Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Processing and Management 41 (2005) 75–95
INFORMATION PROCESSING & MANAGEMENT
www.elsevier.com/locate/infoproman

Text summarization using a trainable summarizer and latent semantic analysis ☆

Jen-Yuan Yeh [a,*], Hao-Ren Ke [b], Wei-Pang Yang [a], I-Heng Meng [a]

[a] *Department of Computer & Information Science, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu 30050, Taiwan, ROC*
[b] *Digital Library & Information Section of Library, National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu 30050, Taiwan, ROC*

Available online 14 May 2004

Abstract

This paper proposes two approaches to address text summarization: modified corpus-based approach (MCBA) and LSA-based T.R.M. approach (LSA + T.R.M.). The first is a trainable summarizer, which takes into account several features, including position, positive keyword, negative keyword, centrality, and the resemblance to the title, to generate summaries. Two new ideas are exploited: (1) sentence positions are ranked to emphasize the significances of different sentence positions, and (2) the score function is trained by the genetic algorithm (GA) to obtain a suitable combination of feature weights. The second uses latent semantic analysis (LSA) to derive the semantic matrix of a document or a corpus and uses semantic sentence representation to construct a semantic text relationship map. We evaluate LSA + T.R.M. both with single documents and at the corpus level to investigate the competence of LSA in text summarization. The two novel approaches were measured at several compression rates on a data corpus composed of 100 political articles. When the compression rate was 30%, an average f-measure of 49% for MCBA, 52% for MCBA + GA, 44% and 40% for LSA + T.R.M. in single-document and corpus level were achieved respectively.

Keywords: Text summarization; Corpus-based approach; Latent semantic analysis; Text relationship map

1. Introduction

With the advent of the information age, people are beset with unprecedented problems because of the abundance of information. One of these problems is the lack of an efficient and effective method to find the required information. Text search and text summarization are two essential technologies to address this problem. Text search engines serve as information filters to sift out an initial set of relevant documents,

☆ Note: An earlier version of this paper was presented at the ICADL2002 Conference.

* Corresponding author. Tel.: +886-3-5712121x56647; fax: +886-3-5721490.
E-mail addresses: jyyeh@cis.nctu.edu.tw (J.-Y. Yeh), claven@lib.nctu.edu.tw (H.-R. Ke), wpyang@cis.nctu.edu.tw (W.-P. Yang), ihmeng@iii.org.tw (I.-H. Meng).

0306-4573/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2004.04.003

while text summarizers play the role of information spotters to help users spot a final set of desired documents (Gong & Liu, 2001).

In general, automatic text summarization takes a source text (or source texts) as input, extracts the essence of the source(s), and presents a well-formed summary [1] to the user. Mani and Maybury (1999) formally defined automatic text summarization as the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks). The process can be decomposed into three phases: *analysis*, *transformation*, and *synthesis*. The analysis phase analyzes the input text and selects a few salient features. The transformation phase transforms the results of analysis into a summary representation. Finally, the synthesis phase takes the summary representation, and produces an appropriate summary corresponding to users' needs. In the overall process, *compression rate*, which is defined as the ratio between the length of the summary and that of the original, is an important factor that influences the quality of the summary. As the compression rate decreases, the summary will be more concise; however, more information is lost. While the compression rate increases, the summary will be more copious; relatively, more insignificant information is contained. In fact, when the compression rate is 5–30%, the quality of the summary is acceptable (Hahn & Mani, 2000; Kupiec, Pedersen, & Chen, 1995; Mani & Maybury, 1999).

Text summarization had its inception in 1950s. Due to the lack of powerful computers and difficulty in nature language processing (NLP), early work focused on the study of text genres such as sentence position and cue phrase (Edmundson, 1969; Luhn, 1958). From 1970s to early 1980s, artificial intelligence (AI) had been applied (Azzam, Humphreys, & Gaizauskas, 1999; DeJong, 1979; Graesser, 1981; McKeown & Radev, 1995; Schank & Abelson, 1977; Young & Hayes, 1985). The idea is to exploit knowledge representations, for example, frames or templates, to identify conceptual entities from a text and to extract relationships between entities by inference mechanisms. The major drawback is that limitedly-defined frames or templates may lead to incomplete analysis of conceptual entities. Since the early 1990s to present, information retrieval (IR) is employed (Aone, Okurowski, Gorlinsky, & Larsen, 1997; Goldstein, Kantrowitz, Mittal, & Carbonell, 1999; Gong & Liu, 2001; Hovy & Lin, 1997; Kupiec et al., 1995; Mani & Bloedorn, 1999; Salton, Singhal, Mitra, & Buckley, 1997; Teufel & Moens, 1997; Yeh, Ke, & Yang, 2002). Similar to the tasks of IR, text summarization can be regarded as how to find out significant sentences from a document. However, most IR techniques that have been exploited in text summarization focus on symbolic-level analysis, and they do not take into account semantics such as synonymy, polysemy, and term dependency (Hovy & Lin, 1997).

In this paper, we propose two novel methods to achieve automatic text summarization: modified corpus-based approach (MCBA), and LSA-based T.R.M. approach (LSA + T.R.M.). The first is based on a score function combined with the analysis of salient features, and the genetic algorithm (GA) (Russell & Norvig, 1995) is employed to discover suitable combinations of feature weights. The second one exploits latent semantic analysis (LSA) (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990; Landauer, Foltz, & Laham, 1998) and a text relationship map (T.R.M.) (Salton et al., 1997) to derive semantically salient structures from a document. Both approaches concentrate on single-document summarization and generate indicative, [2] extract-based summaries. Table 1 shows the outputs of our approaches in different phases of the text-summarization process.

The remainder of this paper is organized as follows. Section 2 introduces some related studies. Sections 3 and 4 give a detail description of our proposed approaches. Section 5 presents the empirical results achieved by our proposed methods and compared with that of previous work. Finally, Section 6 concludes this paper.

[1] A well-formed summary can be an extract or an abstract. Hovy and Lin (1997) defined an extract as portions extracted from the original, and an abstract as novel phrasings describing the content of the original.

[2] A summary can be in three distinct forms: indicative, informative, and evaluative (Mani & Bloedorn, 1999).

Table 1
Outputs of our approaches in different phases of the text-summarization process

Approaches	Analysis phase	Transformation phase	Synthesis phase
MCBA	Physical features ■ Position ■ Positive keyword ■ Negative keyword ■ Centrality ■ Resemblance to the title	Score function	Extract
LSA + T.R.M.	Word-by-sentence matrix	Semantic matrix and text relationship map	Extract

2. Related work

In recent years, a variety of text summarization methods has been proposed and evaluated. According to the level of text processing, Mani and Maybury (1999) categorized text summarization approaches into *surface*, *entity*, and *discourse* levels. Surface-level approaches represent information as shallow features—term frequency, sentence position, cue word, etc., and combine these features to yield a salience function that measures the significance of information (Kupiec et al., 1995; Lin, 1999; Mayeng & Jang, 1999; Teufel & Moens, 1997; Yeh et al., 2002). Entity-level approaches model text entities and their relationships—co-occurrence, co-reference, etc., and determine salient information based on the text-entity model (Azzam et al., 1999; McKeown & Radev, 1995). Discourse-level approaches model the global structure—document format, rhetorical structure, etc.—of the text and its relation to communicative goals (Barzilay & Elhadad, 1997; Silber & McCoy, 2000).

On the other hand, text summarization can be roughly classified into two categories according to how much domain-knowledge is involved (Hahn & Mani, 2000). *Knowledge-poor* approaches do not consider any knowledge pertaining to the domain to which text summarization is applied; therefore, knowledge-poor approaches can be easily applied to any domain (Abracos & Lopes, 1997; Gong & Liu, 2001; Hovy & Lin, 1997; Kupiec et al., 1995; Lin, 1999; Mayeng & Jang, 1999; Salton et al., 1997; Mitra, Singhal, & Buckley, 1997). The foundation of *knowledge-rich* approaches is the assumption that understanding of the meaning of a text can benefit the generation of a good summary (Aone et al., 1997; Azzam et al., 1999; Barzilay & Elhadad, 1997; Hovy & Lin, 1997; McKeown & Radev, 1995). Knowledge-rich approaches rely on a sizeable knowledge base of rules, which must be acquired, maintained, and then adapted to any domain. In general, surface-level approaches are known as knowledge-poor approaches, while entity-level and discourse-level approaches are known as knowledge-rich approaches.

