Extracting Important Sentences with Support Vector Machines

Tsutomu HIRAO and **Hideki ISOZAKI** and **Eisaku MAEDA**
NTT Communication Science Laboratories
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237 Japan
{hirao,isozaki,maeda}@cslab.kecl.ntt.co.jp

Yuji MATSUMOTO
Graduate School of Information and Science, Nara Institute of Science and Technology
8516-9, Takayama, Ikoma, Nara 630-0101 Japan
matsu@is.aist-nara.ac.jp

Abstract

Extracting sentences that contain important information from a document is a form of text summarization. The technique is the key to the automatic generation of summaries similar to those written by humans. To achieve such extraction, it is important to be able to integrate heterogeneous pieces of information. One approach, parameter tuning by machine learning, has been attracting a lot of attention. This paper proposes a method of sentence extraction based on Support Vector Machines (SVMs). To confirm the method's performance, we conduct experiments that compare our method to three existing methods. Results on the Text Summarization Challenge (TSC) corpus show that our method offers the highest accuracy. Moreover, we clarify the different features effective for extracting different document genres.

1 Introduction

Extracting important sentences means extracting from a document only those sentences that have important information. Since some sentences are lost, the result may lack coherence, but important sentence extraction is one of the basic technologies for generating summaries that are useful for humans to browse. Therefore, this technique plays an important role in automatic text summarization.

Many researchers have been studied important sentence extraction since the late 1950's (Luhn, 1958). Conventional methods focus on sentence features and define significance scores. The features include key words, sentence position, and certain linguistic clues. Edmundson (1969) and Nobata et al. (2001) have proposed scoring functions to integrate heterogeneous features. However, we can not tune the parameter values by hand when the number of features is large.

When a large quantity of training data is available, tuning can be effectively realized by machine learning. In recent years, machine learning has attracted attention in the field of automatic text summarization. Aone et al. (1998) and Kupiec et al. (1995) employed Bayesian classifiers, Mani et al. (1998), Nomoto et al. (1997), Lin (1999), and Okumura et al. (1999) used decision tree learning. However, most machine learning methods overfit the training data when many features are given. Therefore, we need to select features carefully.

Support Vector Machines (SVMs) (Vapnik, 1995) is robust even when the number of features is large. Therefore, SVMs have shown good performance for text categorization (Joachims, 1998), chunking (Kudo and Matsumoto, 2001), and dependency structure analysis (Kudo and Matsumoto, 2000).

In this paper, we present an important sentence extraction technique based on SVMs. We verified the technique against the Text Summarization Challenge (TSC) (Fukushima and Okumura, 2001) corpus.

2 Important Sentence Extraction based on Support Vector Machines

2.1 Support Vector Machines (SVMs)

SVM is a supervised learning algorithm for 2-class problems.

Training data is given by

$$(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_u, y_u), \quad \mathbf{x}_j \in \mathbf{R}^n,\ y_j \in \{+1, -1\}.$$

Here, $\mathbf{x}_j$ is a feature vector of the j-th sample; y_j is its class label, positive($+1$) or negative(-1). SVM separates positive and negative examples by a hyperplane defined by

$$\mathbf{w} \cdot \mathbf{x} + b = 0,\ \mathbf{w} \in \mathbf{R}^n, b \in \mathbf{R}, \tag{1}$$

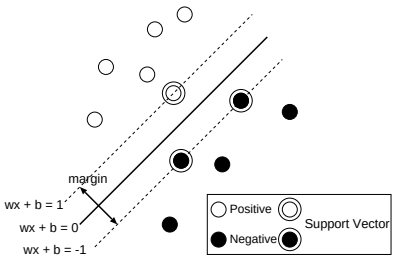

Figure 1: Support Vector Machines.

where "·" represents the inner product.

In general, such a hyperplane is not unique. Figure 1 shows a linearly separable case. The SVM determines the optimal hyperplane by maximizing the margin. A margin is the distance between negative examples and positive examples.

