Computational Linguistics and Chinese Language Processing
Vol. 12, No. 2, June 2007, pp. 195-216

Improve Parsing Performance by Self-Learning

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

Abstract

There are many methods to improve performance of statistical parsers. Resolving structural ambiguities is a major task of these methods. In the proposed 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 constructed 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 information continuously from web.

Keywords: Parsing, Word association, Knowledge Extraction, PCFG, PoS Tagging, Semantic.

1. Introduction

How to solve structural ambiguity is an important task in building a high-performance statistical parser, particularly for Chinese. Since Chinese is an analytic language, words can play different grammatical functions without inflection. A great deal of ambiguous structures would be produced by parsers if no structure evaluation were applied. There are three main steps in our approach that aim to disambiguate the structures. The first step is to have the parser produce *n*-best structures. Second, we extract word-to-word associations from large corpora and build semantic information. The last step is to build a structural evaluator to find the best tree structure from the *n*-best candidates.

There have been some approaches proposed in the past to resolve structure ambiguities. For instance:

* Institute of Information science, Academia Sinica, Taipei, Taiwan
E-mail: {morris, ydc, kchen}@iis.sinica.edu.tw

[Received February 7, 2007; Revised July 2, 2007; Accepted July 9, 2007]

Adding 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 incorporate word-to-word association in their parsing models, such as the Dependency Parsing in Chen *et al.* [2004]. They take advantage of statistical information of word dependency in the parsing process to produce dependency structures. However, word association methods suffer low coverage when lacking very large tree-annotated training corpora while checking dependency relationships between word pairs.

Adding on word semantic knowledge where CiLin and HowNet information are used in the statistic model in the experiment [Xiong *et al.* 2005]. Their results work to solve common parsing mistakes efficiently.

Using a re-annotation method in grammar rules. 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, and tag-splitting annotation strategies to obtain better results.

Building an evaluator. Some people re-rank the structure values and find the best parse [Collins 2000; Charniak *et al.* 2005]. At first, the parser produces a set of candidate parses for each sentence. Later, the re-ranker finds the best tree through relevance features. The performance is better than without the re-ranker.

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 the 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 the 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 experiments 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 improvement on parsing performance. However, the automatically 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 [Hsieh *et al*. 2005]. 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. The binarized grammars have better coverage than the original grammars directly extracted from Treebank. Features are embedded into 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:皮球)))))
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:皮球)))))
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:皮球)))))
Linguistic motivation:	To enforce unique phrasal head in each phrase.

There are two functions of 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:皮球)))". The explanations of feature sets are as follow.

Rule type-1:

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:皮球)))))

Rule type-2:

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:皮球)))))

Rule type-3:

Intermediate node: 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:皮球)))))

[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 calculation 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[2]. 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. Black [1991] proposed two structural evaluating systems in 1991; the more strictly based is named PARSEVAL, and the less strictly based is crossing. We adopt PARSEVAL measures to evaluate the bracketed *f*-score. The formula represents as follows:

$$\text{bracketed precision (BP)} = \frac{\#\text{ brack correct constituents in parser's parse of testing data}}{\#\text{ bracket constituents in parser's parse of testing data}}$$

$$\text{bracketed recall (BR)} = \frac{\#\text{ brack correct constituents in parser's parse of testing data}}{\#\text{ bracket constituents in treebank's parse of testing data}}$$

$$\text{bracketed } f\text{-score (BF)} = \frac{BP * BR * 2}{BP + BR}$$

A bracket represents the phrasal scope. The reason we don't use a labeled *f*-score is that we aim to evaluate the phrasal scope, rather than the effect brought by the phrasal category. For example, the dependency information is much more related to the structure.

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

[2] http://treebank.sinica.edu.tw/

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 best bracketed *f*-score values of different *n*-best parse trees. From the results, 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

For each candidate tree, its syntactic plausibility is obtained by rule probabilities produced by PCFG parser. In addition to this, we need semantic related information to help with finding the best tree structure among candidate trees. In the next section, we will look at some methods of attaining semantic related information.

3. Auto-Extracting World Knowledge

We could extract word knowledge from Treebanks, but the availability of a very large set of trees with rich linguistic annotations has long been a problem. A cheaper way to extract word knowledge is to automatically parse a large amount of data. We believe that with good parsing performance, we could get sufficient information.

