Learning a Robust Word Sense Disambiguation Model using Hypernyms in Definition Sentences

Kiyoaki Shirai, Tsunekazu Yagi
School of Information Science, Japan Advanced Institute of Science and Technology
1-1, Asahidai, Tatsunokuchi, 923-1292, Ishikawa, Japan
{kshirai,t-yagi}@jaist.ac.jp

Abstract

This paper proposes a method to improve the robustness of a word sense disambiguation (WSD) system for Japanese. Two WSD classifiers are trained from a word sense-tagged corpus: one is a classifier obtained by supervised learning, the other is a classifier using hypernyms extracted from definition sentences in a dictionary. The former will be suitable for the disambiguation of high frequency words, while the latter is appropriate for low frequency words. A robust WSD system will be constructed by combining these two classifiers. In our experiments, the F-measure and applicability of our proposed method were 3.4% and 10% greater, respectively, compared with a single classifier obtained by supervised learning.

1 Introduction

Word sense disambiguation (WSD) is the process of selecting the appropriate meaning or sense for a given word in a document. Obviously, WSD is one of the fundamental and important processes needed for many natural language processing (NLP) applications. Over the past decade, many studies have been made on WSD of Japanese. Most current research uses machine learning techniques (Li and Takeuchi, 1997; Murata et al., 2001; Takamura et al., 2001), has achieved good performance. However, as supervised learning methods require word sense-tagged corpora, they often suffer from data sparseness, i.e., words which do not occur frequently in a training corpus can not be disambiguated. Therefore, we cannot use supervised learning algorithms alone in practical NLP applications, especially when it is necessary to disambiguate both high frequency and low frequency words.

To tackle this problem, this paper proposes a method to combine two WSD classifiers. One is a classifier obtained by supervised learning. The learning algorithm used for this classifier is the Support Vector Machine (SVM); this classifier will work well for the disambiguation of high frequency words. The second classifier is the Naive Bayes model, which will work well for the disambiguation of low frequency words. In this model, hypernyms extracted from definition sentences in a dictionary are considered in order to overcome data sparseness.

The details of the SVM classifier are described in Section 2, and the Naive Bayes model in Section 3. The combination of these two classifiers is described in Section 4. The experimental evaluation of the proposed method is reported in Section 5. We mention some related works in Section 6, and conclude the paper in Section 7.

2 SVM Classifier

The first classifier is the SVM classifier. Since SVM is a supervised learning algorithm, a word sense-tagged corpus is required as training data, and the classifier can not be used to disambiguate words which do not occur frequently in the data. However, as the effectiveness of SVM has been widely reported for a variety of NLP tasks including WSD (Murata et al., 2001; Takamura et al., 2001), we know that it will work well for disambiguation of high frequency words.

When training the SVM classifier, each training instance should be represented by a feature vector. We used the following features, which are typical for WSD.

- $S(0), S(-1), S(-2), S(+1), S(+2)$
  Surface forms of a target word and words just before or after a target word. A number in parentheses indicates the position of a word from a target word.
- $P(-1), P(-2), P(+1), P(+2)$
  Parts-of-speech (POSs) of words just before or after a target word.

- $S(-2)\cdot S(-1)$, $S(+1)\cdot S(+2)$, $S(-1)\cdot S(+1)$
  Pairs of surface forms of words surrounding a target word.
- $P(-2)\cdot P(-1)$, $P(+1)\cdot P(+2)$, $P(-1)\cdot P(+1)$
  Pairs of POSs of words surrounding a target word.
- B_{sent}
  Base forms of content words in a sentence[1].
- C_{sent}
  Semantic classes of content words in a sentence. Semantic classes used here are derived from the Japanese thesaurus "Nihongo-Goi-Taikei" (Ikehara et al., 1997).
- B_{head}, B_{mod}
  Base forms of the head (B_{head}) or modifiers (B_{mod}) of a target word.
- $(B_{case}; B_{noun})$
  A pair of the base forms of a case marker (B_{case}) and a case filler noun (B_{noun}) when the target word is a verb.
- $(B_{case}; C_{noun})$
  A pair of the base form of a case marker (B_{case}) and the semantic class of a case filler noun (C_{noun}) when the target word is a verb.
- $(B_{case}; B_{verb})$
  A pair of the base forms of a case marker (B_{case}) and a head verb (B_{verb}) when the target word is a case filler noun of a certain verb.