Since training data is not necessarily linearly separable, slack variables (ξ_j) are introduced for all $\mathbf{x}_j$. These ξ_j incur misclassification error, and should satisfy the following inequalities:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_j + b &\geq 1 - \xi_j \\ \mathbf{w} \cdot \mathbf{x}_j + b &\leq -1 + \xi_j. \end{aligned} \tag{2}$$

Under these constraints, the following objective function is to be minimized.

$$\frac{1}{2}||\mathbf{w}||^2 + C\sum_{j=1}^{u}\xi_j. \tag{3}$$

The first term in (3) corresponds to the size of the margin and the second term represents misclassification.

By solving a quadratic programming problem, the decision function $f(\mathbf{x}) = \mathrm{sgn}(g(\mathbf{x}))$ can be derived where

$$g(\mathbf{x}) = \left(\sum_{i=1}^{\ell}\lambda_i y_i \mathbf{x}_i \cdot \mathbf{x} + b\right). \tag{4}$$

The decision function depends on only support vectors ($\mathbf{x}_i$). Training examples, except for support vectors, have no influence on the decision function.

Non-linear decision surfaces can be realized by replacing the inner product of (4) with a kernel function $K(\mathbf{x} \cdot \mathbf{x}_i)$:

$$g(\mathbf{x}) = \left(\sum_{i=1}^{\ell}\lambda_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right). \tag{5}$$

In this paper, we use polynomial kernel functions that have been very effective when applied to other tasks, such as natural language processing (Joachims, 1998; Kudo and Matsumoto, 2001; Kudo and Matsumoto, 2000):

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d. \tag{6}$$

2.2 Sentence Ranking by using Support Vector Machines

Important sentence extraction can be regarded as a two-class problem: important or unimportant. However, the proportion of important sentences in training data will differ from that in the test data. The number of important sentences in a document is determined by a summarization rate that is given at run-time. A simple solution for this problem is to rank sentences in a document. We use $g(\mathbf{x})$ the distance from the hyperplane to $\mathbf{x}$ to rank the sentences.

2.3 Features

We define the boolean features discussed below that are associated with sentence S_i by taking past studies into account (Zechner, 1996; Nobata et al., 2001; Hirao et al., 2001; Nomoto and Matsumoto, 1997).

We use 410 boolean variables for each S_i. Where $\mathbf{x} = (x[1], \cdots, x[410])$. A real-valued feature normalized between 0 and 1 is represented by 10 boolean variables. Each variable corresponds to an internal $[i/10,(i+1)/10)$ where $i = 0$ to 9. For example, $Posd = 0.75$ is represented by "0000000100" because 0.75 belongs to [7/10,8/10).

Position of sentences

We define three feature functions for the position of S_i. First, *Lead* is a boolean that corresponds to the output of the lead-based method described below[1] . Second, *Posd* is S_i's position in a document. Third, *Posp* is S_i's position in a paragraph. The first sentence obtains the highest score, the last obtains the lowest score:

[1] When a sentence appears in the first N of document, we assign 1 to the sentence. An N was given for each document by TSC committee.

$$Posd(S_i) = 1 - BD(S_i)/|D(S_i)|$$
$$Posp(S_i) = 1 - BP(S_i)/|P(S_i)|.$$

Here, $|D(S_i)|$ is the number of characters in the document $D(S_i)$ that contains S_i; $BD(S_i)$ is the number of characters before S_i in $D(S_i)$; $|P(S_i)|$ is the number of characters of the paragraph $P(S_i)$ that contains S_i, and $BP(S_i)$ is the number of characters before S_i in the paragraph.

Length of sentences

We define a feature function that addresses the length of sentence as

$$Len(S_i) = |S_i| / \max_{S_z \in D(S_i)} |S_z|.$$

Here, $|S_i|$ is the number of characters of sentence S_i, and $\max_{S_z \in D} |S_z|$ is the maximum number of characters in a sentence that belongs to $D(S_i)$.

In addition, the length of a previous sentence $Len_{-1}(S_i) = Len(S_{i-1})$ and the length of a next sentence $Len_{+1}(S_i) = Len(S_{i+1})$ are also features of sentence S_i.

