Japanese Dependency Structure Analysis based on Lexicalized Statistics

MASAKAZU **Fujio**
NAIST
8916-5 Takayama, Ikoma
Nara, 630-0101 JAPAN
masaka-h@is.aist-nara.ac.jp

YUJI **Matsumoto**
NAIST
8916-5 Takayama, Ikoma
Nara, 630-0101 JAPAN
matsu@is.aist-nara.ac.jp

Abstract

We present statistical models of Japanese dependency analysis and report results of some experiments to investigate the performance of the models for the use fo a partical parsing system. The statistical models are rather simple compared with the recent complex models and intesively use lexical level information, such as morphemes, and part-of-speech tags..

We conducted several experiments to show the following properties of the models:

- performance of the models according to feature selection
- performance of the models as a partial parsing system.

The EDR[6] corpus was used for both training and evaluation of the system.

1. Introduction

A number of statistical parsing methods have been proposed. most of the systems focus on full parsing of sentences, and do not discuss the performance of partial parses, which is crucial for some applications, such as information retrieval or pre processing of corpus annotation.

Early approaches of statistical parsing [15, 10, 13] conditioned probabilities on syntactic rules. To take more contextual information into account, word collocation is applied to syntactic formalization, such as lexicalized PCFG, lexicalized tree adjoining grammar, and lexicalized link grammar.

The length of phrases or the distance between headwords were also considered in the several models [16, 8]

There are parsing methods that do not require a grammar. Collins [3] proposes a statistical parser based on probabilities of dependencies between head-words in parse trees. Yasuhara [18], constructs a system based on collocation counts as the only source of grammatical information. He uses co-occurrence patterns of the POS tags of head-words. The method, however, is not statistical, in that it only accumulates correct patterns for direct use.

Magerman [4] proposes a statistical parser based on a decision tree model, in which the probabilities are conditioned on the derivation history of the parse trees [4, 10]. He compares the decision tree model with the n-gram model, and claims that the amount of parameters in the resulting model remains relatively constant, depending mostly on the number of training examples.

Charniak [5] proposes a new model and compared it with Collins', and Magerman's models and shows what aspects of these systems affect their relative performance.

In general, statistical models suffer from the problem of data sparseness.

Instead of using a complex statistical model combined with various smoothing techniques [1, 2, 7, 9], We stick to a statistical model of simple setting aiming at an easy implementation, and pursue a way to select useful information for achieving higher parse accuracy.

The basic model is close to Collins' model[3] Japanese dependency structure are usually based on phrasal units (called "*bunsetsu*"). A *bunsetsu* basically consists of one (or a sequence of) content word(s) and its succeeding function words (that forms the smallest phrase, such as a simple noun phrase.).

We consider the dependency structure such that every *bunsetsu* in a sentence except the right most one modifies one of its following *bunsetsu*'s in the sentence and no two modifications may cross each other.

The difference of our model to Collins' model principally comes from the property of Japanese sentence structure. First, the type of modification relation (dependency relations) is uniqly determined by the function words or the ending form of the modifier. Second, the modification always direct from left to right since Japanese is a head-final language.

There are various features that may affect the parsing precision. We test a number of possible setting and try to find out the best combination of features. We also test the performance of partial parsing in several settings. 200,00 parsed Japanese sentences in EDR corpus is used for evaluation.

In the next section, the statistical model is described. Section 3 outlines the parsing algorithm is outlined. section 4 presents the evaluation method. Final section is for conclusion and future work.

2. The Statistical Model

We propose a statistical model based on the features of *bunsetsu*'s. Those features usually defined by the result of morphological analysis, such as part-of-speech (POS) tags, inflection types, punctuations, and other grammatical or surface information. Some features are determined not directly from the modifier and modifiee *bunsetsu's* For instance, the number of *bunsetsu* between a modifier and a modifiee can be a feature.

