Learning Preference of Dependency between Japanese Subordinate Clauses and its Evaluation in Parsing

Takehito Utsuro

Department of Information and Computer Sciences, Toyohashi University of Technology
Tenpaku-cho, Toyohashi, 441-8580, Japan
utsuro@ics.tut.ac.jp

Abstract

(Utsuro et al., 2000) proposed statistical method for learning dependency preference of Japanese subordinate clauses, in which scope embedding preference of subordinate clauses is exploited as a useful information source for disambiguating dependencies between subordinate clauses. Following (Utsuro et al., 2000), this paper presents detailed results of evaluating the proposed method by comparing it with several closely related existing techniques and shows that the proposed method outperforms those existing techniques.

1. Introduction

In dependency analysis of a Japanese sentence, among various source of ambiguities in a sentence, dependency ambiguities between subordinate clauses are one of the most problematic ones, partly because word order in a sentence is relatively free. In general, dependency ambiguities between subordinate clauses cause scope ambiguities of subordinate clauses, which result in enormous number of syntactic ambiguities of other types of phrases such as noun phrases.[1] In the Japanese linguistics, a theory of (Minami, 1974) regarding scope embedding preference of subordinate clauses is well-known. (Minami, 1974) classifies Japanese subordinate clauses according to the breadths of their scopes and claim that subordinate clauses which inherently have narrower scopes are embedded within the scopes of subordinate clauses which inherently have broader scopes (details are in section 2.). In the Japanese computational linguistics community, (Shirai et al., 1995) employed (Minami, 1974)'s theory on scope embedding preference of Japanese subordinate clauses and applied it to rule-based Japanese dependency analysis. However, in their approach, since categories of subordinate clauses are obtained by manually analyzing a small number of sentences, their coverage against a large corpus such as EDR bracketed corpus (EDR, 1995) is quite low.[2]

In order to realize a broad coverage and high performance dependency analysis of Japanese sentences which exploits scope embedding preference of subordinate clauses, we proposed a corpus-based alternative to

Table 1: Statistics of Test Sentences (5%, 10,320) Extracted from EDR Corpus

	Subsets (# of Subordinate Clauses)		Full
	≥ 2	≤ 1	Set
Ratio in # of Sentences	30.3%	69.7%	100%
Ratio in # of Chunks	39.9%	60.1%	100%
Ave. # of Chunks / Sentence	10.2	6.7	7.8
# of Subordinate Clauses	8,789	—	—
Dependency Analysis Precision (Fujio and Matsumoto, 1998)			
Chunk Level	85.3%	86.7%	86.1%
VP Chunk (Ambiguous)	65.7%	—	65.7%
Sentence Level (Best One)	25.4%	47.5%	40.8%
Sentence Level (Best Five)	35.8%	60.2%	52.8%

The author worked on this research while he was an Instructor at Nara Institute of Science and Technology, Japan. The author would like to thank Prof. Yuji Matsumoto, Dr. Masakazu Fujio, and Shigeyuki Nishiokayama for valuable supports to this research.

[1]Table 1 shows the results of our preliminary corpus analysis against 5% (10,320) test sentences extracted from EDR bracketed corpus (EDR, 1995), using the stochastic dependency analyzer of (Fujio and Matsumoto, 1998). About 30% of the sentences have dependency ambiguities of subordinate clauses, for which the precision of chunk (bunsetsu) level dependencies is about 85.3% and that of sentence level is about 25.4% (for best one) $\sim$ 35.8% (for best five), while for the rest 70% of EDR bracketed corpus, the precision of chunk (bunsetsu) level dependencies is about 86.7% and that of sentence level is about 47.5% (for best one) $\sim$ 60.2% (for best five). Dependency ambiguities between subordinate clauses are among the most difficult ones for those 30% sentences, as shown in Table 1 that the precision of the dependency analysis for the head vp chunks of subordinate clauses is much lower (65.7%) than the average (85.3%). In addition to that, when assuming that those ambiguities of subordinate clause dependencies are initially resolved in some way, the chunk level precision increases to 90.4%, and the sentence level precision to 40.6% (for best one) $\sim$ 67.7% (for best five). This result of our preliminary analysis clearly shows that dependency ambiguities between subordinate clauses are among the most problematic source of syntactic ambiguities in a Japanese sentence.

[2]In our implementation, the coverage of the categories of (Shirai et al., 1995) is only 30% for all the subordinate clauses included in the whole EDR corpus.

Table 2: Comparison of Statistical Dependency Analysis Models

		Dependency Relations of Statistical Dependency Model	
		"modify"/"not-modify"	"modify"/"beyond"
Feature Selection	Hand	(Collins, 1996; Fujio and Matsumoto, 1998)	—
	Decision Tree Learning	(Haruno et al., 1998)	Model (a)
	Decision List Learning	Model (b)	**Our Model**

the rule-based manual approach (Utsuro et al., 2000). This paper presents detailed results of evaluating the proposed method by comparing it with several closely related existing techniques and shows that the proposed method outperforms those existing techniques.

