Hindawi Publishing Corporation
Scientific Programming
Volume 2016, Article ID 5130603, 10 pages
http://dx.doi.org/10.1155/2016/5130603

Research Article

Text Summarization Using FrameNet-Based Semantic Graph Model

Xu Han,[1,2] Tao Lv,[1,2] Zhirui Hu,[3] Xinyan Wang,[4] and Cong Wang[1,2]

[1]*School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*Key Laboratory of Trustworthy Distributed Computing and Service, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[3]*Department of Statistics, Harvard University, Cambridge, MA, USA*
[4]*Air Force General Hospital, Beijing, China*

Correspondence should be addressed to Xinyan Wang; wangxinyan@china.com

Received 8 August 2016; Accepted 30 October 2016

Academic Editor: Xiong Luo

Text summarization is to generate a condensed version of the original document. The major issues for text summarization are eliminating redundant information, identifying important difference among documents, and recovering the informative content. This paper proposes a Semantic Graph Model which exploits the semantic information of sentence using FSGM. FSGM treats sentences as vertexes while the semantic relationship as the edges. It uses FrameNet and word embedding to calculate the similarity of sentences. This method assigns weight to both sentence nodes and edges. After all, it proposes an improved method to rank these sentences, considering both internal and external information. The experimental results show that the applicability of the model to summarize text is feasible and effective.

1. Introduction

With the era of big data, text resources are becoming more and more abundant. Natural Language Processing (NLP) techniques have developed rapidly. Text summarization is a research tool that has widely applied in NLP. It gives a summary which can help us understand the whole article immediately with least words. Text summarization is to generate a condensed version of the original documents by reducing documents in size while retaining the main characteristics [1]. But artificial text summarization needs much background knowledge and often requires long processing time, while qualities of summaries are not always good enough. For this reason, people began to focus on automatic text summarization. Automatic text summarization was first studied almost 60 years ago by Luhn [2] and has gained much attention in recent years. This method is faster than artificial ones and performs better than the average level. With the rapid growth of text resources online, various domains of text summarization are applied. For instance, a Question Answering (QA) System produces a question-based summary to offer information. Another example is the search result snippets in web search engine which can assist users to explore more [3]; short summaries for News Articles can help readers to obtain useful information about an event or a topic [4]; speech summarization automatically selects indicative sentences from original spoken document to form a concise summary [5].

Automatic text summarization is mainly facing the following two problems: one is to reduce redundancy and the other is to provide more information with less words. Sentences in-summary are those which can stand for parts of the whole article. So the repeated information should be reduced, while the main information should be maintained. The major issues for text summarization are as follows. To begin with, information included in text is often redundant, so it is crucial to develop a method to eliminate redundancy. It is very common that different words are used to describe the same object in a text. For that reason, naive similarity measures between words cannot faithfully describe the content

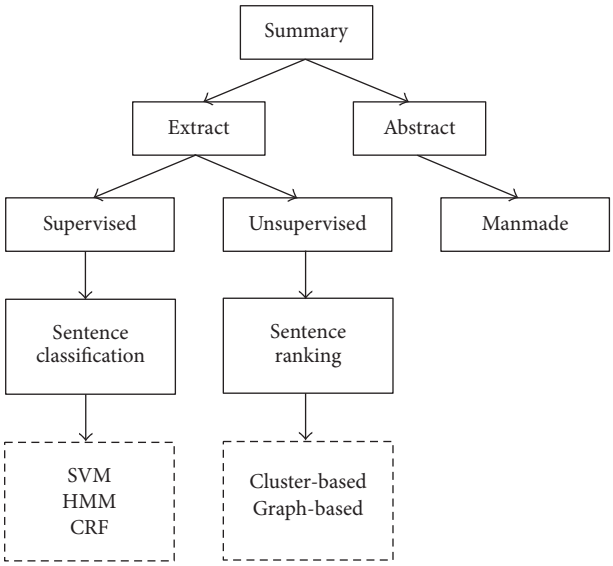

Figure 1: Summarization Approach Categories.

similarity. Another issue is identifying important difference among documents and covering the informative content as much as possible [6].