We first introduce notational conventions. $S = w_1, \ldots, w_n$ is a sentence, where w_i is the i-th word. T is a sequence of words and tag pairs, that is, $T = < w_1, t_1 >, \ldots, < w_n, t_n >$. F is a sequence of *bunsetsu* and feature pairs, that is, $F =< b_1, \mathbf{f}_1 >, \ldots, < b_m, \mathbf{f}_\mathrm{m} >$. We use the notation $Dep(i) = j$ to indicate that the i-th *bunsetsu* in the sequence is a modifier to the j-th *bunsetsu*. Here, the symbol $w_i, t_i, andb_i$ stand for word, tag, and *bunsetsu* respectively, and $\mathbf{f}_\mathrm{i}$ represents the set of features assigned to *bunsetsu* b_i. The subscripts m, *and* n stand for the number of *bunsetsu's* and words, respectively. L is the sequence of dependencies: $L = \langle Dep(1), Dep(2), \ldots, Dep(m-1) \rangle$.

In general, a statistical parsing model estimates the conditional probability, $P(P_t \mid S)$, for each candidate parse tree P_t for a sentence S. In Japanese dependency structure analysis, the final goal is to identify L rather than P_t, and we try to maximize the probability $P(L, F, T \mid S)$.

The most likely dependency structure analysis under the model is then:

$$\begin{aligned} L_{best} &= \operatorname*{argmax}_{L,F,T} P(L, F, T|S) \\ &= \operatorname*{argmax}_{L,F,T} P(L|F, T, S)\ P(F|T, S)\ P(T|S) \end{aligned}$$

We assume that *bunsetsu* construction only depend on word/tag pairs, hence $P(F \mid T, S) = P(F \mid T)$, and assume that a dependency structure can be determined only by *bunsetsu* features, thus $P(L \mid F, T, S) = P(L \mid F)$. The equation (1) is now written:

$$L_{best} = \operatorname*{argmax}_{L,F,T} P(L|F)\ P(F|T)\ P(T|S)$$

For simplicity, we assume that the morphological analysis and the *bunsetsu* construction are both deterministic. For the morphological analysis, we use the most likely output of the Japanese morphological analyzer ChaSen [11].

For the *bunsetsu* construction, we use a finite state transducer constructed from regular expressions of word/tag pairs.

What we need to do therefor is to estimate $P(L \mid F)$, and find L for each S that maximizes the conditional probability $P(L \mid F)$.

We assume that dependencies are mutually independent, that is,

$$P(L \mid F) = \prod_{i=1}^{m-1} P(Dep(i){=}j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \qquad (1)$$

and no two modifications may cross each other.

$\mathbf{f}_1, .., \mathbf{f}_\mathrm{m}$ stands for the sequence of *bunsetsu* features assigned to the *bunsetsu*. Thus, $P(L \mid F)$ can be defined as the product of the probability of dependency pairs.

One point that differs from the Collins' model is that our model does not estimate the type of dependency relations. It only estimate the existence of the dependency relations. This is because the type of dependency is determined uniquely by the modifier in Japanese sentences.

The model estimate the probability of each dependency pair directly by maximum likelihood estimation based on *bunsetsu* features. Head-words, POS tags, word classes, function words, punctuations, and distance measure such as the number of *bunsetsu*'s are used available for the probability estimation.

We can expand each item of the equation (1) by using those features, and assuming independence of the co-occurrence of some features. In the following, we discriminate the *bunsetsu* features that directory relate to the modifier and modifiee and the distance features that relate to relative positions of the modifier and the modifiee.

$$\begin{aligned} &P\ \ (Dep(i){=}j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \\ &\approx\ \ P_h(Dep(i){=}j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \qquad (2) \\ &\qquad \times P_d(Dep(i){=}j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \qquad (3) \end{aligned}$$

In the second equation, we assume independence of two kinds of probabilities. The first is the collocation probability between *bunsetsu features*, and the second one is the distance feature between two *bunsetsu*'s. The independency of these two probabilities reduce the size of the model.

We refer to the probability (2) as the collocation probability, and the probability (3) as the distance probability.

The remainder of this section explains these probabilities in detail.

Head Collocation Probability

Japanese language has dependency relations expressed by the function words or the ending form, and they play a crucial role in determining the dependency structure. The relation name (type) is usually determined by the function words.

If a *bunsetsu* has no function words, we use POS tag (and inflection type) of the right most content word of the *bunsetsu*.

Head word is basically defined by the right most content word in the each *bunsetsu*.