Weight of sentences

We defined the feature function that weights sentences based on frequency-based word weighting as

$$Wf(S_i) = \sum_t tf(t, S_i) \cdot w(t, D(S_i)).$$

Here, $Wf(S_i)$ is the summention of weighting $w(t, D(S_i))$ of words that appear in a sentence. $tf(t, S_i)$ is term frequency of t in S_i. We used only nouns. In addition, we define word weight $w(t, D(S_i))$ based on a *specific field* (Hara et al., 1997):

$$w(t, D(S_i)) = \alpha \left(\frac{1}{T} \sum_{z=1}^{T} \frac{\varepsilon_z}{V_z} \right) + \beta \left(\frac{tf(t, D(S_i))}{\sum_{t'} tf(t', D(S_i))} \right).$$

Here, T is the number of sentence in a document, and V_z is the number of words in sentence $S_z \in D(S_i)$ (repetitions are ignored). Also, ε_z is a boolean value: that is 1 when t appears $in S_z$.

The first term of the equation above is the weighting of a word in a specific field. The second term is the occurrence probability of word t.

We set parameters α and β as 0.8, 0.2, respectively. The weight of a previous sentence $Wf_{-1}(S_i) = Wf(S_{i-1})$, and the weight of a next sentence $Wf_{+1}(S_i) = Wf(S_{i+1})$ are also features of sentence S_i.

Density of key words

We define the feature function $Den(S_i)$ that represents density of key words in a sentence by using Hanning Window function ($f_H(k, m)$):

$$Den(S_i) = \max_m \sum_{k=m-Win/2}^{m+Win/2} f_H(k, m) \cdot a(k, S_i),$$

where $f_H(k, m)$ is given by

$$f_H(k, m) = \begin{cases} \frac{1}{2}\left(1 + \cos 2\pi \frac{k-m}{Win}\right) & (|k-m| \le Win/2) \\ 0 & (|k-m| > Win/2). \end{cases}$$

The key words (KW) are the top 30% of words in a document according to $w(t, D(S_i))$. Also, m is the center position of the window, $Win = |S_i|/2$. In addition, $a(k, S_i)$ is defined as follows:

$$a(k, S_i) = \begin{cases} w(t, D) & \text{Where a word } t\ (\in KW) \text{ begins at } k \\ 0 & k \text{ is not the beginning position of a word in } KW. \end{cases}$$

Named Entities

$x[r]$=1 ($1 \le r \le 8$) indicates that a certain Named Entity class appears in S_i. The number of Named Entity classes is 8 (Sekine and Eriguchi, 2000), e.g., PERSON, LOCATION. We use Isozaki's NE recognizer (Isozaki, 2001).

Conjunctions

$x[r]$=1 ($9 \le r \le 61$) if and only if a certain conjunction is used in the sentence. The number of conjunctions is 53.

Functional words

$x[r]$=1 ($62 \le r \le 234$) if and only if a certain functional word such as `ga`, `ha`, and `ta` is used in the sentence. The number of functional words is 173.

Part of speech

$x[r]$=1 ($235 \le r \le 300$) if and only if a certain part of speech such as "Noun-*jiritsu*" and "Verb-*jiritsu*" is used in the sentence. The number of part of speech is 66.

Semantical depth of nouns

$x[r]$=1 ($301 \leq r \leq 311$) if and only if S_i contains a noun at a certain semantical depth according to a Japanese lexicon, Goi-Taikei (Ikehara et al., 1997). The number of depth levels is 11. For instance, *Semdep=2* means that a noun in S_i belongs to the second depth level.

Document genre

$x[r]$=1 ($312 \leq r \leq 315$) if and only if the document belongs to a certain genre. The genre is explicitly written in the header of each document. The number of genres is four: General, National, Editorial, and Commentary.

Symbols

$x[r]$=1 (r=316) if and only if sentence includes a certain symbol (for example: $\bullet$,$\star$,$\diamond$).

Conversation

$x[r]$=1 (r=317) if and only if S_i includes a conversation style expression.

Assertive expressions

$x[r]$=1 (r=318) if and only if S_i includes an assertive expression.

3 Experimental settings

3.1 Corpus

We used the data set of TSC (Fukushima and Okumura, 2001) summarization collection for our evaluation. TSC was established as a subtask of NTCIR-2 (NII-NACSIS Test Collection for IR Systems). The corpus consists of 180 Japanese documents[2] from the Mainichi Newspapers of 1994, 1995, and 1998. In each document, important sentences were manually extracted at summarization rates of 10%, 30%, and 50%. Note that the summarization rates depend on the number of sentences in a document not the number of characters. Table 1 shows the statistics.

