Proceedings of the 40th Annual Meeting of the Association for
Computational Linguistics (ACL), Philadelphia, July 2002, pp. 505-512.

An Empirical Study of Active Learning with Support Vector Machines for Japanese Word Segmentation

Manabu Sassano
Fujitsu Laboratories Ltd.
4-1-1, Kamikodanaka, Nakahara-ku,
Kawasaki 211-8588, Japan
sassano@jp.fujitsu.com

Abstract

We explore how active learning with Support Vector Machines works well for a non-trivial task in natural language processing. We use Japanese word segmentation as a test case. In particular, we discuss how the size of a pool affects the learning curve. It is found that in the early stage of training with a larger pool, more labeled examples are required to achieve a given level of accuracy than those with a smaller pool. In addition, we propose a novel technique to use a large number of unlabeled examples effectively by adding them gradually to a pool. The experimental results show that our technique requires less labeled examples than those with the technique in previous research. To achieve 97.0 % accuracy, the proposed technique needs 59.3 % of labeled examples that are required when using the previous technique and only 17.4 % of labeled examples with random sampling.

1 Introduction

Corpus-based supervised learning is now a standard approach to achieve high-performance in natural language processing. However, the weakness of supervised learning approach is to need an annotated corpus, the size of which is reasonably large. Even if we have a good supervised-learning method, we cannot get high-performance without an annotated corpus. The problem is that corpus annotation is labour intensive and very expensive. In order to overcome this, some unsupervised learning methods and minimally-supervised methods, e.g., (Yarowsky, 1995; Yarowsky and Wicentowski, 2000), have been proposed. However, such methods usually depend on tasks or domains and their performance often does not match one with a supervised learning method.

Another promising approach is *active learning*, in which a classifier selects examples to be labeled, and then requests a teacher to label them. It is very different from *passive learning*, in which a classifier gets labeled examples randomly. Active learning is a general framework and does not depend on tasks or domains. It is expected that active learning will reduce considerably manual annotation cost while keeping performance. However, few papers in the field of computational linguistics have focused on this approach (Dagan and Engelson, 1995; Thompson et al., 1999; Ngai and Yarowsky, 2000; Hwa, 2000; Banko and Brill, 2001). Although there are many active learning methods with various classifiers such as a probabilistic classifier (McCallum and Nigam, 1998), we focus on active learning with Support Vector Machines (SVMs) because of their performance.

The Support Vector Machine, which is introduced by Vapnik (1995), is a powerful new statistical learning method. Excellent performance is reported in hand-written character recognition, face detection, image classification, and so forth. SVMs have been recently applied to several natural language tasks, including text classification (Joachims, 1998; Dumais et al., 1998), chunking (Kudo and Matsumoto, 2000b; Kudo and Matsumoto, 2001), and dependency analysis (Kudo and Matsumoto, 2000a). SVMs have been greatly successful in such tasks.

Additionally, SVMs as well as boosting have good theoretical background.

The objective of our research is to develop an effective way to build a corpus and to create high-performance NL systems with minimal cost. As a first step, we focus on investigating how active learning with SVMs, which have demonstrated excellent performance, works for complex tasks in natural language processing. For text classification, it is found that this approach is effective (Tong and Koller, 2000; Schohn and Cohn, 2000). They used less than 10,000 binary features and less than 10,000 examples. However, it is not clear that the approach is readily applicable to tasks which have more than 100,000 features and more than 100,000 examples.

We use Japanese word segmentation as a test case. The task is suitable for our purpose because we have to handle combinations of more than 1,000 characters and a very large corpus (EDR, 1995) exists.