By using these features, we define two models of head-collocation probabilities. The first is the generation probability of features and the second is the collocation probability of features.

In the first model, we assume Japanese dependency structure is the result of selectional process of which each modifier selects a modifiee. The selectional probability is written as $F_g(h_j, r_j, p_j \mid h_i, r_i, p_i)$. In this expression, the modifiee's features are h_j, r_j, p_j given that modifier's features are h_i, r_i, p_i. The symbols $h_i, r_i, and p_i$ stand for head feature, relation type, and punctuation, respectively. With this setting, we make the following approximation:

$$\begin{aligned} & P_h \ (Dep(i) = j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \\ \stackrel{\mathrm{def}}{=} \ & F_g(h_j, r_j, p_j \mid h_i, r_i, p_i) \end{aligned}$$

The maximum-likelihood estimate of F_g is given as follows:

$$\begin{aligned} & F_g \ (h_j, r_j, p_j \mid h_i, r_i, p_i) \\ = \ & \frac{C(Dep(i) = j, h_i, r_i, p_i, h_j, r_j, p_j)}{C(Dep(i) = j', h_i, r_i, p_i)} \end{aligned}$$

$C(Dep(i) = j, \ h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times that feature pairs of h_i, r_i, p_i and h_j, r_j, p_j are in a dependency relation in the training data.

In the second model, we define the the selectional probability as $F_c(Dep(i) = j \mid h_i, r_i, p_i, h_j, r_j, p_j)$. This is the probability that *bunsetsu* b_i modifies *bunsetsu* b_j when those *bunsetsu*'s appear in the same sentence.

$$\begin{aligned} & P_h \ (Dep(i) = j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathrm{m}) \\ \stackrel{\mathrm{def}}{=} \ & F_c(Dep(i) = j \mid h_i, r_i, p_i, h_j, r_j, p_j) \end{aligned}$$

The maximum-likelihood estimate of F_c is given as follows:

$$\begin{aligned} & F_c \ (Dep(i) = j \mid h_i, r_i, p_i, h_j, r_j, p_j) \\ = \ & \frac{C(Dep(i) = j, \ \ h_i, r_i, p_i, h_j, r_j, p_j)}{C(h_i, r_i, p_i, h_j, r_j, p_j)} \end{aligned}$$

$C_s(h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times h_i, r_i, p_i and h_j, r_j, p_j appear in the same sentence in the training data. $C_s(Dep(i) = j, \ h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times h_i, r_i, p_i and h_j, r_j, p_j are seen in the same sentence in the training data and b_i modifies b_j with the relation r_i.

For the head feature h_i, we can use the head word, as well as the POS tag or the word class of a head word. We use the Japanese thesaurus ' Bunrui Goi Hyou'(BGH)[12] to define word classes. BGH has a six-layered abstraction hierarchy, in which more than 80,000 words are assigned at the leaves.

For each of those probabilities explained above, we tested the following models for feature selection.

POS model	uses POS tags for the head feature.
LEX model	uses POS tags and lexical forms for the head feature.
BGH model	uses POS tags, lexical forms, and word classes for the head feature.

To acquire the statistics, we have to resolve the following ambiguities:

- Which level of thesaurus hierarchy is appropriate as the class for head-word
- How much information from the function words should be considered to define the dependency relation names.

For the limitation of computer resources, we could not use all the combination of word classes (the combination of modifier and modifiee). The collocation of word classes in the same layer in BGH was learned (from the 2nd to 6th layer) and used separately.

In the current implementation, we count the statistics for various length of dependency relation names. Consider the examples in Table 1.

Relation feature of modifier in 3 → 4 may be "までに" or "に". Relation feature of modifiee in 3 → 4 may be "せる" or empty.