In this paper, to handle these issues, it proposes a Semantic Graph Model using FrameNet (FSGM) for text summarization that exploits the semantic information of sentence. FSGM treats sentences as vertexes while the semantic relationship between sentences as the edges. Firstly, it uses FrameNet to classify all sentences and then calculates the relevance values between sentences using semantic analysis and takes the values as weights of edges. Word embedding is used to combine similar sentences more accurately. Then based on the FSGM, sentences' values are scored by a variant of the improved PageRank graph ranking algorithm [7], considering both internal and external information. The experiments have been conducted on the DUC2004 data sets, and the results show that our model improves graph-based sentence extraction algorithms under various settings. So it can be used to enhance the performance. The experimental results show our model can be used for sentence extraction effectively.

The rest of the paper is organized as follows. Sections 2 and 3 briefly review the related work of current approaches about text summarization and the prepare work for experiments. The proposed Semantic Graph Model using FrameNet and word embedding is presented in Section 4 in detail. Section 5 is about the experiment, results, and relevant discussion. Finally, conclusion for this paper is in Section 6.

2. Related Work

Figure 1 shows the main summarization approaches [8]. There are two categories of summarization: abstract-based and extract-based. The abstractive summarization methods mostly generate the summary by compressing and reformulating the sentence. By this way, summary can keep its original length but has more effective information. However, the method requires complex linguistic processing, while the extract-based summarization measures various statistical significance for the words to locate the most central sentences in documents [9].

Extract-based approaches can be further categorized by supervised and unsupervised approaches. Supervised method proposes summarizing text as a classification problem in which all the sentences can be divided into either in-summary or not-in-summary. These methods are based on an essential assumption that each sentence should be independent. Naturally, it ignores the dependence between sentences. And then, there are additional approaches to build the relationship between sentences, such as using Hidden Markov Model (HMM) or Conditional Random Field (CRF).

Unsupervised method is to generate summaries by ranking sentences. Cluster-based approaches can classify sentences into groups and then choose the important ones in each group to form the final summary. Radev et al. [10] present MEAD with cluster centroids leverage. MEAD makes score for sentences by various features, assigning a representative centroid-based approach. Sentence-level semantic analysis (SLSS) and Symmetric Non-Negative Matrix Factorization (SNMF) are used in summarization by Wang et al. [11]. Mei and Chen [12] adapt fuzzy medoid-based clustering approach to produce subsets of similar sentences. Moreover, Latent Semantic Analysis (LSA), which helps adding and analyzing hidden topic features, is employed by Gong and Liu's work [13]. Wang et al. [14] generate summaries via discriminative sentence selection. He et al. [15] consider sentences in documents as a kind of signals from the perspective of compressive sensing. However, cluster-based approaches only considered the relationship between sentences, which may cause the semantic gap.

Another approach of unsupervised extract-based summarization [11, 16] uses graph-based model. TextRank [17] and LexRank [18] are first two graph-based models applied in text summarization, which use the PageRank-like algorithms to mark sentences. Then, other researchers have integrated the statistical and linguistic features to drive the sentence selection process, for example, the sentence position [19], term frequency [20], topic signature [21], lexical chains [22], and syntactic patterns [7, 23]. Ko and Seo [24] composed two sentences nearby into a bigram. Those bigrams were supposed to be context information. First, they extracted the bigrams by using the sentence extraction model. Then they used another extraction module to extract sentences from them. The ClusterCMRW and ClusterHITS models calculated the sentences scores by considering the cluster-level information in the graph-based ranking algorithm. Canhasi and Kononenko [25] improve matrix decomposition by employing the archetypal analysis for generic multidocument summarization. While coming to the document set, there must be more consideration about the document-level influence. But it did not consider the relationship between words and sentences. The DsR model [26] achieved it by using document-sensitive graph-based ranking model. But this method did not get a satisfied result. Yin et al. improved the summarization quality by adding extra information which

came from the query-extraction scenario. Goyal et al. [27] take Bernoulli model of randomness to index weights of sentences taking the context into consideration. The method proposed in [28] decomposed sentences by semantic role analysis, but while building the model, it did not use graph-based algorithms.