Therefore, in our experiments, we use a Gigaword Chinese corpus to extract word dependence pairs. The Gigaword corpus contains about 1.12 billion Chinese characters, including 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 and 93% tagging accuracies. Then we parse each sentence[4] in the corpus and assign semantic roles to each constituent. Based on the head word information, we extract

[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.

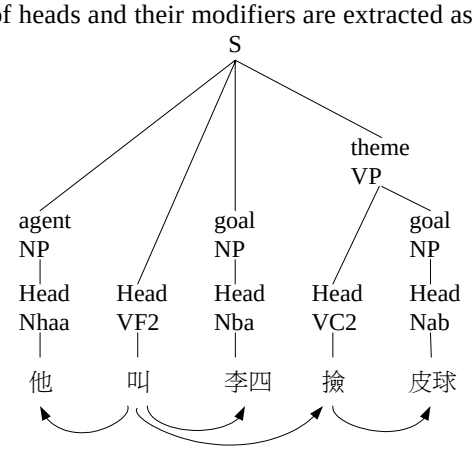
dependence word-pairs between head words and their arguments or modifiers. The following illustrates how the automatic knowledge extraction works. We input a Chinese sentence to the parser:

他 叫 李四 撿 皮球
Ta jiao Li-si jian qiu
He ask Li-si 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) 皮球(Na)

The parser analyzes the sentence structure and assigns roles to each phrase as follows. Then, word-pair knowledge of heads and their modifiers are extracted as shown in Figure 1.



Role1	PoS1	Word1	Role2	PoS2	Word2
agent[NP]	Nhaa	他	Head[S]	VF2	叫
Head[S]	VF2	叫	goal[NP]	Nba	李四
Head[S]	VF2	叫	theme[NP]	VC2	撿
Head[S]	VC2	撿	goal[NP]	Nab	皮球

Figure 1. A sample for word association extraction.

Figure 1 shows the examples of extracted word associations. “Role1/PoS1/Word1 and Role2/Role2/Word2” represent the right- and left-part of the word-pairs. “Role”, “PoS”, and “Word” here mean semantic role, part-of-speech and word respectively. To reduce the number of word association types, we transform the original word-pairs into three simplified 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 the simplified PoS tag[5], “X” refers to any semantic role other than Head role. So, we get basic information of experimental data as follows:

Role1	PoS1	Word1	Role2	PoS2	Word2
X	Nh	他	H	VF	叫
H	VF	叫	X	Nb	李四
H	VF	叫	X	VC	撿
H	VC	撿	X	Na	皮球

The processes above are repeated for each new input sentence from the Gigaword corpus.

Finally, we obtain a great deal of knowledge about dependent word pairs and their association strengths. In our experiments, we have 37,489,408 sentences that are successfully parsed and contain word association information. 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 and precise 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. As mentioned in the last section, we automatically segment, tag, parse and assign roles in CNA data, and then extract word associations. 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. The 5 levels of word associations derived from Figure 1 are as follows:

[5] The simplified way please refer to CKIP 93-05 Technical Report.

Level	Type	Word Associations
Level-1	HWC_WC	(他/Nh _H/叫/VF) (H/叫/VF_李四/Nb) (H/叫/VF_撿/VC) (H/撿/VC_球/Na)
Level-2	HW_W	(他_H/叫) (H/叫_李四) (H/叫_撿) (H/撿_皮球)
Level-3	HC_WC	(他/Nh_H/VF) (H/VF_李四/Nb) (H/VF_撿/VC) (H/VC_皮球/Na)
Level-4	HW_C	(Nh_H/叫) (H/叫_Nb) (H/叫_VC) (H/撿_Nb)
Level-5	HC_C	(Nh_H/VF) (H/VF_Nb) (H/VF_VC) (H/VC_Na)

Theoretically, the precision of fine-grain level like HWC_WC is much better, but it suffers the problem of data sparseness, hence, its coverage rate is low; on the other hand, the coarse-grain level has best coverage rate but relatively low precision. This is the trade-off between precision and coverage. Therefore, we carry out a series of experiments to find a balanced measurement by linear combination of different level associations. There will be experimental results in the following sections.