Then, head collocation feature combinations defined fo 3 → 4 are as follows (in the case of LEX model):

[私 は]$_1$ [それ を]$_2$ [春 まで – に]$_3$ [完成 させる]$_4$ (I complete it untill this spring)

	modifier's features		modifiee's features	
	relation name	head	head	relation name
1 →4	私	は (particle)	完成	させる
2 →4	それ (demonstrative pronoun)	を (case particle)	完成	させる
3 →4	春	まで (particle)–に (case particle)	完成	させる

Table 1: Example of dependency relations. Each square bracket represents a *bunsetsu*

modifier's feature		modifiee's feature	
relation name	head	head	relation name
まで-に	春	完成	させる
まで-に	春	完成	-
に	春	完成	させる
に	春	完成	-
まで-に	Noun	完成	させる
まで-に	Noun	完成	-
に	Noun	完成	させる
に	Noun	完成	-
まで-に	春	Noun	させる
まで-に	春	Noun	-
に	春	Noun	させる
に	春	Noun	-
まで-に	Noun	Noun	させる
まで-に	Noun	Noun	-
に	Noun	Noun	させる
に	Noun	Noun	-

The Distance Probability

Distance measure of dependency relations is an important factor to disambiguate dependency structure. For instance, relation type "ha/particle" has a tendency to modify a distant phrasal unit.

For the distance measure of a pair of bunsetsu's, we use the numbers of the *bunsetsu's* and punctuations between them.

Two types of probabilities are considered for the probabilities of head-collocation described above.

Generation probability model of the distance features is as follows:

$$P_d(Dep(i)=j \mid \mathrm{f}_1, \ldots, \mathrm{f}_\mathrm{m}) \approx F_g^d(r_i, d_{ij}, p_{ij} \mid r_i) = \frac{C(Dep(i)=j, r_i, d_{ij}, p_{ij})}{C(Dep(i) = j', r_i)}$$

Collocation probability version of the distance features is as follows:

$$P_d(Dep(i)=j) \approx F_c^d(Dep(i)=j \mid r_i, d_{if}, p_{ij}) = \frac{C(Dep(i)=j, r_i, d_{ij}, p_{ij})}{C(r_i, d_{ij}, p_{ij})}$$

d_{ij}, and p_{ij} indicate the number of *bunsetsu*'s and the number of punctuations, respectively.

Same as the case of estimation of head collocation probabilities, modification relations of various length was extracted from each modification pair.

3. The Algorithm

Full Parse

1. Tokenization and POS-tagging is applied to the input
2. Construct *bunsetsu'* and define its features,
3. Calculate the probabilities of every *bunsetsu* pair, by using statistics derived from the EDR corpus.
4. Compose the most likely (or n-best) dependency structure based on the statistical model described in section 2.

For the first step, we use the morphological analyzer, ChaSen[11].

For the second step, tokens are analized into *bunsetsu'* based on pre-defined regular expressions, and then *bunsetsu* features are extracted. The basic rules for assigning features are as follows:

- The right most content word in the *bunsetsu* becomes the head feature.
- Morphological information (such as word, tag, and inflection form) of function words in the *bunsetsu* defines the dependency relation.

There is a room to customize the rules by a user to cope with exceptional cases which do not fall into a general pattern, and to cope with conceptual differences between system designs.

For the fourth step, we consider the dependency structure such that:

- Every *bunsetsu* in S except the right most one modifies one of its succeeding *bunsetsu*'s in the sentence
- No two modifications may cross each other (crossing constraint)

Under those constraints, we use CYK algorithm to effectively select the most likely (n-best) combination of dependency relations.

Partial Parse

We propose three types of partial parsing, which focuses on the probabilities of each dependency pairs (p0), the probabilities of whole dependency structure (p1), and some specific dependency relations (p2).

(p0) Output dependency relations of which probability is higher than a particular threshold. The result is the set of dependencies.

(p1) N-best parses are firstly obtained. Then, the dependencies that are included in all of the N-best parses are selected as the result.

(p2) Only the dependencies of the specified relations are produced.

In the *p0* algorithm, we do not use CYK algorithm. If there are more than two modifiees whose dependency probabilities are higher than the threshold, the highest one is chosen (in other words, do not care about "crossing constraint"). Although this method is very simple, it is useful, for example, to help interactive correction procedure of tree-bank construction.

To use the *p2* algorithm, we must evaluate the precision for each relation typef. Some experiments are given in the following section.

4. System Evaluation

For the training and test corpora, we used EDR Japanese bracketed corpus [6], which contains about 208,000 sentences collected from articles of newspapers and magazines.

