Sentence Extraction with Information Extraction technique

Satoshi Sekine
Computer Science Department
New York University
715 Broadway, 7th floor
New York, N.Y. 10003 USA
sekine@cs.nyu.edu

Chikashi Nobata
Computational Linguistics Group
Communications Research Laboratory
2-2-2 Hikaridai Seika-cho
Soraku-gun Kyoto 619-0289 JAPAN
nova@crl.go.jp

Abstract

We have introduced information extraction technique such as named entity tagging and pattern discovery to a summarization system based on sentence extraction technique, and evaluated the performance in the Document Understanding Conference 2001 (DUC-2001). We participated in the Single Document Summarization task in DUC-2001 and achieved one of the best performance in subjective evaluation of summarization results.

1 Introduction

Our aim in this research is to use information extraction (IE) technologies for summarization. The NE task is the lowest level of IE task, defined in the Message Understanding Conference (MUC). An NE tagger tries to identify proper names like person, organization, location and numeral expressions such as temporal or monetary expressions. Excellent NE taggers in English newspaper articles have the performance comparable to human ability [2] and we think NEs are useful to find key expressions for summarization.

We also used automatic pattern discovery procedure, which is currently a hot topic in information extraction (IE) research ([7], [10], [8]). Patterns here mean typical phrases which express specific events in a given domain. A method of pattern matching has been used in most of IE systems. For example, in executive succession domain, the system tries to find the events in given documents using phrase patterns like "Organization hires person". Automatic pattern discovery is a method to find such phrase patterns which is useful for IE. We think phrase patterns are also useful to summarize documents when the domain is specified.

We used those two IE methods combining with the standard summarization technologies. Our summarization system is based on a sentence extraction technique, which is one of useful methods used for automatic text summarization. Various clues have been used for sentence extraction. The lead-based method, which is simple but still effective, uses the sentence location in a given document. Statistical information, like word frequency and document frequency, has also been used to estimate the significance of sentences. Linguistic clues that indicate the structure of a document are also useful for extracting important sentences.

Edmundson [3] proposed a method of integrating several clues to extract sentences. He manually assigned parameter values to integrate evidence for estimating the significance score of sentences. On the other hand, machine learning methods can be applicable to integrate clues. Aone et al. [1] use Bayes' rule, and Lin [4] and Nomoto & Matsumoto [5] generate a decision tree [6] for sentence extraction from training data.

We integrated IE technique into our summarization system based on sentence extraction, which has performed one of the best results in the Text Summarization Challenge (TSC), the evaluation workshop for Japanese summarization [9]. We participated in the Single Document Summarization in Document Understanding Conference in 2001

(DUC-2001) to evaluate the performance of our system.

The organizers of DUC-2001 provided 30 sets of documents, each set containing about 10 documents as training data before evaluation. Since summaries for each document were also created by hand and provided, we used the training data to tune weights for scoring functions. In the evaluation, another 30 sets of documents were provided as test data.

In the following sections, we explain our summarization system and show the evaluation results in DUC-2001.

2 System overview

We used five components in our sentence extraction system. For each sentence, each component output a score. The system combines these independent scores by interpolation. The weights and function types to be used were decided by optimizing the performance of the system on training data. We explain each function and show the contribution of each component in the following sections.

2.1 Sentence position

We prepared three functions for sentence position. The first function returns 1 if the position of the sentence within a given threshold T from the beginning, and returns 0 otherwise:

$$\text{P1. } Score_{\text{pst}}(S_i)(1 \leq i \leq n) = 1(\text{if} \quad i < T)$$
$$= 0(\text{otherwise})$$

The threshold T is decided by the number of words for the summary. The first scoring function was selected in a training stage.

The second function is a reciprocal function to the position of the sentence, by which the first sentence receives the highest score and gradually decrease and go to the minimum at the final sentence.

$$\text{P2. } Score_{\text{pst}}(S_i) = \frac{1}{i}$$

The third function is the maximum of the reciprocal to the position from the beginning or the end of the document.

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

This function was the optimal function in summarizing the editorials in the TSC evaluation workshop for Japanese summarization. We obtained the top score in the 10% summary of the TSC project, subjective judgment in the evaluation.

