Japanese Zero Pronoun Resolution based on Ranking Rules and Machine Learning

Hideki Isozaki and **Tsutomu Hirao**
NTT Communication Science Laboratories
Nippon Telegraph and Telephone Corporation
2-4 Hikaridai, Seika-cho, Souraku-gun, Kyoto, Japan, 619-0237
(isozaki,hirao)@cslab.kecl.ntt.co.jp

Abstract

Anaphora resolution is one of the most important research topics in Natural Language Processing. In English, overt pronouns such as *she* and definite noun phrases such as *the company* are anaphors that refer to preceding entities (*antecedents*). In Japanese, anaphors are often omitted, and these omissions are called *zero pronouns*. There are two major approaches to zero pronoun resolution: the heuristic approach and the machine learning approach. Since we have to take various factors into consideration, it is difficult to find a good combination of heuristic rules. Therefore, the machine learning approach is attractive, but it requires a large amount of training data. In this paper, we propose a method that combines ranking rules and machine learning. The ranking rules are simple and effective, while machine learning can take more factors into account. From the results of our experiments, this combination gives better performance than either of the two previous approaches.

1 Introduction

Anaphora resolution is an important research topic in Natural Language Processing. For instance, machine translation systems should identify antecedents of anaphors (such as *he* or *she*) in the source language to achieve better translation quality in the target language.

We are now studying open-domain question answering systems[1], and we expect QA systems to benefit from anaphora resolution. Typical QA systems try to answer a user's question by finding relevant phrases from large corpora. When a correct answer phrase is far from the keywords given in the question, the systems will not succeed in finding the answer. If the system can correctly resolve anaphors, it will find keywords or answers represented by anaphors, and the chances of finding the answer will increase. From this motivation, we are developing our system toward the ability to resolve anaphors in full-text newspaper articles.

In Japanese, anaphors are often omitted and these omissions are called *zero pronouns*. Since they do not give any hints (e.g., number or gender) about antecedents, automatic zero pronoun resolution is difficult. In this paper, we focus on resolving the zero pronoun, which is shortened for simplicity to 'zero.'

Most studies on Japanese zero pronoun resolution have not tried to resolve zeros in full-text newspaper articles. They have discussed simple sentenses (Kameyama, 1986; Walker et al., 1994; Yamura-Takei et al., 2002), dialogues (Yamamoto et al., 1997), stereotypical lead sentences of newspaper articles (Nakaiwa and Ikehara, 1993), intrasentential resolution (Nakaiwa and Ikehara, 1996; Ehara and Kim, 1996) or organization names in newspaper articles (Aone and Bennett, 1995).

There are two approaches to the problem: the heuristic approach and the machine learning ap-

[1] http://trec.nist.gov/data/qa.html

proach. The Centering Theory (Grosz et al., 1995) is important in the heuristic approach. Walker et al. (1994) proposed forward center ranking for Japanese. Kameyama (1986) emphasized the importance of a *property-sharing constraint*. Okumura and Tamura (1996) experimented on the roles of conjunctive postpositions in complex sentences.

However, these improvements are not sufficient for resolving zeros accurately. Murata and Nagao (1997) proposed complicated heuristic rules that take various features of antecedents and anaphors into account. We have to take even more factors into account, but it is difficult to maintain such heuristic rules. Therefore, recent studies employ machine learning approaches. However, it is also difficult to prepare a sufficient number of annotated corpora.

In this paper, we propose a method that combines these two approaches. Heuristic ranking rules give a general preference, while a machine learning method excludes inappropriate antecedent candidates. From the results of our experiments, the proposed method shows better performance than either of the two approaches alone.

Before giving a description of our methodology, we briefly introduce the grammar of the Japanese language here. A Japanese sentence is a sequence of *bunsetsus*: $b_1, \ldots, b_n$. A *bunsetsu* is a sequence of content words (e.g., nouns, adjectives, and verbs) followed by zero or more functional words (e.g., particles and auxiliary verbs): $b = w_1, \ldots, w_h, f_1, \ldots, f_k$. A bunsetsu modifies one of the following bunsetsus. A particle (*joshi*) marks the grammatical case of the noun phrase immediately before it. For example, *ga* is nominative (subject), *wo* is accusative (object), *ni* is dative (object2), and *wa* marks a topic.