The format of summaries is another criterion to differentiate text-summarization approaches. Usually, a summary can be an extract or an abstract. In fact, a majority of researches have been focused on summary extraction, which selects salient pieces (keywords, sentences or paragraphs) from the source to yield a summary. Hovy and Lin (1997) further distinguished summaries as indicative vs. informative; generic vs. query-oriented; background vs. just-the-news; single-document vs. multi-document; neutral vs. evaluative.

Lin (1999) exploited a selection function for extraction, and used a machine-learning algorithm to automatically learn a good function to combine several heuristics. Aone et al. (1997), Kupiec et al. (1995), and Mayeng and Jang (1999) regarded the task as a classification problem, and employed Bayesian classifiers to determine which sentence should be included in the summary. Barzilay and Elhadad (1997), and Silber and McCoy (2000) created summaries by finding lexical chains, relying on word distribution and lexical links among them to approximate content, and providing a representation of the lexical cohesive structure of the text. Azzam et al. (1999) used co-reference chains to model the structure of a document and to indicate sentences for inclusion in a summary. Gong and Liu (2001) proposed two methods: one used relevance measure to rank sentence relevance, and the other used latent semantic analysis to identify

semantically important sentences. Hovy and Lin (1997) attempted to create a robust summarization system, based on the equation: *summarization = topic identification + interpretation + generation*. The identification stage is to filter the input and retain the most important topics. In the interpretation stage, two or more extracted topics are fused into one or more unifying concept(s). The generation stage reformulates the extracted and fused concepts and then generates an appropriate summary.

The following subsections briefly describe the two approaches on the basis of which our algorithms are developed.

2.1. Corpus-based approaches

Nowadays, corpus-based approaches play an important role in text summarization (Hovy & Lin, 1997; Kupiec et al., 1995; Lin, 1999; Lin & Hovy, 1997; Teufel & Moens, 1997; Yeh et al., 2002). By exploiting technologies of machine learning, it becomes possible to learn rules from a corpus of documents and their corresponding summaries. In general, most corpus-based methods adopt a weighting model. [3] The process of corpus-based text summarization consists of two phases: *the training phase* and *the test phase*. The training phase extracts salient features from the training corpus and then generates rules by a learning algorithm. The test phase applies the rules learned from the training phase on the test corpus and generates the corresponding summaries. The major advantage is that corpus-based approaches are easy to implement. However, a trainable summarizer cannot guarantee that the summaries are useful, due to its deficiency of coherence and cohesion.

Kupiec et al. (1995) proposed a trainable summarizer based on Bayesian classifiers. For each sentence s, the probability that it belongs to the summary S, given k features F_j, is computed. The probability is expressed as Eq. (1), where $P(F_j|s \in S)$ is the probability that F_j appears in a summary sentence, $P(s \in S)$ is the ratio of the number of summary sentences to the total number of sentences in the original corpus, and $P(F_j)$ is the probability that F_j appears in the training corpus. All of them can be computed from the training corpus.

$$P(s \in S|F_1, F_2, \ldots, F_k) = \frac{\prod_{j=1}^{k} P(F_j|s \in S)P(s \in S)}{\prod_{j=1}^{k} P(F_j)} \tag{1}$$

The features used in their experiments were sentence length, fixed phrase, the position of a sentence in a paragraph, thematic word, and uppercase word. Their results showed that position was the most important individual feature, and the best combination of features was position, fixed phrase, and sentence length.

2.2. Text summarization using a text relationship map

Salton et al. (1997) employed the techniques for inter-document link generation to produce intra-document links between passages of a document, and obtained a *text relationship map* according to the intra-document links. Fig. 1 illustrates an example of the map. Each node on the map stands for a paragraph and is represented by a vector of weighted terms. A link is created between two nodes if they have high similarity, which is typically computed as the inner product between the vectors of the corresponding paragraphs. In other words, if there is a link between two nodes, they are said to be "semantically related".

[3] A weighting model is used to evaluate the weight of the text unit U:

$$Weight(U) = Combination(F_1(U), F_2(U), \ldots, F_K(U))$$

where F_i is a salient feature (Hahn & Mani, 2000).

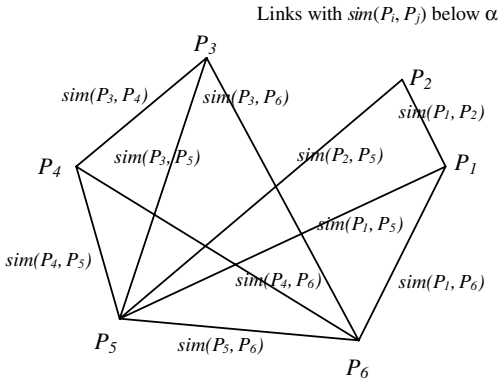

Fig. 1. An example of text relationship map.

Bushiness of a paragraph is defined to measure its significance. The bushiness of a paragraph is the number of links connecting it to other paragraphs. For example, the bushiness of P_5 in Fig. 1 is 5 because it links to P_1, P_2, P_3, P_4, and P_6. A highly bushy node links to many other nodes; in other words, it has a lot of overlapping vocabulary with others; therefore, a highly bushy node is likely to discuss main topics that are covered in many other paragraphs.

As for text summarization, they proposed three heuristic methods to generate the summary according to the bushiness of paragraphs: *global bushy path*, *depth-first path*, and *segmented bushy path*. These three heuristic methods all identify paragraphs with high bushiness but traverse them in different text order.

On the basis of the method proposed by Salton et al. (1997), Kim, Kim, and Hwang (2000) measured the importance of a paragraph by aggregate similarity. Instead of counting the number of links connecting a node (paragraph) to other nodes, aggregate similarity sums the weights on the links. They found that the performance of aggregate similarity surpassed that of bushy paths in the domain of technique articles, but was close to that of bushy paths in the domain of news. The attractiveness of aggregate similarity is that it is easy to adapt to new applications since it does not have to set link-threshold parameters.

3. Modified corpus-based approach

In this section, we propose a novel trainable summarizer, which takes into account several kinds of document features, including *position*, *positive keyword*, *negative keyword*, *centrality*, and *the resemblance to the title*, to generate summaries. Two new ideas are employed to improve conventional corpus-based text summarization. First, sentence positions are ranked to emphasize the significances of different sentence positions; second, the score function is trained by the genetic algorithm (GA) (Russell & Norvig, 1995) to obtain a suitable combination of feature weights.

3.1. Feature extraction

For a sentence s, the belief or score indicating the degree that it belongs to the summary is calculated given the following features.

f_1: *Position*—important sentences, which should be included in the summary, are usually located at some particular positions. For example, the first sentence in each paragraph always introduces the main topics that the paragraph describes, and the last sentence always summarizes what the paragraph discusses. Moreover, it is believed that even for two sentences both in the summary, their significances are different because of their

positions. To emphasize the significances of different sentence positions, each sentence in the summaries of the training corpus is given a rank ranging from 1 to R (in our implementation, R is 5), where a smaller ranking number implies less importance. The position score of a sentence s is defined as Eq. (2), where s comes from position PiSj. For example, "P1S1" indicates that s is the first sentence of the first paragraph.

$$Score_{f_1}(s) = P(s \in S|PiSj) \times \frac{AvgRank(PiSj)}{R} \tag{2}$$

f_2: *Positive keyword*—since words are the basic elements of a sentence, the more content-bearing keywords a sentence has, the more important it is. Hence, positive keywords are defined as the keywords frequently included in the summary. Suppose that a sentence s contains n different keywords, $Keyword_1$, $Keyword_2, \ldots, Keyword_n$, then the positive-keyword score of s is defined as Eq. (3), where tf_i is the occurrence frequency of $Keyword_i$ in s.

$$Score_{f_2}(s) = \frac{1}{length(s)} \sum_{i=1}^{n} tf_i \times P(s \in S|Keyword_i) \tag{3}$$

In Eq. (3), to avoid the bias of the sentence length, [4] this score is normalized by the length of s, which is the number of keywords in s.

f_3: *Negative keyword*—in contrast to f_2, negative keywords are the keywords that are unlikely to occur in the summary. Suppose that a sentence s contains n different keywords, $Keyword_1, Keyword_2, \ldots, Keyword_n$, then the negative-keyword score of s is defined as Eq. (4), where tf_i indicates the occurrence frequency of $Keyword_i$ in s.

$$Score_{f_3}(s) = \frac{1}{length(s)} \sum_{i=1}^{n} tf_i \times P(s \notin S|Keyword_i) \tag{4}$$

Similar to the positive-keyword score, this score is normalized by the length of s to avoid the bias of the sentence length.

f_4: *Centrality*—the centrality of a sentence implies its similarity to others, which can be measured as the degree of vocabulary overlapping between it and other sentences. In other words, if a sentence contains more concepts identical to those of other sentences, it is more significant. Generally, the sentence with the highest centrality denotes the centroid of the document. For a sentence s, this score is defined as Eq. (5).

$$Score_{f_4}(s) = \frac{|keywords\ in\ s\ \cap\ keywords\ in\ other\ sentences|}{|keywords\ in\ s\ \cup\ keywords\ in\ other\ sentences|} \tag{5}$$

f_5: *Resemblance to the title* – it is no doubt that the title always sums up main themes of the document. Hence, the more overlapping keywords a sentence has with the title, the more important it is. For a sentence s, this score is defined as Eq. (6).

$$Score_{f_5}(s) = \frac{|keywords\ in\ s\ \cap\ keywords\ in\ the\ title|}{|keywords\ in\ s\ \cup\ keywords\ in\ the\ title|} \tag{6}$$

In our implementation for those keyword-based features (f_2, f_3, f_4, and f_5), a dictionary is used to identify keywords. The major drawback of dictionary look-up is that it is easy to misidentify keywords because of new keywords that are not in the dictionary. To ameliorate this shortcoming, *mutual information* as shown in Eq. (7) (Maosong, Dayang, & Tsou, 1998) is computed for adjacent keywords to determine if they can constitute a new keyword.