3.2 Evaluated methods

We compared four methods: decision tree learning, boosting, lead, and SVM. At each summarization rate, we trained classifiers and classified test documents.

Decision tree learning method

We used C4.5 (Quinlan, 1993) for our experiments with the default settings. We used the features described in section 2. Sentences were ranked according to their certainty factors given by C4.5.

Boosting method

We used C5.0, which applies boosting to decision tree learning. The number of rounds was set to 10. Sentences were ranked according to their certainty factors given by C5.0.

Lead-based method

The first N sentences of a document were selected. N was determined according to the summarization rates.

SVM method

This is our method as outlined in section 2. We used the second-order polynomial kernel, and set C (in equation (3)) as 0.0001. We used TinySVM[3] .

3.3 Measures for evaluation

In the TSC corpus, the number of sentences to be extracted was explicitly given by the TSC committee. When we extract sentences according to that number, Precision, Recall, and F-measure become the same value. We call this value Accuracy. Accuracy is defined as follows:

$$\text{Accuracy} = b/a \times 100,$$

where a is the specified number of important sentences, and b is the number of true important sentences that were contained in system's output.

4 Results

Table 2 shows the results of five-fold cross validation by using all 180 documents.

For all summarization rates and all genres, SVM achieved the highest accuracy, the lead-based method the lowest. Let the null hypothesis be "There are no differences among the scores of the four methods". We tested this null hypothesis at a significance level of 1% by using Tukey's method. Although the SVM's performance was best, the differences were not statistically significant at 10%. At 30% and 50%, SVM performed better than the other methods with a statistical significance.

[2] Each document is presented in SGML style with sentence and paragraph separators attached.

[3] http://cl.aist-nara.ac.jp/~taku-ku/software/TinySVM/

Table 1: Details of data sets.

	General	National	Editorial	Commentary
# of documents	16	76	41	47
# of sentences	342	1721	1362	1096
# of important sentences (10%)	34	172	143	112
# of important sentences (30%)	103	523	414	330
# of important sentences (50%)	174	899	693	555

Table 2: Evaluation results of cross validation.

Summarization rate 10%

Genre	SVM	C4.5	C5.0	Lead
General	**55.7**	55.2	52.4	47.9
Editorial	**34.2**	33.6	27.9	31.6
National	**61.4**	52.0	56.3	51.8
Commentary	**28.7**	27.4	21.4	15.9
Average	**46.2**	41.4	40.4	37.4

Summarization rate 30%

Genre	SVM	C4.5	C5.0	Lead
General	**51.0**	45.7	50.4	50.5
Editorial	**47.8**	41.6	43.3	36.7
National	**55.9**	44.1	49.3	54.3
Commentary	**48.7**	39.4	40.1	32.4
Average	**51.6**	42.4	45.7	44.2

Summarization rate 50%

Genre	SVM	C4.5	C5.0	Lead
General	**65.2**	63.0	60.2	60.4
Editorial	**60.6**	54.1	54.6	51.0
National	**63.3**	58.7	58.7	61.5
Commentary	**65.7**	59.6	60.6	50.4
Average	**63.5**	58.2	58.4	56.1

5 Discussion

Table 2 shows that Editorial and Commentary are more difficult than the other genres. We can consider two reasons for the poor scores of Editorial and Commentary:

- These genres have no feature useful for discrimination.
- Non-standard features are useful in these genres.

Accordingly, we conduct an experiment to clarify genre dependency[4] .

[4] We did not use General because the number of documents in this genre was insufficient.

1 Extract 36 documents at random from genre i for training.

2 Extract 4 documents at random from genre j for test.

3 Repeat this 10 times for all combinations of (i, j).

Table 3 shows that the result implies that non-standard features are useful in Editorial and Commentary documents.

Now, we examine effective features in each genre. Since we used the second order polynomial kernel, we can expand $g(\mathbf{x})$ as follows:

$$g(\mathbf{x}) = b + \sum_{i=1}^{\ell} w_i + 2\sum_{i=1}^{\ell} w_i \sum_{k=1}^{u} x_i[k]x[k] + \sum_{i=1}^{\ell} w_i \sum_{h=1}^{u}\sum_{k=1}^{u} x_i[h]x_i[k]x[h]x[k], \quad (7)$$

where ℓ is the number of support vectors, and w_i equals $\lambda_i y_i$.

