NTT/NAIST's Text Summarization Systems for TSC-2

Tsutomu Hirao† Kazuhiro Takeuchi‡ Hideki Isozaki† Yutaka Sasaki†
Eisaku Maeda†
†NTT Communication Science Laboratories, NTT Corp.
2-4 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, 619-0237, Japan
{hirao,isozaki,sasaki,maeda}@cslab.kecl.ntt.co.jp
‡Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-city, Nara, 630-0101, Japan
kazuh-ta@is.aist-nara.ac.jp

Abstract

In this paper, we describe the following two approaches to summarization: (1) only sentence extraction, (2) sentence extraction + $bunsetsu$ elimination. For both approaches, we use the machine learning algorithm called Support Vector Machines. We participated in both Task-A (single-document summarization task) and Task-B (multi-document summarization task) of TSC-2.

Keywords: *Sentence extraction, $Bunsetsu$ elimination, Support Vector Machines*

1 Introduction

In this paper, we describe the following two approaches to summarization:

(1) only sentence extraction,

(2) sentence extraction + $bunsetsu$ elimination.

The first system is based on important sentence extraction by using Support Vector Machines (SVMs). The second is important sentence extraction and also $bunsetsu$ elimination by using SVMs. The difference between these two systems (System (1) and System (2)) is illustrated in Figure 1.

We participated in both Task-A (single-document summarization task) and Task-B (multi-document summarization task) of TSC-2.

The remainder of this paper is organized as follows. Section 2 describes the machine learning algorithm, Support Vector Machines (SVMs), that we apply to our systems. In Section 3, we explain our sentence extraction method. Section 4 describes our $bunsetsu$ elimination method. In Section 5, we give our evaluation results at TSC-2.

‡Currently with Communication Research Laboratories.

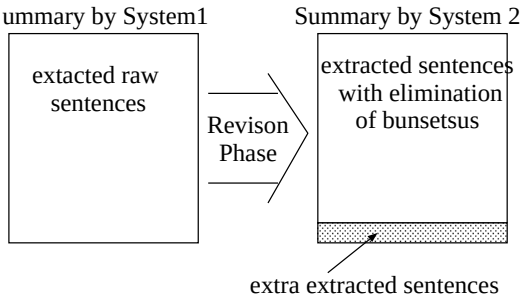

Figure 1. Difference between two systems

2 Support Vector Machines

SVM is a supervised learning algorithm for two-class problems [8].

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 and y_j is its class label, positive (+1) or negative (-1). SVM separates positive and negative examples by a hyperplane given by

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

In general, such a hyperplane is not unique. The SVM determines the optimal hyperplane by maximizing the margin. The margin is the distance between negative examples and positive examples, *i.e.*, the distance between $\mathbf{w} \cdot \mathbf{x} + b = 1$ and $\mathbf{w} \cdot \mathbf{x} + b = -1$. The examples for $\mathbf{w} \cdot \mathbf{x} + b = \pm 1$ compose what is called the Support Vector, which represents both positive and negative examples.

Here, the hyperplane must satisfy the following constraints:

$$y_i(\mathbf{w} \cdot \mathbf{x}_j + b) - 1 \geq 0.$$

Hence, the size of the margin is $2/||\mathbf{w}||$. In order to maximize the margin, we assume the following objective function.

$$\begin{aligned} \underset{\mathbf{w},b}{\text{Minimize}} \quad & J(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_j(\mathbf{w} \cdot \mathbf{x}_j + b) - 1 \geq 0. \end{aligned} \tag{2}$$

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

$$g(\mathbf{x}) = \sum_{i=1}^{u} \lambda_i y_i \mathbf{x}_i \cdot \mathbf{x} + b. \tag{3}$$

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

$$y_i(\mathbf{w} \cdot \mathbf{x}_j + b) - (1 - \xi_j) \geq 0.$$

Hence, we assume the following objective function to maximize margin.

$$\begin{aligned} \underset{\mathbf{w},b,\xi}{\text{Minimize}} \quad & J(\mathbf{w},\xi) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{j=1}^{u} \xi_j \\ \text{s.t.} \quad & y_j(\mathbf{w} \cdot \mathbf{x}_j + b) - (1 - \xi_j) \geq 0. \end{aligned} \tag{4}$$

Here, $||\mathbf{w}||/2$ indicates the size of the margin, $\sum_{j=1}^{u} \xi_j$ indicates the penalty for misclassification, and C is the cost parameter that determines the trade-off for these two arguments. By solving a quadratic programming problem, the decision function $f(\mathbf{x}) = \text{sgn}(g(\mathbf{x}))$ is derived in the same way as linear separation (equation (3)).