[4] A longer sentence may contain more content-bearing keywords than a shorter sentence. Therefore, without normalization, a longer sentence may have a larger feature score.

$$MI(x,y) = \log \frac{P(x,y)}{P(x)P(y)} \tag{7}$$

In Eq. (7), x, y are keywords, $P(x)$ is the probability that x occurs in the corpus, and $P(x,y)$ is the probability that x and y occurs adjacently in the corpus. We consider those new keywords when computing scores of keyword-based features, in order to alleviate the impacts of keyword misidentification.

3.2. Summary generation

For a sentence s, a weighted score function, as shown in Eq. (8), is employed to integrate all the feature scores mentioned above, where w_i indicates the weight of f_i.

$$Score(s) = w_1 \times Score_{f_1}(s) + w_2 \times Score_{f_2}(s) + w_3 \times Score_{f_3}(s) + w_4 \times Score_{f_4}(s) + w_5 \times Score_{f_5}(s) \tag{8}$$

As to the selection of sentences to generate a summary, all sentences are ranked according to their scores calculated from Eq. (8), and a designated number of the top-score sentences are picked out to form the summary.

Moreover, the genetic algorithm (GA) is exploited to obtain an appropriate set of feature weights. A chromosome is represented as the combination of all feature weights in the form as $(w_1, w_2, w_3, w_4, w_5)$. To measure the effectiveness of each genome, we define fitness as the average F-measure [5] obtained with the genome when the summarization process is applied on the training corpus. When performing the genetic algorithm, we produce 1000 genomes for each generation, evaluate fitness of each genome, and retain the fittest 10 genomes to mate for new ones in the next generation. In our experiments, 100 generations are evaluated to obtain steady combinations of feature weights.

By applying GA, a suitable combination of feature weights could be found. It cannot be guaranteed that the score function whose feature weights are obtained by GA definitely performs well for the test corpus; nevertheless, if the genre of the test corpus is close to that of the training corpus, we can make a prediction that the score function will work well. In other words, the performance of such a score function depends on the similarity of the genre of the test corpus and that of the training corpus.

4. LSA + T.R.M. approach

Latent semantic analysis (LSA) is a mathematical technique for extracting and inferring relations of expected contextual usage of words in passages of discourse (Deerwester et al., 1990; Landauer et al., 1998). LSA is employed in this paper to derive latent structures from a document. In this section, the method used to derive semantic representation by LSA is elaborated, and a method to generate the summary according to semantic representation is proposed.

4.1. Process of LSA + T.R.M.

The overall process shown in Fig. 2 consists of four phases: (1) preprocessing, (2) semantic model analysis, (3) text relationship map construction, and (4) sentence selection.

Preprocessing delimits each sentence by punctuation. Furthermore, it segments each sentence into keywords with a toolkit, named AutoTag (Academia Sinica, 1999). *Semantic model analysis* represents the

[5] F-measure is defined as $F = 2PR/(P+R)$, where P and R are precision and recall respectively (Baeza-Yates & Ribeiro-Neto, 1999).

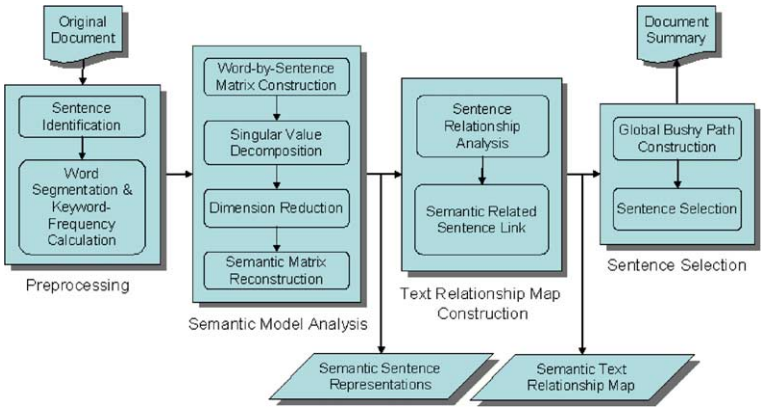

Fig. 2. Overall process of our LSA + T.R.M. approach.

input document (LSA in single-document level) or the whole corpus (LSA in corpus level) as a word-by-sentence matrix and reconstructs the corresponding semantic matrix via singular value decomposition (SVD) and dimension reduction. *Text relationship map construction* creates the text relationship map based on semantic sentence representation derived from the semantic matrix. Finally, *sentence selection* establishes a global bushy path (Salton et al., 1997) according to the map and selects important sentences to compose a summary.

4.2. Semantic sentence/word representation

To analyze the impacts of expected contextual usage of words in different levels, we construct two semantic matrices, one for single-document level, and the other for corpus level. On one hand, single-document level considers words and sentences in the same document to construct the semantic matrix; on the other hand, corpus level considers words and sentences in the whole corpus to model the word-by-sentence matrix.

The following elucidates how to construct the word-by-sentence matrix for the single-document level. Let D be a document, $W(|W| = M)$ be the set of keywords in D, and S $(|S| = N)$ be the set of sentences in D. A word-by-sentence matrix, [6] A, is constructed as Eq. (9), where S_i indicates a sentence and W_i indicates a keyword. In our work, only nouns and verbs are taken into account in that they carry essential information about the meaning of a sentence.

$$A = \begin{array}{c|cccc} & S_1 & S_2 & \cdots & S_N \\ \hline W_1 & a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ W_2 & a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_M & a_{M,1} & a_{M,2} & \cdots & a_{M,N} \end{array} \tag{9}$$

In A, $a_{i,j}$ is defined as Eq. (10), where L_{ij} is the local weight of W_i in S_j, and G_i is the global weight of W_i in D. L_{ij} is defined as $L_{ij} = \log\left(1 + \frac{c_{ij}}{n_j}\right)$, and G_i is defined as $G_i = 1 - E_i$, where c_{ij} is the frequency of W_i occurring in S_j, n_j is the number of words in S_j, and E_i is the normalized entropy of W_i, which is defined as $E_i = -\frac{1}{\log(N)} \sum_{j=1}^{N} f_{ij} \times \log(f_{ij})$ (Bellegarda, Butzberger, Chow, Coccaro, & Naik, 1996).

[6] In the matrix, without loss of generality, we assume that $|W| \geqslant |S|$.

$$a_{i,j} = G_i \times L_{ij} \tag{10}$$

We then perform singular value decomposition (SVD) to A. The SVD of A is defined as $A = UZV^{\mathrm{T}}$, where U is an $M \times N$ matrix of left singular vectors, Z is an $N \times N$ diagonal matrix of singular values, and V is an $N \times N$ matrix of right singular vectors.

Finally, the process of *dimension reduction* is applied to Z by deleting a few entries in it, and the result of dimension reduction is a matrix Z'. A new matrix, A', is reconstructed by multiplying three component matrixes. A' is defined as Eq. (11), where Z' is the semantic space that derives latent semantic structures from A, $U' = [u'_i]$ is a matrix of left singular vectors whose ith vector u'_i represents W_i in Z', and $V' = [v'_j]$ is a matrix of right singular vectors whose jth vector v'_j represents S_j in Z'.

$$A' = U'Z'V'^{\mathrm{T}} \approx A \tag{11}$$

Each column of A' denotes the semantic sentence representation, and each row denotes the semantic word representation. We use the semantic sentence representations for summary generation.