We can rewrite it as follows when all vectors are boolean:

$$g(\mathbf{x}) = W_0 + \sum_{k=1}^{u} W_1[k]x[k] + \sum_{h=1}^{u-1}\sum_{k=h+1}^{u} W_2[k,h]x[h]x[k] \quad (8)$$

where
$W_0 = b + \sum_{i=1}^{\ell} w_i$,$W_1[k] = 3\sum_{i=1}^{\ell} w_i x_i[k]$, and $W_2[h,k] = 2\sum_{i=1}^{\ell} w_i x_i[h]x_i[k]$.

Therefore, $W_1[k]$ indicates the significance of an individual feature and $W_2[h,k]$ indicates the significance of a feature pair. When $|W_1[k]|$ or $|W_2[h,k]|$ was large, the feature or the feature pair had a strong influence on the optimal hyperplane.

Table 3: Evaluation results for three genres.

Training \ Test	National			Editorial			Commentary		
	10%	30%	50%	10%	30%	50%	10%	30%	50%
National	**63.4**	**57.6**	**65.5**	32.8	39.4	53.6	24.0	39.5	60.8
Editorial	49.3	46.8	58.4	**33.9**	**49.1**	**64.4**	24.9	43.6	62.1
Commentary	37.4	43.3	61.1	18.4	41.8	57.8	**30.6**	**49.6**	**67.0**

Table 4: Effective features and their pairs

Summarization rate 10%

National	Editorial	Commentary
Lead $\wedge$ ga	$0.9 \leq Posd \leq 1.0 \wedge 0.7 \leq Wf < 0.8$	$0.9 \leq Posd \leq 1.0 \wedge$ *Semdep*=2
$0.9 \leq Posd \leq 1.0 \wedge$ ga	*NE* $\wedge$ de	$0.5 \leq Len_{+1} < 0.6 \wedge$ Noun-*hijiritsu*
Lead $\wedge$ ta	$0.9 \leq Posd \leq 1.0 \wedge$ de	$0.0 \leq Posp < 0.1 \wedge 0.5 \leq Wf_{+1} < 0.6$
$0.9 \leq Posd \leq 1.0 \wedge$ ta	*Lead* $\wedge$ $0.7 \leq Wf < 0.8$	$0.8 \leq Posd < 0.9 \wedge$ Particle

Summarization rate 30%

National	Editorial	Commentary
Lead $\wedge$ *Semdep*=6	$0.0 \leq Posp < 0.1 \wedge$ ga	Aux verb $\wedge$ *Semdep*=2
$0.9 \leq Posd \leq 1.0 \wedge$ *Semdep*=6	$0.9 \leq Posd \leq 1.0 \wedge$ *NE*	Verb-*jiritsu* $\wedge$ *Semdep*=2
Lead $\wedge$ ga	*Lead* $\wedge$ *NE*	*Semdep*=2
$0.9 \leq Posd \leq 1.0$	$0.0 \leq Posd < 0.1$	$0.0 \leq Posp < 0.1 \wedge 0.5 \leq Den < 0.6$

Summarization rate 50%

National	Editorial	Commentary
Lead	$0.0 \leq Posp < 0.1 \wedge$ *Semdep*=6	$0.0 \leq Posp < 0.1 \wedge$ Particle
Lead $\wedge$ ha	$0.0 \leq Posp < 0.1 \wedge$ ga	$0.2 \leq Posd < 0.3$
Lead $\wedge$ Verb-*jiritsu*	$0.0 \leq Posp < 0.1$	$0.4 \leq Len < 0.5$
Lead $\wedge$ ta	$0.0 \leq Posd < 0.1$	$0.0 \leq Posp < 0.1$

Table 4 shows some of the effective features that had large weights $W_1[k]$, $W_2[h, k]$ for each genre.

Effective features common to three genres at three rates were sentence positions. Since National has a typical newspaper style, the beginning of the document was important. Moreover, "ga" and "ta" were important. These functional words are used when a new event is introduced.

In Editorial and Commentary, the end of a paragraph and that of a document were important. The reason for this result is that subtopic or main topic conclusions are common in those positions. This implies that National has a different text structure from Editorial and Commentary.