Tomu ga /	*Bobu ni* /	*hon wo* /	*okutta.*
Tom=subj	Bob=object2	book=object	sent

(Tom sent a book to Bob.)

Bunsetsu dependency is represented by a list of bunsetsu pairs (modifier, modified). For instance, $\{(1,4),(2,3),(3,4),(4,-1)\}$ indicates that there are four bunsetsus in this sentence and that the first bunsetsu modifies the fourth bunsetsu and so on. The last bunsetsu modifies no bunsetsu, which is indicated by -1.

It takes a long time to construct high-quality annotated data, and we want to compare our results with conventional methods. Therefore, we obtained Seki's data (Seki et al., 2002a; Seki et al., 2002b), which are based on the Kyoto University Corpus[2] 2.0. These data are divided into two groups: *general* and *editorial*. *General* contains 30 general news articles, and *editorial* contains 30 editorial articles. According to his experiments, *editorial* is harder than *general*. Perhaps this is caused by the difference in rhetorical styles and the lengths of articles. The average number of sentences in an editorial article is 28.7, while that in a general article is 13.9.

However, we found problems in his data. For instance, the data contained ambiguous antecedents like *dou-shi* (the same person) or *dou-sha* (the same company) as correct antecedents. We replaced these 'correct answers' with their explicit names. We also removed zeros in quoted sentences because they are quite different from other sentences.

In addition, we decided to use the output of ChaSen 2.2.9[3] and CaboCha 0.34[4] instead of the morphological information and the dependency information provided by the Kyoto Corpus since classification of the *joshi* (particles) in the Corpus was not satisfactory for our purpose. Since CaboCha was trained by Kyoto Corpus 3.0, CaboCha's dependency output is very similar to that of the Corpus.

2 Methodology

In this paper, we combine heuristic ranking rules and machine learning. First, we describe how we extract possible antecedents (candidates). Second, we describe the rule-based ranking system and the machine learning system. Finally, we describe how to combine these two methods.

We consider only anaphors for noun phrases following Seki and other studies. We assume that zeros are already detected. We also assume zeros are located at the starting point of a bunsetsu that contains a *yougen* (a verb, an adjective, or an auxiliary verb). From now on, we use 'verb' instead of 'yougen' for readability. A *zero's bunsetsu* is a bunsetsu that contains the zero. We further assume that each zero's grammatical case is already determined by a zero detector and represented by corresponding particles.

[2] http://pine.kuee.kyoto-u.ac.jp/nl-resource/courpus-e.html
[3] http://chasen.aist-nara.ac.jp/
[4] http://cl.aist-nara.ac.jp/~taku-ku/software/cabocha/

If a zero is the subject of a verb, its case is represented by the particle *ga*. If it is an object, it is represented by *wo*. If it is an object2, it is represented by *ni*. We consider only these three cases. A *zero's particle* means such a particle.

Since complex sentences are hard to analyze, each sentence is automatically split at conjunctive postpositions (*setsuzoku joshi*) (Okumura and Tamura, 1996; Ehara and Kim, 1996). In order to distinguish the original complex sentence and the simpler sentences after the split, we call the former just a '*sentence*' and the latter '*post-split sentences*'. When a conjunctive postposition appears in a relative clause, we do not split the sentence at that position. In the examples below, we split the first sentence at 'and' but do not split the second sentence at 'and'.

She bought the book and sold it to him.
She bought the book that he wrote and sold.

A *zero's sentence* is the (original) sentence that contains the zero. From now on, z stands for a zero and c stands for a candidate of z's antecedent. z's particle is denoted **ZP**, and **CP** stands for c's next word that is c's particle or a punctuation symbol.