As to constructing the word-by-sentence matrix for corpus level, D represents all documents in the corpus, W ($|W| = M$) is the set of keywords in the corpus, and S ($|S| = N$) is the set of sentences in the corpus. Furthermore, the value of G_i for $a_{i,j}$ in Eq. (10) is the global weight of W_i in the whole corpus.

4.3. Summary generation

Salton et al. (1997) proposed a text relationship map to represent the structure of a document and generated the summary according to the map. One problem of their map is the lack of the type or the context of a link. To take into account the context of a link, we integrate the map and the above-mentioned semantic sentence representation to promote text summarization from keyword-level analysis to semantic-level analysis.

In our method, a sentence S_i is represented by the corresponding semantic sentence representation (Section 4.2), instead of the original keyword-based frequency vector. The similarity between a pair of sentences S_i and S_j is evaluated to determine if they are semantic ally related. The similarity is defined as Eq. (12).

$$sim(S_i, S_j) = \frac{\vec{S}_i \cdot \vec{S}_j}{|\vec{S}_i||\vec{S}_j|} \tag{12}$$

We also use a *map density* parameter (Salton et al., 1997) to decide whether a link should be considered as a valid semantic link. In our implementation, we set the map density parameter as $1.5 \times n$ (i.e., retain the $1.5 \times n$ best links on the map, where n is the number of sentences).

The significance of a sentence is measured by counting the number of links that it has. A global bushy path (Salton et al., 1997) is established by arranging the k bushiest sentences in the order that they appear in the original document. Finally, a designated number of sentences are selected from the global bushy path to generate a summary.

5. Evaluation

In this section, we report our experimental results.

5.1. Data corpus

One hundred articles in the domain of politics were collected from New Taiwan Weekly (New Taiwan Inc., 2003) and were partitioned into five sub-collections, named Set 1, Set 2, ..., and Set 5.

Table 2
Statistics of the data corpus

Document statistics	Set 1	Set 2	Set 3	Set 4	Set 5
Documents per collection	20	20	20	20	20
Sentences per document	27.5	27.9	30.8	25.5	27.4
Sentences per manual summary	8.7	9.0	10.0	8.0	8.7
Manual compression rate per document	30%	30%	30%	30%	30%
Average rank of manual summary sentence	4.1	3.8	4.1	3.8	4.0

The summaries of the articles were created by three independent human annotators. Each sentence included in the summaries was annotated with a rank to indicate its significance. To create a final set of summaries, each sentence received a maximum rank among the three annotated ranks. Then, the top 30% of sentences for each document were selected as its summary. Table 2 shows the statistics of the data corpus.

5.2. Evaluation methods

There are two sorts of methods to evaluate the performance of text summarization: *extrinsic evaluation* and *intrinsic evaluation* (Mani & Bloedorn, 1999; Mani & Maybury, 1999). Extrinsic evaluation judges the quality of a summary based on how it affects other task(s), and intrinsic evaluation judges the quality of a summary based on the coverage between it and the manual summary. We chose intrinsic evaluation and used recall (R), precision (P) and F-measure (F) to judge the coverage between the manual and the machine-generated summaries. Assume that T is the manual summary and S is the machine-generated summary, the measurements are defined as Eq. (13) (Baeza-Yates & Ribeiro-Neto, 1999).

$$P = \frac{|S \cap T|}{|S|}, \quad R = \frac{|S \cap T|}{|T|}, \quad F = \frac{2PR}{R + P} \tag{13}$$

5.3. Modified corpus-based approach

First of all, we explain the symbols used in the following evaluation results.

- *CR*: compression rate,
- *POS*: position (f_1),
- *PK*: positive keyword (f_2),
- *NK*: negative keyword (f_3),
- *C*: centrality (f_4),
- *TR*: resemblance to the title (f_5),
- *CBA*: original corpus-based approach,
- *MCBA*: modified corpus-based approach,
- *MCBA + GA*: MCBA with GA to find a suitable score function.

We measured the performance of our first approach in the way called k-fold cross-validation (Han & Kamber, 2001). In our experiments, training and testing was performed five times (i.e., $k = 5$). In iteration i, Set i was selected as the test corpus, and the other sets are collectively used to train the values for each feature. Table 3 shows the effects of different features. Consider POS first; it can be observed that MCBA

Table 3
Influence of each feature between CBA and MCBA when considering F-measure

F-measure	CR = 10%		CR = 20%		CR = 30%	
	CBA	MCBA	CBA	MCBA	CBA	MCBA
POS	0.2282	0.2327	0.2988	0.3092	0.3563	0.3671
PK	0.1785	0.1687	0.2813	0.2919	0.3547	0.3548
NK	0.1439	0.1293	0.2468	0.2259	0.3307	0.3031
C	0.3039	0.2971	0.3913	0.3935	0.4478	0.4476
TR	0.2345	0.2515	0.3313	0.3382	0.3920	0.4056

In this table, we only list the average performance for each feature.

Table 4
Performance evaluation (F-measure) compared between CBA and MCBA when considering different heuristic functions

F-measure	CR = 10%		CR = 20%		CR = 30%	
	CBA	MCBA	CBA	MCBA	CBA	MCBA
All features	0.3046	0.3029	0.4122	0.4136	0.4688	0.4839
Without POS	0.3051 (+)	0.3109 (+)	0.4003 (−)	0.4129 (−)	0.4686 (−)	0.4824 (−)
Without PK	0.3071 (+)	0.3025 (−)	0.4048 (−)	0.3989 (−)	0.4583 (−)	0.4619 (−)
Without NK	0.3083 (+)	0.3166 (+)	0.4045 (−)	0.4294 (+)	0.4736 (+)	0.4908 (+)
Without C	0.2585 (−)	0.2234 (−)	0.3346 (−)	0.3297 (−)	0.4029 (−)	0.4024 (−)
Without TR	0.2745 (−)	0.2974 (−)	0.3829 (−)	0.3773 (−)	0.4418 (−)	0.4599 (−)

In this table, "+" means the result outperforms that of the heuristic function when all features are considered, and "−" means the opposite.

outperforms CBA about 2.0%, 3.5%, and 3.0% [7] when CR is 10%, 20%, and 30%, respectively. The result justifies our assumption that the significance of a sentence depends on its position in the text. With regard to PK, MCBA outperforms CBA when CR is 20% and 30%. The result shows that the sentence length may bias the score, and make the summarizer misjudge the importance of sentences. Furthermore, for C, and TR, the result reveals that considering new keywords when computing scores of keyword-based features may influence the importance of keywords, because both C and TR in CBA do not consider new keywords.

Table 4 gives the performance results when different heuristic functions are considered. For example, "All features" indicates that POS, PK, NK, C, and TR are all considered, while "Without POS" means all but the POS are used. In this experiment, the weight of each feature is set to be 1. The result shows that C and TR are two essential features. Without C and TR, the performance deteriorates severely. For example, when CR is 30%, C declines about 14.1% for CBA and 16.8% for MCBA, while TR declined about 5.8% for CBA and 5.0% for MCBA. On the other hand, the result indicates that POS and PK are two less important features, because the performance tends to worsen without them, but the differences are not great. We conjecture that POS and PK are neutral features for the test corpus since C and TR dominate the performance. In addition, NK is not a good feature either for CBA or MCBA because without NK, the overall performance improves. Therefore, we conclude that POS + PK + C + TR is the best combination of features. Tables 5–7 show the performance when POS, PK, C, and TR are considered. It can be seen that F-measure of MCBA outperforms that of CBA about 2.7%, 6.2%, and 3.6%, when CR is 10%, 20%, and 30% respectively.

Table 8 lists feature weights obtained by GA when CR is 30%. In this table, we list as well the fitness (F-measure) when the feature weights are applied to the training corpus. Tables 9–11 show the performance

[7] Hereafter, we use relative improvement for comparison. Here, relative improvement is calculated as $100 \times (MCBA - CBA)/CBA$.

