A summarization system with categorization of document sets

Chikashi NOBATA† Satoshi SEKINE‡
Kiyotaka UCHIMOTO† Hitoshi ISAHARA†

†Computational Linguistics Group
Communications Research Laboratory
2-2-2 Hikaridai Seika-Cho Soraku-Gun
Kyoto, 619-0289 Japan
{nova,uchimoto,isahara}@crl.go.jp

‡Computer Science Department, New York University,
715 Broadway, 7th floor, New York, NY 10003 USA
sekine@cs.nyu.edu

Abstract

We participated in both the single-document and multi-document summarization tasks at the TSC 2002. We have incorporated two modules into our earlier summarization system, which is based on a sentence-extraction technique, so that we could apply the system to the multi-document summarization task. One is a module to categorize document sets and the other is to estimate the similarity between sentences. The categorization of document sets is done based on extended named entity classes that include event or facility types as well as original classes. Our system uses the category information to decide how to use similarity information. The similarity between sentences is measured according to the Dice coefficient, and the results are used to either select a representative sentence from among similar sentences or to extract typical sentences from a given document set.

Keywords: *Multi-document summarization, Sentence extraction, Document set categorization, Sentence similarity*

1 Introduction

Our summarization system is based on a system that we developed for TSC2001[7] and uses a sentence-extraction technique, a frequently used method for summarization ([9], [2], [12], [8], [1], [4]). Since our previous system could only produce a summary for a single document, we added two modules so that we could apply the system to a multi-document summarization task. One module is to categorize document sets to allow multi-document summarization, and the other is to estimate the similarity between sentences. We describe how these two modules work with the sentence-extraction system in Section 2. At TSC2002, we participated in both single-document summarization (Task A) and multi-document summarization (Task B) [11]. The evaluation results of our system for these tasks are discussed in Section 3.

2 System Overview

In this section, we explain the scoring functions used for extracting key sentences, which are revised forms of those used in our previous system. Then, we introduce the two added modules: one to categorize document sets and one to measure sentence similarity.

2.1 Score function

Our system uses four types of metrics to estimate the significance of sentences: sentence position, sentence length, term frequency, and similarity to the title. The significance of sentences is given by the sum of the values of these metrics which is used with parameters. Each metric is explained below.

2.1.1 Sentence position

Our system has a function that uses sentence position to establish the significance of sentences. In this function, three methods are used to handle sentence position. The first is to give a score of 1 to the first N sentences and 0 to the others, where N is a given threshold for the number of sentences. That is, the score of the ith sentence (S_i) is:

$$\text{P1. } \mathit{Score}_{\text{pst}}(S_i)(1 \leq i \leq n) = 1 \quad (\text{if} \quad i < N)$$
$$= 0 \quad (\text{otherwise})$$

where n is the number of sentences in a given article. The second method is to give a score that is the reciprocal of the sentence position, so that the score of the ith sentence (S_i) is:

$$\text{P2. } \mathit{Score}_{\text{pst}}(S_i) \quad = \quad \frac{1}{i}.$$

These two methods are based on the hypothesis that sentences at the beginning of an article are more important than those elsewhere in the article.

The third method is a modified version of the second in that it checks the sentence position from the end of the article as well as from the beginning:

$$\text{P3. } \mathit{Score}_{\text{pst}}(S_i) \quad = \quad \max(\frac{1}{i}, \frac{1}{n-i+1}).$$

This method is based on the hypothesis that sentences at either the beginning or the end of an article are more important than those in the middle.

The third method (referred to as P3) performed best in the training stage using the dry run data.