We used the LIBSVM package [2] for training the SVM classifier. The SVM model is $\nu-$SVM (Schölkopf, 2000) with a linear kernel, where the parameter $\nu = 0.0001$. The pairwise method is used to apply SVM to multi classification.

3 Naive Bayes Classifier using Hypernyms in Definition Sentences

In this section, we will describe the details of the WSD classifier using hypernyms of words extracted from definition sentences in a dictionary.

[1]We tried using the special symbol "NUM" as a feature for any numbers in a sentence, but the performance was slightly worse in our experiment. We thank the anonymous reviewer who gave us the comment about this.

[2]http://www.csie.ntu.edu.tw/%7Ecjlin/libsvm/

CID	Definition sentence
3c5631	こっけいな話をする 演芸(a comic story-telling entertainment)
1f66e3	とりとめもない 話(a rambling story)

Figure 1: Sense Set of "漫談"

3.1 Overview

Let us explain the basic idea of the model by considering the case in which the word "漫談" (*mandan*; comic chat) in the following example sentence (A) should be disambiguated:

(A) 坂野さんは 漫談 の世界に入り、...
(Mr. Sakano was initiated into the world of comic chat ...)

In this paper, word senses are defined according to the EDR concept dictionary (EDR, 1995). Figure 1 illustrates two meanings for "漫談" (comic chat) in the EDR concept dictionary. "CID" indicates a concept ID, an identification number of a sense.

One of the ways to disambiguate the senses of "漫談" (comic chat) is to train the WSD classifier from the sense-tagged corpus, as with the SVM classifier. However, when "漫談" (comic chat) occurs infrequently or not at all in the training corpus, we can not train any reliable classifiers.

To train the WSD classifier for low frequency words, we looked at hypernyms of senses in definition sentences. For Japanese, in most cases the last word in a definition sentence is a hypernym. For example, the hypernym of sense 3c5631 in Figure 1 is the last underlined word "演芸" (*engei*; entertainment), while the hypernym of 1f66e3 is "話" (*hanashi*; story).

In the EDR concept dictionary, there are senses whose hypernyms are also "演芸" (entertainment) or "話" (story). For example, as shown in Figure 2, 10d9a4, 3c3fbb and 3c5ab3 are senses whose hypernyms are "演芸" (entertainment), while the hypernym of 3cf737, 0f73c1 and 3c3071 is "話" (story). If these senses occur in the training corpus, we can train a classifier that determines whether the hypernym of "漫談" (comic chat) is "演芸" (entertainment) or "話" (story). If we can determine the correct hypernym, we can also determine which is the correct sense, 3c5631 or 1f66e3. Notice that we can train such a model even when "漫談" (comic

【落語】	10d9a4	こっけいな話を続け, 最後に落ちをつける寄席 演芸 (a monologue-style, comic story-telling entertainment always ending with a punch line)
【猿楽】	3c3fbb	猿楽という中世の民衆 演芸 (a type of medieval folk entertainment of Japan, called 'Sarugaku')
【紙切】	3c5ab3	紙を切り抜いていろいろの形を作る 演芸 (entertainment of cutting shapes out of paper)
【伝説】	3cf737	昔から民間に語り伝えられた 話 (a story passed down among people since ancient times)
【実話】	0f73c1	実際にあった本当の 話 (a true story)
【童話】	101156	子供向けに作られた 話 (a story for children)

Figure 2: Examples of Senses whose Hypernyms are "演芸" (entertainment) or "話" (story)

0efb60	競技/や/試合/の/回数/を/表す/語 (a word representing the number of competitions or contests)

Figure 3: Definition Sentence of the sense 0efb60 of the word "ゲーム"

chat) itself does not occur in the training corpus.

As described later, we train the probabilistic model that predicts a hypernym of a given word, instead of a word sense. Much more training data will be available to train the model predicting hypernyms rather than the model predicting senses, because there are fewer types of hypernyms than of senses. Figure 2 illustrates this fact clearly: all words labeled with 10d9a4, 3c3fbb and 3c5ab3 in the training data can be used as the data labeled with the hypernym "演芸" (entertainment). In this way, we can train a reliable WSD classifier for low frequency words. Furthermore, hypernyms will be automatically extracted from definition sentences, as described in Subsection 3.2, so that the model can be automatically trained without human intervention.