Table 5
Performance evaluation (recall) compared between CBA and MCBA when not considering NK, and the weight of each feature is set as 1

POS + PK + C + TR (recall)	CR = 10%		CR = 20%		CR = 30%	
	CBA	MCBA	CBA	MCBA	CBA	MCBA
Set 1	0.1893	0.2173	0.3636	0.3732	0.5264	0.5330
Set 2	0.1775	0.1825	0.3256	0.3683	0.4456	0.4989
Set 3	0.2347	0.2272	0.3105	0.3241	0.4578	0.4667
Set 4	0.2119	0.2029	0.3114	0.3204	0.4143	0.4210
Set 5	0.2388	0.2506	0.3789	0.4120	0.5237	0.5342
Avg.	0.2104	0.2161	0.3380	0.3596	0.4736	0.4908

Table 6
Performance evaluation (precision) compared between CBA and MCBA when not considering NK, and the weight of each feature is set as 1

POS + PK + C + TR (precision)	CR = 10%		CR = 20%		CR = 30%	
	CBA	MCBA	CBA	MCBA	CBA	MCBA
Set 1	0.5292	0.6083	0.5319	0.5474	0.5264	0.5330
Set 2	0.5108	0.5208	0.4928	0.5570	0.4456	0.4989
Set 3	0.6308	0.6058	0.4720	0.4735	0.4578	0.4667
Set 4	0.5750	0.5500	0.4586	0.4727	0.4143	0.4210
Set 5	0.6367	0.6742	0.5631	0.6132	0.5237	0.5342
Avg.	0.5765	0.5918	0.5037	0.5328	0.4736	0.4908

Table 7
Performance evaluation (F-measure) compared between CBA and MCBA when not considering NK, and the weight of each feature is set as 1

POS + PK + C + TR (F-measure)	CR = 10%		CR = 20%		CR = 30%	
	CBA	MCBA	CBA	MCBA	CBA	MCBA
Set 1	0.2789	0.3202	0.4319	0.4438	0.5264	0.5330
Set 2	0.2635	0.2703	0.3921	0.4434	0.4456	0.4989
Set 3	0.3421	0.3305	0.3746	0.3848	0.4578	0.4667
Set 4	0.3097	0.2964	0.3709	0.3819	0.4143	0.4210
Set 5	0.3473	0.3654	0.4530	0.4929	0.5237	0.5342
Avg.	0.3083	0.3166	0.4045	0.4294	0.4736	0.4908

when the feature weights in Table 8 are applied to the test corpus (i.e., MCBA + GA). It can be observed that when F-measure is considered, on average, MCBA + GA outperforms MCBA about 12.3%, 9.6%, and 5.0% [8] when CR is 10%, 20%, and 30% respectively. This illustrates the benefits of employing GA in training. The primary advantage of GA lies in providing a preliminary analysis of the corpus and provides a referral to tune the score function. However, GA does not guarantee that the obtained score function always performs well for the test corpus. For example, when CR is 20%, the performance of Set 2 declines about 0.4%, and when CR is 30%, the performance of Set 5 declines about 1.0% (Table 11). We conclude

[8] The relative improvement is calculated as $100 \times (MCBA + GA - MCBA)/MCBA$.

Table 8
Features weights obtained by GA when CR = 30%

CR = 30%	POS	PK	NK	C	TR	Fitness (F-measure)
Combination 1	0.140	0.857	0.000	0.281	0.278	0.9264
Combination 2	0.225	0.995	0.000	0.299	0.336	0.9358
Combination 3	0.066	0.909	0.000	0.196	0.169	0.9626
Combination 4	0.173	0.961	0.000	0.333	0.039	0.9554
Combination 5	0.172	0.794	0.000	0.248	0.085	0.9587

In the training phase, we just consider the best combination (i.e., POS + PK + C + TR).

Table 9
Performance evaluation (recall) compared between MCBA and MCBA + GA when NK is not considered, and the weight of each feature is set as that in Table 8

POS + PK + C + TR (recall)	CR = 10%		CR = 20%		CR = 30%	
	MCBA	MCBA + GA	MCBA	MCBA + GA	MCBA	MCBA + GA
Set 1	0.2173	0.2487	0.3732	0.4153	0.5330	0.5446
Set 2	0.1825	0.2101	0.3683	0.3667	0.4989	0.5244
Set 3	0.2272	0.2569	0.3241	0.4031	0.4667	0.5108
Set 4	0.2029	0.2463	0.3204	0.3562	0.4210	0.4668
Set 5	0.2506	0.2535	0.4120	0.4304	0.5342	0.5289
Avg.	0.2161	0.2431	0.3596	0.3943	0.4908	0.5151

Table 10
Performance evaluation (precision) compared between MCBA and MCBA + GA when NK is not considered, and the weight for each feature is set as that in Table 8

POS + PK + C + TR (precision)	CR = 10%		CR = 20%		CR = 30%	
	MCBA	MCBA + GA	MCBA	MCBA + GA	MCBA	MCBA + GA
Set 1	0.6083	0.6875	0.5474	0.6081	0.5330	0.5446
Set 2	0.5208	0.5975	0.5570	0.5556	0.4989	0.5244
Set 3	0.6058	0.6825	0.4735	0.5893	0.4667	0.5108
Set 4	0.5500	0.6750	0.4727	0.5233	0.4210	0.4668
Set 5	0.6742	0.6725	0.6132	0.6395	0.5342	0.5289
Avg.	0.5918	0.6630	0.5328	0.5832	0.4908	0.5151

Table 11
Performance evaluation (F-measure) compared between MCBA and MCBA + GA when NK is not considered, and the weight of each feature is set as that in Table 8

POS + PK + C + TR (F-measure)	CR = 10%		CR = 20%		CR = 30%	
	MCBA	MCBA + GA	MCBA	MCBA + GA	MCBA	MCBA + GA
Set 1	0.3202	0.3653	0.4438	0.4935	0.5330	0.5446
Set 2	0.2703	0.3109	0.4434	0.4418	0.4989	0.5244
Set 3	0.3305	0.3733	0.3848	0.4787	0.4667	0.5108
Set 4	0.2964	0.3609	0.3819	0.4239	0.4210	0.4668
Set 5	0.3654	0.3682	0.4929	0.5145	0.5342	0.5289
Avg.	0.3166	0.3557	0.4294	0.4705	0.4908	0.5151

that the performance of the GA-trained score function depends on the similarity of the genre of the test corpus and that of the training corpus. In spite of this, GA does provide a way to address the situation when we are indecisive about a good combination of feature weights.

5.4. LSA + T.R.M. approach when considering in single-document level

In this experiment, the feasibility of applying LSA to text summarization is evaluated. Tables 12–14 show the performance of our approach. For different test corpuses, the dimension reduction ratios (DR) differ; for example, when CR is 10%, the best DR for Set 1 is 0.7 (i.e., if the rank of the singular value matrix

Table 12
Performance evaluation (recall) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (recall)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.2088 (0.7)	0.1447	0.3365 (0.7)	0.2908	0.4400 (0.7)	0.3818
Set 2	0.1977 (0.8)	0.1650	0.3211 (0.8)	0.2728	0.4297 (0.7)	0.3834
Set 3	0.1853 (0.9)	0.1530	0.3014 (0.7)	0.2753	0.4272 (0.8)	0.3399
Set 4	0.1887 (0.7)	0.1324	0.3117 (0.4)	0.2241	0.4311 (0.4)	0.3296
Set 5	0.2016 (0.8)	0.1688	0.3371 (0.8)	0.2902	0.4932 (0.6)	0.4085
Avg.	0.1964 (0.8)	0.1522	0.3215 (0.7)	0.2707	0.4442 (0.6)	0.3686

Table 13
Performance evaluation (precision) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (precision)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.5875 (0.7)	0.4042	0.4984 (0.7)	0.4313	0.4400 (0.7)	0.3818
Set 2	0.5608 (0.8)	0.4650	0.4837 (0.8)	0.4104	0.4297 (0.7)	0.3834
Set 3	0.4942 (0.9)	0.3992	0.4407 (0.7)	0.4030	0.4272 (0.8)	0.3399
Set 4	0.5042 (0.7)	0.3542	0.4611 (0.4)	0.3285	0.4311 (0.4)	0.3296
Set 5	0.5383 (0.8)	0.4542	0.4972 (0.8)	0.4282	0.4932 (0.6)	0.4085
Avg.	0.5370 (0.8)	0.4153	0.4762 (0.7)	0.4003	0.4442 (0.6)	0.3686

Table 14
Performance evaluation (F-measure) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (F-measure)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.3076 (0.7)	0.2127	0.4016 (0.7)	0.3472	0.4400 (0.7)	0.3818
Set 2	0.2920 (0.8)	0.2433	0.3858 (0.8)	0.3276	0.4297 (0.7)	0.3834
Set 3	0.2689 (0.9)	0.2180	0.3578 (0.7)	0.3270	0.4272 (0.8)	0.3399
Set 4	0.2740 (0.7)	0.1922	0.3718 (0.4)	0.2663	0.4311 (0.4)	0.3296
Set 5	0.2928 (0.8)	0.2457	0.4013 (0.8)	0.3456	0.4932 (0.6)	0.4085
Avg.	0.2870 (0.8)	0.2224	0.3837 (0.7)	0.3227	0.4442 (0.6)	0.3686