First, in (Utsuro et al., 2000), we formalized the problem of deciding scope embedding preference as a classification problem, in which various types of linguistic information of each subordinate clause are encoded as features and used for deciding which one of given two subordinate clauses has a broader scope than the other. We classified the dependency relations of two vp chunks into the following two cases: the case where dependency relation holds between the given two vp chunks, and the case where dependency relation does not hold but the anterior vp chunk modifies another vp chunk which *follows* the posterior vp chunk. Our modeling is different from those of other standard approaches to statistical dependency analysis (Collins, 1996; Fujio and Matsumoto, 1998; Haruno et al., 1998) which simply distinguish the two cases: the case where dependency relation holds between the given two vp chunks, and the case where dependency relation does not hold.[3] In contrast to those standard approaches, we ignore the case where the anterior vp chunk modifies the head vp chunk of another subordinate clause which *precedes* the posterior vp chunk. This is because we assume that this case is loosely related to the scope embedding preference of subordinate clauses. In the experimental evaluation, we show that our modeling of the dependency relations outperforms the standard approach (Model (b) in Table 2, see section 5.2.).

Second, as a statistical learning method, (Utsuro et al., 2000) employed the decision list learning method of (Yarowsky, 1994). One of the advantages of our formalization of decision list learning is that at each feature selection step it considers every possible pair of the subsets of the features of the two subordinate clauses. This is especially true when compared with the decision tree learning (Quinlan, 1993) approach to feature selection of dependency analysis (Haruno et al., 1998), where the utility of each feature is evaluated independently, and thus the utility of the combination of more than one features is not evaluated directly at each feature selection step.[4] In the experimental evaluation, we show that our formalization of decision list learning outperforms the decision tree learning approach (Model (a) in Table 2, see section 5.1.).

Finally, we evaluate the estimated dependencies of subordinate clauses in (Fujio and Matsumoto, 1998)'s framework of the statistical dependency analysis of a whole sentence, in which we successfully increase the precisions of both chunk level and sentence level dependencies thanks to the estimated dependencies of subordinate clauses (section 5.3.).

The rest of the paper is organized as follows: section 2. describes the basic idea of analyzing dependencies between Japanese subordinate clauses utilizing scope embedding preference. Section 3. presents a technique of the decision list learning of dependency preference of Japanese subordinate clauses. Section 4. presents how to analyze dependencies of subordinate clauses in a sentence according to the probabilities of the dependencies between two subordinate clauses. Section 5. describes the results of experimentally evaluating the proposed method.

2. Analyzing Dependencies between Japanese Subordinate Clauses based on Scope Embedding Preference

2.1. Dependency Analysis of A Japanese Sentence

First, we overview dependency analysis of a Japanese sentence. Since words in a Japanese sentence are not segmented by explicit delimiters, input sentences are first word segmented, part-of-speech tagged, and then chunked into a sequence of segments called *bunsetsu*s.[5] Each chunk (bunsetsu) generally consists of a set of content words and function words. Then, dependency relations among those chunks are estimated, where most practical dependency analyzers for the Japanese language usually assume the following two constraints:

1. Every chunk (bunsetsu) except the last one modifies only one posterior chunk (bunsetsu).
2. No modification crosses to other modifications in a sentence.

Table 3 gives an example of word segmentation, part-of-speech tagging, and bunsetsu segmentation (chunking) of a Japanese sentence, where the verb and the adjective are tagged with their parts-of-speech as

[3]Table 2 classifies and compares several statistical dependency analysis models according to two dimensions. The dimension of the "Dependency Relations of Statistical Dependency Model" distinguishes our "modify"/"beyond" model and the standard "modify"/"not-modify" model.

[4]In Table 2, the dimension of the "Feature Selection" distinguishes feature selection by hand, by decision tree learning, and by decision list learning.

[5]Word segmentation and part-of-speech tagging are performed by the Japanese morphological analyzer Chasen (Matsumoto et al., 1997), and chunking is done by the preprocessor used in (Fujio and Matsumoto, 1998).

Table 3: Word Segmentation, POS tagging, and *Bunsetsu* Segmentation of A Japanese Sentence

Word Segmentation	Tenki	ga	yoi	kara	dekakeyou
POS (+ conjugation form) Tagging	noun	case-particle	adjective (base)	predicate-conjunctive-particle	verb (volitional)
Bunsetsu Segmentation (Chunking)	Tenki-ga		yoi-kara		dekakeyou
English Translation	*weather*	subject	*fine*	*because*	*let's go out*
	(*Because the weather is fine, let's go out.*)				

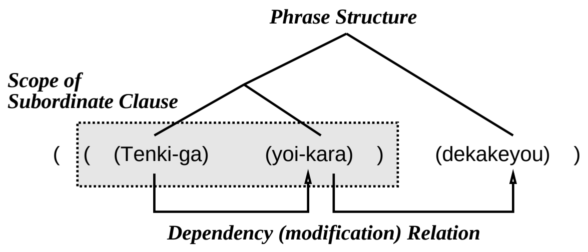

Phrase Structure
Scope of Subordinate Clause
(((Tenki-ga) (yoi-kara)) (dekakeyou))
Dependency (modification) Relation

Figure 1: An Example of Japanese Subordinate Clause (taken from the Sentence of Table 3)

well as conjugation forms. Figure 1 shows the phrase structure, the bracketing, and the dependency relation of the chunks (bunsetsus) within the sentence.

2.2. Japanese Subordinate Clause

The following gives the definition of what we call a "Japanese *subordinate clause*" throughout this paper. A *clause* in a sentence is represented as a sequence of chunks. Since the Japanese language is a head-final language, the clause head is the final chunk in the sequence. A grammatical definition of a Japanese *subordinate clause* is given in Figure 2.[6] For example, the Japanese sentence in Table 3 has one subordinate clause, whose scope is indicated as the shaded rectangle in Figure 1.