3.2 Extraction of Hypernyms

In this subsection, we will describe how to extract hypernyms from definition sentences in a dictionary. In principle, we assume that the hypernym of a sense is the last word of a definition sentence. For example, in the definition sentence of sense 3c5631 of "漫談" (comic chat), the last word "演芸" (entertainment) is the hypernym, as shown in Figure 1. However, we cannot always regard the last word as a hypernym. Let us consider the definition of sense 0eb70d of the word "ゲーム" (*gêmu*; game). In the EDR concept dictionary, the expression "A/を/表わす/語" (a word representing A) often appears in definition sentences. In this case, the hypernym of the sense is not the last word but A. Thus the hypernym of 0efb60 is not the last word "語" (*go*; word) but "回数" (*kaisuu*; number) in Figure 3.

When we extract a hypernym from a definition sentence, the definition sentence is first analyzed morphologically (word segmentation and POS tagging) by ChaSen [3]. Then a hypernym in a definition sentence is identified by pattern matching. An example of patterns used here is the rule extracting A when the expression "A/を/表わす/語" is found in a definition sentence. We made 64 similar patterns manually in order to extract hypernyms appropriately.

Out of the 194,303 senses of content words in the EDR concept dictionary, hypernyms were extracted for 191,742 senses (98.7%) by our pattern matching algorithm. Furthermore, we chose 100 hypernyms randomly and checked their validity, and found that 96% of the hypernyms were appropriate. Therefore, our method for extracting hypernyms worked well. The major reasons why acquisition of hypernyms failed were lack of patterns and faults in the morphological analysis of definition sentences.

3.3 Naive Bayes Model

We will describe the details of our probabilistic model that considers hypernyms in definition sentences. First of all, let us consider the following probability:

$$P(s, c|F) \quad (1)$$

In (1), s is a sense of a target word, c is a hypernym extracted from the definition sentence of s, and F is the set of features representing an input sentence including a target word.

[3] ChaSen is the Japanese morphological analyzer. http://chasen.aist-nara.ac.jp/hiki/ChaSen/

Next, we approximate Equation (1) as (2):

$$P(s, c|F) = P(s|c, F)P(c|F) \simeq P(s|c)P(c|F) \quad (2)$$

The first term, $P(s|c, F)$, is the probabilistic model that predicts a sense s given a feature set F (and c). It is similar to the ordinary Naive Bayes model for WSD (Pedersen, 2000). However, we assume that this model can not be trained for low frequency words due to a lack of training data. Therefore, we approximate $P(s|c, F)$ to $P(s|c)$.

Using Bayes' rule, Equation (2) can be computed as follows:

$$P(s|c)P(c|F) = \frac{P(s)P(c|s)}{P(c)} \frac{P(c)P(F|c)}{P(F)} \quad (3)$$

$$= \frac{P(s)P(F|c)}{P(F)} \quad (4)$$

Notice that $P(c|s)$ in (3) is equal to 1, because a hypernym c of a sense s is uniquely extracted by pattern matching (Subsection 3.2).

As all we want to do is to choose an s' which maximizes (4), $P(F)$ can be eliminated:

$$s' = \arg\max_s \frac{P(s)P(F|c)}{P(F)} \quad (5)$$

$$= \arg\max_s P(s)P(F|c) \quad (6)$$

Finally, by the Naive Bayes assumption, that is all features in F are conditionally independent, Equation (6) can be approximated as follows:

$$s' = \arg\max_s P(s) \prod_{f_i \in F} P(f_i|c) \quad (7)$$

In (7), $P(s)$ is the prior probability of a sense s which reflects statistics of the appearance of senses, while $P(f_i|c)$ is the posterior probability which reflects collocation statistics between an individual feature f_i and a hypernym c. The parameters of these probabilistic models can be estimated from the word sense-tagged corpus. We estimated $P(s)$ by Expected Likelihood Estimation and $P(f_i|c)$ by linear interpolation.

Feature Set

The features used in the Naive Bayes model are almost same as ones used in the SVM classifier except for the following features:

[Features **not** used in the Naive Bayes model]

- $S(-2), S(+2), P(-2), P(+2)$
- $S(-2)\cdot S(-1)$, $S(+1)\cdot S(+2)$, $S(-1)\cdot S(+1)$
- $P(-2)\cdot P(-1)$, $P(+1)\cdot P(+2)$, $P(-1)\cdot P(+1)$
- C_{sent}, $(B_{case}; C_{noun})$

According to the preliminary experiment, the accuracy of the Naive Bayes model slightly decreased when all features in the SVM classifier were used. This was the reason why we did not use the above features.

3.4 Discussion

The following discussion examines our method for extracting hypernyms from definition sentences.