Why not use HWC_W or HC_W? From our observation, we have found that these two show similar performance with HWC_WC and HC_WC respectively; therefore, we exclude them. Besides, there are some asymmetric representations, such as the use of "HW_C". They are used to raise the coverage rate in word association while not being too general.

We like to see the bi-gram coverage rates for each level of representation. After CNA producing word associations in each level, we observe the relationship between the amount of word associations and the coverage rates of the three texts: Sinica, Sinorama, and Textbook. We extracted word associations from the three data sets in each level and calculated their coverage rates.

We tested the coverage rates for 10 different size word association data, of which each was extracted from different size corpora. Figure 2 shows coverage relationships between five levels and sizes of word association data for three testing data.

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 only about 70%, which is far from saturation. Nonetheless, the coverage rate can be improved by reading more texts from the web. The coarse-grained level associations, *e.g*. Level-5 (HC_C), cover the most bi-gram categories. However, it may not be very useful, since syntactic associations which are partially embedded in the PCFG are redundant. To attain a better evaluation model, we derived new associations between semantic classes. Criteria for semantic classification are discussed in the following section.

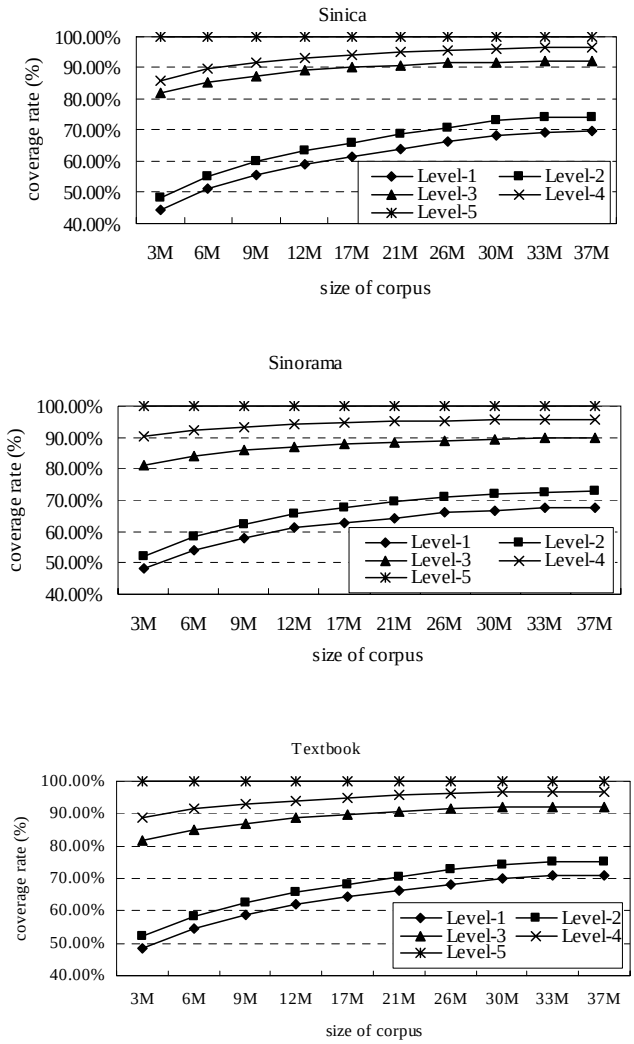


Figure 2. Coverage rates vs. size of Corpus: (a) Sinica; (b) Sinorama; (c) Textbook.

3.2 Incorporating Semantic Knowledge

For precision and coverage tradeoffs, we face a dilemma of using word or PoS category. We find that the coverage of word is low, though its precision is high; on the contrary, the coverage of PoS is too high to be discriminative. We hope to find a classification that covers enough information and is discriminative as well; that is, a classification system that falls between word and PoS category. A semantic classification is the solution.