2.3. Scope Embedding Preference of Subordinate Clauses

We introduce the concept of (Minami, 1974)'s classification of Japanese subordinate clauses by describing the more specific classification by (Shirai et al., 1995). From 972 newspaper summary sentences, (Shirai et al., 1995) manually extracted 54 clause final function words of Japanese subordinate clauses and classified them into the following three categories according to the embedding relation of their scopes.

Category A: Seven expressions representing simultaneous occurrences such as "$Verb_1$ *to-tomoni* ($Clause_2$)" and "$Verb_1$ *nagara* ($Clause_2$)".

Category B: 46 expressions representing cause and discontinuity such as "$Verb_1$ *te* ($Clause_2$)" (in English "$Verb_1$ *and* ($Clause_2$)") and "$Verb_1$ *node*" (in English "*because* (subject) $Verb_1$ ...,").

[6]This definition includes adnominal or noun phrase modifying clauses "$Clause_1$ (NP_1)" (in English, relative clauses "(NP_1) that $Clause_1$"). Since an adnominal clause does not modify any posterior subordinate clauses, but modifies a posterior noun phrase, we regard adnominal clauses only as modifees when considering dependencies between subordinate clauses.

Category C: One expression representing independence, "$Verb_1$ *ga*" (in English, "although (subject) $Verb_1$ ...,").

The category A has the narrowest scope, while the category C has the broadest scope, i.e.,

$$\text{Category A} \prec \text{Category B} \prec \text{Category C}$$

where the relation '$\prec$' denotes the embedding relation of scopes of subordinate clauses. Then, scope embedding preference of Japanese subordinate clauses can be stated as below:

Scope Embedding Preference of Japanese Subordinate Clauses

1. A subordinate clause can be embedded within the scope of another subordinate clause which inherently has a scope of the same or a broader breadth.
2. A subordinate clause can not be embedded within the scope of another subordinate clause which inherently has a narrower scope.

For example, a subordinate clause of 'Category B' can be embedded within the scope of another subordinate clause of 'Category B' or 'Category C', but not within that of 'Category A'. Figure 3 (a) gives an example of an anterior Japanese subordinate clause (*"kakimaze-nagara"*, Category A), which is embedded within the scope of a posterior one with a broader scope (*"ni-mashita-ga-,"*, Category C). Since the posterior subordinate clause inherently has a broader scope than the anterior, the anterior is embedded within the scope of the posterior. On the other hand, Figure 3 (b) gives an example of an anterior Japanese subordinate clause (*"ari-masu-ga-,"*, Category C), which is not embedded within the scope of a posterior one with a narrower scope (*"kakimaze-nagara"*, Category A). Since the posterior subordinate clause inherently has a narrower scope than the anterior, the anterior is not embedded within the scope of the posterior.

2.4. Preference of Dependencies between Subordinate Clauses based on Scope Embedding Preference

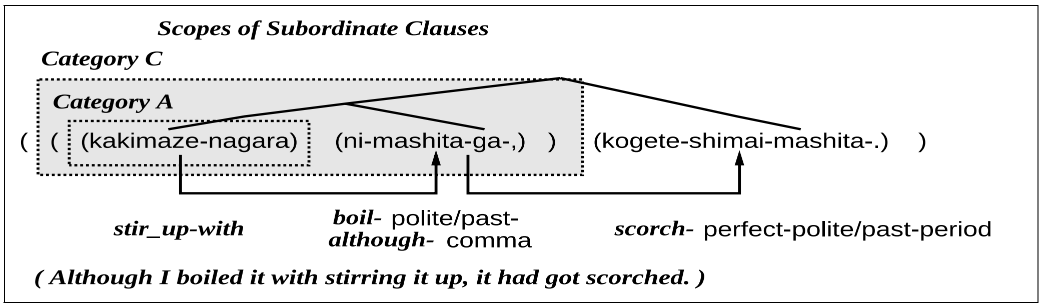
Following the scope embedding preference of Japanese subordinate clauses proposed by (Minami, 1974), (Shirai et al., 1995) applied it to rule-based Japanese dependency analysis, and proposed the following preference of deciding dependencies between subordinate clauses. Suppose that a sentence has two subordinate clauses $Clause_1$ and $Clause_2$, where the head vp chunk of $Clause_1$ precedes that of $Clause_2$.

A Japanese *subordinate clause* is a clause whose head chunk satisfies the following properties.

1. The content words part of the chunk (bunsetsu) is one of the following types:
   (a) A predicate (i.e., a verb or an adjective).
   (b) nouns and a copula like "$Noun_1$ *dearu*" (in English, "*be* $Noun_1$").
2. The function words part of the chunk (bunsetsu) is one of the following types:
   (a) Null.
   (b) Adverb type such as "$Verb_1$ *ippou-de*" (in English, "(subject) $Verb_1$ ..., *on the other hand*,").
   (c) Adverbial noun type such as "$Verb_1$ *tame*" (in English, "*in order to* $Verb_1$").
   (d) Formal noun type such as "$Verb_1$ *koto*" (in English, gerund "$Verb_1$-ing").
   (e) Temporal noun type such as "$Verb_1$ *mae*" (in English, "*before* (subject) $Verb_1$ ...").
   (f) A predicate conjunctive particle such as "$Verb_1$ *ga*" (in English, "although (subject) $Verb_1$ ...,").
   (g) A quoting particle such as "$Verb_1$ *to* (*iu*)" (in English, "*(say) that* (subject) $Verb_1$ ...").
   (h) (a)∼(g) followed by topic marking particles and/or sentence-final particles.

