Decision Lists for Determining Adjective Dependency in Japanese

Taiichi Hashimoto†, Kosuke Nishidate†, Kiyoaki Shirai‡,
Takenobu Tokunaga† and Hozumi Tanaka†

†Department of Computer Science Tokyo Institute of Technology
‡School of Information Science, Japan Advanced Institute of Science and Technology

†2-12-1 Ookayama, Meguro-ku, Tokyo 152, Japan
‡l-1 Asahidai, Tatsunokuchi, Ishikawa 923-1292, Japan

taiichi@cl.cs.titech.ac.jp, nishidy@cl.cs.titech.ac.jp, kshirai@jaist.ac.jp,
take@cl.cs.titech.ac.jp, tanaka@cl.cs.titech.ac.jp

Abstract

In Japanese constructions of the form [N_1 *no Adj* N_2], the adjective *Adj* modifies either N_1 or N_2. Determing the semantic dependencies of adjective in such phrase is an important task for machine translation. This paper describes a method for determining the adjective dependency in such constructions using decision lists, and inducing decision lists from training contexts with correct semantic dependencies and without. Based on evaluation, our method is able to determine adjective dependency with an precision of about 94%. We further analyze rules in the induced decision lists and examine effective features to determine the semantic dependencies of adjectives.

Keywords
Determining Adjective Dependency, Japanse, Decision List,
Lexical Preferences, Exploration of Effective Feature

1 Introduction

1.1 Background

In Japanese constructions of the form [N_1 no *Adj* N_2], *Adj* can modify either N_1 or N_2, where N_1 and N_2 are nouns, *Adj* is an adjective and "*no*" is an adnominal function word. The following are examples of this construction type:

a) *zo* (*elephant*) *no* *nagai* (*long*) *hana* (*trunk*)

"an elephant's long trunk"

b) *kami* (*hair*) *no* *nagai* (*long*) *shojo* (*girl*)

"a girl with long hair"

The adjective "*nagai* (*long*)" modifies the proceeding noun "*hana* (*trunk*)" in noun phrase a), while the adjective "*nagai* (*long*)" modifies the preceding noun "*kami* (*hair*)" in noun phrase b). In terms of syntactic dependency, both adjectives modify the proceeding noun, because all words syntactically modify the succeeding word in Japanese. In terms of semantic dependency, however, adjectives can modify the preceding noun as in b). Obviously, determining the semantic dependencies of adjective is an important task for machine translation. Notice that the syntactic composition of a) and b) are the same in Japanese, but different in English.

As the part-of-speech (POS hereafter) context of the two example noun phrases is the same, the determination of whether *Adj* semantically modifies N_1 or N_2 is strongly influenced by lexical items in the noun phrase. Therefore, lexical preferences are necessary to analyse semantic dependencies between adjectives and nouns.

1.2 Related Work

Tanaka and Ogino (1980) observed that the semantic dependency of the adjective is highly correlated with the semantic dependency between the two nouns in constructions of the form [N_1 no *Adj* N_2], and proposed a set of principles to determine the modifiee of the adjective ("Noun-Noun Dependency Principles") as follows:

In Japanese constructions of the form

$[N_1$ no Adj $N_2]$,

Principle 1
If $[N_1$ no $N_2]$ is felicitous,[1] then Adj modifies N_2.

Principle 2
If $[N_2$ no $N_1]$ is felicitous, then Adj modifies N_1.

Principle 3
If both $[N_1$ no $N_2]$ and $[N_2$ no $N_1]$ are felicitous, then Adj modifies the noun which it is most plausibly dependent on.

In noun phrase a), for example, "*nagai* (*long*)" modifies "*hana* (*trunk*)" according to Principle 1, because "*zo no hana (an elephant's trunk)*" is semantically felicitous. On the other hand, in noun phrases b), "*nagai* (*long*)" modifies "*kami* (*hair*)" according to Principle 2, because "*shoujo no kami (hair of a girl)*" is felicitous. Principle 1 and 2 only consider the semantic dependency between the nouns surrounding the adjective, not the adjective itself, in determining the modifiee of the adjective. One fascinating characteristic of the Noun-Noun Dependency Principles is that Principle 1 and 2 are preferred to Principle 3 which directly considers the semantic dependency between the adjective and each noun.