2.1.2 Sentence length

The second scoring function uses sentence length to establish the significance of sentences. The length here means the number of characters in the sentence. Two methods are defined here. The first returns the length of each sentence (L_i) relative to the maximum length of the sentence (L_{max}). Because we would like to use a uniform setting for all document sets, we fixed the value of (L_{max}) in advance to 200.

$$\begin{aligned} \text{L1. } \mathit{Score}_{\text{len}}(S_i) &= \frac{L_i}{L_{max}} (\text{if} \quad L_i \leq L_{max}) \\ &= 1 \quad (\text{otherwise}). \end{aligned}$$

The second method sets the score to a negative value as a penalty when a sentence is shorter than a certain length (L_{min}):

$$\begin{aligned} \text{L2. } \mathit{Score}_{\text{len}}(S_i) &= 0 \quad (\text{if} \quad L_i \geq L_{min}) \\ &= \frac{L_i - L_{min}}{L_{min}} \quad (\text{otherwise}). \end{aligned}$$

Since we set L_{min} to 20 in the following evaluation, a sentence with 20 or fewer characters received a penalty score. The second method (L2) performed better in the training stage using the dry run data.

2.1.3 Tf*idf

The third scoring function is based on term frequency and document frequency. The hypothesis here is that sentences containing more words that are specific to an article are more likely to be relevant. The target words are nouns (excluding temporal or adverbial nouns), and the system calculates the tf*idf score for each of these nouns in a sentence,. The total score indicates the significance of the sentence. The word segmentation is done using Juman ver. 3.61 [3].

When a set of documents is given in advance, our system counts the term frequency (*tf*) and the document frequency (*df*) for each word w, then calculates the tf*idf score as follows:

$$\text{tf*idf}(w) \quad = \quad \mathit{tf(w)} \log \frac{\mathit{DN}}{\mathit{df(w)}}$$

where *DN* is the number of given documents. We used articles that appeared in the Mainichi newspaper from 1994 to 2001 to count the document frequency.

The sentence score with tf*idf values of words is calculated with normalization [5]. When a document D is given, our system calculates the Euclidean norm for tf*idf values for all words in D (D_{norm}).

$$D_{\text{norm}} \quad = \quad \sqrt{\sum_{w \in D} \text{tf*idf}(w)^2}$$

The score for the ith sentence (S_i) in D is then calculated as follows:

$$\mathit{Score}_{\text{tf*idf}}(S_i) \quad = \quad \frac{1}{D_{\text{norm}}} \sqrt{\sum_{w \in S_i} \text{tf*idf}(w)^2}$$

2.1.4 Headline

The fourth scoring function is to use the headline of an article to establish the significance of sentences. The basic idea is that the greater the number of words in a sentence that match those in the headline, the more important the sentence is likely to be. This function estimates the relevance between a headline (H) and a sentence (S_i) using the tf*idf values of words (w) (except for the stop words) in the headline:

$$\text{H1. } \mathit{Score}_{\text{hl}}(S_i) = \frac{\sum_{w \in H \cap S_i} \text{tf*idf}(w)}{\sum_{w \in H} \text{tf*idf}(w)}$$

We also evaluated this scoring function using only named entities (NEs) instead of the nouns. Named entities were annotated by a pattern-based named entity extraction program developed to annotate extended named entity categories [10]. For NEs, only the term frequency was used because we expected the document frequency for entities (e) to usually be quite small, thereby making the difference between entities negligible:

$$\text{H2. } \mathit{Score}_{\text{hl}}(S_i) = \frac{\sum_{e \in H \cap S_i} \frac{\mathit{tf(e)}}{\mathit{tf(e)}+1}}{\sum_{e \in H} \frac{\mathit{tf(e)}}{\mathit{tf(e)}+1}}$$

From the training data, we found that the scoring function using only NEs worked better than that using all words.

2.2 Parameters

Our system uses parameters to integrate the results of each scoring function in order to calculate the total score for a sentence. The total score for a sentence (S_i) is determined using a scoring function (*Score*$_j()$) and a parameter (α_j) as follows:

$$\text{Total-Score}(S_i) = \sum_j \alpha_j \textit{Score}_j(S_i)$$

Our system calculates a score for all of the sentences and sets the ranking of each sentence in descending order of score. The order of the extracted sentences is the same as in the original articles when the system outputs a summary.

We approximated the optimal values of these parameters using the extraction data created from the dry run data of Task A. The parameter sets for Task A were also used for Task B: the parameter set for 20% compression in task A was applied to the short summary task in Task B, and that for 40% was applied to the long summary task.