We splitted the sentences into twenty files. One of these files is held out for evaluation and others are used for training.

Full parse accuracy is evaluated by the precision of correct dependency pairs. Partial parse accuracy is evaluated by the precision and recall of correct dependency pairs.

Precision and recall are defined as follows:

$$Precision = \frac{\text{Number of correct dependencies generated by the system}}{\text{Number of system's output of dependencies}}$$

$$Recall = \frac{\text{Number of correct dependencies generated by the system}}{\text{Total number of dependencies}}$$

Evaluation of Full Parse

The precision of the number of dependency pairs was calculated under the following models.

(a) Base-line

(b) POS model

(c) LEX model

(d) BGH model

The model (a) is used as the base-line, in which all modifiers modify its immediate right *bunsetsu*. "POS model" means that POS tags of head-words are used as the head feature. "LEX model" means that POS tags of head-words and lexical items are used. "BGH model" means that POS tags, lexical items, and word classes are used as the head feature. The level of the layers in the thesaurus is altered from 2 to 6 (leaf layer).

For each of (b), (c), (d) models, we applied two probability models described in section 2 (generation probability and collocation probability) to each of head-collocation probability and distance probability. Then each (a), (b), (c), and (d) models has four different models. But we only shows the result of the following two models, for the each POS, LEX, and BGH model.

- head-collocation (collocation model) + distance (generation model) → model-1
- head-collocation (collocation model) + distance (collocation model) → model-2

Since the other two models give the performance (precision) as low as 70 %, we will not go into more detail of those models. The amount of training data was changed and evaluated in terms of the precision of correct dependency relations.

Figure 1 shows the result of the precision for the inside and outside data under "model-1"[1] . Figure 2 shows the result of the precision for the inside and the outside data under "model-2".

"BGH:6" in the figure means that the sixth-layer of the thesaurus is used for the word class. It slightly outperforms other models that use higher layers in the thesaurus.

When evaluating with outside data, we imposed certain frequency threshold on the statistical data, that is, the collocation data whose occurrence frequency is less than *i*-times was discarded, where *i* is a predetermined threshold.

Figure 3 show the resulting change of precisions under the POS, LEX, BGH models. The value of "*i*" was changed from 2 to 10.

[1] By "inside data", we mean that the training data is used also for the test data, whereas "outside data" means that the held-out data is used for the test data.

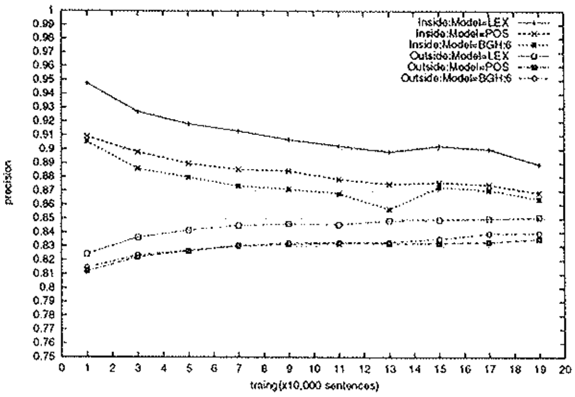

Figure 1: Precision under model-1.

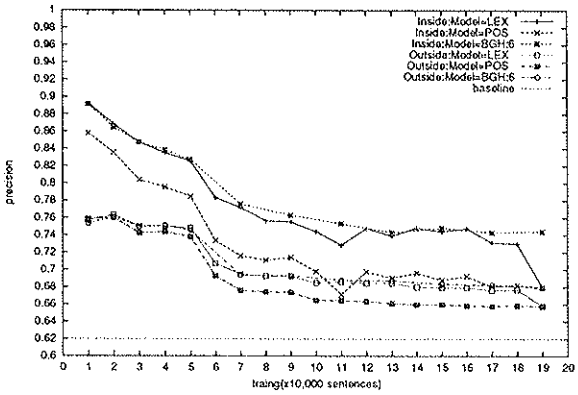

Figure 2: Precision under model-2.