Hashimoto et al. (2000) proposed a statistical method for determining the modifiee of Adj according to the Noun-Noun Dependency Principles. First, they compare the relative occurrence of "N_1 *no* N_2" with that of "N_2 *no* N_1" in a training corpus. If the former is greater than the latter, they analyze "N_1 *no* N_2" to be more felicitous and Adj as modifying N_1 according to Principle 1. Otherwise they analyze Adj as modifying N_2 according to Principle 2. While empirically verifying the validity of Principles 1 and 2, they do not consider semantic dependency between the adjective and each noun (Principle 3).

Kikuchi and Itoh (1999) also proposed a method for determining semantic dependencies between words in constructions of the form $[N_1$ no Adj $N_2]$. They manually analyzed 2,029 example constructions, and derived seven rules for determining semantic dependencies. Their rules are based on the syntactic and semantic features of words appearing in the construction. They achieved an precision of 96.5% over 791 test phrases. The disadvantage of their method is that the number of adjectives covered by rules is highly restricted, because their rules are derived manually. In fact, their method relies on the semantic features of only 22 adjectives. There are considerable advantages to using automatic means to learn such rules to determine semantic dependencies.

[1] That is to say N_1 semantically modifies N_2.

1.3 Purpose

In this paper, we propose a method which automatically learns semantic dependencies between adjectives and nouns in Japanese constructions of the form $[N_1$ no Adj $N_2]$. We base our method on decision lists (Rivest, 1987). Decision list have been applied to various tasks in natural language processing, such as accent restoration (Yarowsky, 1994), word sense disambiguation (Yarowsky, 1995), detection of homophone errors (Shinnou, 1999), and a named entity task (Utsuro and Sassano, 2000).

Another goal of this paper is to explore features which are effective for determining the modifiee of adjectives. Especially, we are interested in examining the validity of lexical preferences between N_1 and N_2, supported by the Noun-Noun Dependency Principles. One of the reason why we choose to use a decision lists is that the induced decision list can be interpreted easily in examining effective features in this task.

2 Decision List Induction

In this section, we describe the method for inducing a decision list (Rivest, 1987) to determine the modifiee of the adjective in constructions of the form $[N_1$ no Adj $N_2]$. The algorithm described here is essentially the same as in (Yarowsky, 1995), except as detailed in Section 2.2.2.

2.1 Deriving Decision Rules

As training data, we extracted constructions of the form $[N_1$ no Adj $N_2]$ from the RWC POS-tagged corpus (Hasida et al., 1998) by way of simple heuristics, finding POS sequences of type "$N_1{}^+$ no Adj $N_2{}^+$". $N_i{}^+$ is a sequence of one or more nouns, which we regard as a single com-

pound noun. We manually annotated each extracted construction for *Adj* dependency.

Next, we derived six rule types from the training noun phrases, as detailed in Table 1.

Rule ($C \rightarrow d$)			Type
$Adj = x$	$\rightarrow$	N_1 or N_2	A
$N_1 = x$	$\rightarrow$	N_1 or N_2	N1
$N_2 = x$	$\rightarrow$	N_1 or N_2	N2
$Adj = x$ & $N_1 = y$	$\rightarrow$	N_1 or N_2	A+N1
$Adj = x$ & $N_2 = y$	$\rightarrow$	N_1 or N_2	A+N2
$N_1 = x$ & $N_2 = y$	$\rightarrow$	N_1 or N_2	N1+N2

Table 1: Rule Templates

In Table 1, "$C \rightarrow d$" means : determine the modifiee of *Adj* as d (N_1 or N_2) when [N_1 no *Adj* N_2] satisfies condition C.

For example, the following rules are derived from the noun phrase [$zo(elephant)$ no $nagai(long)$ $hana(trunk)$]:

- $Adj = nagai(long) \rightarrow N_2$
- $N_1 = zo(elephant) \rightarrow N_2$
- $N_2 = hana(trunk) \rightarrow N_2$
- $Adj = nagai(long)$ & $N_1 = zo(elephant) \rightarrow N_2$
- $Adj = nagai(long)$ & $N_2 = hana(trunk) \rightarrow N_2$
- $N_1 = zo(elephant)$ & $N_2 = hana(trunk) \rightarrow N_2$

Actually, N_1 or N_2 in [N_1 no *Adj* N_2] can be a sequence of nouns as described above. Treating a sequence of nouns as a single compound noun leads to data sparseness, because the number of rule instances increases dramatically. To overcome this problem, we extracted rules which refer to only N_1^h and N_2^h, where N_i^h is the semantic head of compound noun $N_i{}^+$. In general, the semantic head of a Japanese compound noun is the last noun of that noun sequence, such that we were able to identify the semantic head of each compound noun $\{N_1 \ldots N_k\}$ simply as the last noun N_k.