Multiple Hypernyms

In general, two or more hypernyms can be extracted from a definition sentence, when the definition of a sense consists of several sentences or a definition sentence contains a coordinate structure. However, for this work we extracted only one hypernym for a sense, because definitions of all senses in the EDR concept dictionary are described by a single sentence, and most of them contain no coordinate structure.

In order to apply our model for multiple hypernyms, we must consider the probabilistic model $P(s, C|F)$ instead of Equation (1), where C is a set of hypernyms. Unfortunately, the estimation of $P(s, C|F)$ is not obvious, so investigation of this will be done in future.

Ambiguity of hypernyms

The fact that hypernyms may have several meanings does not appear to be a major problem, because most hypernyms in definition sentences of a certain dictionary have a single meaning according to our rough observation. So for this work we ignored the possible ambiguity of hypernyms.

Using other dictionaries

As described in Subsection 3.2, hypernyms are extracted by pattern matching. We would have to rebuild these patterns when we use other dictionaries, but we do not expect to require too much labor. Generally, in Japanese the last word in a definition sentence can be regarded as a hypernym. Furthermore, many extraction patterns for the EDR concept dictionary may also be applicable for other dictionaries. We are already building patterns to extract hypernyms from the other major Japanese dictionary, the *Iwanami Kokugo Jiten*, and developing the WSD system that will use them.

4 Combined Model

The details of two WSD classifiers are described in the previous two sections: one is the SVM classifier for high frequency words, and the other is the Naive Bayes classifier for low frequency words. These two classifiers are combined to construct the robust WSD system. We developed two kinds of combined models, described below in subsections 4.1 and 4.2.

4.1 Simple Ensemble

In this model, the process combining the two classifiers is quite simple. When only one of classifiers, SVM or Naive Bayes, outputs senses for a given word, the combined model outputs senses provided by that classifier. When both classifiers output senses, the ones provided by the SVM classifier are always chosen for the final output.

In the experiment in Section 5, SVM classifiers were trained for words which occur more than 20 times in the training corpus. Therefore, the simple ensemble described here is summarized as follows: we use the SVM classifier for high frequency words those which occur more than 20 times and the Naive Bayes classifier for the low frequency words.

4.2 Ensemble using Validation Data

First, we prepare validation data, which is a sense-tagged corpus, as common test data for the classifiers. The performance of the classifiers for a word w is evaluated by *correctness* C_w, defined by (8).

$$C_w = \frac{\begin{array}{c}\#\text{ of words in which one of the senses}\\ \text{selected by a classifier is correct}\end{array}}{\begin{array}{c}\#\text{ of words for which a classifier selects}\\ \text{one or more senses}\end{array}} \quad (8)$$

The main reason for combining two classifiers is to improve the recall and applicability of the WSD system. Note that a classifier which often outputs a correct sense would achieve high correctness C_w, even though it also outputs wrong senses. Thus, the higher the C_w of a classifier, the more it improves the recall of the combined model.

Next, the correctness C_w of each classifier for each word w is measured on the validation data. When two classifiers output senses for a given word, their C_w scores are compared. Then, the word senses provided by the better classifier are selected as the final outputs.

When the number of words in the validation data is small, comparison of the classifiers' C_w is unreliable. For that reason, when the number of words in the validation data is less that a certain threshold O_h, a sense output by the SVM classifier is chosen for the final output. This is because the correctness for all words in the validation data is higher for the SVM classifier than for the Naive Bayes classifier. In the experiment in Section 5, we set O_h to 10.

5 Experiment

In this section, we will describe the experiment to evaluate our proposed method. We used the EDR corpus (EDR, 1995) in the experiment. It is made up of about 200,000 Japanese sentences extracted from newspaper articles and magazines. In the EDR corpus, each word was annotated with a sense ID (CID). We used 20,000 sentences in the EDR corpus as the test data, 20,000 sentence as the validation data, and the remaining 161,332 sentences as the training data. The training data was used to train the SVM classifier and the Naive Bayes classifier, while the validation data was used for the combined model described in Subsection 4.2. The target instances used for evaluation were all ambiguous content words in the test data; the number of target instances was 91,986.

We evaluated three single WSD classifiers and two combined models:

- BL

  The baseline model. This is the WSD classifier which always selects the most frequently used sense. When there is more than one sense with equally high frequency, the classifier chooses all those senses.

- NB

  The Naive Bayes classifier (Section 3).

- SVM

  The SVM classifier (Section 2).