2.2 Sentence length

The second scoring function uses sentence length to set the significance of sentences. The length here means the number of words in the sentence. The first method only returns the length of each sentence (L_i):

$$\text{L1. } Score_{\text{len}}(S_i) = L_i$$

The second method sets the score to a negative value as a penalty when the sentence is shorter than a certain length (C), like:

$$\text{L2. } Score_{\text{len}}(S_i) = 0 \quad (\text{if} \quad L_i \geq C)$$
$$L_i - C \quad (\text{otherwise})$$

Since we set C to 10 in the following evaluation, a sentence with 10 or less words received a penalty score. The scoring function with penalty was selected in a training stage.

2.3 tf*idf

The third scoring function is based on term frequency (*tf*) and document frequency (*df*). The hypothesis here is that the more words that are specific to a transcription that exist in a sentence, the more important the sentence is.

The score of a sentence (S_i) is the average of the tf*idf scores of each word (w) in the sentence:

$$Score_{\text{tf*idf}}(S_i) = \frac{1}{|S_i|} \sum_{w \in S_i} \text{tf*idf}(w)$$

We have three scoring functions for tf*idf, where term frequencies were calculated differently. The first one used the raw term frequencies, and the others are two ways of normalizing the figure:

$$\text{T1. tf*idf}(w) = \textit{tf(w)} \log \frac{DN}{\textit{df(w)}}$$

$$\text{T2. tf*idf}(w) = \frac{\textit{tf(w)-1}}{\textit{tf(w)}} \log \frac{DN}{\textit{df(w)}}$$

$$\text{T3. tf*idf}(w) = \frac{\textit{tf(w)}}{\textit{tf(w)+1}} \log \frac{DN}{\textit{df(w)}}$$

where *DN* is the number of given documents. We used all the articles in the Wall Street Journal in 1994 and 1995 to count document frequencies. The scoring function with the raw term frequency was selected in a training stage.

2.4 Headline

We used the similarity measure of the sentence to the headline. The basic idea is that the more words in the sentence overlap with the words in the headline, the more important the sentence is. This function estimates the relevance between a headline(H) and a sentence(S_i) using the tf*idf values of words(w) except stop words in the headline:

$$\text{H1. } Score_{\text{hl}}(S_i) = \frac{\displaystyle\sum_{w \in H \cap S_i} \frac{tf(w)}{tf(w)+1} \log \frac{DN}{df(w)}}{\displaystyle\sum_{w \in H} \frac{tf(w)}{tf(w)+1} \log \frac{DN}{df(w)}}$$

We also evaluated this scoring function using only named entities (NEs). instead of the nouns. For NEs, only the term frequency was used because we judged that the document frequency for entities(e) was usually quite small thereby making the difference between entities negligible:

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

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

2.5 Patterns

In this section, we explain a scoring function that introduces the other IE technology in our system. The assumption of the IE pattern discovery is that patterns that appears often in the domain is important. For example, in the earthquake report domain, we may find a lot of patterns like "There was an earthquake in LOCATION at TIME on DATE". Then it is regarded as an important pattern in the domain.

We used the given document set provided by DUC to extract patterns. There were 30 sets of documents and about 10 documents per set. We assumed each set of documents as a domain, and extracted patterns at each set. In the pattern instances, each type of named entities was treated as a class rather than the literal words. The introduced procedure of pattern discovery follows the one used for Japanese IE [8]. A Basic procedure of pattern discovery has the following processes, as shown in Figure 1.

1. Analyze sentences
   The system analyzes all the sentences in the domain documents, such as named entity tagging and dependency analysis.

2. Extract sub-trees
   The system extracts all sub-trees from the dependency trees in the domain.

3. Score sub-trees
   The system scores sub-trees is set by the multiplication of the frequency of the tree and the idf values average among the words in the tree. Score trees with high score are regarded as important patterns.

Extracted patterns are stored with the scores prior to summarization. In the summarization stage, the system applies named entity tagging and dependency analysis to each sentence (S_i) so that it can be compared with stored patterns. If a stored pattern (P_j) matches the sentence, the score of the pattern is accumulated, and the logarithm of the accumulated pattern scores is set as the score for the sentence:

$$\begin{aligned}
Score_{\text{pat}}(S_i) &= \log(PatScore(S_i) + 1) \\
PatScore(S_i) &= \sum_j F_{P_j} \frac{\sum_{w \in P_j} \log \frac{DN}{df(w)}}{|P_j|} \\
&\quad \text{(if } P_j \text{ matches } S_i\text{)} \\
&= 0 \text{ (otherwise)}
\end{aligned}$$

where F_{P_j} is the frequency of the pattern P_j in a given domain, $|P_j|$ is the number of words in P_j.