2 Support Vector Machines

In this section we give some theoretical definitions of SVMs. Assume that we are given the training data

$$(\boldsymbol{x}_i, y_i), \ldots, (\boldsymbol{x}_l, y_l), \boldsymbol{x}_i \in \boldsymbol{R}^n, y_i \in \{+1, -1\}$$

The decision function g in SVM framework is defined as:

$$g(\boldsymbol{x}) = \mathrm{sgn}(f(\boldsymbol{x})) \quad (1)$$

$$f(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b \quad (2)$$

where K is a kernel function, $b \in \boldsymbol{R}$ is a threshold, and α_i are weights. Besides the α_i satisfy the following constraints:

$$0 \leq \alpha_i \leq C, \forall i \text{ and } \sum_{i=1}^{l} \alpha_i y_i = 0,$$

where C is a missclassification cost. The $\boldsymbol{x}_i$ with non-zero α_i are called Support Vectors. For linear SVMs, the kernel function K is defined as:

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = \boldsymbol{x}_i \cdot \boldsymbol{x}.$$

In this case, Equation 2 can be written as:

$$f(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{x} + b \quad (3)$$

1. Build an initial classifier
2. While a teacher can label examples
   (a) Apply the current classifier to each unlabeled example
   (b) Find the m examples which are most *informative* for the classifier
   (c) Have the teacher label the subsample of m examples
   (d) Train a new classifier on all labeled examples

Figure 1: Algorithm of pool-based active learning

where $\boldsymbol{w} = \sum_{i=1}^{l} y_i \alpha_i \boldsymbol{x}_i$. To train an SVM is to find the α_i and the b by solving the following optimization problem:

$$\text{maximize} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C, \forall i \text{ and } \sum_{i=1}^{l} \alpha_i y_i = 0.$$

3 Active Learning for Support Vector Machines

3.1 General Framework of Active Learning

We use *pool-based active learning* (Lewis and Gale, 1994). SVMs are used here instead of probabilistic classifiers used by Lewis and Gale. Figure 1 shows an algorithm of pool-based active learning[1]. There can be various forms of the algorithm depending on what kind of example is found informative.

3.2 Previous Algorithm

Two groups have proposed an algorithm for SVMs active learning (Tong and Koller, 2000; Schohn and Cohn, 2000)[2]. Figure 2 shows the selection algorithm proposed by them. This corresponds to (a) and (b) in Figure 1.

[1] The figure described here is based on the algorithm by Lewis and Gale (1994) for their sequential sampling algorithm.

[2] Tong and Koller (2000) propose three selection algorithms. The method described here is simplest and computationally efficient.

1. Compute $f(\boldsymbol{x}_i)$ (Equation 2) over all $\boldsymbol{x}_i$ in a pool.
2. Sort $\boldsymbol{x}_i$ with $|f(\boldsymbol{x}_i)|$ in decreasing order.
3. Select top m examples.

Figure 2: Selection Algorithm

1. Build an initial classifier.
2. While a teacher can label examples
   (a) Select m examples using the algorithm in Figure 2.
   (b) Have the teacher label the subsample of m examples.
   (c) Train a new classifier on all labeled examples.
   (d) Add new unlabeled examples to the primary pool if a specified condition is true.

Figure 3: Outline of Tow Pool Algorithm

3.3 Two Pool Algorithm

We observed in our experiments that when using the algorithm in the previous section, in the early stage of training, a classifier with a larger pool requires more examples than that with a smaller pool does (to be described in Section 5). In order to overcome the weakness, we propose two new algorithms. We call them "Two Pool Algorithm" generically. It has two pools, i.e., a primary pool and a secondary one, and moves gradually unlabeled examples to the primary pool from the secondary instead of using a large pool from the start of training. The primary pool is used directly for selection of examples which are requested a teacher to label, whereas the secondary is not. The basic idea is simple. Since we cannot get good performance when using a large pool at the beginning of training, we enlarge gradually a pool of unlabeled examples.

The outline of Two Pool Algorithm is shown in Figure 3. We describe below two variations, which are different in the condition at (d) in Figure 3.

Our first variation, which is called Two Pool Algorithm A, adds new unlabeled examples to the primary pool when the increasing ratio of support vectors in the current classifier decreases, because the gain of accuracy is very little once the ratio is down. This phenomenon is observed in our experiments (Section 5). This observation has also been reported in previous studies (Schohn and Cohn, 2000).

In Two Pool Algorithm we add new unlabeled examples so that the total number of examples including both labeled examples in the training set and unlabeled examples in the primary pool is doubled. For example, suppose that the size of a initial primary pool is 1,000 examples. Before starting training, there are no labeled examples and 1,000 unlabeled examples. We add 1,000 new unlabeled examples to the primary pool when the increasing ratio of support vectors is down after t examples has been labeled. Then, there are the t labeled examples and the $(2,000 - t)$ unlabeled examples in the primary pool. At the next time when we add new unlabeled examples, the number of newly added examples is 2,000 and then the total number of both labeled in the training set and unlabeled examples in the primary pool is 4,000.