The decision function depends only on support vectors ($\lambda_i \neq 0$). Training examples, except for support vectors ($\lambda_i = 0$), have no influence on the decision function.

Moreover, SVMs can handle non-linear decision surfaces by simply substituting every occurrence of the inner product in equation (3) with the kernel function $K(\mathbf{x}_i \cdot \mathbf{x})$. Therefore, the decision function can be rewritten as follows.

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

In this paper, we use polynomial kernel functions:

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

3 Sentence Extraction Phase

In this section, we describe the sentence extraction method based on SVMs.

3.1 Sentence Ranking

Important sentence extraction can be regarded as a two-class problem. However, the proportion of important sentences in training data will differ from that in test data. The number of important sentences in a document is determined by a summarization rate or word limit, which is given at run-time.

A simple solution to this problem is to rank sentences in a document and then select the top N sentences. We used $g(\mathbf{x})$, the normalized distance from the hyperplane to $\mathbf{x}$, to rank the sentences.

3.2 Features for Single-Document Summarization

We define the boolean features discussed below in relation to a sentence S_i. We took past studies into account and added a new feature that represents the TF·IDF value by considering the dependency structure and presence of named entities in a sentence.

Position of sentences

We define two feature functions for the position of S_i. First, Posd(S_i) is S_i's position in a document. Second, Posp(S_i) is S_i's position in a paragraph. The first sentence obtains the highest score, the last obtains the lowest score:

$$\begin{aligned} \text{Posd}(S_i) &= 1 - \frac{BD(S_i)}{|D|} \\ \text{Posp}(S_i) &= 1 - \frac{BP(S_i)}{|P|}. \end{aligned}$$

Here, $|D|$ is the number of characters in the document D that contains S_i; $BD(S_i)$ is the number of characters before S_i in $D(S_i)$; $|P|$ is the number of characters in the paragraph P 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 related to the length of sentence as

$$Len(S_i) = |S_i|.$$

Here, $|S_i|$ is the number of characters of sentence S_i.

TF·IDF

We define the feature function Score(S_i), which weights sentences based on TF·IDF term weighting, as

$$\text{Score}(S_i) = \sum_{t \in S_i} tf(t, S_i) \cdot w(t, D).$$

Here, Score(S_i) is the summation of weighting $w(t, D)$ of terms, appearances in a sentence. $tf(t, S_i)$ is the term frequency of t in S_i.

In addition, we define the term weight $w(t, D)$ based on TF·IDF:

$$w(t, D) = 0.5 \left(1 + \frac{tf(t, D)}{tf_{max}(D)}\right) \cdot \log \left(\frac{|DB|}{df(t)}\right).$$

Here, $tf(t, D)$ is the term frequency of t in D, $tf_{max}(D)$ is the maximum term frequency in D, and $df(t)$ is the frequency of documents that contain term t. $|DB|$ is the number of documents in the database.

We use the term t, which was judged to be a noun or unknown by the morphological analyzer ChaSen[4]. The database indicates MAINICHI newspaper articles by year, *i.e.*, 1994, 1995 and 1999.

Density of keywords

We define the feature function Den(S_i), which represents density of keywords in a sentence, as follows[3].

$$\text{Den}(S_i) = \frac{\sum_{t \in KW(S_i)} w(t, D)}{d(S_i)}.$$

$d(S_i)$ is defined as

$$d(s_i) = \frac{\sqrt{\sum_{k=2}^{|KW(S_i)|} (dist_k)^2}}{|KW(S_i)| - 1}.$$

Here, $KW(S_i)$ is the set of keywords in the sentence S_i, $|KW(S_i)|$ is the number of keywords in the sentence S_i, and $dist_k$ is the distance between the k-th keyword and the $(k-1)$-th keyword in S_i.