There are many ways to classify semantic properties of words. Xiong *et al*. [2005] adopt CiLin and HowNet as their semantic classes in their experiment; however, the data sparseness is still a problem to be solved. Here, we propose a simple approach to build a semantic-class-based association strength for word pairs, which will be our 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, it can 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 the affix database of high frequency verbs and nouns [Chiu *et al*. 2004] to set up noun and verb classes. There are 34,857 examples of compound words in the database. 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 the 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 set up the transformation rules to transform a word-PoS pair into its semantic class. The transformation algorithm is shown at Appendix A. Take “李四, Nb” as example, its semantic class is “PersonalName(人名)” in our classification. In another instance, the semantic class of “皮球, Na” is “Na_球”. The transformation rules are PoS dependent. Each PoS is referred to CKIP [1993], which explains the PoS with words and examples. We set up discriminative subcategorization on some parts-of-speech: N/P/D/A according to the distribution of PoS and word frequency. As to the verbs, we use an initial step to assign initial value. Take PoS as "A" for example, adding prefix information is more useful than using "A" alone.

Role1	PoS1	Word1	Class1	Role2	PoS2	Word2	Class2
X	Nh	他	他	H	VF	叫	叫
H	VF	叫	VF_叫	X	Nb	李四	PersonalName
H	VF	叫	VF_叫	X	VC	撿	VC_撿
H	VC	撿	VC_撿	X	Na	皮球	Na_球

The following example is the result of DM, prefix and affix, through a function in Level-6 (HS_S):

S(theme:NP(quantifier:DM:兩個|Head:Nab:人)|deontics:Dbab:能|Head:VC1:在|goal:GP(DUMMY:NP(property:Nad:人生|Head:Nad:旅途)|Head:Ng:中))

Role1	PoS1	Word1	Class1	Role2	PoS1	Word1	Class2
X	DM	兩個	general_DM	H	Na	人	Na_人
X	Na	人	Na_人	H	VCL	在	VCL_在
X	D	能	D_能	H	VCL	在	VCL_在
H	VCL	在	VCL_在	X	Na	旅途..中	Na..Ng
X	Na	人生	N_人	H	Na	旅途	Na_途
X	Na	旅途	Na_途	H	Ng	中	Location

It is necessary to discriminate syntactic head from semantic head in word association extraction of GPs and PPs. From the table above, Row 4, signified by the different color shows that "旅途" is the semantic head of the GP "旅途..中", while the word "中" is the syntactic head of the phrase.

We estimate the word association coverage rate of the Level-6 associations. From the results shown in Figure 3, the coverage rate of Level-6 is higher than Level-2, and the problem of data sparseness is indeed moderately smoothed.

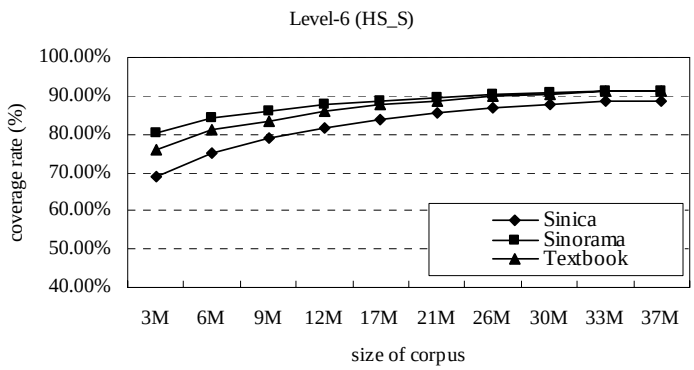


Figure 3. WA coverage rate of Level-6.

Next, we will use different levels of associations to construct an evaluation model to find the best structure among the numerous ambiguous candidates.

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 its n-best trees $\{y_1(s),...,y_n(s)\}$. The evaluating model finds out the best structure

according to the rule probability (syntactic) and corresponding word association probability (semantic). Rule probabilities are generated by the PCFG parser when n-best trees are produced. We will estimate word association probabilities in the following formula. In the formula, "Head" means the Head of a word association, notated as HWC, HC, or HW. "Modify" means dependent daughter, notated as WC, W, or C.

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

Data sparseness is a common problem in dealing with corpora. A minimal value σ is used to smooth data sparseness:

$$\sigma = \frac{1}{total\ number\ of\ WA\ token} = \frac{1}{221482591}$$

The evaluation value $Value(y_n(s))$ to each candidate tree *Yn(S)* is defined as:

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

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 $y^*(s)$.

$$y^*(s) = \arg\max Value(y_n(s)) \quad /*Yi \text{ on all } i \tag{5}$$

We calculate related λ and θ values from the 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.