2.1 Enumeration of possible antecedents

Candidates (possible antecedents) are enumerated on the fly by using the following method.

1. We extract a content word sequence $w_1, \ldots, w_h$ as a candidate c if it is followed by a case marker (*kaku-joshi*, e.g., *ga*, *wo*), a topic marker (*wa* or *mo*), or a period.

2. If c's w_h is a verb, an adjective, an auxilary verb, an adverb, or a relative pronoun (ChaSen's *meishi-hijiritsu*, e.g., *koto* (*what* he did) and *toki* (*when* she married)), c is excluded. (If w_h is a closing quotation mark, w_{h-1} is checked instead.)

3. If c's w_h is a pronoun or an adverbial noun (a noun that can also be used as an adverb, i.e., ChaSen's *meishi-fukushi-kanou*), c is excluded.

4. If c is *dou-shi* (the person), it is replaced by the latest person name. If c is *dou-sha* (the company), it is replaced by the latest organization name. If c is *dou*+suffix, it is replaced by the latest candidate that has the same suffix.

   For this task, we use a named entity recognizer (Isozaki and Kazawa, 2002).

The first step extracts a content word sequence from a bunsetsu. The second step excludes verb phrases, adjective phrases, and clauses. As a result, we obtain only noun phrases. The third step excludes adverbial expressions like *kotoshi* (this year). The forth step resolves anaphors like definite noun phrases in English. We should also resolve pronouns, but we did not because useful pronouns are rare in newspaper articles.

In addition, we register a resolved zero as a new candidate. If z's antecedent is determined to be c_a, a new candidate c'_a is created for future zeros. c'_a is a copy of c_a except that c'_a's particle is **ZP** and c_a's location is z's location. In the training phase of the machine learning approach, we consider a correct answer as c_a. Then, we can remove far candidates from the list.

In this way, our zero resolver creates a 'general purpose' candidate list. However, some of the candidates are inappropriate for certain zeros. A verb usually does not have the same entity in two or more cases (Murata and Nagao, 1997). Therefore, our resolver excludes candidates that are filled in other cases of the verb. When a verb has two or more zeros, we resolve *ga* first, and its best candidate is excluded from the candidates of *wo* or *ni*.

2.2 Ranking rules

Various heuristics have been reported in past literature. Here, we use the following heuristics.

1. Forward center ranking (Walker et al., 1994): (topic $>$ empathy $>$ subject $>$ object2 $>$ object $>$ others).

2. Property-sharing (Kameyama, 1986): If a zero is the subject of a verb, its antecedent is perhaps a subject in the antecedent's sentence. If a zero is an object, its antecedent is perhaps an object.

3. Semantic constraints (Yamura-Takei et al., 2002; Yoshino, 2001): If a zero is the subject of 'eat,' its antecedent is probably a person or an animal, and so on. We use Nihongo Goi Taikei (Ikehara et al., 1997), which has 14,730 English-to-Japanese translation patterns

for 6,103 verbs, to check the acceptability of a candidate. Goi Taikei also has 300,000 words in about 3,000 semantic categories. (See Appendix A for details.)

4. Demotion of candidates in a relative clause (*rentai shuushoku setsu*): Usually, Japanese zeros do not refer to noun phrases in relative clauses (Ehara and Kim, 1996). (See Appendix B for details.)

Since sentences in newspaper articles are often complex and relative clauses are sometimes nested, we refine this rule in the following way.

- A *candidate's relative clause* is the inmost relative clause that contains the candidate.
- A relative clause *finishes* at the noun modified by the clause.
- If z appears before the finishing noun of c's relative clause, the clause is still *unfinished* at z. Otherwise, the clause is already *finished*.
- A quoted clause (with or without quotation marks " ") indicated by a quotation marker '*to*' ('that' in 'He said that she is ...') is also regarded as a relative clause.
- We demote c after c's relative clause finishes.