Moreover, in Editorial, "de" and sentence weight was important. In Commentary, semantically shallow words, sentence weight and the length of a next sentence were important.

In short, we confirmed that the feature(s) effective for discriminating a genre differ with the genre.

6 Conclusion

This paper presented a SVM-based important sentence extraction technique. Comparisons were made using the lead-based method, decision tree learning method, and boosting method with the summarization rates of 10%, 30%, and 50%. The experimental results show that the SVM-based method outperforms the other methods at all summarization rates. Moreover, we clarified the effective features for three gen-

res, and showed that the important features vary with the genre.

In our future work, we would like to apply our method to trainable Question Answering System SAIQA-II developed in our group.

Acknowledgement

We would like to thank all the members of the Knowledge Processing Research Group for valuable comments and discussions.

References

C. Aone, M. Okurowski, and J. Gorlinsky. 1998. Trainable Scalable Summarization Using Robust NLP and Machine Learning. *Proc. of the 17th COLING and 36th ACL*, pages 62–66.

H. Edmundson. 1969. New methods in automatic abstracting. *Journal of ACM*, 16(2):246–285.

T. Fukushima and M. Okumura. 2001. Text Summarization Challenge Text summarization evaluation in Japan. *Proc. of the NAACL2001 Workshop on Automatic summarization*, pages 51–59.

M. Hara, H. Nakajima, and T. Kitani. 1997. Keyword Extraction Using a Text Format and Word Importance in a Specific Filed (in Japanese). *Transactions of Information Processing Society of Japan*, 38(2):299–309.

T. Hirao, M. Hatayama, S. Yamada, and K. Takeuchi. 2001. Text Summarization based on Hanning Window and Dependency structure analysis. *Proc. of the 2nd NTCIR Workshop*, pages 349–354.

S. Ikehara, M. Miyazaki, S. Shirai, A. Yokoo, H. Nakaiwa, K. Ogura, Y. Ooyama, and Y. Hayashi. 1997. *Goi-Taikei – A Japanese Lexicon (in Japanese)*. Iwanami Shoten.

H. Isozaki. 2001. Japanese Named Entity Recognition based on Simple Rule Generator and Decision Tree Learning. *Proc. of the 39th ACL*, pages 306–313.

T. Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proc. of ECML*, pages 137–142.

T. Kudo and Y. Matsumoto. 2000. Japane Dependency Structure Analysis Based on Suport Vector Machines. *Proc. of EMNLP and VLC*, pages 18–25.

T. Kudo and Y. Matsumoto. 2001. Chunking with Support Vector Machine. *Proc. of the 2nd NAACL*, pages 192–199.

J. Kupiec, J. Pedersen, and F. Chen. 1995. A Trainable Document Summarizer. *Proc. of the 18th ACM-SIGIR*, pages 68–73.

Chin-Yew Lin. 1999. Training a Selection Function for Extraction. *Proc. of the 18th ACM-CIKM*, pages 55–62.

H. Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

I. Mani and E. Bloedorn. 1998. Machine Learning of Generic and User-Focused Summarization. *Proc. of the 15th AAAI*, pages 821–826.

C. Nobata, S. Sekine, M. Murata, K. Uchimoto, M. Utiyama, and H. Isahara. 2001. Sentence Extraction System Assembling Multiple Evidence. *Proc. of the 2nd NTCIR Workshop*, pages 319–324.

T. Nomoto and Y. Matsumoto. 1997. The Reliability of Human Coding and Effects on Automatic Abstracting (in Japanese). *The Special Interest Group Notes of IPSJ (NL-120-11)*, pages 71–76.

M. Okumura, Y. Haraguchi, and H. Mochizuki. 1999. Some Observations on Automatic Text Summarization Based on Decision Tree Learning (in Japanese). *Proc. of the 59th National Convention of IPSJ (5N-2)*, pages 393–394.

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

S. Sekine and Y. Eriguchi. 2000. Japanese Named Entity Extraction Evaluation - Analysis of Results -. *Proc. of the 18th COLING*, pages 1106–1110.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. New York.

K. Zechner. 1996. Fast Generation of Abstracts from General Domain Text Corpora by Extracting Relevant Sentences. *Proc. of the 16th COLING*, pages 986–989.