For instance, the following example has eight different ambiguous parsing results produced by the parser.

Input segmentation with PoS tag: 我們(Nh) 都(D) 喜歡(VK) 蝴蝶(Na)
Parsing results:
#1:1.[0] S(NP(Head:Nh:我們)|D:都|Head:VK:喜歡|NP(Head:Na:蝴蝶))#
#1:2.[0] NP(Nh:我們|Head:NP(VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶))#
#1:3.[0] VP(PP(Head:Nh:我們)|VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶)#
#1:4.[0] NP(VP(Head:Nh:我們)|Head:NP(VP(D:都|Head:VK:喜歡)|Head:Na:蝴蝶))#
#1:5.[0] VP(Head:VP(VP(Head:Nh:我們)|VP(D:都|Head:VK:喜歡))|NP(Head:Na:蝴蝶))#
#1:6.[0] NP(S(NP(Head:Nh:我們)|D:都|Head:VK:喜歡)|Head:Na:蝴蝶)#
#1:7.[0] VP(PP(Head:Nh:我們)|Head:VP(VP(D:都|Head:VK:喜歡)|VP(Head:Na:蝴蝶)))#
#1:8.[0] VP(Head:VP(VP(Head:Nh:我們)|VP(Head:D:都))|Head:VP(Head:VK:喜歡|NP(Head:Na:蝴蝶)))#

	Prob (log_2)
Rule	*-23.74*

Type	**WA**	**Prob (log_2)**
Level-1 (HWC_WC)	(我們/Nh_H/喜歡/VK) (都/D_H/喜歡/VK) (H/喜歡/VK_蝴碟/Na)	$log_2(76/21528)+log_2(578/21528)+$ $log_2(2/12200)$ = *-25.9395936826742*
Level-2 (HW_W)	(我們_H/喜歡) (都_H/喜歡) (H/喜歡_蝴碟)	$log_2(76/21528)+log_2(578/21528)+$ $log_2(2/12200)$ = *-25.9395936826742*
Level-3 (HC_WC)	(我們/Nh_H/VK) (都/D_H/VK) (H/VK_蝴碟/Na)	$log_2(25520/3235010)+log_2(49025/3235010)+$ $log_2(8/2501420)$ = *-31.2844226460991*
Level-4 (HW_C)	(Nh_H/喜歡) (D_H/喜歡) (H/喜歡_Na)	$log_2(3257/21528)+log_2(6160/21528)+$ $log_2(2927/11741)$ = -6.53387135079941
Level-5 (HC_C)	(Nh_H/VK) (D_H/VK) (H/VK_Na)	$log_2(230163/3235010)+log_2(1086580/3235010)+$ $log_2(575635/2601356)$ = *-7.56305573913316*
Level-6 (HS_S)	(我們_H/VK 喜) (D 都_H/VK 喜) (H/VK 喜_Na 碟)	$log_2(81/23809)+ log_2(586/23809)+$ $log_2(2/13986)$ = *-26.3155277463539*

Figure 4. An Example of Rule calculationand and WA probability.

Figure 4 shows the WA values of the first sentence at each level. Similarly the WA data are produced for all other input sentences. Then, we derive the evaluation values $Value(y_n(s))$ for each ambiguous sentence and find the best result with respect to different weights.

5. Experimental Results

The parsing performance and our evaluating model are evaluated by standard PARSEVAL metrics. In our experiments, we only use sentences longer than 6 words for our testing, since Hsieh *et al*. [2005] found that the bracketing *f*-score of short sentence (the length of a sentence is from 1 to 5 words) is over 90%. We use the *n*-best tree structures produced from "Rule type-3" mentioned in the Section 2. The oracle 50-best and the top 1-best bracketed f-scores of "Rule type-3" are listed in Table 4. Take the data of Sinica for example, we find that for the 50-best results, the oracle score is 90.11%. In contrast the 1-best *f*-score is 83.09%.

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	77.54	83.19
50-best	90.11	87.44	89.94

To simplify our evaluation model, we try to find the most effective levels of associations first. In turn, the parser 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 bracketing *f*-scores of testing data for each different level of association. 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, in considering type of information, data coverage, and dimension reduction only three levels (Level-1, Level-4 and Level-6) are taken into consideration to form the final evaluation model.

