        if POS is pre/post-verbal adverb of degree then CLASS='Df'+suffix(Word) /*e.g. Dfa/Dfb */
        if POS is VD/VCL/VL then CLASS=POS+suffix(WORD)
```

Figure 3. Transformation algorithm.

We estimate the word association coverage rate as the above mentioned. From the results shown in Figure 4, the coverage rate of Level-6 is higher than Level-2, and the problem of data sparseness is indeed moderated.

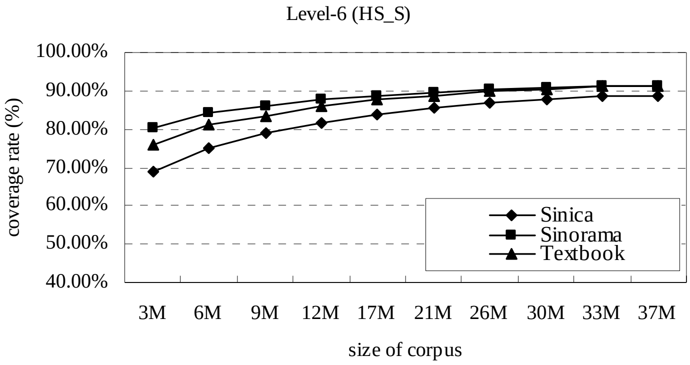

Figure 4. WA coverage rate of Level-6.

Now we have semantic information. How it works with rule probability to find the best structure among the numerous ambiguous candidates will be discussed in Section 4.

4. Building evaluation model

A sentence structure is evaluated by its syntactic and semantic plausibility. The syntactic plausibility is modeled by products of phrase rule probabilities of its syntactic tree. The semantic plausibility is modeled by the word association strengths between head words and their arguments or modifiers. For an input sentence s, the feature-embedded PCFG parser produces *n*-best trees of $\{y_1(s),...,y_n(s)\}$. The evaluating model finds out the best structure according to the rule probability (syntactic) and corresponding word association probability (semantic). Rule probabilities are marked when *n*-best trees are produced. We will estimate word association probabilities in the following formula. In the formula, “Head” means the Head member word association, as HWC, HC, HW. “Modify” means modify or argument member, as in WC, W, C. “freq(Head)” means Head word frequency in the corpus and “freq(Head, Modify)” refers to the co-occurrence frequency of “Head” and “Modify”.

$$P(Modify \mid Head) = \frac{freq(Head, Modify)}{freq(Head)} \quad (1)$$

Data sparseness is a common problem in dealing with corpus. A minimal value σ is used to smooth data sparseness, such as $\sigma = \frac{1}{total\ number\ of\ WA\ token}$.

$Value(y_n(s))$ in the formula below means the final evaluation value to each candidate tree.

$$\begin{aligned}&Value(y_n(s)) = \\ &\lambda * RuleValue(y_n(s)) + (1-\lambda) WAValue(y_n(s))\end{aligned}, \tag{2}$$

Where $RuleValue(y_n(s))$ is the rule probability of the sentence and $WAValue(y_n(s))$ is the total word association value in different level *n*. RuleValue and WAValue are normalized, i.e. *(i-min)/(max-min)*. The following shows weighting in different levels and explanation of formula:

$$WAValue(y_n(s)) = \sum_{level=1}^{6} \theta_{level} * WA_{level}(y_n(s)) \tag{3}$$

$$WA_{level}(y_n(s)) = \prod_{all\_word\_association\_for\_y_n(s)} P(Modify \mid Head) \tag{4}$$

After semantic probability collocating with rule probability, we hope to find the best tree $y*(s)$.

$$y*(s) = \arg\max Value(y_n(s)) \tag{5}$$

where $y*(s)$ has the best bracketed *f*-score. We calculate relating λ and θ values from development sets. The development sets are adopted from trees in training data. In evaluation, we substitute λ and θ for every interval of 0.1 from 0 to 1. Then we find out the best results in certain probability. The experiment results will be shown in the following section. Moreover, we justify whether the word associations are reasonable.

5. Experimental results

We evaluated the performance of our evaluating model using the standard PARSEVAL metrics. Hsieh et al. (2005) state that the bracketed *f*-score of short sentence parsing (the length of a sentence is from 1 to 5 words) is over 90% in their experiment. As a result, the following experiments are on sentences more than 6 words. The oracle 50-best bracketed *f*-scores of “Rule type-3" are listed in Table 4.