However, most of these graph-based methods only consider the relation of keyword cooccurrence, without considering the sentence-level dependency syntax. Those papers which use semantic information do not utilize the semantic information in the sentence-level. Thus, how to take advantage of the relationship between sentences needs further research. In this paper, it proposes sentence-level Semantic Graph Model. FSGM can build the relationships between sentences in a better way. Several experiments show ideal results in our model.

3. Preliminaries

In this section, it introduces related works for the experiments: firstly, the graph sorting algorithm, then the FrameNet, and finally word embedding.

3.1. Graph Sorting Algorithm. Graph sorting algorithm can calculate the nodes importance based on the graph structure. PageRank is one of its classical algorithms. The basic idea concerns the webpages' weight impacted both by usual links and reverse links, which means the more reverse links are, the heavier webpages' weight is. It builds graphs by using links structure, while the links are edges and webpages are nodes. The formula is as follows:

$$\mathrm{PR}\left(p_i\right) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{\mathrm{PR}\left(p_j\right)}{L\left(p_j\right)}, \tag{1}$$

where $p_1, p_2, \ldots, p_n$ are all nodes, $M(p_i)$ is a set which connect to p_i, $L(p_i)$ is p_i out-degree, N is the total number of nodes, and d is damping coefficient, usually set by 0.85.

In automatic text summarization, the graph is built by Natural Language Processing, text units are nodes, and the relationship of text units is edges. These relevancies often have its meaning, so the graph is basically unweighted undirected graph.

In undirected graph, every node has same in-degree and out-degree. In sparse condition, the convergence speed in undirected graph is faster than directed graph. The stronger the graph can connect, the faster it can converge. If the graph connectivity is strong enough, the convergence curves undirected and directed are nearly overlapped [29].

In weighted graph, the weight between nodes can be used; the formula is as follows:

$$\mathrm{PR}\left(p_i\right) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{w_{ij}\mathrm{PR}\left(p_j\right)}{L\left(p_j\right)}. \tag{2}$$

In graph sorting algorithm, the relevance between nodes is very important. In LexRank, if Cosine similarity between sentences is larger than the threshold, there is an edge. TextRank has another weight parameter comparing with PageRank. It gives edges weights and builds weighted graph. These two methods are simple and ignore some effective factors, such as semantic information and linguistic knowledge. Above all [30, 31], these methods are all considered more similar to judge the relevance between nodes. Corresponding solutions of these limitations are proposed in this paper. It used FrameNet combining the word embedding, to calculate the relevance between sentences; it will give more details in next section.

3.2. FrameNet. Traditional semantic sentence representations employ WordNet [17] or a corpus-based measure [9] in a classic bag of words, without considering sentence-level semantic information and word order. The proposed semantic sentence representation uses semantic frames instead of words to take both sentence-level semantic information and word order into account.

FrameNet, a lexical database based on a theory called frame semantics, contains a wealth of semantic lexical resources. Frame semantics studying the meaning of words and syntactic structure based on real corpus is proposed by Charles J. Fillmore. This theory uses empirical methods to find the close relationship between language and the human experience and tries to describe this relationship with a possible way. The basic idea is straightforward: the meanings of most words can best be understood on the basis of a semantic frame, a description of a type of event, relation, or entity, and the participants in it.

In the context of FrameNet, "frame" is a linguistic term that refers to a set of concepts to understand when people use natural language to describe an event or a semantic scene. Words that evoke a frame are called "lexical units" (LUs); for example, bike, carry, and drive are the LUs of the BRINGING frame. Each frame contains a series of semantic roles containers which called Frame Elements (FEs). FEs are related to words described in the context of events or objects in real corpus. There are two classes of FEs, core and noncore elements, which are determined by their importance to the corresponding frame. Different frames have different types and numbers of Frame Elements, and these differences can reflect semantic information in natural language. Figure 2 is an illustration of the BRINGING frame.