2.6 Optimal weight

Our system set weights at each scoring function to calculate the total score of a sentence. The total

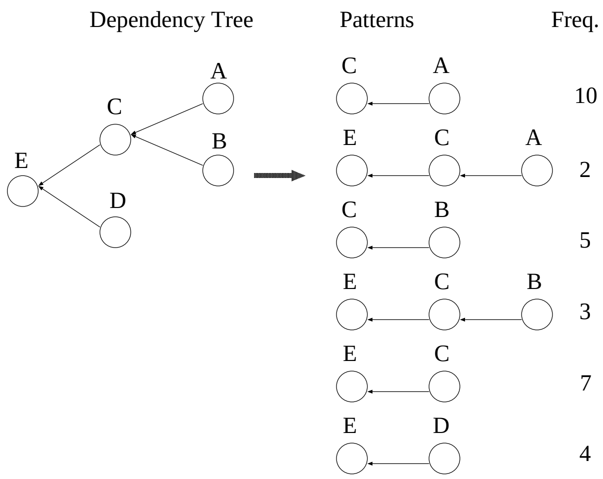


Figure 1: Processes of pattern discovery

score of a sentence (S_i) is set using scoring functions ($Score_j()$) and weights (α_j) as follows:

$$\text{TotalScore}(S_i) = \sum_j \alpha_j Score_j(S_i)$$

We approximated the optimal values of these weights with training data. After the range of each weight was set manually, the system changed the values of the weights within the range and performed a summarization on training data. Each score was recorded whenever the weight were changed, and the weight having the best score were stored.

Table 1 shows the contribution of each component, which is multiplication of the optimized weight and the standard deviation of the score. The selected function type in the training stage is also shown in the table. We can see that the biggest contribution is the sentence position and the second is the tf*idf. Contribution from the other features is relatively small.