After the range of each parameter was set manually, the system changed the values of the parameters within the range and performed a summarization based on the dry run data. Each score was recorded whenever the parameter values were changed, and the parameter values resulting in the best score were stored.

Table 1 shows the contribution of each feature that was the basis of a scoring function. The contribution was the product of the optimized weight and the standard deviation of the score. The weights were normalized by the norm of all weights; i.e.

$$\sum_j \alpha_j = 1.$$

The function types selected in the training stage are also shown in the table.

When the compression ratio was 20%, the largest contribution was from the tf*idf feature and the next largest was from the length feature. On the other hand, when the compression ratio was 40%, the length feature was dominant.

2.3 Categorization of document sets

For multi-document summarization, our system assigns a given document set to a category before generating summaries. We defined 13 categories based on the NE classes, which are shown in Table 2. They are also an extension of the categories used in [6].

Table 1. Contribution of each feature

		Contribution ($\times 10^{-2}$)	
Features	Type	20%	40%
Position	P3	8.51	4.99
Length	L2	10.11	14.83
tf*idf	-	12.57	1.84
Headline	H2	1.16	0.34

Table 2. Types of categories

Single-Person	Multi-Person
Single-Organization	Multi-Organization
Single-Location	Multi-Location
Single-Facility	Multi-Facility
Single-Product	Multi-Product
Single-Event	Multi-Event
Other	

"Single" means that all the articles are talking about a single event, person, or organization, whereas "multiple" articles are talking about several different events, people, or organizations. Also, we assign the entity type of the central topic to one of the NE categories. We used the definition developed in [10].

The categorization module uses the frequency and document frequency of words and NE tags. The NE classes to be tagged are those that we used in the definition; i.e., 'ORGANIZATION', 'PERSON', 'LOCATION', 'EVENT', 'PRODUCT', and 'FACILITY'.

Besides NE tags, a 'class-term' list is also used. Class-terms are nouns or compound nouns which are closely related to a particular NE class. For example, "president" is a person class-term, and "earthquake" is an event class-term. When terms from the list are found in the given document set, the system stores the frequency and the document frequency of the terms and the class, respectively. The lists contain about 16,000 class-terms that were created using a thesaurus and some human labor for a different task.

Our categorization method contains three stages. First, we try to determine if a given set belongs in a single-class category. This is done by checking the document frequency of the most frequent (in terms of document frequency) NE word. If a particular NE word appears throughout the documents in the set, the word is likely to be the main topic and the NE type of the word might indicate the category. We empirically found that different criteria are needed between EVENT type and the other classes (Figure 1).

The second step is that if no single-class category is found, we try to find if the set is a multi-class category by using the most frequent (again in terms of the document frequency) NE class. If the document frequency of words in a particular NE class is high (i.e.,

w = the word with the highest DF
p_w = the NE class assigned to w
if DF(w) $> T_d$ then
  if $p_w =$ Event then
    category = Single-Event
  else if $\frac{|w|}{|p_w|} > T_w$ then
    category = Single-p_w
  else goto *Algorithm 2*
else goto *Algorithm 2*

Figure 1. Pseudo code for the 'single-class' category (Algorithm 1)

p = NE class with the highest DF
P = all NE classes
if DF(p) $> T_d$ then
  if $p =$ Event then
    category = Multi-Event
  else if $p =$ (L or O or P) then
    if $\frac{|p|}{|P|} > T_{C1}$ then
      category = Multi-p
    else goto *Algorithm 3*
  else if $\frac{|p|}{|P|} > T_{C2}$ then
    category = Multi-p
  else goto *Algorithm 3*
else goto *Algorithm 3*

Figure 2. Pseudo code for the 'multi-class' category (Algorithm 2)

words of a particular NE class appear throughout the documents), that class is likely to be the multi-class category of that NE type (Figure 2).

Third, we use the class-term to find if the set belongs in a multi-class category. The system repeats the first and the second step explained above, looking for class terms instead of NEs, and always outputs multi-classes instead of single-classes. All categories created at this step are multi-class, as the target we are looking at is common nouns, rather than proper nouns. In other words, if a class-term word appears throughout the documents, it is likely to belong to a multi-class category (Figure 3).