Frame-to-frame relation describes an overview semantic relationship, which is an asymmetric directional relation. Eight frame relations are defined, which are inheritance, perspective_on, subframe, precedes, causative_of, inchoative_of, using, and see_also.

Each frame is directly related to two frames, based on the orientation relationship; one is called super_frame and the other is called sub_frame. The BRINGING frame-to-frame relation is as follows:

Inherits from:

Is Inherited by: *Smuggling*

Perspective on:

Is Perspectivized in:

Uses: *Cause_motion*, *Motion*

BRINGING

Definition:

This frame concerns the movement of a Theme and an Agent and/or Carrier. The Agent, a person or other sentient entity, controls the shared Path by moving the Theme during the motion. In other words, the Agent has overall motion in directing the motion of the Theme. The Carrier may be a separate entity, or it may be the Agent's body. The Constant_location may be a subregion of the Agent's body or (a subregion of) a vehicle that the Agent uses.

Karl CARRIED the books across campus to the library on his head.

The FEs include Path, Goal, and Source. Area is an area that contains the motion when the path is understood as irregular. This frame emphasizes the path of movement as opposed to the FEs Source or Goal as in Filling or Placing.

FE core set(s):

{Goal, Path, Source}, {Agent, Carrier}

Lexical units:

airlift.v, bear.v, bike.v, bring.v, bus.v, carry.v, cart.v, convey.v, drive.v, ferry.v, fetch.v, fly.v, get.v, haul.v, hump.v, jet.v, lug.v, mobile.a, motor.v, paddle.v, portable.a, row.v, schlep.v, shuttle.v, take.v, tote.v, transport.n, transport.v, truck.v, trundle.v, wheel.v

Figure 2: Definition of the frame.

Is Used by: *Convoy*, *Sending*

Subframe of:

Has Subframe(s):

Precedes:

Is Preceded by:

Is Inchoative of:

Is Causative of:

See also: *Cause_motion*, *Motion*, *Passing*, *Sending*

To sum up, the strengths of FrameNet are as follows.

(a) It contains a wealth of semantic information (semantic roles) and specifies the scene of semantic roles appeared under predicates.

(b) The definitions of semantic roles are intuitive and straightforward.

(c) It defines relationships between the predicates.

The weaknesses are listed below.

(a) Frames are determined by predicates, so new predicates need to define new frame.

(b) The expression of the semantic predicates is determined by other related words; the selection of frames is a thorny problem, especially in the case of polysemy.

(c) FrameNet may parse some phrases directly into semantic roles, but these phrases may be separable.

3.3. Word Embedding. Word embedding is a core technique, which brings deep learning into NLP research. In the past, it often uses a simple method, called one-hot representation, to transfer words into vectors. Every word is shown as a long vector; the dimension of the vector is the length of the thesaurus. It is represented by sparse matrix. For example, word "mic" is saved as $[0, 0, 0, 0, 1, \ldots, 0, 0]$ and word "microphone" is saved as $[0, 0, 0, 1, 0, \ldots, 0, 0]$. It is simple but has a fatal weakness; it cannot distinguish synonym, such as "microphone" and "mic." This phenomenon is called semantic gap. Meanwhile, the curse of dimensionality will appear when the dimension is large enough [32].

Hinton [33] proposed "distributed representation" in 1986. It uses low dimension real numbers to represent word vectors; the length of the dimension used to be 50 or 100. For example, word "mic" can be saved as $[0.58, 0.45, -0.15, 0.18, \ldots, 0.52]$. This method avoids the semantic gap by calculating the distances between words. Word embedding can give words similarity at the semantic level, which is an effective addition for FrameNet. At first, it used FrameNet to compute sentence similarity at a sentence-level. Although it achieved good results, the similarity between sentences calculation relies only on simple comparison within the framework of words. Word embedding transfers all the words in the sentence. In the calculation of sentence similarity, it can effectively reduce the redundancy of sentences by combining the subject or object [34]. In this paper, it uses the lexicon generated by Wikipedia words.