Table 4. The bracketed *f*-scores of 1-best and oracle performance of 50-best. (sentence length ≥ 6)

Top *n*-best	Testing data		
	Sinica	Sinorama	Textbook
1-best	83.09	77.545	83.195
50-best	90.11	87.445	89.945

To simplify our evaluation model, we try to find the most effective levels of associations first. In turn, the evaluation model uses only one level of association and rule probabilities to select the best structure from n candidates. That is,

$$WAValue(y_n(s)) = WA_{level}(y_n(s)) = \prod_{all\_word\_association\_for\_y_n(s)} P(Modify \mid Head) \tag{6}$$

Figure 5 displays the results of testing data. The best results of Level-1 slightly surpass that of Level-2; results of Level-6 overtake that of Level-3; Level-6 has better performance than Level-5. Therefore, only three levels (Level-1, Level-4 and Level-6) are chosen to be calculated, for dimension reduction.

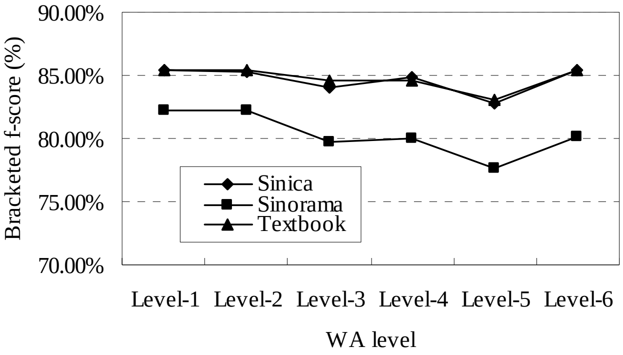

Figure 5. Matching rule with WA value in each level (sentence length$\geq$6).

Finally we use the combination of L1, L4, and L6 associations and rule probabilities to evaluate plausibility of structures. Results of experiments on the three testing data are shown in Table 5.

Table 5. The bracketed *f*-scores of 50-best parses (sentence length$\geq$6)

Models	Testing data		
	Sinica	Sinorama	Textbook
R, L1, L4, L6	86.59	82.81	85.97

In Table 5, we see that semantic information is effective in finding correct structure. If we justify the rationality of WA, about 3.5%~5.2% of the performance is raised. In our experiments, λ =0.7, θ_1=0.7, θ_4=0.3, and θ_6=0.5. In Charniak et al. (2005), the f-score was improved from 89.7% (without reranking) to 91.02% (with reranking) for English; the oracle f-score was 96.8% for n-best in their paper. From the result, we see an improvement in the testing data. With the more data parsed, better word-association values are obtained. This enhances the parsing performance and reaches our goal of self-learning.

6. Further Experiments on Sentences with Automatic PoS Tagging

Perfect testing data was used in the above experiments without considering PoS tagging errors. However, in reality, PoS tagging errors will degenerate parsing performances. The real parsing performances of accepting input from PoS tagging system are shown in the Table 6(1). In this table, "Autotag" mean to markup the best PoS on the segmented data. The naïve approach to overcome the PoS tagging errors is to delay some of the ambiguous PoS resolution for words with lower confidence tagging scores and leave the ambiguous PoS to be resolves at parsing stage. The tagging confidence of each word is measured by the following value.

$$\text{Confidence value} = \frac{P(c_{1,}w)}{P(c_{1,}w) + P(c_{2,}w)}, \tag{7}$$

where P(c1,w) and P(c2,w) are probabilities assigned by the tagging model for the best candidate “c1,w” and the second best candidate “c2,w”.

In Table 6(2), "Autotag with confidence value=1.0" means that if confidence value <= 1.0, we list all possible PoSs for parser to decide. The experimental results, Table 6(2), show that delaying ambiguous PoS resolution does not improve parsing performances, since PoS ambiguities increase structure ambiguities and the PCFG parser is not robust enough to select better syntactic structures.

Table 6. Oracle bracketed *f*-scores of different autotag for parsing:
(1)Autotag; (2)Autotag with confidence value = 1.0.

Top *n*-best		Testing data		
		Sinica	Sinorama	Textbook
(1)	1-best	75.31	72.05	79.27
	50-best	84.09	83.36	87.54
(2)	1-best	73.41	68.34	77.83
	50-best	86.45	83.99	88.83

We then apply our evaluation model to select the best structure from 50-best parses. The results are shown in Table 7. The experiment above takes “Rule type-3” for *n*-best parses. The bracketed *f*-score is raised from the original 73.41% to 79.34%, about 4% of improvement in the Sinica testing data. Sinorama data is improved from 68.34% to 74.78%. Textbook data is from 77.83% to 82.59%. All these results are raised up to 2%~4%. We can see that our evaluating model finds better results than Autotag. In solving the ambiguous POS, our evaluating model produces better tree structures than Autotag.