Our second variation, which is called Two Pool Algorithm B, adds new unlabeled examples to the primary pool when the number of support vectors of the current classifier exceeds a threshold d. The d is defined as:

$$d = N\frac{\delta}{100}, 0 < \delta \leq 100 \tag{4}$$

where δ is a parameter for deciding when unlabeled examples are added to the primary pool and N is the number of examples including both labeled examples in the training set and unlabeled ones in the primary pool. The δ must be less than the percentage of support vectors of a training set[3]. When deciding how many unlabeled examples should be added to the primary pool, we use the strategy as described in the paragraph above.

4 Japanese Word Segmentation

4.1 Word Segmentation as a Classification Task

Many tasks in natural language processing can be formulated as a classification task (van den Bosch

[3]Since typically the percentage of support vectors is small (e.g., less than 30 %), we choose around 10 % for δ. We need further studies to find the best value of δ before or during training.

et al., 1996). Japanese word segmentation can be viewed in the same way, too (Shinnou, 2000). Let a Japanese character sequence be $s = c_1 c_2 \cdots c_m$ and a boundary b_i exist between c_i and c_{i+1}. The b_i is either $+1$ (word boundary) or -1 (non-boundary). The word segmentation task can be defined as determining the class of the b_i. We use an SVM to determine it.

4.2 Features

We assume that each character c_i has two attributes. The first attribute is a character type (t_i). It can be hiragana[4], katakana, kanji (Chinese characters), numbers, English alphabets, kanji-numbers (numbers written in Chinese), or symbols. A character type gives some hints to segment a Japanese sentence to words. For example, kanji is mainly used to represent nouns or stems of verbs and adjectives. It is never used for particles, which are always written in hiragana. Therefore, it is more probable that a boundary exists between a kanji character and a hiragana character. Of course, there are quite a few exceptions to this heuristics. For example, some proper nouns are written in mixed hiragana, kanji and katakana.

The second attribute is a character code (k_i). The range of a character code is from 1 to 6,879. JIS X 0208, which is one of Japanese character set standards, enumerates 6,879 characters.

We use here four characters to decide a word boundary. A set of the attributes of c_{i-1}, c_i, c_{i+1}, and c_{i+2} is used to predict the label of the b_i. The set consists of twenty attributes: ten for the character type ($t_{i-1}t_it_{i+1}t_{i+2}$, $t_{i-1}t_it_{i+1}$, $t_{i-1}t_i$, t_{i-1}, $t_it_{i+1}t_{i+2}$, t_it_{i+1}, t_i, $t_{i+1}t_{i+2}$, t_{i+1}, t_{i+2}), and another ten for the character code ($k_{i-1}k_ik_{i+1}k_{i+2}$, $k_{i-1}k_ik_{i+1}$, $k_{i-1}k_i$, k_{i-1}, $k_ik_{i+1}k_{i+2}$, k_ik_{i+1}, k_i, $k_{i+1}k_{i+2}$, k_{i+1}, and k_{i+2}).

5 Experimental Results and Discussion

We used the EDR Japanese Corpus (EDR, 1995) for experiments. The corpus is assembled from various sources such as newspapers, magazines, and textbooks. It contains 208,000 sentences. We selected randomly 20,000 sentences for training and 10,000 sentences for testing. Then, we created examples using the feature encoding method in Section 4. Through these experiments we used the original SVM tools, the algorithm of which is based on SMO (Sequential Minimal Optimization) by Platt (1999). We used linear SVMs and set a missclassification cost C to 0.2.

First, we changed the number of labeled examples which were randomly selected. This is an experiment on *passive learning*. Table 2 shows the accuracy at different sizes of labeled examples.