The details of the algorithms are described in pseudo code as shown in Figures 1 to 3. Here, "DF" means "document frequency" and "L or O or P" means "Person or Location or Organization".

2.4 Similarity between sentences

We added a module to estimate the similarity between sentences. Similarity values are used to either select one key sentence from among semantically similar sentences or output a set of similar sentences with

t = class-term with the highest DF
q_t = NE class assigned to t
q = NE class with the highest DF made from all class-terms in the class
Q = all NE classes made from class-terms
if DF(t) $> T_d$ then
  if $q_t =$ Event then
    category = Multi-Event
  else if $\frac{|t|}{|q_t|} > T_w$ then
    category = Multi-q_t
  else goto NECLASS
else goto NECLASS

NECLASS:
if DF(q) $> T_d$ then
  if $q =$ Event then
    category = Multi-Event
  else if $q =$ (P or L or O) then
    if $\frac{|q|}{|Q|} > T_{C1}$ then
      category = Multi-q
    else category = Multi-Event (DEFAULT)
  else if $\frac{|q|}{|Q|} > T_{C2}$ then
    category = Multi-q
  else category = Multi-Event (DEFAULT)
else category = Multi-Event (DEFAULT)

Figure 3. Pseudo code for 'multi-class' category using class-terms (Algorithm 3)

high sentence scores.

The system calculates the Dice coefficient as a similarity measure based on the number of words between two sentences S_x and S_y. For the words $\{x_i|x_i \in S_x\}$ and $\{y_j|y_j \in S_y\}$, the similarity between S_x and S_y is calculated as

$$Dice(S_x, S_y) = \frac{2 \sum_{x_i=y_j} ave(f(x_i), f(y_j))}{\sum_i f(x_i) + \sum_j f(y_j)}.$$

The function $f()$ gives the weight for each word. Three types of weight can be selected:

Binary: if the word appears in the sentence, the weight is set to 1. Otherwise, the weight is set to 0.

Tf: the tf value of the word.

Tf*idf: the tf*idf value of the word.

The system uses one of the weights to calculate similarities. We used the simplest similarity measure for the formal run, i.e., the Dice coefficient with the binary weight, because we didn't observe any significant difference among the weights in the experiment using the dry run data.

Sentence pairs that have a coefficient value higher than a threshold T_s are regarded as similar to each other. The value of the threshold T_s is set to 0.5 at the training stage.

Our system has two methods to use the similarity information. One method is to select only the one sentence that has the highest sentence score among similar sentences. When this sentence is selected, other sentences that are similar to the selected one are discarded. This is intended to remove redundancy from the generated summary. For example, in articles about a criminal case the established facts of the case are repeatedly described. Such repetition should be removed.

The other method is to output a set of similar sentences when one of them is selected. This is intended to output typical expressions in the given document sets. For example, when each article in a given document set describes an earthquake that occurred at a different place, expressions in the articles are typical and similar, but provide us with different information. These expressions should be included in the summary of the document set.

3 Results and discussion

In this section, we discuss the evaluation results of our system for each task of the TSC formal run. We consider the subjective ranking evaluation in Task A and Task B. In Table 3, which shows the results for Task A, 'C' means the evaluation of 'contents', 'R' means 'Readability', and 20% and 40% are compression ratios. The actual score of our system (Sys) and rank among eight systems (Rank) are shown for a baseline system using the term frequency (Baseline(tf)) and a manually created summary (Manual). The average scores (Ave.) among eight systems are also shown.