Figure 2: Definition of Japanese Subordinate Clause

(a) Category A $\prec$ Category C


Scopes of Subordinate Clauses
Category C
Category A
(((kakimaze-nagara) (ni-mashita-ga-,)) (kogete-shimai-mashita-.))
stir_up-with
boil- polite/past-
although- comma
scorch- perfect-polite/past-period
(Although I boiled it with stirring it up, it had got scorched.)

(b) Category C $\succ$ Category A

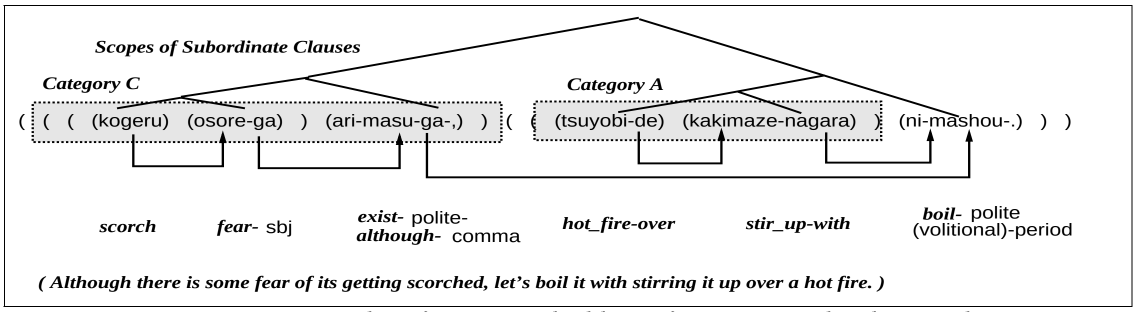

Scopes of Subordinate Clauses
Category C
Category A
((((kogeru) (osore-ga)) (ari-masu-ga-,)) (((tsuyobi-de) (kakimaze-nagara)) (ni-mashou-.)))
scorch
fear- sbj
exist- polite-
although- comma
hot_fire-over
stir_up-with
boil- polite
(volitional)-period
(Although there is some fear of its getting scorched, let's boil it with stirring it up over a hot fire.)

Figure 3: Examples of Scope Embedding of Japanese Subordinate Clauses

Dependency Preference of Japanese Subordinate Clauses

1. The head vp chunk of $Clause_1$ can modify that of $Clause_2$ if $Clause_2$ inherently has a scope of the same or a broader breadth compared with that of $Clause_1$.
2. The head vp chunk of $Clause_1$ can not modify that of $Clause_2$ if $Clause_2$ inherently has a narrower scope compared with that of $Clause_1$.

3. Learning Dependency Preference of Japanese Subordinate Clauses

3.1. Task Definition

Considering the dependency preference of Japanese subordinate clauses described in section 2.4., the following gives the definition of our task of deciding the dependency of Japanese subordinate clauses. Suppose that a sentence has two subordinate clauses $Clause_1$ and $Clause_2$, where the head vp chunk of $Clause_1$ precedes that of $Clause_2$. Then, our task of deciding the dependency of Japanese subordinate clauses is to distinguish the following two cases:

1. The head vp chunk of $Clause_1$ modifies that of $Clause_2$.
2. The head vp chunk of $Clause_1$ does not modify that of $Clause_2$, but modifies that of another subordinate clause or the matrix clause which *follows* $Clause_2$.

Roughly speaking, the first corresponds to the case where $Clause_2$ inherently has a scope of the same or a broader breadth compared with that of $Clause_1$, while the second corresponds to the case where $Clause_2$ inherently has a narrower scope compared with that of $Clause_1$.

3.2. Decision List Learning

A decision list (Yarowsky, 1994) is a sorted list of the decision rules each of which decides the value of a *decision D* given some *evidence E*. Each decision rule in a decision list is sorted in descending order with respect to the following *log of likelihood ratio*:

$$\log_2 \frac{P(D{=}x_1 \mid E{=}1)}{P(D{=}\neg x_1 \mid E{=}1)}$$

Table 4: Features of Japanese Subordinate Clauses

Feature Type	# of Features	Each Binary Feature
Punctuation	2	with-comma, without-comma
Grammatical (some features have distinction of chunk-final/middle)	17	adverb, adverbial-noun, formal-noun, temporal-noun, quoting-particle, copula, predicate-conjunctive-particle, topic-marking-particle, sentence-final-particle
Conjugation form of chunk-final conjugative word	12	stem, base, mizen, ren'you, rentai, conditional, imperative, *ta*, *tari*, *te*, conjecture, volitional
Lexical (lexicalized forms of 'Grammatical' features, with more than 9 occurrences in EDR corpus)	235	adverb (e.g., *ippou-de, irai*), adverbial-noun (e.g., *tame, baai*) topic-marking-particle (e.g., *ha, mo*), quoting-particle (*to*), predicate-conjunctive-particle (e.g., *ga, kara*), temporal-noun (e.g., *ima, shunkan*), formal-noun (e.g., *koto*), copula (*dearu*), sentence-final-particle (e.g., *ka, yo*)

The final line of a decision list is defined as 'a default', where the likelihood ratio is calculated as the ratio of the largest marginal probability of the decision $D = x_1$ to the marginal probability of the rest $D = \neg x_1$:

$$\log_2 \frac{P(D = x_1)}{P(D = \neg x_1)}$$

Then, rules with higher preference values are applied first when applying the decision list to some new data.