In addition to these rules, we also added the default rule (1), which is applicable to any input noun phrase.

$$true \rightarrow N_2 \quad (1)$$

Note that the default rule always analyzes the modifiee of *Adj* as N_2, because N_2 is preferred to N_1 as the modifiee of *Adj* in the training data described in Section 3.

2.2 Sorting Decision Rules

The next step in constructing the decision list is to sort the decision rules, i.e. to determine the rule order. We constructed two decision lists: one based on log-likelihood (Section 2.2.1), and the other on the Noun-Noun Dependency Principles (Section 2.2.2).

2.2.1 Decision list based on log-likelihood

We define the log-likelihood $L(r)$ of a rule $r(C \rightarrow d)$ as :

$$L(r) = \log \frac{P(d|C)}{P(\bar{d}|C)} \quad (2)$$

$P(d|C)$ is the probability that *Adj* modifies d when condition C is true, while $P(\bar{d}|C)$ is the probability that *Adj* doesn't modify d. $P(d|C)$ is estimated as :

$$P(d|C) = \frac{O(C,d) + \alpha}{O(C) + \alpha} \quad (3)$$

$O(C)$ is the frequency of occurrence of training noun phrases where condition C is true, while $O(C,d)$ is the frequency of occurrence of training noun phrases where condition C is true and *Adj* modifies d. α is a smoothing parameter, which we set α to 0.5. $P(\bar{d}|C)$ is estimated in the same way.

The rule r which is most strongly indicative of the modifiee of *Adj* will have the largest log-likelihood $L(r)$. Sorting by $L(r)$ will list the rules in order of reliablity.

After sorting the decision rules, we removed some rules from the list. First, we discarded rules for which the log-likelihood was smaller than that of the default rule (1). In other words, the last rule in the list is always the default rule. Next, we removed "redundant rules". Redundant rules are the rules which can never be applied. For example, consider the following rules, r_a and r_b:

r_a $Adj = a_1 \rightarrow d_1$

r_b $Adj = a_1$ & $N_1 = n_1 \rightarrow d_2$

If the log-likelihood of rule r_a is greater than that of r_b, rule r_a is always used earlier than

r_b for determining the modifiee of Adj, and r_b would never be used. As such, r_b is redundant and can be removed from the list.

2.2.2 Decision List based on the Noun-Noun Dependency Principles

According to the Noun-Noun Dependency Principles described in Section 1.2, we should first check whether [N_1 no N_2] and [N_2 no N_1] are felicitous or not (Principles 1 and 2), then check for semantic dependencies between Adj and N_1 and Adj and N_2 (Principle 3). In the framework of decision lists, Noun-Noun Dependency Principles correspond to rules of type N1+N2 being applied first, followed by rules of type A+N1 and A+N2. Simulating Noun-Noun Dependency Principles by a decision list, we sort the rules by their type in the following order:

$$N1+N2 > A+N2 > A+N1 > N1 > A > N2 \quad (4)$$

Rules of the same type are sorted in decreasing order of log-likelihood based on Equation (2). We removed rules for which the log-likelihood is lower than that of the default rule. Unlike decision lists sorted in order of log-likelihood, there is no redundancy in a decision list ordered based on (4), because rules associated with two words (types N1+N2, A+N2 and A+N1) are always ranked higher than rules associated with a single word (types N1, A and N2).