Because $d(S_i)$ represents the mean of square distance, density is high if its value is small.

The keyword is the term t that satisfies the following.

$$\mu + 0.5\sigma \leq w(t, D).$$

Note that μ is mean and σ is standard deviation of all $w(t, D)$ in D.

Similarity between Headline and Sentence

We define feature function Sim(S_i), which is the similarity between the headlines of documents that contain S_i, as follows.

$$\text{Sim}(S_i) = \frac{\vec{v}(S_i) \cdot \vec{v}(H)}{\|\vec{v}(S_i)\| \, \|\vec{v}(H)\|}.$$

Here, $\vec{v}(H)$ is a boolean vector in the Vector Space Model (VSM), the elements of which represent terms in the headline. $\vec{v}(S_i)$ is also a boolean vector, the elements of which represent terms in the sentence.

TF·IDF considering dependency structure

We define feature functions Score$_{dep}(S_i)$ and Score$_{wid}(S_i)$ by considering the dependency structure of the sentence:

$$\begin{aligned} \text{Score}_{dep} &= \sum_{t \in t_d} w(t, S_i) \\ \text{Score}_{wid} &= \sum_{t \in t_w} w(t, S_i). \end{aligned}$$

Here, t_d is the set of terms in all $bunsetsu$ that modify the last $bunsetsu$ in the deepest path in the dependency tree, and t_w is the set of terms in all $bunsetsu$ that directly modify the last $bunsetsu$. We use Cabocha[1] for dependency structure analysis.

Named Entities

Boolean value: 1 indicates that a certain Name Entity class appears in S_i. There re eight Named Entity classes[6]:

> PERSON, LOCATION, ORGANIZATION, ARTIFACT, DATE, MONEY, PERCENT, TIME.

We use Isozaki's NE recognizer [2].

Conjunctions

Boolean value: 1 indicates that a certain conjunction appears in S_i The number of conjunctions is 53.

Functional words

Boolean value: 1 indicates that a certain functional word appears in S_i. The number of functional words is 13.

Modality

Boolean value: 1 indicates that S_i has a certain modality that belongs to a certain major or minor category.

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

Rhetorical relations

Boolean value: 1 indicates that S_i has a certain rhetorical relation between S_i and S_{i-1}. The number of relations is four.

Verbs

Boolean value: 1 indicates that S_i has a verb that belongs to a certain class. A verb is classified into one of 36 basic classes by $Goi - taikei$[1]. However, some verbs belong to multiple basic classes. We classified verbs into 366 classes while taking multiple meanings into account.

3.3 Features for Multi-Document Summarization

Here, we define the extra features for multi-document summarization.

Sentence Position in a Document Set

First, the documents in document set E are sorted by their date stamps. Then, we define a feature function Post(S_i) for the position of a sentence S_i in E. The first sentence in the document set E obtains the highest score and the last sentence obtains the lowest score.

$$\text{Post}(S_i) = 1 - \text{BE}(S_i)/M(E).$$

Here, $M(S_i)$ is the number of characters in E. BE(S_i) is the number of characters before S_i in the sorted E.

MDL-based Significant Word Selection

We aim to find a set of significant words that are useful for important sentence extraction from a document set. [7] proposed a method of significant words selection from a document set based on χ^2 metrics, and [5] proposed another method based on AIC metrics. In this paper, we propose an MDL-based significant word selection method.

We defined the feature function that carries our weighting of sentences based on TF·IDF as

$$\text{Score}_E(S_j) = \sum_{t \in T(S_j) \cap T'(E)} tf(t, S_j) \cdot w(t, D_i),$$

$$\text{Score}_C(S_j) = \sum_{t \in T(S_j) \cap T'(C_i)} tf(t, S_j) \cdot w(t, D_i).$$

where $S_j \in C_i$ and $T'(E)$ or $T'(C_i)$ indicates the set of significant words extracted by the MDL-based method. Note that C_i is a subset of E, which includes only documents written on the same day.

Genres of Documents

Sometimes various documents in different genres mention a certain event. However, some genres are useless for summarization. Therefore, we define boolean-valued features for document genres. Here, the documents are classified into seven genres:

> News, Editorial, Commentary, Review, General, Feature, Science.