Table 15
Influence of different DR's ranging from 0.1 to 0.9 when F-measure is considered, and CR is 30%

F-measure (CR = 30%)	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Set 1	0.2901	0.3500	0.4304	0.3430	0.3859	0.4282	0.4400	0.4268	0.3859
Set 2	0.3451	0.4095	0.4079	0.3812	0.3955	0.3860	0.4297	0.4190	0.3750
Set 3	0.3407	0.3557	0.3311	0.3360	0.3973	0.3985	0.4107	0.4272	0.3593
Set 4	0.3629	0.3179	0.3672	0.4311	0.3674	0.3826	0.4132	0.4167	0.3284
Set 5	0.2779	0.3823	0.3547	0.4569	0.4744	0.4932	0.4614	0.4288	0.4149
Avg.	0.3059	0.3446	0.3989	0.4431	0.4420	0.4584	0.4583	0.4379	0.4090

Table 16
The upper-bound performance (F-measure) of LSA + T.R.M. when consider the best DR for each document

LSA + T.R.M. (F-measure)	CR = 10%	CR = 20%	CR = 30%
Set 1	0.4245	0.5373	0.5773
Set 2	0.4192	0.4944	0.5315
Set 3	0.4183	0.4925	0.5169
Set 4	0.4223	0.4841	0.5228
Set 5	0.4394	0.5114	0.5901
Avg.	0.4247	0.5039	0.5477

is n, then only the first $0.7 \times n$ of diagonal entries are kept for semantic matrix reconstruction). When the compression rate is 10%, 20%, and 30%, the average DR is about 0.8, 0.7, and 0.6 respectively. Regarding F-measure on average LSA + T.R.M. outperforms Keyword + T.R.M. (Salton et al., 1997) about 29.0%, 18.9%, and 20.5% [9] when CR is 10%, 20%, and 30% respectively.

Table 15 shows the impact of different DR's when CR is 30%. The result indicates that when DR ranges from 0.6 to 0.8, LSA + T.R.M. achieves better performance. Similarly, when CR is 10% and 20%, LSA + T.R.M. achieves better performance as well when DR ranges from 0.6 to 0.8. Table 16 shows the performance of different sub-collections when the best DR is considered for each document. The result reveals that when CR is 10%, 20%, and 30%, we get an F-measure of 0.4247, 0.5039, and 0.5477 respectively. This indicates an upper-bound performance of our LSA + T.R.M. method. To sum up, with the appropriate DR, LSA + T.R.M. works well. Thus, we conclude that LSA can be employed to promote text summarization from keyword-level analysis to semantic-level analysis.

The effect of LSA in text summarization is illustrated with an example shown in Table 17. The precedent 1 means that the following sentence belongs to the manual summary, the precedent 2 means a summary sentence created by Keyword + T.R.M., and precedent 3 means a summary sentence created by LSA + T.R.M. Table 18 shows the text relationship maps created by Keyword + T.R.M. and LSA + T.R.M. respectively.

In this example, when CR is 30%, LSA + T.R.M. achieves F-measure of 0.8, and Keyword + T.R.M. achieves F-measure of zero. Probing into the cause of the result, we find that LSA + T.R.M. sorts all sentences in the order of {P9S1, P1S1, P8S3, P8S2, P4S2, P3S1, P6S1, P10S1, P3S2, P11S2, P11S1, P8S1, P7S1, P5S1, P4S1, P2S1}, while Keyword + T.R.M. sorts all sentences in the order of {P3S1, P7S1, P5S1, P3S2, P10S1, P8S3, P2S1, P11S2, P9S1, P8S2, P4S2, P4S1, P1S1, P8S1, P6S1, P11S1}. LSA + T.R.M. has four overlapping sentences with the manual summary in the top 30% sentences, but Keyword + T.R.M. has

[9] The relative improvement is calculated as $100 \times (LSA + T.R.M. - Keyword + T.R.M.)/Keyword + T.R.M.$

Table 17
An example article from New Taiwan Weekly (New Taiwan Inc., 2003) that has 16 sentences in total

Title	只有時間能解決問題？軍公教退休金優惠利率，財政噩夢。
Content	[1,3]<P1S1>「62、83、116、157、206」這不是新一期的樂透獎開獎號碼，遞增的數字代表的是從民國八十一年以來，每隔三四年政府對軍公教優惠利率以億為單位的補貼金額，其中二百零六億是九十一年度所編列的預算。 <P2S1>相較之下，政府九十年度對身心障礙者和低收入戶補助的社會福利預算，不過也才一百五十一億元，所以新任的考試委員李慶雄近日對外發言表示，就任後將會提案要求重新檢討關於軍公教退休金百分之十八的優惠利率。 [2]<P3S1>軍公教的優惠利息補貼政策，是在民國四十九年時政府因考量到公務人員待遇過於微薄，所以以一紙行政命令讓退休的公務人員享有年息百分之十八的優惠利率，等於是一種月退休金的補助制度。[2]<P3S2>不過經過數十年的待遇調整，現在的軍公教人員的待遇有時比任職於民間企業還好，當時政府的德政成了現在執政者沉重的包袱，部份縣市甚至從八十三年起就開始付不出補貼優惠利率的費用，欠款則全由台灣銀行代墊，預估今年台銀就會替地方政府墊上約四百億的欠款。 <P4S1>其實在退輔新制實施後，已經限定必須要是八十四年六月三十日前的年資，才能適用百分之十八的優惠利率。[3]<P4S2>可是隨著政府為刺激景氣走向寬鬆的貨幣政策，使得國內存款利率不斷走低，結果造成十年內政府對軍公教優惠存款利息補貼約成長二點五倍，並以每年十億元速度持續增長著。 [2]<P5S1>此外根據人事行政局的統計，目前共有卅三萬多的軍公教退休人員適用優惠利率，若再加上往後陸續退休的軍公教人員八十四年前的年資，極有可能要到民國九十九年，政府負擔的優惠利率才會逐年降低。 [1]<P6S1>雖然軍公教優惠利率的問題已經有愈來愈多人重視，但對於這樣的議題，不論新舊政府誰都不敢貿然觸及，行政院和考試院過去均曾評估取消的可行性，但在考量到可能違背「信賴保護原則」下，最後都不了了之。 [2]<P7S1>主計長林全也對外直言，只能透過時間才能解決這樣的爭議，銓敘部也在近日發佈新聞稿強調，在八十四年後已有所謂的斷源措施，政府不會失信於數十萬退休的公務人員。 <P8S1>但這些說法看在新任的考試委員李慶雄眼裏，只能算是推諉之詞。[1,3]<P8S2>他指出，許多公務人員退休後事實上仍在民間兼職，特別像是軍人常常是四五十歲就可以吃終生俸，當初為照顧公務人員而制定的優惠利率政策已失去了原意。[13]<P8S3>他覺得基於公平正義的原則，在優惠利率上應該加上年齡的限制，也應該限制退休金超出二百萬的人不具適用資格，不然像一些勞工恐怕辛苦工作了一輩子還不見得有退休金，軍公教人員實在享有太多的保障。 [1,3]<P9S1>但是也有學者認為，政府的誠信還是應該擺第一，前政大公共行政學系教授薄慶玖就在一場公聽會中強調「奉足以養廉」，他認為如果貿然取消優惠利率，只會讓公務人員在職位上拿不該拿的錢。 [2]<P10S1>薄慶玖說，現在高普考錄取率只有百分之二，代表在文官體制上是成功的，若讓公務人員感到政府說話都不算話，搞不好以後都沒有人要來政府作事。 <P11S1>軍公教優惠利率的問題已逐漸受到大眾的注意，不論是在全國財政會議或是全國社福會議上都是重要的課題。<P11S2>年年增加的財政負擔是否只能靜待時間解決，還是可以像取消軍公教免稅一樣，以中央概括承受的方式一次解決歷史所留下的包袱，仍有待執政者的智慧來權衡。

no overlapping sentences with the manual summary. Hence, LSA + T.R.M. gets a better result. This phenomenon explains that LSA can derive more precise semantic meanings from a text.

5.5. LSA + T.R.M. when considering in corpus level

In contrast to the previous experiments that consider LSA + T.R.M. in single-document level (Section 5.4), in this experiment, we apply LSA + T.R.M. in corpus level in order to analyze the impacts of expected contextual usage of words in the whole corpus. Tables 19–21 show the evaluation results. When CR is 10%, 20%, and 30%, the average DR is about 0.5, 0.5, and 0.4 respectively. Similar to Tables 12–14, regarding *F*-measure, our LSA + T.R.M. outperforms Keyword + T.R.M. about 19.6%, 16.7%, and 13.4% when CR is 10%, 20%, and 30% respectively. However, the scale of relative improvements does not surpass that of LSA + T.R.M. in single-document level (Table 14, Table 21).