- SVM+NB(simple)

  The combined model by simple ensemble (Subsection 4.1).

- SVM+NB(valid)

  The combined model using the validation data (Subsection 4.2).

Table 1 reveals the precision(P), recall(R), F-measure(F) [4], applicability(A) and number of word types(T) of these five classifiers on the test

[4] $\frac{2PR}{P+R}$ where P and R represent the precision and recall, respectively.

Table 1: Results of WSD Classifiers

	R	P	F	A	T
1)	.6047	.6036	.6042	.9962	10,310
2)	.6274	.6543	.6406	.9568	10,501
3)	.6366	**.7080**	.6704	.8992	4,575
4)	.7016	.7010	.7013	**.9993**	**10,592**
5)	**.7050**	.7043	**.7046**	**.9993**	**10,592**

1)=BL, 2)=NB, 3)=SVM,
4)=SVM+NB(simple), 5)=SVM+NB(valid)

data. A(applicability) indicates the ratio of the number of instances disambiguated by a classifier to the total number of target instances; T indicates the number of word types which could be disambiguated by a classifier.

The two combined models outperformed the SVM classifier, for all criteria except precision. The gains in recall and applicability were especially remarkable. Notice the figures in column "T" in Table 1: the SVM classifiers could be applied only to 4,575 words, while the Naive Bayes classifiers were applicable to 10,501 words, including low frequency words. Thus, the ensemble of these two classifiers would significantly improve applicability and recall with little loss of precision.

Comparing the performance of the two combined models, "SVM+NB(validation)" slightly outperformed "SVM+NB(simple)", but there was no significant difference between them. The correctness, C_w, of the SVM classifier on the validation data was usually greater than that of the Naive Bayes classifier, so the SVM classifier was preferred when both were applicable. This was the almost same strategy for the simple ensemble, and we think this was the reason why the performance of two combined models were almost the same. In the rest of this section, we will show the results for the combined model using the validation data only.

Our goal was to improve the robustness of the WSD system. The naive way to construct a robust WSD system is to create an ensemble of a supervised learned classifier and a baseline classifier. So, we compared our proposed method (SVM+NB) with the combined model of the SVM and baseline classifier (SVM+BL). The results are shown in Table 2 and Figure 4. Table 2 shows the same criteria as in Table 1, indicating that "SVM+NB" outperformed "SVM+BL" for all criteria. Figure 4 shows the relation between the F-measure of the classifiers and word frequency in the training data. The horizontal axis

Table 2: Results of the Combined Models (1)

	R	P	F	A	T
5)	.7050	.7043	.7046	.9993	10,592
6)	.6977	.6976	.6977	.9962	10,310

5)=SVM+NB, 6)=SVM+BL

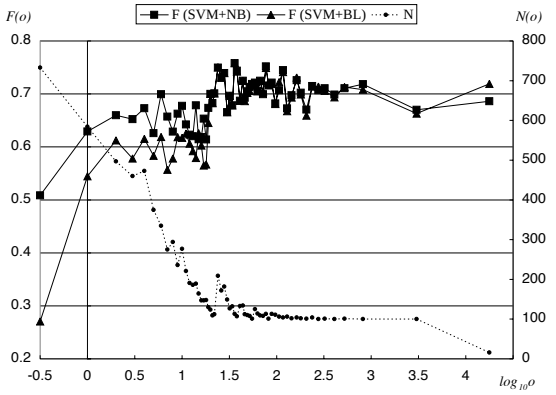

Figure 4: Results of the Combined Models (2)

indicates the occurrence of words in the training data (o) in log scale. Squares and triangles with lines indicates the $F(o)$ of the "SVM+NB" and "SVM+BL", respectively, where $F(o)$ is the macro average of F-measures for words which occur o times in the training data. The broken line indicates $N(o)$, the number of word types which occur o times [5]. For convenience, when $o = 0$, we plot $F(0)$ and $N(0)$ at $x = -0.5$ instead of $-\infty(= \log_{10} 0)$. As shown in Figure 4, "SVM+NB" significantly outperformed "SVM+BL" for low frequency words, and the number of word types ($N(o)$) became obviously greater when o was small. In other words, the Naive Bayes classifier proposed here could probably handle many more low frequency words than the baseline classifier. Therefore, it was more effective to combine the Naive Bayes classifier with the SVM classifier rather than the baseline classifier in order to improve the robustness of the overall WSD system.

Finally, we constructed a combined model of all three classifiers, the SVM, Naive Bayes and baseline classifiers. As shown in Table 3, this model slightly outperformed the two-classifier combined models shown in Table 2.