Table 7. The bracketed *f*-scores in Autotag
with confidence value=1.0 and 50-best parses (sentence length ≥ 6).

Models	Testing data		
	Sinica	Sinorama	Textbook
R, L1, L4, L6	79.34	74.78	82.59

7. Conclusion

Parsers of any language aim to correctly analyze the syntactic structure of a sentence, often with the help of semantics. This paper shows a self-learning method to produce imperfect (due to errors produced by automatic parsing) but unlimited amount of word association data to evaluate the *n*-best trees produced by a feature-extended PCFG grammar. We prove that although the statistical association strengths produced by automatic parsing are not perfect, still the extracted data is reliable enough in measuring plausibility of ambiguous structures. The parser with this WA evaluation is considerably superior to those without evaluation. We believe that the above iterative learning processes can improve parsing performances automatically by learning word-dependence knowledge continuously from web. We also propose an easy method to produce *n*-best of a sentence. First of all we slightly modify the parser to produce *n*-best. Then different feature sets in grammar rules are used to bring forth different results. There is one feature set that covers more structures than the original 1-best. In our experiments, we use 50-best to estimate the efficiency of our evaluating model.

On the other hand, we offer a general syntactic and semantic evaluation model. We input *n*-best parses to our evaluating model. The evaluating model selects the best parse from this set of parses using a rule and semantic probability. The system we described, using the standard PARSEVAL framework, has a bracketed *f*-score of the selected trees, which is 86.59% higher to the original 1-best. Furthermore, ambiguous PoS of a word is also parsed and evaluated on *n*-best. We can see that our evaluating model finds better results than Autotag.

In the future research, we plan to improve the quality of word-association. Three aspects need to done: improving the accuracy of PoS tagger; enhancing the parser's ability to solve common mistakes, such as parsing conjunctive structures; extracting more word associations by reading and parsing text from web. As to the evaluating model, a properly corresponding semantic classifications from coarse to fine-grained category are needed in Level-6.

8. Acknowledgements

This research was supported in part by National Science Council under Grant NSC 95-2422-H-001-008- and National Digital Archives Program Grant 95-0210-29-戊-13-09-00-2.

9. References

Eugene Charniak and Mak Johnson. 2005. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173-180, Ann Arbor, MI.

Keh-Jiann Chen, Chu-Ren Huang, Feng-Yi Chen, Chi-Ching Luo, Ming-Chung Chang, Chao-Jan Chen, and Zhao-Ming Gao. 2003. Sinica Treebank: design criteria, representational issues and implementation. In Anne Abeille, (ed.): *Building and Using Parsed Corpora. Text, Speech and Language Technology*. 20:231-248, pp231-248.

Yuchang Chen, Masayuki Asahara, and Yuji Matsumoto. 2004. Deterministic Dependency Structure Analyzer for Chinese. In *Proceedings of the First International Join Conference on Natural Language Processing*, pages 135-140, Sanya City, Hainan Island, China.

Chih-Ming Chiu, Ji-Qing Luo, and Keh-Jiann Chen. 2004. Compositional semantics of mandarin affix verbs. In *Proceedings of ROCLING XVI: Conference on Computational Linguistics and Speech Processing*, pages 131-139, Taipei.

CKIP (Chinese Knowledge Information processing). 1993. The categorical analysis of Chinese. Technical Report no 93-05. Taipei: Academia Sinica.

Michael Collins. 1999. Head-driven statistical models for natural language parsing. PhD thesis, University of Pennsylvania.

Michael Collins. 2000. Discriminative reranking for natural language parsing. *In Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 175-182, Morgan Kaufmann, San Francisco, CA.

Yu-Ming Hsieh, Duen-Chi Yang, and Keh-Jiann Chen. 2005. Linguistically-motivated grammar extraction, generalization and adaptation. In *Proceedings of the Second International Join Conference on Natural Language Processing*, pages 177-187, Jeju Island, Republic of Korea.

Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4): 613-632.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423-430, Sapporo, Japan.

Yu-Fang Tsi and Keh-Jiann Chen. 2003. Context-rule model for PoS tagging. In *Proceedings of 17th Pacific Asia Conference on Language, Information and Computation (PACLIC 17)*, pages 146-151, COLIPS, Sentosa, Singapore.

Deyi Xiong , Shuanglong Li, Qun Liu, Shouxun Lin, and Yueliang Qian. 2005. Parsing the Penn Chinese Treebank with semantic knowledge. In *Proceedings of the Second International Join Conference on Natural Language Processing*, pages 70-81, Jeju Island, Republic of Korea.