4 Revision Phase

In this section, we describe the revision phase in which the system rewrites the extracts to improve the quality of the extraction as a summary.

Although the processing in the revision phase of our system is only a trial program, the purpose of submitting such a system is to confirm the present effects of an automated revision process to advance the technology to the next stage.

4.1 Implementation of Elimination Process

The revision phase in our system is based on the elimination of *bunsetsu*, which are the very elemental phrases used as the basic unit in the syntactic structure of Japanese sentences.

Candidates for elimination in our revision phase are particular types of *bunsetsus*. We define the following two conditions in choosing the candidates. First, we restrict a candidate part to the *bunsetsus* in the particular position shown in Figure 2. A sentence including the candidate is divided into three parts (Part A, candidate part for elimination, and Part B in Figure 2) from the information on the sentence's dependency structure. The dependency structure is a simple representation of the syntactic structure of the sentence, each *bunsetsu* except for the last *bunsetsu* in the sentence has a dependency (syntactic) relation to another. Note that Part A does not have a dependency relation to the candidate part, but both these parts have the dependency relations to part B. We permit the case in which the candidate part is the beginning part of the sentence. In this case, Part A is nil.

Second, the candidate part is also restricted to one of three types: adverbs, conjunctive expressions, and adjunctive clauses. We prepared a list of adverbs and conjunctive expressions in which the item serves as a candidate. For an adjunctive clause to serve as a candidate, the type of the last segment of the clause is also restricted. Such a last segment must be on a list that is prepared beforehand.

Training Data

We prepared 600 examples to train SVMs. Half of the examples (300 examples) are for the conjunctive

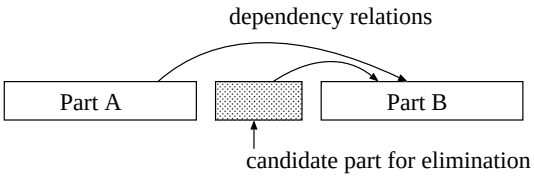

Figure 2. Structural Condition of a Candidate in the Dependency Structure

expressions and adverbs, and the rest of them are for the adjunctive clauses.

Each sentence in the training examples has a part that is a candidate for elimination. In order to train the SVMs, the candidates in the examples have been classified into two classes: to eliminate and not to eliminate. The class of each candidate is manually judged according to its importance in the sentence. When the judges classify each candidate, they can refer to the sentences that come before the sentence in which the candidate occurs to check the backward context.

Features

The features used to represent a candidate consist of the three parts in the dependency structure (Part A, candidate part, Part B in Figure 2) and extra features of the candidate. Each part has the following three features that provide clues to eliminate the candidate part:

- the number of *bunsetsus* that are included in the part
- the occurrence of topic/subject marker in the part
- the type of the last *bunsetsu* of the part

Extra features for the candidate consist of the position information and context information. The position information represents the position where the candidate appears in the sentence. The context information represents the co-occurrence of nouns between the candidate part and other parts. To represent backward contexts, we check the co-occurrence not only in another part of the sentence, but also in the previous sentences. The occurrence of reference expressions in the candidate part is also represented as a part of the context information.

ad-hoc **Elimination Rule**

Our trained SVMs do not eliminate parts very frequently, and the length of an eliminated part is not as long as half of a sentence. In order to investigate elimination of longer clauses (e.g., the subordinate of a complex sentence describing cause-effect), we experimentally make an *ad hoc* rule that always eliminates a certain type of long subordinate. The rule states that a subordinate is eliminated if it has the conjunctive particle 'ga' and a comma separates it from a following main clause that has a topic/subject marker.

5 Results

The TSC-2 Committee evaluates the quality of each set of summaries from various points of view. In TSC-2, two summaries for each source article or document set, a short summary and a long summary, are evaluated. The number of source articles, or document set, is 30.

5.1 Results on Content and Readability Metrics

Table 1, Table 2 show the results of Task-A, Task-B, respectively.

In these tables, "System(1)" denotes the results of the important sentence extraction and "System(2)" denotes the results of the important sentence extraction with $bunsetsu$ elimination.