[5]To be more accurate, $F(o)$ and $N(o)$ are the figures for words which occurred more than or equal o times and less than $o+t$ times, where $o+t$ is the next point at the horizontal axis. t was chosen as the smallest integer so that $N(o)$ would be more than 100.

Table 3: Results of SVM+NB+BL

	R	P	F	A	T
7)	.7079	.7066	.7072	1	10,636

7)=SVM+NB+BL

6 Related Work

As described in Section 1, the goal of this project was to improve robustness of the WSD system. One of the promising ways to construct a robust WSD system is unsupervised learning as with the EM algorithm (Manning and Schütze, 1999), i.e. training a WSD classifier from an unlabeled data set. On the other hand, our approach is to use a machine readable dictionary in addition to a corpus as knowledge resources for WSD. Notice that we used hypernyms of definition sentences in a dictionary to train the Naive Bayes classifier, and this process worked well for words which did not occur frequently in the corpus. However, we did not compare our method and the unsupervised learning method empirically. This will be one of our future projects.

Using hypernyms of definition sentences is similar to using semantic classes derived from a thesaurus. One of the advantages of our method is that a thesaurus is not obligatory when word senses are defined according to a machine readable dictionary. Furthermore, our method is to train the probabilistic model that predicts a hypernym of a word, while most previous approaches use semantic classes as features (i.e., the condition of the posterior probability in the case of the Naive Bayes model). In facts, we also use features associated with semantic classes derived from the thesaurus, C_{sent} and $(B_{case}; C_{noun})$, as described in Section 2.

Several previous studies have used both a corpus and a machine readable dictionary for WSD (Litkowski, 2002; Rigau et al., 1997; Stevenson and Wilks, 2001). The difference between those methods and ours is the way we use information derived from the dictionary for WSD. Training the probabilistic model that predicts a hypernym in a dictionary is our own approach. However, these various methods are not in competition with our method. In fact, the robustness of the WSD system would be even more improved by combining these methods with that described in this paper.

7 Conclusion

This paper has proposed a method to develop a robust WSD system. We combined a WSD classifier obtained by supervised learning for high frequency words and a classifier using hypernyms in definition sentences in a dictionary for low frequency words. Experimental results showed that both recall and applicability were remarkably improved with our method. In future, we plan to investigate the optimum way to combine these two classifiers or to train a single probabilistic model using hypernyms in definition sentences, which is suitable for both high and low frequency words.

References

EDR. 1995. EDR electronic dictionary technical guide (second edition). Technical Report TR–045, Japan Electronic Dictionary Research Institute.

Satoshi Ikehara et al. 1997. *Nihongo Goi Taikei (in Japanese)*. Iwanami Shoten, Publishers.

Hand Li and Jun-ichi Takeuchi. 1997. Using evidence that is both strong and reliable in Japanese homograph disambiguation. In *SIG-NL, Information Processing Society of Japan*, pages 53–59.

Kenneth C. Litkowski. 2002. Sense information for disambiguation: Confluence of supervised and unsupervised methods. In *Proceedings of the SIGLEX/SENSEVAL Workshop on Word Sense Disambiguation*, pages 47–53.

Christopher D. Manning and Hinrich Schütze, 1999. *Foundations of Statistical Natural Language Processing*, chapter 7. MIT Press.

Masaki Murata, Masao Utiyama, Kiyotaka Uchimoto, Qing Ma, and Hitoshi Isahara. 2001. Japanese word sense disambiguation using the simple bayes and support vector machine methods. In *Proceedings of the SENSEVAL-2*, pages 135–138.

Ted Pedersen. 2000. A simple approach to building ensembles of naive baysian classifiers for word sense disambiguation. In *Proceedings of the NAACL*, pages 63–69.

German Rigau, Jordi Atserias, and Eneko Agirre. 1997. Combining unsupervised lexical knowledge methods for word sense disambiguation. In *Proceedings of the ACL*, pages 48–55.

Bernhard Schölkopf. 2000. New support vector algorithms. *Neural Computation*, 12:1083–1121.

Mark Stevenson and Yorick Wilks. 2001. The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics*, 27(3):321–349.

Hiroya Takamura, Hiroyasu Yamada, Taku Kudoh, Kaoru Yamamoto, and Yuji Matsumoto. 2001. Ensembling based on feature space restructuring with application to WSD. In *Proceedings of the NLPRS*, pages 41–48.