It is not clear how to combine the above heuristics consistently. Here, we sort the candidates in a lexicographical order based on the above features of candidates. For instance, we can use a lexicographically increasing order defined by $(\mathbf{Vi}, \mathbf{Re}, \mathbf{Ag}, \mathbf{Di}, \mathbf{Sa})$, where

- **Vi** (for violation) is 1 if the candidate violates the semantic constraint. Otherwise, **Vi** is 0.
- **Re** (for relative) is 1 if the candidate is in a relative clause that has already finished before z. Otherwise, **Re** is 0.
- **Ag** (for agreement) is 0 if **CP=ZP** holds. (Since most of *wa* and *mo* are subjects, they are regarded as *ga* here.) Otherwise, **Ag** is 1.
- **Di** (for distance) is a non-negative integer that represents the number of post-split sentences between c and z. If a candidate's **Di** is larger than *maxDi*, it is removed from the candidate list.
- **Sa** (for salience) is 0 if **CP** is *wa*. **Sa** is 1 if **CP** is *ga*. **Sa** is 2 if **CP** is *ni*. **Sa** is 3 if **CP** is *wo*. Otherwise, **Sa** is 4. We did not implement *empathy* because it makes the program more complex, and empathy verbs are rare in newspaper articles.

For instance, $(0, 0, 0, 2, 1) < (1, 0, 0, 0, 0)$ holds. The first ranked (lexicographically smallest) candidate is regarded as the best candidate. We employ lexicographical ordering because it seems the simplest way to rank candidates. We put **Vi** in the first place because **Vi** was often regarded as a constraint in the past literature. We put **Ag** before **Sa** because Kameyama's method was better than Walker's in Okumura and Tamura (1996). Therefore, $(\mathbf{Vi}, .., \mathbf{Ag}, .., \mathbf{Sa}, ..)$ is expected to be a good ordering. The above ordering is an instance of this.

2.3 Machine Learning

Although we can consider various other features for zero pronoun resolution, it is difficult to combine these features consistently. Therefore, we use machine learning. Support Vector Machines (SVMs) have shown good performance in various tasks in Natural Language Processing (Kudo and Matsumoto, 2001; Isozaki and Kazawa, 2002; Hirao et al., 2002).

Yoshino (2001) and Iida et al.(2003b) also applied SVM to Japanese zero pronoun resolution, but the usefulness of each feature was not clear. Here, we add features for complex sentences and analyze useful features by examining the weights of features. We use the following features of c as well as **CP**.

CSem c's semantic categories. (See Appendix A.)

CPPOS **CP**'s part-of-speech (POS) tags (rough and detailed).

CPOS The POS tags of the last word of c.

Siblings When **CP** is *wa* or *mo*, it is not clear whether c is a subject. However, a verb rarely has the same entity in two or more cases. Therefore, if c modifies a verb that has a subject, c is not a subject. In the next example, *hon* is an object of *katta*.

Ano / hon wa / Tomu ga / katta.
that book=topic Tom=subj bought
(As for that book, Tom bought it.)

In order to learn such things, we use sibling case-markers that modify the same verb as c's features.

We also use the following features of z as well as **ZP**.

Conjunct The latest conjunctive postposition in the sentence and its classification (Okumura and Tamura, 1996; Yoshimoto, 1986).

ZSem Semantic categories of the verb that z modifies. We use them only when the verb is *sahen meishi* + 'suru.' *Sahen meishi* is a kind of noun that can be an object of the verb '*suru*' (do) (e.g., 'shopping' in 'do the shopping').

We also use the following relations between z and c as well as **Ag**, **Vi**, and **Di**.

Relative Whether c is in a relative clause.

Unfinished Whether the relative clause is unfinished at z.

Intra (for intrasentential coreference) Whether c *explicitly* appears in z's sentence.

Sometimes it is difficult to distinguish cataphora from anaphora. Even if an antecedent appears in a preceding sentence, it is sometimes easier to find a candidate after z, as illustrated by the case of 'his' in the next English example.