Second, we changed the number of examples in a pool and ran the active learning algorithm in Section 3.2. We use the same examples for a pool as those used in the passive learning experiments. We selected 1,000 examples at each iteration of the active learning. Figure 4 shows the learning curve of this experiment and Figure 5 is a close-up of Figure 4. We see from Figure 4 that active learning works quite well and it significantly reduces labeled examples to be required. Let us see how many labeled examples are required to achieve 96.0 % accuracy. In active learning with the pool, the size of which is 2,500 sentences (97,349 examples), only 28,813 labeled examples are needed, whereas in passive learning, about 97,000 examples are required. That means over 70 % reduction is realized by active learning. In the case of 97 % accuracy, approximately the same percentage of reduction is realized when using the pool, the size of which is 20,000 sentences (776,586 examples).

Now let us see how the accuracy curve varies depending on the size of a pool. Surprisingly, the performance of a larger pool is worse than that of a smaller pool in the early stage of training[5]. One reason for this could be that support vectors in selected examples at each iteration from a larger pool make larger clusters than those selected from a smaller pool do. In other words, in the case of a larger pool, more examples selected at each iteration would be similar to each other. We computed variances[6] of each 1,000 selected examples at the learning iteration from 2 to 11 (Table 1). The variances of se-

[4] Hiragana and katakana are phonetic characters which represent Japanese syllables. Katakana is primarily used to write foreign words.

[5] Tong and Koller (2000) have got the similar results in a text classification task with two small pools: 500 and 1000. However, they have concluded that a larger pool is better than a smaller one because the final accuracy of the former is higher than that of the latter.

[6] The variance σ^2 of a set of selected examples $\boldsymbol{x}_i$ is defined

Table 1: Variances of Selected Examples

Iteration	2	3	4	5	6	7	8	9	10	11
1,250 Sent. Size Pool	16.87	17.25	17.85	17.63	17.24	17.37	17.34	17.73	17.94	17.57
20,000 Sent. Size Pool	16.66	17.03	16.92	16.75	16.80	16.72	16.91	16.93	16.87	16.97

lected examples using the 20,000 sentence size pool is always lower than those using the 1,250 sentence size pool. The result is not inconsistent with our hypothesis.

Before we discuss the results of Two Pool Algorithm, we show in Figure 6 how support vectors of a classifier increase and the accuracy changes when using the 2,500 sentence size pool. It is clear that after the accuracy improvement almost stops, the increment of the number of support vectors is down. We also observed the same phenomenon with different sizes of pools. We utilize this phenomenon in Algorithm A.

Next, we ran Two Pool Algorithm A[7]. The result is shown in Figure 7. The accuracy curve of Algorithm A is better than that of the previously proposed method at the number of labeled examples roughly up to 20,000. After that, however, the performance of Algorithm A does not clearly exceed that of the previous method.

The result of Algorithm B is shown in Figure 8. We have tried three values for δ : 5 %, 10 %, and 20 %. The performance with δ of 10 %, which is best, is plotted in Figure 8. As noted above, the improvement by Algorithm A is limited, whereas it is remarkable that the accuracy curve of Algorithm B is always the same or better than those of the previous algorithm with different sizes of pools (the detailed information about the performance is shown in Table 3). To achieve 97.0 % accuracy Algorithm B requires only 59,813 labeled examples, while passive learning requires about 343,000[8] labeled examples and the previous method with the 200,000 sentence size pool requires 100,813. That means 82.6 % and 40.7 % reduction compared to passive learning and the previous method with the 200,000 sentence size pool, respectively.

Table 2: Accuracy at Different Labeled Data Sizes with Random Sampling

# of Sentences	# of Examples	# of Binary Features	Accuracy (%)
21	813	5896	89.07
41	1525	10224	90.30
81	3189	18672	91.65
162	6167	32258	92.93
313	12218	56202	93.89
625	24488	98561	94.73
1250	48701	168478	95.46
2500	97349	288697	96.10
5000	194785	493942	96.66
10000	387345	827023	97.10
20000	776586	1376244	97.40

6 Conclusion

To our knowledge, this is the first paper that reports the empirical results of active learning with SVMs for a more complex task in natural language processing than a text classification task. The experimental results show that SVM active learning works well for Japanese word segmentation, which is one of such complex tasks, and the naive use of a large pool with the previous method of SVM active learning is less effective. In addition, we have proposed a novel technique to improve the learning curve when using a large number of unlabeled examples and have eval-

as:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} ||\boldsymbol{x}_i - \boldsymbol{m}||^2$$

where $\boldsymbol{m} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$ and n is the number of selected examples.