3.3. Feature of Subordinate Clauses

Japanese subordinate clauses defined in section 2.2. are encoded using the following four types of features: i) Punctuation: represents whether the head vp chunk of the subordinate clause is marked with a comma or not, ii) Grammatical: represents parts-of-speech of function words of the head vp chunk of the subordinate clause,[7] iii) Conjugation form of chunk-final conjugative word: used when the chunk-final word is conjugative, iv) Lexical: lexicalized forms of 'Grammatical' features which appear more than 9 times in EDR corpus. Each feature of these four types is binary and its value is '1' or '0' ('1' denotes the presence of the corresponding feature, '0' its absence). The whole feature set shown in Table 4 is designed so as to cover the 210,000 sentences of EDR corpus.

3.4. Decision List Learning of Dependency Preference of Subordinate Clauses

First, in the modeling of the evidence, we consider every possible correlation (i.e., dependency) of the features of the subordinate clauses listed in section 3.3.. Furthermore, since it is necessary to consider the features for both of the given two subordinate clauses, we consider all the possible combination of features of the anterior and posterior head vp chunks of the given two subordinate clauses. Second, in the modeling of the decision, we distinguish the two cases of dependency relations described in section 3.1.. We name the first case as the decision "modify", while the second as the decision "beyond".

Figure 4 illustrates an example of transforming subordinate clauses into feature expression, and then obtaining training pairs of an evidence and a decision from a bracketed sentence. Figure 4 (a) shows an example sentence which contains two subordinate clauses $Clause_1$ and $Clause_2$, with chunking, bracketing, and dependency relations of chunks. Both of the head vp chunks Seg_1 and Seg_2 of $Clause_1$ and $Clause_2$ modify the sentence-final vp chunk. As shown in Figure 4 (b), the head vp chunks Seg_1 and Seg_2 have feature sets $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Then, every possible subsets F_1 and F_2 of $\mathcal{F}_1$ and $\mathcal{F}_2$ are considered,[8] respectively, and training pairs of an evidence and a decision are collected as in Figure 4 (c). In this case, the value of the decision D is "beyond", because Seg_1 modifies the sentence-final vp chunk, which follows Seg_2.

4. Analyzing Dependencies of Subordinate Clauses in a Sentence

This section describes how to analyze dependencies of subordinate clauses in a sentence based on the probabilities of the dependencies between two subordinate clauses. First, we estimate the probability $P(D = x \mid (Seg_i, Seg_j))$ of the decision $D = x$ given a pair of vp chunks (Seg_i, Seg_j) in a sentence as the maximum of the probabilities $P(D = x \mid (F_i, F_j))$ for every possible pair of feature subsets (F_i, F_j) and denote it as $\hat{P}(D = x \mid (Seg_i, Seg_j))$. Then, the *preference value* $Q(D = \text{"modify"} \mid (Seg_i, Seg_k))$ of the dependency of Seg_i's modifying Seg_k is calculated as follows:

1. In the cases where Seg_k is not sentence-final: $Q(D = \text{"modify"} \mid (Seg_i, Seg_k))$ is calculated as the geometric mean of the probability of Seg_i's modifying Seg_k and those of Seg_i's modification being beyond Seg_j $(j = i+1, \ldots, k-1)$.[9]

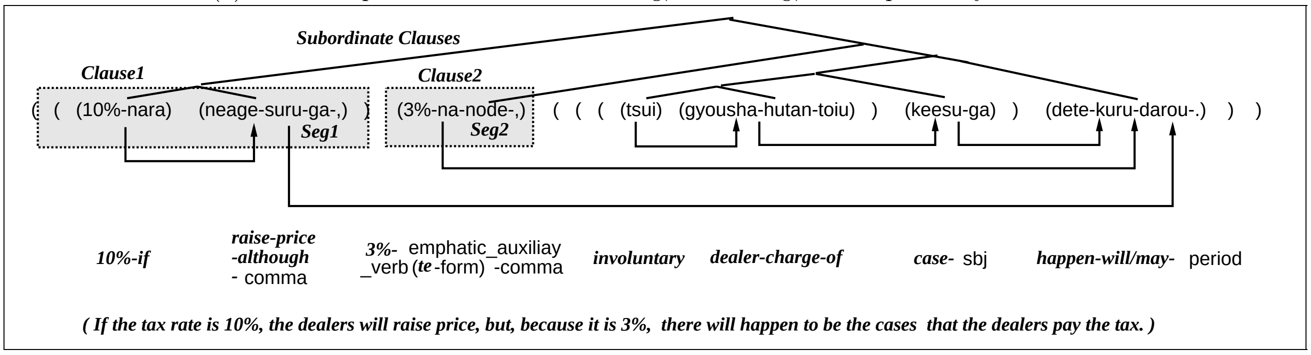
$$Q(D = \text{"modify"} \mid (Seg_i, Seg_k)) = \Big(\hat{P}(D = \text{"modify"} \mid (Seg_i, Seg_k)) \times \prod_{j=i+1}^{k-1} \hat{P}(D = \text{"beyond"} \mid (Seg_i, Seg_j))\Big)^{\frac{1}{k-i}} \quad (1)$$

[7]Terms of parts-of-speech tags and conjugation forms are borrowed from those of the Japanese morphological analysis system Chasen (Matsumoto et al., 1997).

[8]Since the feature 'predicate-conjunctive-particle(chunk-final)' subsumes 'predicate-conjunctive-particle(chunk-final)-"*ga*"', they are not considered together as one evidence.

[9]We calculate the preference value by the geometric mean rather than the product in order to make a fair comparison among the cases of different number of intermediate chunks Seg_js.