Bob and John separately drove to Charlie's house. . . . Since his car broke down, John made a phone call.

Even if **Di** $= 0$ holds, Intra does not necessarily hold because we introduce resolved zeros as new candidaites.

Parallel Whether c appears in a clause parallel to a clause in which a zero appears. This will be useful for the resolution of a zero as with 'it' in the next English sentence.

He turned on the TV set and she turned it off.

Immediate Whether c's bunsetsu appears immediately before z's. In the following sentence, a candidate *ryoushin* is located immediately before the zero.

Kare no / ryoushin wa /
he+'s parents=topic
(z ga) ikiteiru to / shinjiteiru.
(z=subj) alive+that believe
(His parents believe that (z) is still alive.)

Here, we represent all of the above features by a boolean value: 0 or 1. Semantic categories can be represented by a 0/1 vector whose i-th component corresponds to the i-th semantic category. Similarly, POS tags can be represented by a 0/1 vector whose i-th component corresponds to the i-th POS tag. On the other hand, **Di** has a non-negative integer value. We also encode the distance by a 0/1 vector whose i-th component corresponds to the fact that the distance is i. The distance has an upper bound *maxDi*.

In this way, we can represent a candidate by a boolean feature vector. A candidate c_i's feature vector is denoted $\boldsymbol{x}_i$. If a boolean feature appears only once in the given data, we remove the feature from the feature vectors.

The training data comprise the set of pairs $\{(y_i, \boldsymbol{x}_i)\}$, where y_i is 1 if c_i is a correct antecedent of a zero. Otherwise, y_i is -1. By using the training data, SVM finds a decision function $g(\boldsymbol{x}) = \sum_i y_i \alpha_i K(\boldsymbol{x}, \boldsymbol{z}_i) + b$, where $\boldsymbol{x}$ is the feature vector of a candidate c and $\boldsymbol{z}_i$s are *support vectors* selected from the training data. α_i is a constant. $K(\cdot, \cdot)$ is called a kernel function. If $g(\boldsymbol{x}) > 0$ holds, $\boldsymbol{x}$ is classified as a correct antecedent.

2.4 Combinations

Here, we use the following method to combine the ordering and SVM.

1. Sort candidates by using the lexicographical order.
2. Classify each candidate by using SVM in this order.
3. If $g(\boldsymbol{x}_i)$ is positive, stop there and sort the evaluated candidates by $g(\boldsymbol{x}_i)$ in decreasing order.
4. If no candidate satisfies $g(\boldsymbol{x}_i) > 0$, return the best candidate in terms of $g(\boldsymbol{x}_i)$.

3 Results

We conducted leave-one(-article)-out experiments. For each article, 29 other articles were used for training. Table 1 compares the scores of the above methods. 'First' picks up the first candidate given by a given lexicographical ordering. The acronym 'vrads' stands for the lexicographical ordering of (**Vi**, **Re**, **Ag**, **Di**, **Sa**). 'Best' picks up the best candidate in terms of $g(\boldsymbol{x})$ without checking whether it

Table 1: Percentage of correctly resolved zeros

$\times$ = The combination is worse than 'first' or 'best.'
a = (Seki et al., 2002a), b = (Seki et al., 2002b)

	general				editorial			
	first	mem	svm1	**svm2**	first	mem	svm1	**svm2**
best		51.0	56.8	55.9		43.4	45.1	45.1
vrads	64.3	$53.0^{\times}$	$58.5^{\times}$	**66.3**	45.3	$44.0^{\times}$	45.9	47.3
vards	64.0	$53.0^{\times}$	$58.5^{\times}$	66.0	45.9	$44.2^{\times}$	45.9	46.9
rvads	63.4	$51.0^{\times}$	$58.5^{\times}$	**66.3**	44.4	$43.4^{\times}$	46.1	47.5
avrds	62.8	$53.0^{\times}$	$58.5^{\times}$	66.0	44.2	$44.0^{\times}$	45.9	46.9
vrdsa	55.9	$53.0^{\times}$	58.5	65.7	43.4	44.0	45.9	48.6
adsvr	53.0	$51.0^{\times}$	57.9	62.8	43.8	$43.4^{\times}$	46.3	48.6
davrs	39.5	53.0	57.6	62.5	34.6	44.2	46.1	**50.2**
Seki	54.0^{a} 50.7^{b}				39.8^{a}			