[7] In order to stabilize the algorithm, we use the following strategy at (d) in Figure 3: add new unlabeled examples to the primary pool when the current increment of support vectors is less than half of the average increment.

[8] We computed this by simple interpolation.

Table 3: Accuracy of Different Active Learning Algorithms

# of Ex.	Algo. A	Algo. B	Pool Size 1250 Sent.	5,000 Sent.	20,000 Sent.
813	89.07	89.07	89.07	89.07	89.07
1813	91.70	91.70	91.48	90.89	90.61
3813	93.82	93.82	93.60	93.11	92.42
6813	94.62	94.93	94.90	94.23	93.53
12813	95.24	95.87	95.29	95.42	94.82
24813	95.98	96.43	95.46	96.20	95.80
48813	96.51	96.88		96.51	96.62

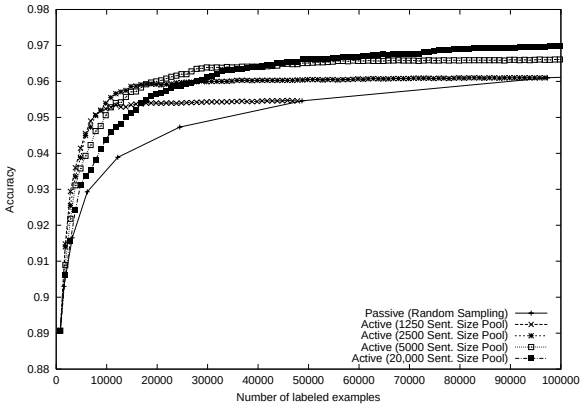
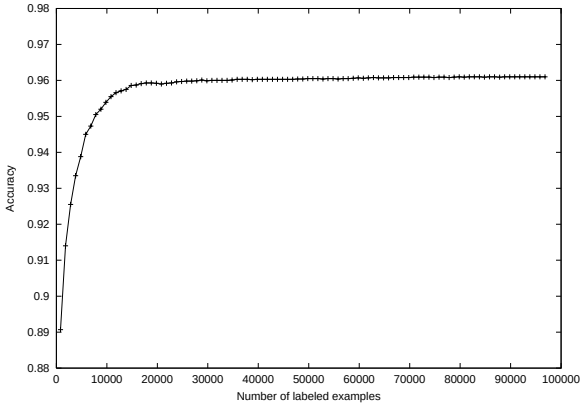


Figure 4: Accuracy Curve with Different Pool Sizes



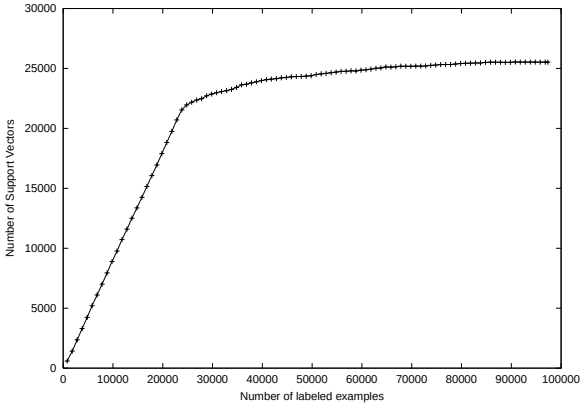


Figure 6: Change of Accuracy and Number of Support Vectors of Active Learning with 2500 Sentence Size Pool

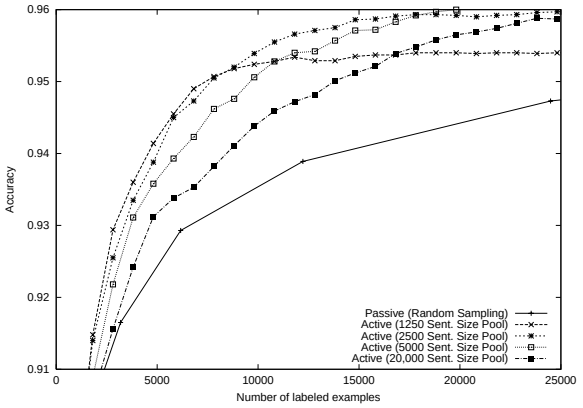


Figure 5: Accuracy Curve with Different Pool Sizes (close-up)