2.3 Bootstrap

The decision list induction method described above needs training data annotated with semantic dependencies between Adj and N_i, and as such constitutes supervised learning. However, the cost in manually annoting semantic dependencies is high. Utilizing unannotated training data enables us to extend the size of the training data. For this reason, we introduce the following bootstrapping algorithm:

1 For annotated training noun phrases with semantic dependencies T_c, and unannotated training noun phrases T_u.

2 Induce decision list dl_1 from T_c.

3 Determine the modifiee of Adj for each noun phrase in T_u using decision list dl_i.

4 Induce decision list dl_{i+1} from both T_c and T_u.

5 Repeat steps 3 and 4 until no classificational change is seen in T_u (using dl_i and dl_{i+1}).

3 Experiment

3.1 Inducing Decision Lists

We used the RWC POS-tagged corpus (Hasida et al., 1998) as our training corpus. This corpus originates from Mainichi Shimbun newspaper articles for the years 1991 to 1995. It comprises about 142 million words, and was POS annotated automatically. We extracted 31,541 training noun phrases of the form [N_1 no Adj N_2] from the RWC corpus. We used 3,634 noun phrases as T_c, 27,428 as T_u and 479 as test data. Correct semantic dependencies in T_c and the test data were manually assigned.

We induced two decision lists, dl-L and dl-T: dl-L is the log-likelihood-based decision list (see Section 2.2.1), while dl-T is the Noun-Noun Dependency Principles-based decision list (see Section 2.2.2). With the bootstrapping algorithm (see Section 2.3), the learning process converged after 5 iterations for dl-L, and 2 iterations for dl-T.

3.2 Results

The results of our experiment are given in Table 2. BL (baseline) indicates a naive strategy, where by N_2 is analyzed as modifying Adj for all test cases. The definitions of "Precision" and "Applicability" are as follows:

$$Precision \stackrel{def}{=} \frac{x}{y} \times 100 \quad (5)$$

$$Applicability \stackrel{def}{=} \frac{y}{z} \times 100 \quad (6)$$

x: # of noun phrases where the modifiee of the adjective was correctly identified.
y: # of noun phrases where the modifiee of the adjective was identified by decision rules (other than default rule).
z: total number of noun phrases in the test set.

The precisions of both dl-L and dl-T were about 94%, much higher than that of the baseline, and the applicabilities were almost 100%. On the other hand, the gain achieved by bootstrapping was small.

Comparing these results with previous work, the precision is higher than the 92% figure

Non Bootstrap	BL	dl-L	dl-T
Precision	52.6%	94.77%	94.35%
Applicability	100%	99.79%	99.79%

Bootstrap	BL	dl-L	dl-T
Precision	52.6%	94.78%	93.95%
Applicability	100%	100%	100%