is positive. Consequently, it is independent of the ordering (unless two or more candidates have the best value). 'Svm1' uses the ordinary SVM (Vapnik, 1995) while 'svm2' uses a modified SVM for unbalanced data (Morik et al., 1999), which gives a large penalty to misclassification of a minority (= positive) example.[5] In general, svm2 accepts more cadidates than svm1. According to this table, svm1 is too severe to exclude only bad candidates. We also tried the maximum entropy model [6] (mem) and C4.5, but they were also too severe.

When we use SVM, we have to choose a good kernel for better performance. Here, we used the linear kernel ($K(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x} \cdot \boldsymbol{z}$) for SVM because it was best according to our preliminary experiments. We set *maxDi* at 3 because it gave the best results.

The table also shows Seki's scores for reference, but it is not fair to compare our scores with Seki's scores directly because our data is slightly different from Seki's. The number of zeros in *general* in our data is 347, while Seki resolved 355 detected zeros in (Seki et al., 2002a) and 404 in (Seki et al., 2002b). The number of zeros in our *editorial* is 514, while (Seki et al., 2002a) resolved 498 detected zeros. In order to overcome the data sparseness, Seki used unannotated articles to get co-occurrence statistics. Without the data, their scores degraded about 5 points. We have not conducted experiments that use unannotated corpora; this task is our future work.

As we expected, instances of (**Vi**, .., **Ag**, .., **Sa**, ..) show good performance. Without SVMs, 'vrads' is the best for *general* in the table. It is interesting that such a simple ordering gives better performance than SVMs. However, the combination of 'vrads' and 'svm2' (= vrads+svm2) gives even better results. In general, 'x+svm2' is better than 'first' and 'x+svm1.' With SVM, 'davrs+svm2' gave the best result for *editorial*. Editorial articles sometimes use anthropomorphism (e.g., The report says ...) that violates semantic constraints. Therefore, 'vrads' does not work well for such cases.

Table 2 shows the weights of the above features determined by svm2 for a fold of the leave-one-out experiment of 'vrads+svm2.' The weights can be given by rewriting $g(x[1], \ldots, x[d])$ as $w_0 + \sum_{j=1}^{d} w[j]x[j]$. This table shows that Kameyama's property-sharing (**Ag**), semantic violation (**Vi**), candidate's particle (**CP**), and distance (**Di**) are very important features. Our new features Parallel, Unfinished, and Intra also obtained relatively large weights. Semantic categories 'suggestions' and 'report' reflect the fact that some articles use anthropomorphism. These weights will be useful to design better heuristic rules. The fact that Unfinished's weight almost cancels Relative's weight justifies the

[5] An ordinary SVM minimizes $\|w\|^2/2 + C\sum_i \xi_i$ while the modified SVM minimizes $\|w\|^2/2 + C_+\sum_{i:y_i=1} \xi_i + C_-\sum_{i:y_i=-1} \xi_i$ where C_+/C_- = number of negative examples/number of positive examples.