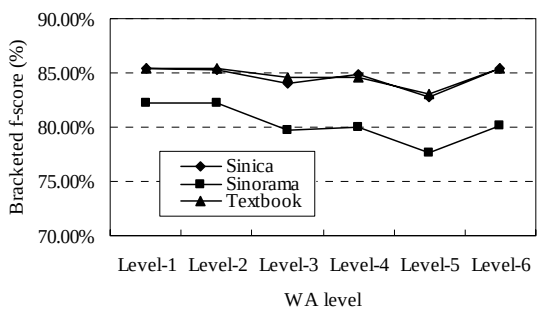


Figure 5. Matching rule with WA value in each level (sentence length ≥ 6).

Finally, we adjust the weights of L1, L4, and L6 associations and rule probabilities to evaluate the plausibility of structures from the 50-best parses tree of the developing data and the results of experiments on the three testing data are shown in Table 5. For our experiments, λ =0.7, θ_1=0.7, θ_4=0.3, and θ_6=0.5.

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
1-best	83.09	77.54	83.19
50-best	90.11	87.44	89.94

From the results shown in Table 5, we see that semantic information is effective in finding better structures. About 3.5%~5.2% of the bracketing *f*-scores are raised. In Charniak *et al.* [2005], the *f*-score was improved from 89.7% (without re-ranking) to 91.02% (with re-ranking) for English[6]; the oracle *f*-score was 96.8% for *n*-best in their paper. We also believe that with more data parsed, better word-association values will be obtained; hence, the parsing performance will be improved by self-learning. Our WA was first extracted from the 1-best result from parser. With the parser producing *n*-best and the evaluating system finding the best structure, we can continuously derive more and better word associations. Similarly, if we have a better WA referent statistic, we should be able to choose the better structure. This is the idea of how self-learning works. The left side of Figure 6 denotes how we produce knowledge initially, and the right side of Figure 6 explains the repeated procedure of automatic knowledge extraction and accumulation. From the results shown in Table 4 and Table 5, we see that there is much space for improvement.

[6] The English parser has better evaluating results than the Chinese one due to the better performance of the parser and language differences. The charateristic of a strictly regulated grammar in English gives an advantage in parsing. Nonetheless, we have to admit that there is plenty of room for improvement in Chinese parsing.

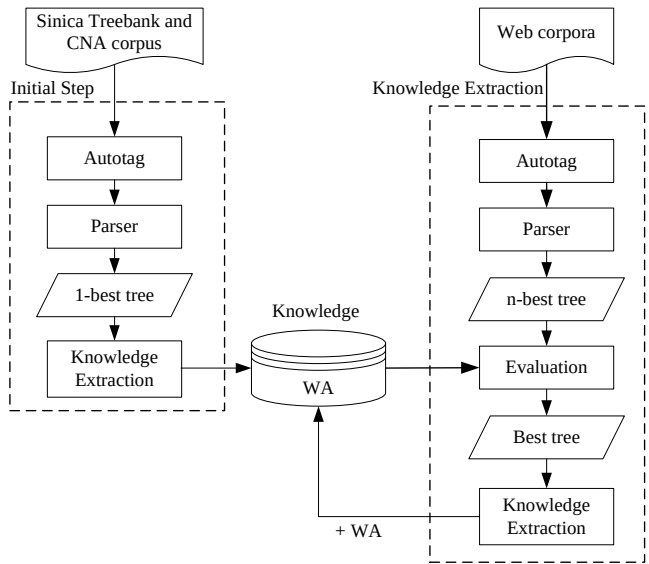


Figure 6. Procedure of self-learning.