Table 2: Experiment results

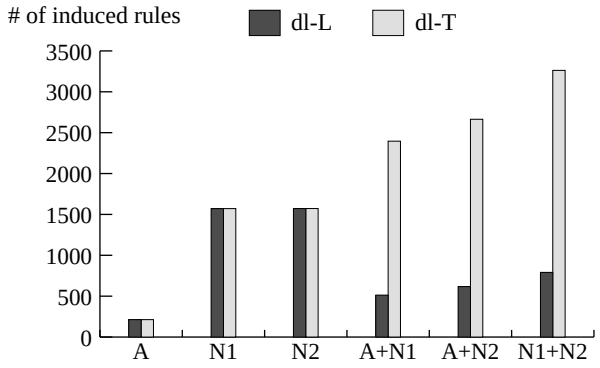

Figure 1: Number of induced rules

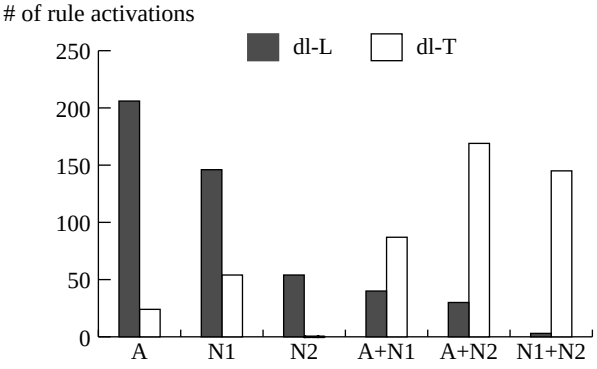

Figure 2: Number of rules activations for the test set

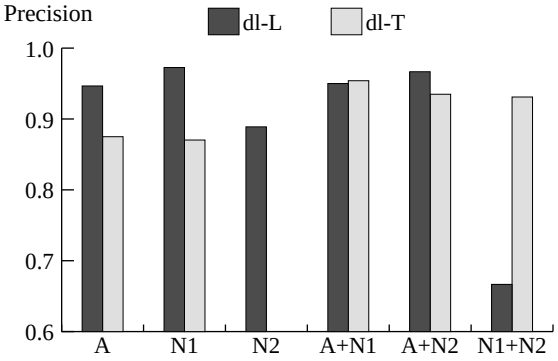

Figure 3: Precision of rule of each type

achieved by Hashimoto et al. (2000), but lower than the 97% figure cited by Kikuchi and Itoh (1999), while the applicability achieved with our method is much higher than that for these two methods. The applicability of Hashimoto et al.'s method was 61%, because they considered only N_1 and N_2 and ignored *Adj*. Kikuchi and Itoh don't indicate the applicability of their method, but the number of adjectives covered by their rules is highly restricted: they used semantic features of only 22 adjectives. In dl-L, on the other hand, 536 adjectives were observed in rules of type A, i.e. the semantic dependencies in noun phrases with these 536 adjectives can be determined by dl-L.

3.3 Effective Features

Next, we examined which types of decision rules are effective in determining the modifiee of adjectives. Figure 1 indicates the number of induced rules in the decision list associated with the 6 rule types. Figure 2 indicates the number of times the various rule types were activated in determining the modifiee of *Adj* with the test noun phrases. Figure 3 indicates the precision of each rule type.

In dl-L, a large propotion of induced rules were of types N1 and N2. The reason is that the log-likelihood of rules associated with two words (types A+N1, A+N2 and N1+N2) tends to be smaller than that of rules associated with a single word (types A, N1 and N2), and many instances of the lower rule type were thus removed from the list on the grounds of being redundant[2]. In fact, 81 of the top 100 rules in dl-L were associated with a single word. This is contrary to the prediction of the Noun-Noun Dependency Principles that rules of type N1+N2 would be most highly ranked. Furthermore, many rules of type A, N1 and N2 were activated during testing as shown in Figure 2. Despite rules of type A being fewest in number, they were activated with the greatest frequency in testing (dl-L). In addition to this, the mean

[2]The number of induced rules before removing redundant rules was the same as for dl-T.

precision of rules of type A is high at about 95%, as shown in Figure 3. It indicates that semantic dependency in constructions of the form [N_1 no *Adj* N_2] is largely determined by *Adj*.

The precision of dl-T, based on the Noun-Noun Dependency Principles, is almost equal to that of dl-L. Compared with dl-L, many rules of type N1+N2 were activated in the test phrase , performing at a relatively high precision of around 93%. Although rules of type N1+N2 ranked lower with dl-L, they were still effective in determining the semantic dependency of [N_1 no *Adj* N_2] constructons, underlining the validity of the Noun-Noun Dependency Principles.

4 Conclusion

In this paper, we proposed two methods for inducing decision lists to determine the semantic dependency of adjectives in Japanese construction of the form [N_1 no *Adj* N_2]. In the first method, the decision list was sorted in decreasing order of log-likelihood, in the second, it was based on Noun-Noun Dependency Principles. For both methods, the precision and applicability of the induced decision list were around 94% and 100%, respectively. We also examined the effectiveness of different rule types. In dl-L (method 1), rules of type N1 and N2 were induced with greatest frequency, and rules of type A were activated most frequently in testing. Furthermore, as suggested by the Noun-Noun Dependency Principles, rules of type N1+N2 were also found to be effective in determining the modifiee of the adjective

One reason why rules of type N1+N2 ranked lower in dl-L is that the number of nouns types is much greater than that of adjectives, leading to data sparseness. In the future, we intend to devise a method to induce a decision list using semantic classes in order to overcome this problem.

References

Koiti Hasida, Hitoshi Isahara, Takenobu Tokunaga, Minako Hashimoto, Shiho Ogino, Wakako Kashino, Jun Toyoura, and Hironobu Takahashi. 1998. The RWC Text Databases. In *Proceedings of the First International Conference on Language Resources and Evaluation*, pages 457–462.

Ronald L. Rivest. 1987. Learning decision lists. *Machine Learning*, 2:229–246.

Hiroyuki Shinnou. 1999. Detection of Japanese homophone errors by a decision list including a written word as a default evidence. In *Proceedings of 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 180–187.

Takehito Utsuro and Manabu Sassano. 2000. Named Entity Chunking Techniques and their Evaluation in Japanese Statistical Named Entity Recognition. *Information Processing Society of Japan SIGNL Notes*, 2000(86):9–16.

David Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of 32nd Annual Meeting of the Association for Computational Linguistics*, pages 88–95.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196.