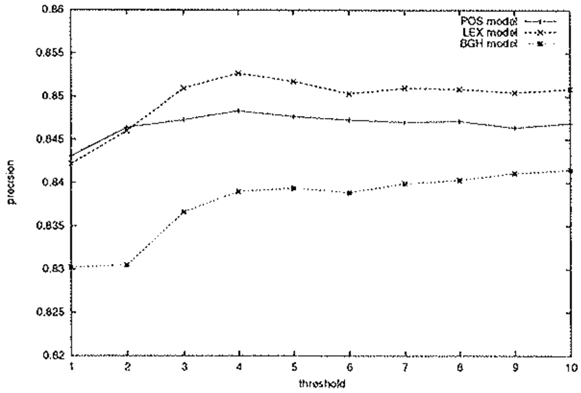

Figure 3: Precision of full parses. Trained from 190,000 sentences. Evaluated by 1,000 sentences

From this experiment, we decided to set the value of i to 4.

The LEX model shows the highest performance in both cases, and the result of model-1 outperforms that of model-2 constantly.

Surprisingly, the BGH model shows poor performance than the POS model. A part of the reason may comes from the fact that we only used one layer of word classes for each experiments. Other reason may be that the hierarchy of "Bunrui Goi Hyou" is not adequate for the syntactic analysis.

The graph shows that the performance of the inside data decreases when the size of training data increases.

The precision of the outside data in "model-1" constantly close up to the precision of the inside data.

We use "model-1" for further analysis.

Contribution of Head-Collocation Probability and Distance Probability

To test which features of head-collocation and distance feature contribute to the accuracy of parsing, the following models are tested.

(e) Distance probability

(f) POS model without the distance probability

(g) LEX without the distance probability

(h) BGH without the distance probability

Each model is trained by 190,000 sentences, and evaluated by 1,000 sentences held out from the training data.

model	precision %	correct/total
(e)	66.07	5087/7610
(f)	79.09	6019/7610
(g)	80.09	6095/7610
(h)	77.58	5819/7610

Table 2: Precision for 1,000 sentences.

The distance probability makes little contribution to the parsing accuracy compared to the head collocation probability. This is because the features used for the distance probability is too simple.

Sentence Level Evaluation

We evaluate sentence level accuracy in this section. A sentence is regarded as correct if the correct structure is found in the n-best parse of the parser, where n is a predetermined value.

Figure 4 shows the rate of correct parses appearing in the n-best parses, where n is changed from 1 to 10. The average number of *bunsetsu*'s in a sentence is 7.

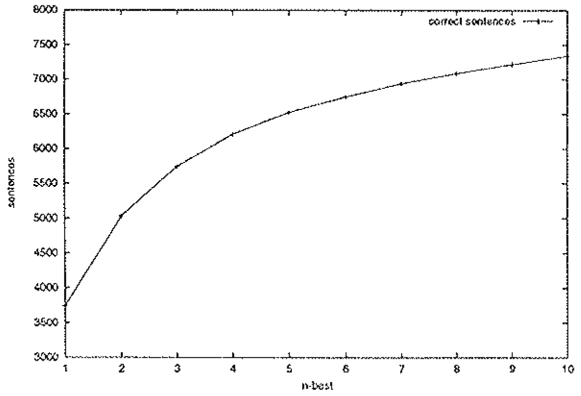

Figure 4: Distribution of correct parses (out of 10,000 sentences). Trained under LEX model by 190,000 sentences.

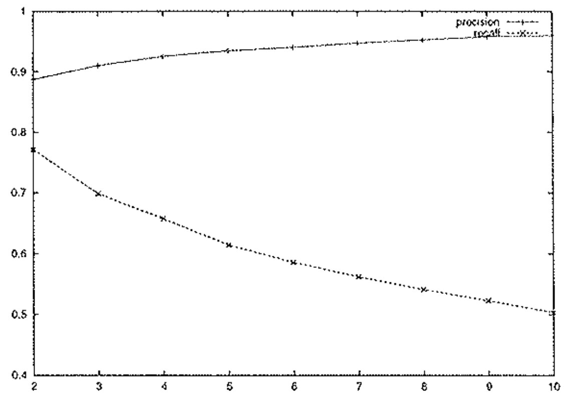

Figure 6: Evaluation of *p1* algorithm. LEX model learned from 190,000 sentences was used.