[6] http://www2.crl.go.jp/jt/a132/members/mutiyama/software.html

Table 2: Weights of features

general	editorial
$+1.22$ **Ag**=0	$+0.76$ **Ag**=0
$+0.46$ **ZP**=*ni*	$+0.52$ Parallel
$+0.39$ concrete∈CSem	$+0.45$ **Di**=0
$+0.37$ **CP**=*ga*	$+0.38$ Intra
$+0.37$ Intra	$+0.36$ **CP**=*ga*
$+0.34$ agents∈CSem	$+0.33$ suggestion∈CSem
$+0.33$ **CP**=*wa*	$+0.33$ report∈CSem
$+0.23$ **Di**=0	$+0.32$ agents∈CSem
$+0.09$ Parallel	$+0.24$ concrete∈CSem
$+0.06$ Unfinished	$+0.24$ Unfinished
-0.07 Relative	$+0.24$ **CP**=*wa*
-0.21 **CP**=*mo*	-0.20 CPPOS='case marker'
-0.23 **CP**=*no*	-0.23 Relative
-0.32 **ZP**=*wo*	-0.40 **CP**=*no*
-0.37 **Di**=3	-0.49 **Di**=3
-0.69 **Vi**=1	-0.59 **Vi**=1

definition of **Re**.

4 Discussion

Yoshino (2001) used an ordinary SVM with $K(\boldsymbol{x}, \boldsymbol{y}) = (1 + \boldsymbol{x} \cdot \boldsymbol{y})^3$. He tried to find useful features by feature elimination. Since features are not completely independent, removing a heavily weighted feature does not necessarily degrade the system's performance. Hence, feature elimination is more reliable for reducing the number of features. However, feature elimination takes a long time. On the other hand, feature weights can give rough guidance. According to the table, our new features (Parallel, Unfinished, and Intra) obtained relatively large weights. This implies their importance. When we eliminated these three features, vrads+svm2's score for editorial dropped by 4 points. Therefore, combinations of these three features are useful.

Recently, Iida et al. (2003a) proposed an SVM-based tournament model that compares two candidates and selects the better one. We would like to compare or combine their method with our method. For further improvement, we have to make the morphological analyzer and the dependency analyzer more reliable because they make many mistakes when they process complex sentences.

SVM has often been criticized as being too slow. However, the above data were small enough for the state-of-the-art SVM programs. The number of examples in each set of training data was about 5,000–6,100, and each training phase took only 5–18 seconds on a 2.4-GHz Pentium 4 machine.

5 Conclusions

In order to make Japanese zero pronoun resolution more reliable, we have to maintain complicated heuristic rules or prepare a large amount of training data. In order to alleviate this problem, we combined simple lexicographical orderings and SVMs. It turned out that a simple lexicographical ordering performed better than SVM, but their combination gave even better performance. By examining feature weights, we found that features for complex sentences are important in zero pronoun resolution. We confirmed this by feature elimination.

References

Chinatsu Aone and Scott William Bennett. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proc. of ACL-1995*, pages 122–129.

Terumasa Ehara and Yeun-Bae Kim. 1996. Zero-subject resolution by probabilistic model (in Japanese). *Journal of Natural Language Processing*, 3(4):67–86.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

Tsutomu Hirao, Hideki Isozaki, Eisaku Maeda, and Yuji Matsumoto. 2002. Extracting important sentences with support vector machines. In *Proc. of COLING-2002*, pages 342–348.

Ryu Iida, Kentaro Inui, Hiroya Takamura, and Yuji Matsumoto. 2003a. Incorporating contextual cues in trainable models for coreference resolution (to appear). In *Proc. of EACL Workshop on the Computational Treatment of Anaphora*.

Ryu Iida, Kentaro Inui, Hiroya Takamura, and Yuji Matsumoto. 2003b. One method for resolving Japanese zero pronouns with machine learning model (in Japanese). In *IPSJ SIG-NL 154*.

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1997. *Goi-Taikei — A Japanese Lexicon (in Japanese)*. Iwanami Shoten.

Hideki Isozaki and Hideto Kazawa. 2002. Efficient support vector classifiers for named entity recognition. In *Proc. of COLING-2002*, pages 390–396.

Megumi Kameyama. 1986. A property-sharing constraint in centering. In *Proc. of ACL-1986*, pages 200–206.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *Proc. of NAACL-2001*, pages 192–199.

Katharina Morik, Peter Brockhausen, and Thorsten Joachims. 1999. Combining statistical learning with a knowledge-based approach — a case study in intensive care monitoring. In *Proc. of ICML-1999*, pages 268–277.