Table 22 shows the performance when the best DR is considered for each document. When CR is 10%, 20%, and 30%, we get an *F*-measure of 0.3064, 0.4047, and 0.4532 respectively. This indicates an upper-bound performance of our LSA + T.R.M. method in corpus-level. Similarly, the result does not surpass that

Table 18
Text relationship maps created by LSA + T.R.M. and Keyword + T.R.M.

	LSA + T.R.M.		Keyword + T.R.M.	
	Connected sentences	Link	Connected sentences	Link
P1S1	P3S1, P3S2, P4S2, P6S1, P8S2, P8S3, P9S1	7	P3S1, P4S2	2
P2S1		0	P3S1, P7S1, P8S1	3
P3S1	P1S1, P4S2, P6S1, P8S2, P8S3, P9S1	6	P1S1, P2S1, P3S2, P4S2, P5S1, P8S2, P8S3, P9S1, P10S1	9
P3S2	P1S1, P6S1	2	P3S1, P5S1, P7S1, P10S1, P11S2	5
P4S1		0	P5S1, P7S1	2
P4S2	P1S1, P3S1, P6S1, P8S2, P8S3, P9S1	6	P1S1, P3S1	2
P5S1		0	P3S1, P3S2, P4S1, P7S1, P8S3	5
P6S1	P1S1, P3S1, P3S2, P4S2, P9S1	5	P8S3	1
P7S1		0	P2S1, P3S2, P4S1, P5S1, P8S2, P11S2	6
P8S1		0	P2S1	1
P8S2	P1S1, P3S1, P4S2, P8S3, P9S1, P10S1	6	P3S1, P7S1	2
P8S3	P1S1, P3S1, P4S2, P8S2, P9S1, P10S1	6	P3S1, P5S1, P6S1	3
P9S1	P1S1, P3S1, P4S2, P6S1, P8S2, P8S3, P10S1	7	P3S1, P10S1	2
P10S1	P8S2, P8S3, P9S1	3	P3S1, P3S2, P9S1	3
P11S1		0		0
P11S2		0	P3S2, P7S1	2

Table 19
Performance evaluation (recall) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (recall)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.1671 (0.4)	0.1450	0.2801 (0.5)	0.2456	0.3986 (0.2)	0.3340
Set 2	0.1507 (0.3)	0.1410	0.2757 (0.6)	0.2310	0.4007 (0.3)	0.3394
Set 3	0.1560 (0.6)	0.1275	0.2914 (0.6)	0.2611	0.3956 (0.4)	0.3473
Set 4	0.1832 (0.4)	0.1382	0.2725 (0.4)	0.2236	0.3530 (0.4)	0.3426
Set 5	0.1900 (0.6)	0.1570	0.3224 (0.4)	0.2741	0.4300 (0.6)	0.3806
Avg.	0.1694 (0.5)	0.1417	0.2885 (0.5)	0.2471	0.3956 (0.4)	0.3488

Table 20
Performance evaluation (precision) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (precision)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.4625 (0.4)	0.3958	0.4104 (0.5)	0.3604	0.3986 (0.2)	0.3340
Set 2	0.4342 (0.3)	0.4050	0.4175 (0.6)	0.3472	0.4007 (0.3)	0.3394
Set 3	0.4217 (0.6)	0.3425	0.4250 (0.6)	0.3816	0.3956 (0.4)	0.3473
Set 4	0.4958 (0.4)	0.3750	0.4011 (0.4)	0.3288	0.3530 (0.4)	0.3426
Set 5	0.4992 (0.6)	0.4142	0.4708 (0.4)	0.4049	0.4300 (0.6)	0.3806
Avg.	0.4627 (0.5)	0.3865	0.4250 (0.5)	0.3646	0.3956 (0.4)	0.3488

of LSA + T.R.M. in single-document level. As mentioned in Hofmann (2001), LSA is not able to explicitly capture multiple senses of a word, nor does it take into account that every word occurrence is typically intended to refer to only one meaning at a time. In other words, conceptual representation obtained by LSA

Table 21
Performance evaluation (F-measure) compared with LSA + T.R.M. and Keyword + T.R.M., where the number in parentheses stands for DR

T.R.M. (F-measure)	CR = 10%		CR = 20%		CR = 30%	
	LSA	Keyword	LSA	Keyword	LSA	Keyword
Set 1	0.2450 (0.4)	0.2117	0.3328 (0.5)	0.2920	0.3986 (0.2)	0.3340
Set 2	0.2236 (0.3)	0.2091	0.3320 (0.6)	0.2772	0.4007 (0.3)	0.3394
Set 3	0.2273 (0.6)	0.1855	0.3456 (0.6)	0.3099	0.3956 (0.4)	0.3473
Set 4	0.2669 (0.4)	0.2015	0.3244 (0.4)	0.2661	0.3530 (0.4)	0.3426
Set 5	0.2748 (0.6)	0.2274	0.3823 (0.4)	0.3265	0.4300 (0.6)	0.3806
Avg.	0.2475 (0.5)	0.2070	0.3434 (0.5)	0.2943	0.3956 (0.4)	0.3488

Table 22
The upper-bound performance (F-measure) of LSA + T.R.M. when consider the best DR for each document

LSA + T.R.M. (F-measure)	CR = 10%	CR = 20%	CR = 30%
Set 1	0.3067	0.4026	0.4570
Set 2	0.3225	0.4053	0.4618
Set 3	0.2673	0.4051	0.4393
Set 4	0.3060	0.3728	0.4266
Set 5	0.3294	0.4377	0.4812
Avg.	0.3064	0.4047	0.4532

is unable to handle polysemy. We conclude two causes that may affect the performance of LSA + T.R.M. in corpus level. Consider the polysemy problem first; since our corpus is in the domain of news, there exists some common words with different meanings existing in different documents. Take "運動" for example, which can mean "*exercise*" or "*activation*". Second, the basis of LSA depends on the relations of co-occurrence among words. A proper name, such as a person's name or an organization's name, may occur in different documents that describe different events; therefore, words with different meaning but co-occurring with the proper name, may be bound together in the semantic space even though they do not have any relations with each other. Consequently, the result of LSA in corpus level cannot reflect the true relations of expected contextual usage of words.

5.6. Comparison between MCBA and LSA + T.R.M.

Comparing our two proposed approaches, we find that the performance of LSA + T.R.M. approach in single-document level is close to that of MCBA though it does not surpass it (Tables 5–7, Tables 12–14). The major attractiveness of LSA + T.R.M. in single-document level over MCBA is that it concerns a single document; hence, it is easy to implement since it needs no preprocessing. Moreover, when the best DR for each document is considered, the upper-bound performance of LSA + T.R.M. even exceeds that of MCBA + GA about 19.4%, 7.1%, and 6.3% when CR is 10%, 20%, and 30% respectively (Table 11, Table 16). This responds to the nature of LSA that if we can find the best DR, LSA + T.R.M. performs better than other keyword-based methods. Hence, we conclude that LSA can be employed to promote text summarization from keyword-level analysis to semantic-level analysis. Furthermore, compare LSA + T.R.M. in single-document and LSA + T.R.M. in corpus level (Tables 12–14, Tables 19–21), the average DR of LSA + T.R.M. in corpus level is smaller that of LSA + T.R.M. in single-document level. This out-

come reveals that when the dimension of the word-by-sentence matrix is large, we have to remove more diagonal entries in the singular value matrix to get an accurate analysis of LSA. Finally, for all methods, CR is still the important factor that affects the quality of the summary. The results reveal that as CR rises, we get better performance in both approaches.