Table 3. Evaluation results for Task A

	C20%	R20%	C40%	R40%	ALL
Sys	2.70	2.60	2.50	2.53	2.58
Rank	5	2	2	1	1
Baseline(tf)	3.30	3.30	3.20	3.10	3.23
Manual	2.33	2.20	2.10	2.03	2.17
Ave.	2.73	2.84	2.70	2.80	2.77

Table 4. Evaluation results for Task B

	Cshort	Rshort	Clong	Rlong	ALL
Sys1	2.93	2.70	2.53	2.80	2.74
Sys1(Rank)	8	3	2	1	2
Sys2	2.83	2.73	2.53	2.87	2.74
Sys2(Rank)	7	4	2	2	2
Manual	2.00	2.17	1.83	2.33	2.08
Ave.	2.73	2.83	2.77	3.08	2.85

In Task A, our system obtained better scores than the baseline system at compression ratios of both 20% and 40%. The 40% summaries showed better results than the 20% summaries, probably because our system is based on a sentence-extraction technique that enables the generated summaries to preserve the structure of sentences in the original documents.

In Task B, we submitted two summary results. One result was a summary without the category information of the document set (Sys1), and the other was a summary using the information (Sys2). The results are shown in Table 4. The influence of the category information was not clear. One reason for this was that the NE program we used was not sufficient to annotate extended classes such as event and facility names. Since the categorization module relies on the quality of the NE annotation, we plan to improve the NE program so that it will enable our categorization module to properly find key entities and NE classes. Another reason was that we have only two methods that were selected using the category information. We need to analyze the characteristics of different document sets to find suitable summarization methods in each category for the document sets.

4 Conclusion

We have implemented two modules in our summarization system, which incorporates a sentence-extraction technique, to allow it to summarize multiple documents. One module is to estimate the similarity between sentences, and the other is to categorize the given document sets. With this system, we participated in both the single-document and the multi-document summarization tasks at the TSC 2002. While the module for categorizing document sets did not improve our evaluation results for Task B, the subjective evaluation using our results was better than the baseline system.

As future work, we plan to improve the NE program to enable our categorization module to properly find key entities and NE classes since the quality of NE annotation is critical to the performance of our categorization module. We would also like to analyze the characteristics of different document sets to find suitable summarization methods in each category for document sets.

Acknowledgment

We thank Dr. Hirao of the NTT Cmputer Sceience Laboratories for helping us to use the TSC dry run data to set parameter values.

References

[1] C. Aone, M. E. Okurowski, J. Gorlinsky, and B. Larsen. A Scalable Summarization System Using Robust NLP. In *Proceedings of the ACL Work shop on Intelligent Scalable Text Summarization*, pages 66–73, 1997.

[2] H. Edmundson. New methods in automatic abstracting. *Journal of ACM*, 16(2):264–285, 1969.

[3] S. Kurohashi and M. Nagao. *Japanese Morphological Analyzing System: JUMAN version 3.61*. Kyoto University, 1999.

[4] C.-Y. Lin. Training a selection function for extraction. In *Proc. of the CIKM'99*, 1999.

[5] Madani. http://classes.seattleu.edu/computer_science/csse470/Madani/ABCs.html. ABCs of Text Categorization.

[6] K. R. McKeown, R. Barzilay, D. Evans, V. Hatzivassilogou, M. Y. Kan, B. Schiffman, and S. Teufel. Columbia Multi-Document Summarization: Approach and Evaluation. In *Online Proceedings of DUC2001*, 2001.

[7] C. NOBATA, S. SEKINE, M. MURATA, K. UCHIMOTO, M. UTIYAMA, and H. ISAHARA. Sentence extraction system assembling multiple evidence. In *Proceedings of the Second NTCIR Workshop Meeting*, pages 5–213–218, March 2001.

[8] T. Nomoto and Y. Matsumoto. The Reliability of Human Coding and Effects on Automatic Abstracting (in Japanese). In *IPSJ-NL 120-11*, pages 71–76, July 1997.

[9] M. Okumura and H. Nanba. Automated Text Summarization: A Survey (in Japanese). *Journal of Natural Language Processing*, 6(6):1–26, 1999.

[10] S. Sekine, K. Sudo, and C. Nobata. Extended Named Entity Hierarchy. In *Proceedings of the LREC-2002 Conference*, pages 1818–1824, 2002.

[11] TSC. http://oku-gw.pi.titech.ac.jp/tsc/. Text Summarization Challenge.

[12] H. Watanabe. A Method for Abstracting Newspaper Articles by Using Surface Clues. In *Proc. of COLING'96*, pages 974–979, 1996.