Masaki Murata and Makoto Nagao. 1997. An estimate of referents of pronouns in Japanese sentences using examples and surface expressions (in Japanese). *Journal of Natural Language Processing*, 4(1):41–56.

Hiromi Nakaiwa and Satoru Ikehara. 1993. Zero pronoun resolution in a Japanese to English machine translation system using verbal semantic attributes (in Japanese). *Transaction of the Information Processing Society of Japan*, 34(8):1705–1715.

Hiromi Nakaiwa and Satoru Ikehara. 1996. Intrasentential resolution of Japanese zero pronouns using pragmatic and semantic constraints (in Japanese). *Journal of Natural Language Processing*, 3(4):49–65.

Manabu Okumura and Kouji Tamura. 1996. Zero pronoun resolution based on centering theory. In *Proc. of COLING-1996*, pages 871–876.

Kazuhiro Seki, Atsushi Fujii, and Tetsuya Ishikawa. 2002a. Japanese zero pronoun resolution using a probabilistic model (in Japanese). *Journal of Natural Language Processing*, 9(3):63–85.

Kazuhiro Seki, Atsushi Fujii, and Tetsuya Ishikawa. 2002b. A probabilistic method for analyzing Japanese anaphora integrating zero pronoun detection and resolution. In *Proc. of COLING-2002*, pages 911–917.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.

Marilyn Walker, Masayo Iida, and Sharon Cote. 1994. Japanese discourse and the process of centering. *Computational Linguistics*, 20(2):193–233.

Kazuhide Yamamoto, Eiichiro Sumita, Osamu Furuse, and Hitoshi Iida. 1997. Ellipsis resolution in dialogues via decision-tree learning. In *Proc. of NLPRS-1997*, pages 423–428.

Mitsuko Yamura-Takei, Miho Fujiwara, Makoto Yoshie, and Teruaki Aizawa. 2002. Automatic linguistic analysis for language teachers: The case of zeros. In *Proc. of COLING-2002*, pages 1114–1120.

Kei Yoshimoto. 1986. Study of Japanese zero pronouns in discourse processing (in Japanese). In *IPSJ SIG notes, NL-56-4*, pages 1–8.

Keiichi Yoshino. 2001. Anaphora resolution of Japanese zero pronouns using machine learning (in Japanese). Master's thesis, Nara Institute of Science and Technology.

Appendix A: Semantic constraint check

One word may belong to two or more semantic categories, and each semantic category has superclasses (e.g., 'father' has the superclass 'parent'). Therefore, we keep all of these categories and their superclasses in a *category list* for the candidate. If the candidate is not registered in Goi Taikei and can be decomposed into shorter words, we use the semantic categories of the last candidate word because the last word is usually the head word.

Furthermore, we use named entity recognition. When the candidate contains a person name, an organization name, or a location name, a corresponding semantic category is added to the list.

A verb may have two or more translation patterns. Here, we use disjunction of the constraints. For instance, the verb '*yomu*' (to read) has three translation patterns. The first and second patterns' subjects are restricted to AGENT, and the third pattern's subject is restricted to PEOPLE. Therefore, the subject of *yomu* is accepted if and only if it is AGENT or PEOPLE.

Appendix B: Relative clause analysis

We have to be careful about parallel structures for this analysis. According to CaboCha, *Kare ga* in the next example modifies a verb *katte*, which modifies another verb *karita*. However, *katte* is contained in a clause that modifies the noun *hon*.

Kare ga /	*katte* /	*kanojo ga* /	*karita* /
he=subj	bought	she=subj	borrowed
hon wa /	*omoshiroi*.		
book=topic	interesting		

(The book that he bought and she borrowed is interesting.)

The particle *no* (= "'s" in English) directly modifies a noun. For instance, *Taro* in *Taro no hon* (Taro's book) is a book that Taro wrote or a book that Taro has. From this point of view, we also mark *A* in *A no B* (A's B) as a candidate in a relative clause.