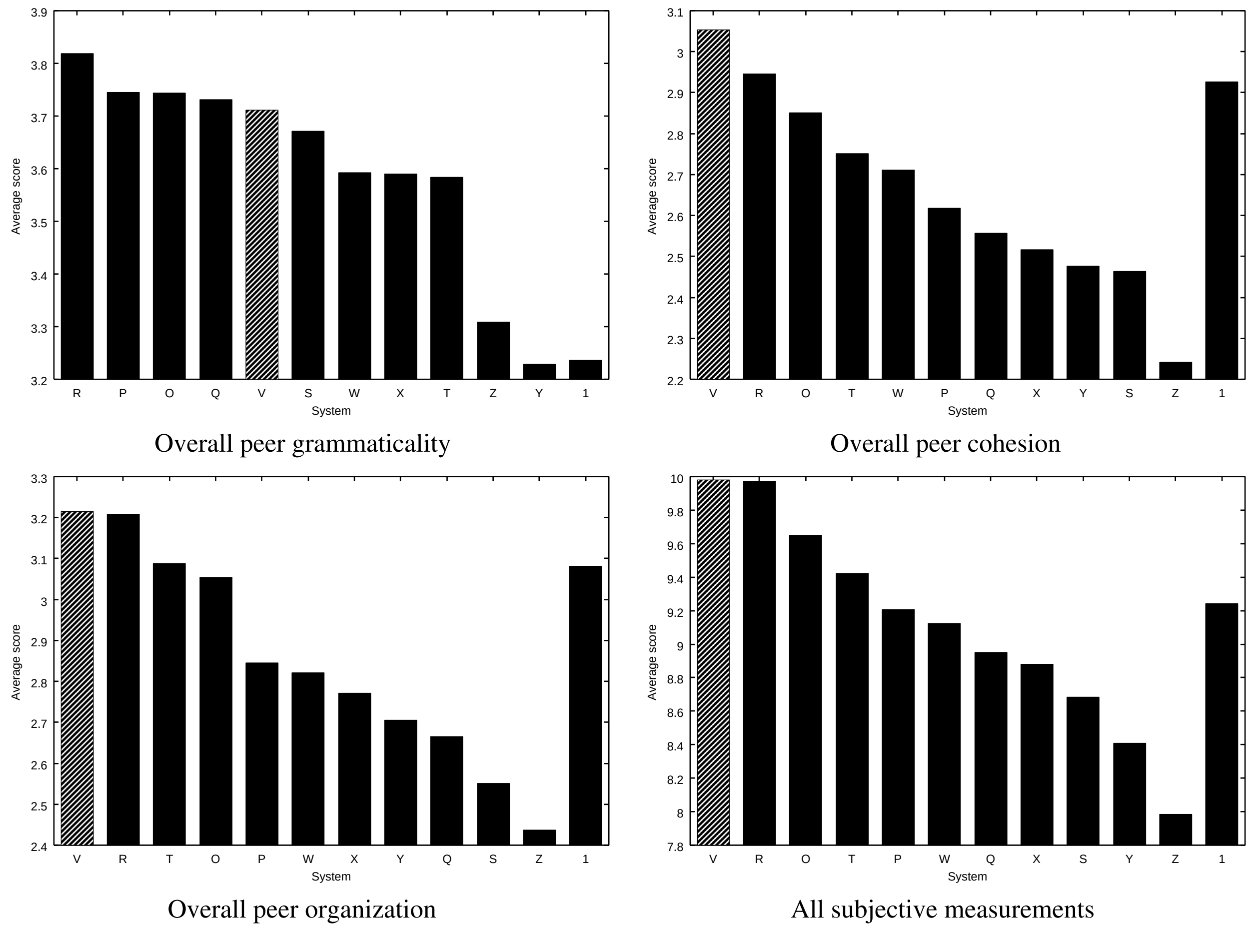
Table 1: Contribution of each feature

Features	Type	Contribution
Position	P1	277
Length	L2	8
tf*idf	T1	96
Headline	H2	18
Pattern	-	2

3 Evaluation results

Table 2 shows the result of subjective evaluation in single document summarization. Subjective evaluation was performed by 10 assessors, who gave a score each system's outputs on a zero-to-four scale (four is the best), compared to human-made summaries. The figures are the average scores over all documents. The 'NYU/CRL' column shows the performance of our system with the rank among 12 systems that participated in Single Document Summarization. The 'Lead' column shows the performance of the baseline system that always outputs the first 100 words of a given document, and the 'Avg.' column shows the average of all systems Our system scores the 5th in grammaticality and the top for the other measurements including total.

Figure 2 shows the performance of all systems in the descending order of each subjective evaluation. The rightmost bar ('1') shows the performance of the baseline system. We see that our system ('V') was especially better than other systems in 'Cohesion', and one of the best systems in total.



Overall peer grammaticality

Overall peer cohesion

Overall peer organization

All subjective measurements

Figure 2: Subjective evaluation results of all systems in single document summarization

Table 2: Evaluation results in single document summarization

	NYU/CRL	Lead	Avg.
Grammaticality	3.711 (5)	3.236	3.580
Cohesion	3.054 (1)	2.926	2.676
Organization	3.215 (1)	3.081	2.870
Total	9.980 (1)	9.243	9.126

4 Concluding remarks

We have introduced information extraction technique such as named entity extraction and pattern discovery to our summarization system which is based on sentence extraction. The performance of our system was better than that of a baseline system and the average of all systems in every subjective evaluation of Single Document Summarization task in DUC-2001. Our system was the best in 'Cohesion', 'Organization' as well as in total.

As a future work, we consider to extract patterns from larger sets of similar documents using IR techniques, We would also like to utilize more information of named entities, since our NE tagging tool with extended class definition is now developing.

References

[1] Chinatsu Aone, Mary Ellen Okurowski, and James Gorlinsky. Trainable, Scalable Summarization Using Robust NLP and Machine Learning. In *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, pages 62–66, 1998.

[2] DARPA. *Proceedings of the Seventh Message Understanding Conference(MUC-7)*, Fairfax, VA, USA, May 1998.

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

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

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

[6] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1993.

[7] Ellen Riloff. Automatically Generating Extraction Patterns from Untagged Text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 1044–1049, 1996.

[8] Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. Automatic pattern acquisition for japanese information extraction. In *Proceedings of Human Language Technology Conference*, San Diego, California, USA, 2001.

[9] TSC. http://oku-gw.pi.titech.ac.jp/tsc/, 2001. Text Summarization Challenge.

[10] Roman Yangarber, Ralgh Grishman, and Pasi Tapanainen. Unsupervised discovery of scenario-level patterns for information extraction. In *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP2000)*, 2000.