(a) An Example Sentence with Chunking, Bracketing, and Dependency Relations


Subordinate Clauses
Clause1
Clause2
(((10%-nara) (neage-suru-ga-,)) (3%-na-node-,) ((((tsui) (gyousha-hutan-toiu)) (keesu-ga)) (dete-kuru-darou-.)))
Seg1
Seg2
10%-if
raise-price -although - comma
3%- emphatic_auxiliay _verb (te-form) -comma
involuntary
dealer-charge-of
case- sbj
happen-will/may- period
(If the tax rate is 10%, the dealers will raise price, but, because it is 3%, there will happen to be the cases that the dealers pay the tax.)

(b) Feature Expression of Head VP Chunk of Subordinate Clauses

Head VP Chunk of Subordinate Clause	Feature Set
Seg_1 : "neage-suru-ga-,"	$\mathcal{F}_1 = \{$ with-comma, predicate-conjunctive-particle(chunk-final), predicate-conjunctive-particle(chunk-final)-"*ga*" $\}$
Seg_2 : "3%-na-node-,"	$\mathcal{F}_2 = \{$ with-comma, chunk-final-conjugative-word-*te*-form $\}$

(c) Evidence-Decision Pairs for Decision List Learning

Evidence E ($E=1$) (feature names are abbreviated)		
F_1	F_2	Decision D
with-comma	with-comma	"beyond"
with-comma	*te*-form	"beyond"
with-comma	with-comma, *te*-form	"beyond"
pred-conj-particle(final)	with-comma	"beyond"
...	...	...
with-comma, pred-conj-particle(final)	with-comma	"beyond"
...	...	...
pred-conj-particle(final)-"*ga*"	with-comma	"beyond"
...	...	...
with-comma, pred-conj-particle(final)-"*ga*"	with-comma	"beyond"
...	...	...

Figure 4: An Example of Evidence-Decision Pair of Japanese Subordinate Clauses

2. In the cases where Seg_k is sentence-final: we can assume $\hat{P}(D = \text{"beyond"} \mid (Seg_i, Seg_k)) = 0$ and $\hat{P}(D = \text{"modify"} \mid (Seg_i, Seg_k)) = 1$. Then, $Q(D = \text{"modify"} \mid (Seg_i, Seg_k))$ is calculated as the geometric mean of the probabilities of Seg_i's modification being beyond Seg_j ($j = i+1, \ldots, k-1$).

$$Q(D = \text{"modify"} \mid (Seg_i, Seg_k)) = \left(\prod_{j=i+1}^{k-1} \hat{P}(D = \text{"beyond"} \mid (Seg_i, Seg_j)) \right)^{\frac{1}{k-i-1}} \quad (2)$$

Then, suppose that $Dep(S_{sb})$ denote the dependencies among the sequence S_{sb} of n vp chunks in a sentence, their preference value $Q(S_{sb}, Dep(S_{sb}))$ is calculated as the product of the preference value of each dependency:

$$Q(S_{sb}, Dep(S_{sb})) = \prod_{i=1}^{n-2} Q(D = \text{"modify"} \mid (Seg_i, mod(Seg_i)))$$

Then, the dependency which gives the highest preference value is selected as the estimation $\hat{Dep}(S_{sb})$ of the dependencies among the sequence S_{sb} of the head vp chunks of subordinate clauses.

5. Experiments and Evaluation

We divided the 210,000 sentences of the whole EDR bracketed Japanese corpus into 95% training sentences and 5% test sentences. Then, we extracted 162,443 pairs of subordinate clauses from the 199,500 training sentences, and learned a decision list for dependency preference of subordinate clauses from those pairs. The default decision in the decision list is $D =$"beyond", where the marginal probability $P(D = \text{"beyond"}) = 0.5378$, i.e., the baseline precision of deciding dependency between two subordinate clauses is 53.78 %. We limit the frequency of each evidence-decision pair to be more than 9. The total number of obtained evidence-decision pairs is 7,812. We evaluate the learned decision list through the following experiments.

5.1. Comparison of Decision List Learning and Decision Tree Learning

First, we compare the performance of the learned decision list applied to deciding dependency between two subordinate clauses with that of the decision tree learning approach (Quinlan, 1993) to feature selection of dependency analysis (Haruno et al., 1998) (Model (a) in Table 2). In the decision tree learning, following the modeling in (Haruno et al., 1998), we design the features of Japanese subordinate clauses as the four feature types for each of the two subordinate clauses in Table 4 (eight features in total), where the numbers of values for those features are 2 for 'Punctuation', 17

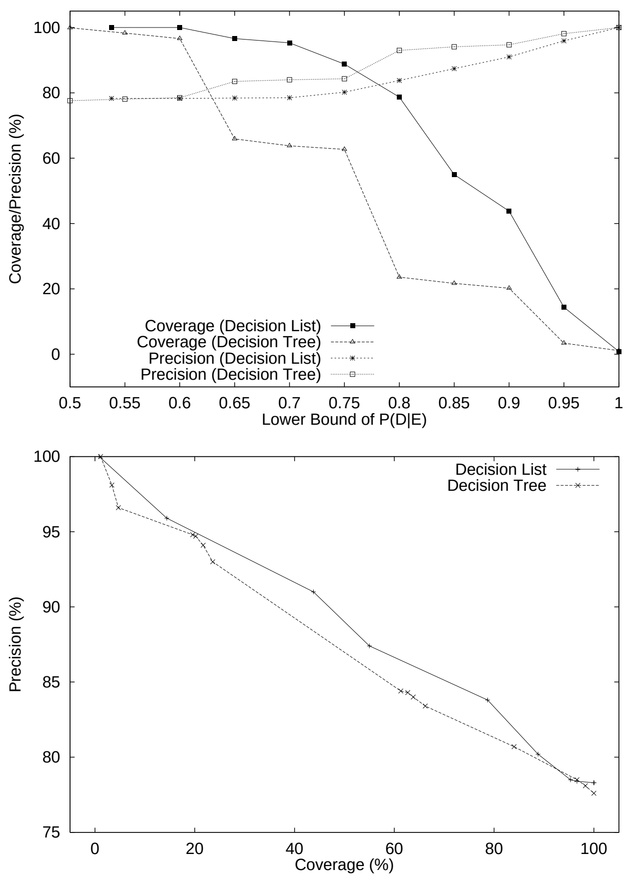


Figure 5: Performance Comparison of Decision List/Tree Learning: Deciding Dependency between Two Subordinate Clauses