When n is 5 the precision is 65.21 %, and when n is 10, it becomes 73.40 %.

Evaluation of Each Relation Types

We also check the precision of relation types. The results are shown in Table 5. The first column specifies the type of dependency, which consists of a word, a tag or an inflection form. The second column in Table 5 indicates the ratio of correct dependencies over the total system output.

It is seen that the frequencies of relation type, noun base-form-verb, and ha-particle are high, and influence system's performance, since the precisions for these relations are bad. The particle "ha', "verb/renyou", and "verb/tekei" can construct subordinate clauses in Japanese, and in some cases, it is difficult even for human to consistently determine its modifiee.

A noun + punctuation pattern is also a problematic case, because it can be a part of conjunction phrases. They behave like adverbs (temperal noun and adverbial noun) or form subordinate clauses.

In these cases, it is reasonable to leave these modifiees unspecified. This doesn't conflict the purpose of using the system for practical fields or preprocessor of higher NLP, because it is favorable to output reliable partial parses rather than output unreliable full parses.

Evaluation of Partial Parsing

The results of full parsing accuracy show that model-1 under the LEX model outperforms other models.

For the model, we further examined partial parsing methods explained in section 3, and evaluated its precision and recall.

Table 3 shows the result of *p0* algorithm. The first column in Table 3 indicates the threshold on the probability of each dependency relation. The degree of the reliability (hence the degree of the precision) can be controlled by the value of the threshold on the probabilities.

threshold	precision %(correct/total)	recall % (correct/total)
0.5	86.16 (6356/7377)	83.52 (6356/7610)
0.6	88.23 (6193/7019)	81.38 (6193/7610)
0.7	90.24 (5999/6648)	78.83 (5999/7610)
0.8	92.33 (5705/6179)	74.97 (5705/7610)
0.9	95.19 (5149/5409)	67.66 (5149/7610)

Table 3: Evaluation of *p0* algorithm. LEX model learned from 190,000 sentences was used.

Figure 5 shows the result of *p1* algorithm. The value of "n" in the *p1* algorithm is varied from 2 to 10.

The degree of the precision can be controlled by the value of "n". Figure 6 depicts the results in graphs.

threshold	precision %(correct/total)	recall % (correct/total)
2	88.77 (5149/5409)	77.14 (5149/7610)
3	91.03 (5705/6179)	69.91 (5705/7610)
4	92.53 (5999/6648)	65.80 (5999/7610)
5	93.47 (6193/7019)	61.46 (6193/7610)
6	93.99 (6356/7377)	58.59 (6356/7610)
7	94.71 (6356/7377)	56.23 (6356/7610)
8	95.26 (6356/7377)	54.14 (6356/7610)
9	95.78 (6356/7377)	52.30 (6356/7610)
10	95.99 (6356/7377)	50.38 (6356/7610)

Table 4: Evaluation of *p1* algorithm. LEX model learned from 190,000 sentences was used.

Table 5 shows the result of *p2* algorithm. *p2* algorithm achieves slightly better precision than full parse, but is not as good as *p0* and *p1* algorithms.

When comparing three methods, *p0* algorithm shows highest performance, in terms of the precision and re-

relation name (lexicon/POS/inflection form)	precision (%)	correct	total
/adjective/rentai	95.41	1019	1068
/demonstrative/	93.72	1329	1418
wo/cp/	93.32	7000	7501
no/p/	92.15	11040	11980
ni/cp/	91.51	5769	6304
/adjective/renyou	88.14	959	1088
ga/cp/	87.94	5025	5714
/verb/base	87.32	1344	1539
to/cp/	85.49	1585	1854
mo/p/	83.54	1680	2011
de/cp/	81.83	991	1211
/verb/tekei	79.55	926	1164
/temporal noun/	78.20	1155	1477
da/declarative/tekei	77.96	902	1157
ha/p/	75.32	5790	7687
/noun/	75.29	1182	1570
/verb/renyou	72.43	796	1099

Figure 5: System's outputs were classified according to the right most constituent of relation type, and sorted with their precisions. The symbol cp, and p in the first column mean case-particle and particle. Renyou, rentai tekei and base are the names of inflection forms.

relation types	precision%
without "ha"	86.21 (5904/6808)
without "verb/renyou,tekei"	85.56 (6333/7402)
without "verb/renyou,tekei, ha"	86.57 (5748/6640)

Table 5: Dependency relations without some types of relations. Trained by 190,000 sentences. Evaluated by other 1,000 sentences.

call. When *p0* and *p1* algorithm shows same precision, *p0* algorithm shows higher recall.

p0 and *p1* algorithms can be controlled by a single parameter.