Another important issue that we want to mention in particular is which approach is preferred, MCBA + GA or LSA + T.R.M. We first consider MCBA + GA in several folds: (1) since it is a learning algorithm, the selection of salient features extremely affects the quality of the summary, (2) the performance depends on the gene of documents (e.g., you cannot expect a brilliant performance when learning the score function from a data corpus in the domain of finance, but applying it in the domain of politics), and (3) the time to compute the summarization for a document is modest because the score function and the feature values are obtained in advance. In contrast to MCBA + GA, LSA + T.R.M. derives semantic structures from documents; we believe that it performs better in terms of semantics. However, it is difficult to determine the best dimension reduction ratio and hard to explain the effects of LSA (Hofmann, 2001). In addition, the computation time of SVD is much expensive. In conclusion, it depends on the situation to choose MCBA + GA or LSA + T.R.M. for achieving a better performance. As the above-mentioned, it is obvious that MCBA + GA is more suitable when you deal with a fixed-domain corpus, and is also appropriate for an on-line application. As for LSA + T.R.M., it is more preferred when you concern the quality of the summary since it takes into account semantics in nature. Besides, both of our approaches are language independent. When applying the algorithms, the most important thing is the selection of salient terms, and to take more semantics, such as named entities and noun phrases, into consideration. [10]

6. Conclusion

In this paper, we propose two text summarization approaches: modified corpus-based approach (MCBA) and LSA-based T.R.M. approach (LSA + T.R.M.). The first is a trainable summarizer, which considers several kinds of document features, including *position*, *positive keyword*, *negative keyword*, *centrality*, and *the resemblance to the title*, to generate summaries. We exploit two ideas to improve the conventional corpus-based approaches. First of all, sentence positions are ranked to emphasize the significance of different sentence positions. In addition, document features are combined by a weighting score function, and the genetic algorithm (GA) is employed to obtain a suitable combination of feature weights. The second uses latent semantic analysis (LSA) to derive the semantic matrix of a document (in single-document level) or a corpus (in corpus level), and uses semantic sentence representation to construct a semantic text relationship map. We evaluate LSA + T.R.M. both in single-document level and corpus level to investigate the competence of LSA in text summarization.

The two approaches were measured at several compression rates on a data corpus composed of 100 political articles. When the compression rate was 30%, the average F-measure of 49% for MCBA, 52% for MCBA + GA, 44% for LSA + T.R.M. in single-document level and 40% for LSA + T.R.M. in corpus level were achieved respectively. The results of MCBA show that centrality and resemblance to the title are the two dominant features, and the best combination of features is sentence positions, positive keywords, centrality, and resemblance to the title. The performance of MCBA + GA indicates that employing GA in training does provide an effective way to address the situation when we are indecisive about a good combination of feature weights. The results of LSA + T.R.M. show that either in single-document level or

[10] As mentioned in Liu, Gong, Xu, and Zhu (2002), an investigation shows that documents belonging to the same topic share named entities and contain many word associations. Hence, considering salient named entities and noun phrases may be the most essential step.

in corpus level, as the appropriate dimension reduction ratio is chosen, LSA + T.R.M. outperforms keyword-based text summarization approaches. We conclude that LSA can be employed to promote text summarization from keyword-level analysis to semantic-level analysis.

Acknowledgements

The research was supported by the Software Technology for Advanced Network Application project of Institute for Information Industry and sponsored by MOEA, ROC.

References

Abracos, J., & Lopes, G. P. (1997). Statistical methods for retrieving most significant paragraphs in newspaper articles. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 51–57), Madrid, Spain.

Academia Sinica (1999). CKIP AutoTag 1.0. Available: http://godel.iis.sinica.edu.tw/CKIP/ws.

Aone, C., Okurowski, M. E., Gorlinsky, J., & Larsen, B. (1997). A scalable summarization system using robust NLP. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 10–17), Madrid, Spain.

Azzam, S., Humphreys, K., & Gaizauskas, R. (1999). Using coreference chains for text summarization. In *Proceedings of the ACL'99 workshop on coreference and its applications* (pp. 77–84), College Park, MD, USA.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Harlow, UK: Addison-Wesley Longman Co Inc.

Barzilay, R., & Elhadad, M. (1997). Using lexical chains for text summarization. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 10–17), Madrid, Spain.

Bellegarda, J. R., Butzberger, J. W., Chow, Y. L., Coccaro, N. B., & Naik, D. (1996). A novel word clustering algorithm based on latent semantic analysis. In *Proceedings of the 1996 international conference on acoustics, speech, signal processing (ICASSP'96)* (pp. 172–175), Atlanta, GA, USA.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391–407.

DeJong, G. F. (1979). Skimming stories in real time: an experiment in integrated understanding. *Doctoral Dissertation*. Computer Science Department, Yale University.

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the ACM (JACM), 16*(2), 264–285.

Goldstein, J., Kantrowitz, M., Mittal, V., & Carbonell, J. (1999). Summarizing text documents: sentence selection and evaluation metrics. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'99)* (pp. 121–128), Berkeley, CA, USA.

Gong, Y., & Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'01)* (pp. 19–25), New Orleans, LA, USA.

Graesser, A. C. (1981). *Prose Comprehension beyond the Word*. NY: Springer-Verlag.

Hahn, U., & Mani, I. (2000). The challenges of automatic summarization. *IEEE-Computer, 33*(11), 29–36.

Han, J., & Kamber, M. (2001). *Data mining: concepts and techniques*. San Diego, CA: Academic Press.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning, 42*(1–2), 177–196.

Hovy, E., & Lin, C. Y. (1997). Automatic text summarization in SUMMARIST. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 18–24), Madrid, Spain.

Kim, J. H., Kim, J. H., & Hwang, D. (2000). Korean text summarization using an aggregative similarity. In *Proceedings of the 5th international workshop on information retrieval with Asian languages (IRAL'00)* (pp. 111–118), Hong Kong, China.

Kupiec, J., Pedersen, J., Chen, & F. (1995). A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'95)* (pp. 68–73), Seattle, WA, USA.

Landauer, T. K, Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes, 25,* 259–284.

Lin, C. Y. (1999). Training a selection function for extraction. In *Proceedings of the 8th international conference on information and knowledge management (CIKM'99)* (pp. 55–62), Kansas City, MO, USA.

Lin, C. Y., & Hovy, E. H. (1997). Identifying topics by position. In *Proceedings of the 5th conference on the applied natural language processing (ANLP'97)* (pp. 283–290), Washington, DC, USA.

Liu, X., Gong, Y., Xu, W., & Zhu, S. (2002). Document clustering with cluster refinement and model selection capabilities. In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'02)* (pp. 191–198), Tampere, Finland.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development, 2*(2), 159–165.

Mani, I., & Bloedorn, E. (1999). Summarizing similarities and differences among related documents. *Information Retrieval, 1*(1–2), 35–67.

Mani, I.& Maybury, M. T. (Eds.). (1999). *Advances in automated text summarization*. Cambridge, MA: The MIT Press.

Maosong, S., Dayang, S., & Tsou, B. K. (1998). Chinese word segmentation without using lexicon and handcrafted training data. In *Proceedings of the 17th international conference on computational linguistics (COLING'98) and the 36th annual meeting of the association for computational linguistics (ACL'98) (COLING-ACL'98)* (pp. 1265–1271), Montreal, Quebec, Canada.

Mayeng, S. H., & Jang, D. (1999). Development and evaluation of a statistically based document summarization system. In I. Mani & M. T. Maybury (Eds.), *Advances in automatic text summarization* (pp. 61–70). Cambridge, MA: The MIT Press.

McKeown, K., & Radev, D. R. (1995). Generating summaries of multiple news articles. In *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR'95)* (pp. 74–82), Seattle, WA, USA.

Mitra, M., Singhal, A., & Buckley, C. (1997). Automatic text summarization by paragraph extraction. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 39–36), Madrid, Spain.

New Taiwan Inc. (2003). New Taiwan Weekly. Available: http://www.newtaiwan.com.tw.

Russell, S. J., & Norvig, P. (1995). *Artificial intelligence: a modern approach*. Englewood Cliffs, NJ: Prentice-Hall International Inc.

Salton, G., Singhal, A., Mitra, M., & Buckley, C. (1997). Automatic text structuring and summarization. *Information Processing & Management, 33*(2), 193–207.

Schank, R., & Abelson, R. (1977). *Scripts, Plans, Goals, and Understanding*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Silber, H. G., & McCoy, K. F. (2000). Efficient text summarization using lexical chains. In *Proceedings of the 5th international conference on intelligent user interfaces (IUI'00)* (pp. 252–255), New Orleans, LA, USA.

Teufel, S. H., & Moens, M. (1997). Sentence extraction as a classification task. In *Proceedings of the ACL'97/EACL'97 workshop on intelligent scalable text summarization* (pp. 58–65), Madrid, Spain.

Yeh, J. Y., Ke, H. R., & Yang, W. P. (2002). Chinese text summarization using a trainable summarizer and latent semantic analysis. In *Lecture Notes in Computer Science (Vol. 2555)*. In *Proceedings of the 5th international conference on Asian digital libraries (ICADL'02)* (pp. 76–87), Singapore. Berlin: Springer-Verlag.

Young, S. R., & Hayes, P. J. (1985). Automatic classification and summarization of banking telexes. In *Proceedings of the 2nd Conference on Artificial Intelligence Applications* (pp. 402–408).