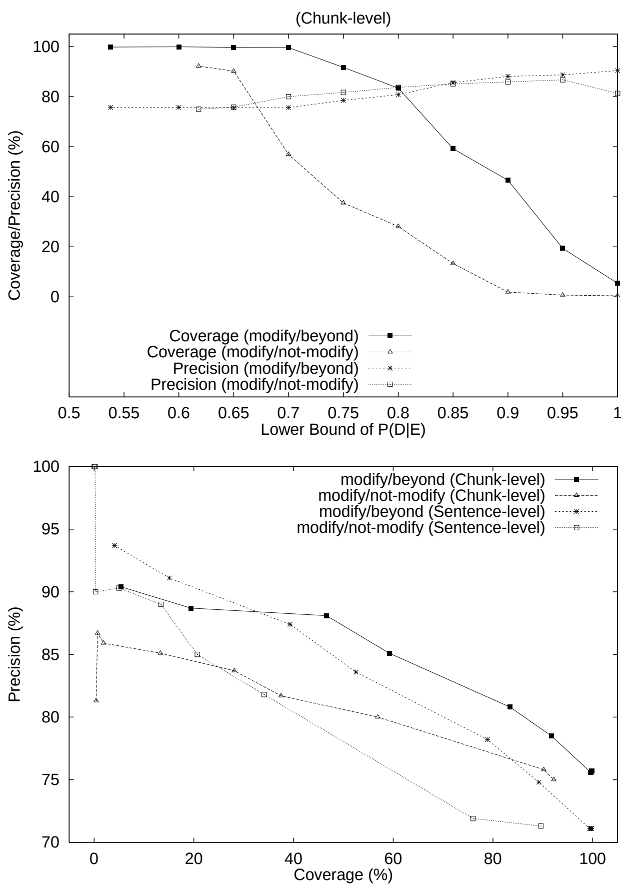


Figure 6: Performance Comparison of "modify"/"beyond" and "modify"/"not-modify" Models: Dependency Analysis among VP Chunks in a Sentence

for 'Grammatical', 12 for 'Conjugation form', and 235 for 'Lexical'. The modeling of the decision is the same as that of the decision list learning in section 3.. The frequency of the training instances at each leaf node of the decision tree is limited to be more than 9, and the decision tree is learned without post-pruning.

The results of performance comparison are shown in Figure 5. We change the threshold of the probability $P(D \mid E)$[10] in the decision list as well as in the leaf nodes of the decision tree and plot the trade-off between coverage and precision.[11] For both of the decision list learning and the decision tree learning, the precision varies from 78 to 100% according to the changes of the threshold of the probability $P(D \mid E)$. However, for the same threshold value of $P(D \mid E)$, the coverage of the decision tree learning is much lower than that of decision list learning, which results in the advantage of the decision list learning over the decision tree learning as shown in the lower part of Figure 5. This is mainly because the total number of nodes in the decision tree is 774 and thus about 10 times smaller than the number of rules in the decision list. This result clearly supports the claim that our modeling of decision list learning has an advantage over the decision tree learning, in that our modeling contributes to learning a fine-grained high coverage model which consists of both general and specific decision rules.

[10] $P(D \mid E)$ is used equivalently to the likelihood ratio.

[11] Coverage: the rate of the pairs of subordinate clauses whose dependencies are decided by the decision list, against the total pairs of subordinate clauses, Precision: the rate of the pairs of subordinate clauses whose dependencies are *correctly* decided by the decision list, against those *covered* pairs of subordinate clauses.

5.2. Comparison of "modify"/"beyond" and "modify"/"not-modify" Models

Next, we compare our model of distinguishing "modify"/"beyond" as the decisions with standard approaches to statistical dependency analysis (Collins, 1996; Fujio and Matsumoto, 1998; Haruno et al., 1998) which simply distinguish "modify"/"not-modify" as the decisions (Model (b) in Table 2). In the "modify"/"not-modify" model, dependency relations are classified into the two cases: "modify" and "not-modify". The procedure of learning dependency preference between two subordinate clauses is the same as that in section 3. except that we add the feature representing sentence-final vp chunks to the four features listed in Table 4. However, the procedure of analyzing dependencies among vp chunks in a sentence is different from that in section 4.. As the preference value $Q(D = \text{"modify"} \mid (Seg_i, Seg_k))$ of the dependency of Seg_i's modifying Seg_k, instead of the equations (1) and (2), the "modify"/"not-modify" model simply uses the probability of Seg_i's modifying Seg_k: $\hat{P}(D = \text{"modify"} \mid (Seg_i, Seg_k))$.