5. Conclusion and Future Works

We showed that the statistical method incorporating lexical level information without any grammar rule is effective in Japanese dependency structure analysis.

Instead of lexical items, we also tested word classes of the thesaurus as head features of phrasal units (BGH model). But that model showed poor performance than the POS model (which uses part-of-speech tags, as head features). This may be because that the hierarchy of applied thesaurus is not appropriate for the syntactic analysis.

85 % of precision (the number of correct dependency relations) is achieved by using LEX model.

In those experiments, the combinations of features are determined manually by human. There is a room to select the combinations of features automatically. One reason of this comes from the fact that we applied various kinds of distance features, such as the number of noun phrases, the number of case particles, the number of verbs and other kinds of grammatical features between two *bunsetsu*'s, but finally it turned out that simple features, such as the number of *bunsetsu*'s and punctuations between two *bunsetsu*'s shows good performance. This may imply the limitaion of manual selection of combinations of features. Automatical selection of appropriate features is one of our future works.

We also proposed several partial parse methods. Among them, *p0* algorithm is exhibited highest performance in terms of precision and recall, in spite of its simplicity of algorithm.

In *p0* algorithm, the degree of reliability (in other word, degree of precision) is controllable by a single parameter.

Partial parse method can be used for other NLP applications, such as information retrieval or preprocessing of corpus annotation.

References

[1] S.F. Chen and. An empirical study of smoothing techniques for language modeling. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 310–318, Jun 1996.

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

[3] Michael John Collins. A new statistical parser based on bigram lexical dependencies. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 184–191, Jun 1996.

[4] D.Magerman. Statistical decision-tree model for parsing. *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics*, pp. 276–283, Jun 1995.

[5] E.Charniak. Statistical parsing with a context-free grammar and word statistics. AAAI, pp. pages 598–603., 1997.

[6] Japan Electronic Dictionary Research Institute, Ltd. *EDR Electronic Dictionary Technical Guide*. 1996.

[7] F.Jelinek and R.L.Mercer. Interpolated estimation of markov source parameters from sparse data. *Proceedings of the Workshop on Pattern Recognition in Practice*, pp. 400–401, March 1987.

[8] H.Li. A probabilistic disambiguation method based on psycholinguistic. *Proceedings of the Forth Workshop on Very Large Corpora*, Aug 1996.

[9] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-35, No. 3*, pp. 400–401, March 1987.

[10] D. Magerman. and M. Marcus. Pearl: A probabilistic chart parser. Proceedings of the 1991 European ACL Conference, 1991. Berlin, Germany.

[11] Y. Matsumoto, O. Imaichi, T. Yamashita, A Kitauchi, and Tomoaki Imamura. *Morphological analysis system ChaSen version 1.0b5 user manual*. Matsumoto lab. Nara Institute of Science and Technology. (in Japanese), 1996.

[12] *Word List by Semantic Principles*, syuei syuppan. (in japanese), 1964,1993.

[13] F. Pereira. and Y. Schabes. Inside-outside reestimation from partially bracketed corpora. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. pages 128–135., 1992.

[14] R Rivest. Learning decision lists. *Machine Learning*, pp. 229–246, 1987.

[15] T.Briscoe and John Carroll. Generalized probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, Vol. Vol.19, No.1, pp. 25–29, Mar 1993.

[16] W.R.Hogenhout and Y.Matsumoto. Training stochastic grammars on semantical categories. *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, 1996.

[17] D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. *Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics*, pp. 88–95, Jun 1994.

[18] H Yasuhara. Kakari-uke dependency analysis with learning function based on reduced type cooccurrence relation. *Journal of Japanese Association for Language Processing*, pp. 87–101, Oct 1996.