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 performance. The real parsing performance of accepting input from a PoS tagging system is shown in Table 6(1). In this table, "Autotag" means 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 resolved in the parsing stage. In Tsai *et al.* [2003], 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". Some examples follow:

confidence value=1.0
他({Nh,Nes}) 叫({VG,VF}) 李四(Nb) 撿({VC,VB}) 皮球(Na)
confidence value=0.8
他(Nh) 叫({VG,VF}) 李四(Nb) 撿(VC) 皮球(Na)
confidence value<0.5
他(Nh) 叫(VF) 李四(Nb) 撿(VC) 皮球(Na)

In Table 6(2), "Autotag with confidence value=1.0" means that if confidence value ≦ 1.0, we list all possible PoSs for the parser to decide. The experimental results of the 1-best, Table 6(2), show that delaying ambiguous PoS resolution does not improve parsing performance, since PoS ambiguities increase structural ambiguities and the PCFG parser is not robust enough to select better syntactic structures. However, for the experiment of 50-best, take the oracle score as the example; the 50-best oracle *f*-scores shown in Table 6(2) are better than the results without leaving ambiguous tags as shown in Table 6(1). Therefore, it is more likely to find better results after applying our evaluation model on the set of data with better oracle scores. Hence, we try to see the power of our evaluation model by leaving ambiguous PoS tags for the testing data.

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 the 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%, for about 4% 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%. This proves that our evaluating model is robust enough to handle ambiguous PoS tagging and produces better results than solely using the unique tag produced by 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
1-best	73.41	68.34	77.83
50-best	86.45	83.99	88.83

7. Conclusion

Parsers of any language aim to correctly analyze the syntactic structure of a sentence, often with the help of semantic information. 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, 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 a method to modify our grammars to increase the oracle scores of the produced *n*-best sentences.

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 than the original 1-best. Furthermore, the ambiguous PoS of a word is also parsed and evaluated on *n*-best, and we prove that our evaluating model is robust enough to improve parsing results on sentences with ambiguous PoS tagging.

From our experiment results, we find that sentences with coordinate structures are more difficult to deal with. The information of semantic parallelism instead of semantic dependencies is required for solving conjunctive structures. The extracted word associations don't have enough discriminative power to resolve both syntactic and semantic symmetry of conjunctive structures. The possible improvement may come from modifying the extraction method or predicting their plausible ranges before parsing. As to other difficult sentences, for example, in Figure 2, the coverage rate of Level 2 (HW_W) associations is only about 70%, which is far less than needed. We may expand our data to read more web texts to resolve this problem.

In future research, we plan to improve the quality of word-association. Four aspects need to be addressed: improving the accuracy of the 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 evaluation model, properly corresponding semantic classifications from coarse to fine-grained categories are needed in Level-6.

Acknowledgments

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.

Reference

Black, E., S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski, "A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars," In *Proceedings of the Workshop on Speech and Natural language*, 1991, pp. 306-311.

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

Chen, K.-J., C.-R. Huang, F.-Y. Chen, C.-C. Luo, M.-C. Chang, C.-J. Chen, and Z.-M. Gao, "Sinica Treebank: design criteria, representational issues and implementation," In *Anne Abeille, (ed.): Building and Using Parsed Corpora. Text, Speech and Language Technology*, 2003, 20, pp. 231-248.

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

Chiu, C.-M., J.-Q. Luo, and K.-J. Chen, "Compositional semantics of mandarin affix verbs." In *Proceedings of ROCLING XVI: Conference on Computational Linguistics and Speech Processing*, 2004, Taipei, pp. 131-139.

CKIP (Chinese Knowledge Information processing), "The categorical analysis of Chinese," Technical Report No. 93-05, Institute of Information Science Academia Sinica, Taipei, 1993.

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

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

Hsieh, Y.-M., D.-C. Yang, and K.-J. Chen, "Linguistically-motivated grammar extraction, generalization and adaptation," In *Proceedings of the Second International Join Conference on Natural Language Processing*, 2005, Jeju Island, Republic of Korea, pp. 177-187.

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

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

Tsai, Y.-F., and K.-J. Chen, "Context-rule model for PoS tagging," In *Proceedings of 17th Pacific Asia Conference on Language, Information and Computation (PACLIC 17)*, 2003, COLIPS, Sentosa, Singapore, pp. 146-151.

Xiong, D., S. Li, Q. Liu, S. Lin, and Y. Qian, "Parsing the Penn Chinese Treebank with semantic knowledge," In *Proceedings of the Second International Join Conference on Natural Language Processing*, 2005, Jeju Island, Republic of Korea, pp. 70-81.

Appendix A. Transformation algorithm

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 then *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)*