The results of comparing performance of depen-

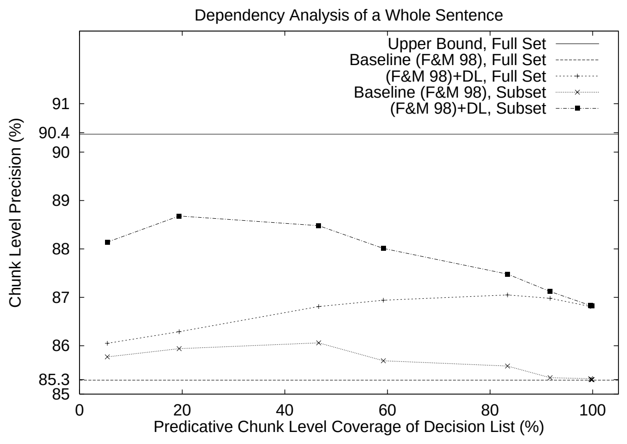


Figure 7: Performance Evaluation: Integrating Dependency Preference of Subordinate Clauses into Statistical Dependency Analysis of a Whole Sentence

dency analysis among vp chunks in a sentence are shown in Figure 6. The chunk level precision varies from 75.7 to 90.4% for the "modify"/"beyond" model, while that for the "modify"/"not-modify" model varies from 75.0 to 86.7%. In the 100% coverage case, the vp chunk level precision of the "modify"/"beyond" model is significantly better than that of (Fujio and Matsumoto, 1998) (75.7% versus 65.7%). For the same threshold value of $P(D \mid E)$, the coverage of the "modify"/"not-modify" model is much lower than that of the "modify"/"beyond" model, which results in the advantage of the "modify"/"beyond"model over the "modify"/"not-modify" model as shown in the lower part of Figure 6. This result clearly supports the claim that the "beyond" probability considered in the "modify"/"beyond"model is quite useful in analyzing dependencies among subordinate clauses.

5.3. Integrating Dependency Preference of Subordinate Clauses into Statistical Dependency Analysis of a Whole Sentence

Finally, we examine whether the estimated dependencies of subordinate clauses improve the precision of (Fujio and Matsumoto, 1998)'s statistical dependency analyzer.[12] First, we estimate the dependencies of subordinate clauses in a sentence by the procedure of section 4., then, regard them as correct dependencies in the statistical dependency analysis of a whole sentence in (Fujio and Matsumoto, 1998). We change the threshold of the probability $P(D \mid E)$ in the decision list and plot the trade-off between coverage and precision. Since the results of comparing chunk level and sentence level performance are similar, we only show chunk level performance in Figure 7. The upper bound as well as the baseline performance are also shown in Figure 7, where the upper bound performance is estimated by providing (Fujio and Matsumoto, 1998)'s statistical dependency analyzer with correct dependencies of subordinate clauses extracted from the bracketing of the EDR corpus, and the baseline performance is that of (Fujio and Matsumoto, 1998).

[12](Fujio and Matsumoto, 1998)'s lexicalized dependency analyzer is similar to that of (Collins, 1996), where various features were evaluated through performance test and an optimal feature set was manually selected.

Depending on the threshold of $P(D \mid E)$, we achieve 0.8∼1.8% improvement over the baseline in chunk level precision (the plot of '(F& M 98)+DL, Full Set'), and 1.6∼4.7% improvement over the baseline in sentence level. We also show the performance against a subset of sentences, where for each threshold of $P(D \mid E)$, a subset is constructed by collecting sentences for which all the vp chunks have dependency probabilities over the threshold. For this subset, depending on the threshold of $P(D \mid E)$, we achieve about 1.5∼2.5% improvement over the baseline in chunk level precision. This result clearly shows that the estimated dependencies of subordinate clauses is quite useful for improving the precision of (Fujio and Matsumoto, 1998)'s statistical dependency analyzer.

6. Conclusion

(Utsuro et al., 2000) proposed statistical method for learning dependency preference of Japanese subordinate clauses, in which scope embedding preference of subordinate clauses is exploited as a useful information source for disambiguating dependencies between subordinate clauses. Following (Utsuro et al., 2000), this paper presented detailed results of evaluating the proposed method by comparing it with several closely related existing techniques and showed that the proposed method outperformed those existing techniques.

7. References

Collins, M., 1996. A new statistical parser based on bigram lexical dependencies. In *Proc. 34th ACL*, pages 184–191.

EDR (Japan Electronic Dictionary Research Institute, Ltd.),1995. *EDR Electronic Dictionary Technical Guide*.

Fujio, M. and Y. Matsumoto, 1998. Japanese dependency structure analysis based on lexicalized statistics. In *Proc. 3rd EMNLP*, pages 88–96.

Haruno, M., et al., 1998. Using decision trees to construct a practical parser. In *Proc. 17th COLING and 36th ACL*, pages 505–511.

Matsumoto, Y., et al., 1997. Japanese morphological analyzer ChaSen 1.0 users manual. Information Science Technical Report NAIST-IS-TR97007, Nara Institute of Science and Technology. (in Japanese).

Minami, F., 1974. *Gendai Nihongo no Kouzou*. Taishuukan Shoten. (in Japanese).

Quinlan, J. R., 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Shirai, S., et al., 1995. A new dependency analysis method based on semantically embedded sentence structures and its performance on Japanese subordinate clauses. *Transactions of Information Processing Society of Japan*, 36(10):2353–2361. (in Japanese).

Utsuro, T., et al., 2000. Analyzing dependencies of Japanese subordinate clauses based on statistics of scope embedding preference. In *Proc. 1st NAACL*.

Yarowsky, D., 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proc. 32nd ACL*, pages 88–95.