"C" indicates content-based metrics for summaries, and a lower score means better performance. "R" indicates readability metrics, and again a lower score means better performance.

5.2 Comparison between our eliminations and TSC-2 revision

One of the novel attempts of TSC-2 is the evaluation based on revision of those summaries, in which human judges revise each submitted summary to improve its quality.

In the TSC-2 revision, a bad summary that is difficult to improve by revising is labeled with the tag 'give up'. Among the long summaries, there was only one summary that was given up for TSC-2 revision in the output of System (2). On the other hand, nearly half of the short summaries of both System (1) and System (2) were labeled 'give up'. This shows that it is difficult to improve the quality of a short summary by revising when only a few extracted sentences are not appropriate to represent the main information of the article.

In the TSC-2 revision, a summary was rewritten by using three operations: insert, delete, and replace. Those three operations were applied to improve the following two types of summary quality. In this section, we show the result of Task-A (40%) to discuss the effects of our revision phase clearly.

Quality of Information: the degree to which the content is sufficient and concise in describing the main information of the text

Table 1. Evaluation results of Task-A.

Rate	C(20%)	R(20%)	C(40%)	R(40%)
System(1)	2.80	2.93	2.90	2.90
System(2)	2.77	2.73	2.80	2.90

Table 2. Evaluation results of Task-B.

Rate	C(Short)	R(Short)	C(Long)	R(Long)
System(1)	2.73	2.70	2.77	2.93
System(2)	2.60	2.33	2.97	3.03

Table 3. Comparison between Our Revision and TSC-2 Revision

Elimination part	TSC-2 Revision		
Type	keep	undo	del
Adverb	6	2	3
Conjunctive Exp.	10	2	1
Clause Level	32	7	8
ad hoc Rule	8	5	2
Total	56	16	14

Quality of text: the degree of readability and naturalness as a summary

To compare our revised sentence with the corresponding part in the TSC-2 revision, we divided the treatment of our eliminated part into the following three categories. Assume that E is to an eliminated part in our revision phase.

keep: the elimination of E is kept through the TSC-2 revision

undo: E is rewritten by eliminated expression or by another alternative expression in the TSC-2 revision.

del: the sentence that includes E is deleted in the TSC-2 revision.

Table 3 shows the number of sub-parts that our system eliminates and the number of the operations that applied for them in the TSC-2 revision.

Although this kind of evaluation is very difficult and a more careful investigation is needed, the results lead us to make the following two assumptions.

- Our elimination process treat different sub-parts from those changed in the revision process of TSC-2. (Our revision process mainly treats the inter-clause level.)
- The effect of the eliminations in our revision does not conflict with the TSC-2 revision.

6 Conclusions

In this paper, we described two machine learning-based summarization systems that participated in Task-A and Task-B at TSC-2 and showed our results.

Acknowledgment

We would like to thank all members of the TSC Committee and the NTCIR-Workshop Committee.

References

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

[2] H. Isozaki. Japanese Named Entity Recognition based on Simple Rule Generator and Decision Tree Learning. *Proc. of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 306–313, 2001.

[3] C. Kwok, O. Etzioni, and D. Weld. Scaling Question Answering to the Web. *Proc. of the 10th International World Wide Web Conference*, pages 150–161, 2001.

[4] Y. Matsumoto, A. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara. Morphological Aanalysis System ChaSen version 2.2.1 Manual. Technical report, Nara Institute Science and Technology, 2000.

[5] S. Ohira, K. Hoashi, K. Matsumoto, K. Hashimoto, and K. Shirai. Proposal and Evaluation of Significant Word Selection Method based on AIC. *Proc. of the Symposeum of Natural Language Processing*, 1999.

[6] S. Sekine and Y. Eriguchi. Japanese Named Entity Extraction Evaluation - Analysis of Results -. *Proc. of the 18th International National Conference on Computational Linguistics*, pages 1106–1110, 2000.

[7] R. Swan and J. Allan. Extracting Significant Time Varying Features from Text. *Proc. of the 8th International Conference on Information and Knowledge Management*, pages 38–45, 1999.

[8] V. Vapnik. *The Nature of Statistical Learning Theory*. New York, 1995.