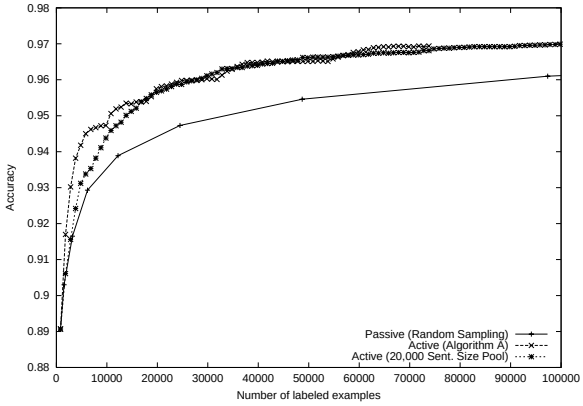


Figure 7: Accuracy Curve of Algorithm A

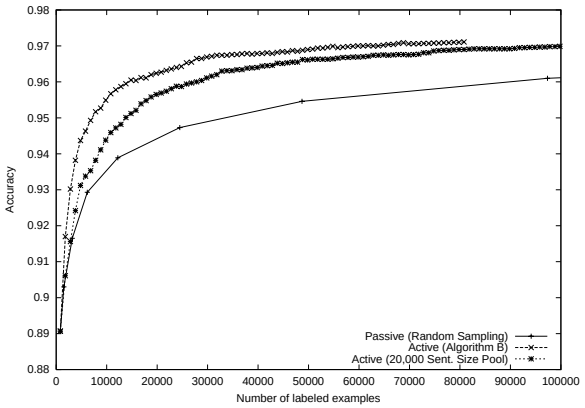


Figure 8: Accuracy Curve of Algorithm B

uated it by Japanese word segmentation. Our technique outperforms the method in previous research and can significantly reduce required labeled examples to achieve a given level of accuracy.

References

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of ACL-2001*, pages 26–33.

Ido Dagan and Sean P. Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *Proceedings of the Tweleveth International Conference on Machine Learning*, pages 150–157.

Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. 1998. Inductive learning algorithms and representations for text categorization. In *Proceedings of the ACM CIKM International Conference on Information and Knowledge Management*, pages 148–155.

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

Rebecca Hwa. 2000. Sample selection for statitical grammar induction. In *Proceedings of EMNLP/VLC 2000*, pages 45–52.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*.

Taku Kudo and Yuji Matsumoto. 2000a. Japanese dependency structure analysis based on support vector machines. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 18–25.

Taku Kudo and Yuji Matsumoto. 2000b. Use of support vector learning for chunk identification. In *Proceedings of the 4th Conference on CoNLL-2000 and LLL-2000*, pages 142–144.

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

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Rettrieval*, pages 3–12.

Andrew Kachites McCallum and Kamal Nigam. 1998. Employing EM and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 359–367.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proceedings of ACL-2000*, pages 117–216.

John C. Platt. 1999. Fast training of support vector machines using sequential minimal optimization. In Bernhard Schölkopf, Christopher J.C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 185–208. MIT Press.

Greg Schohn and David Cohn. 2000. Less is more: Active learning with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning*.

Hiroyuki Shinnou. 2000. Deterministic Japanese word segmentation by decision list method. In *Proceedings of the Sixth Pacific Rim International Conference on Artificial Intelligence*, page 822.

Cynthia A. Thompson, Mary Leaine Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 406–414.

Simon Tong and Daphne Koller. 2000. Support vector machine active learning with applications to text classification. In *Proceedings of the Seventeenth International Conference on Machine Learning*.

Antal van den Bosch, Walter Daelemans, and Ton Weijters. 1996. Morphological analysis as classification: an inductive-learning approach. In *Proceedings of the Second International Conference on New Methods in Natural Language Processing*, pages 79–89.

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

David Yarowsky and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of ACL-2000*, pages 207–216.

David Yarowsky. 1995. Unsupervised word sence disambiguation rivaling supvervised methods. In *Proceedings of ACL-1995*, pages 189–196.