4. FrameNet-Based Semantic Graph Model

Sarkar [35] uses selected key phrases as key concepts identified from a document to create summary of document, while Sankarasubramaniam et al. [36] leverage Wikipedia as the knowledge base to understand the salient concepts of documents. However, the above techniques may fail to semantically analyze newly coined words; Heu et al. [37] employ the tag clusters used by Flickr, a Folksonomy system. Inspired by above researchers, FSGM adapts FrameNet to detect key concepts from documents.

In FSGM, it utilizes the semantic information while considering automatic text summarization, to avoid semantic gap as well as ignorance of grammatical structure by other methods. FrameNet is the basis of FSGM to represent semantic information. The Frame Elements in FrameNet can express rich semantic information of the framework. And FSGM requires the relationship between frames to calculate similarities between sentences. In order to make full use of semantic information, word embedding is applied in computation of sentence similarity. In construction of the semantic graph, sentences should be aggregated when sentence similarity value is greater than the threshold α, and the aggregated sentence should assign new aggregated weights. Thus, the number of nodes in the semantic graph n is not greater than the number of sentences in the text. Graph

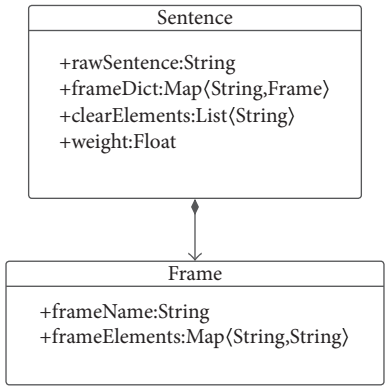

Figure 3: UML form of the structure.

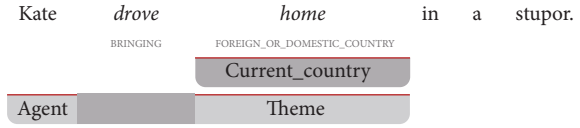

Figure 4: Semantic parsing.

sorting algorithm is then applied to the semantic graph. After the calculation, the aggregation and nodes weights are combined to select the most representative sentence. Detailed steps are as follows.

(a) Use FrameNet to parse the text, to identify the frame, and annotate the sentences with LUs and FEs, and generate the sentence node.

(b) Calculate the similarity between sentence nodes by combining the FrameNet with the word embedding.

(c) Aggregate sentences when the similarity value is greater than threshold α.

(d) Calculate the weight of sentence nodes by graph ranking algorithm.

(e) The final weights of the sentence nodes are calculated by using the method of combination of aggregate weights and relationship weights.

(f) Select the most representative sentence to generate summary.

4.1. Structure of Sentence Node. In order to facilitate further steps after preprocessing, sentences should be stored in specific structure. The structure of sentence node in FSGM must include original text, recognized frame, Frame Elements, lexical elements after preprocessing, and sentence weight. It should be noted that a sentence may contain multiple frames, and it designs the most basic data structure for storing the semantic representation of the sentence. Figure 3 shows the UML form of the structure.

In Figure 3, frame structure includes frame name and collection of Frame Elements. Sentence structure includes original sentence, collection of frames, collection of lexical elements after preprocessing, and sentence weight.

After computing sentence similarity, FSGM will aggregate appropriate sentences, for the purpose of reducing redundancy and promoting diversity while generating summary.

4.2. Sentence Semantic Parsing. When FSGM parses sentences, it first annotates LUs in sentences to identify frames inside, then locates other FEs in sentences according to the FEs defined in the frames, and stores parsed semantic information in predefined sentence structure. FSGM uses an open-source semantic parser SEMAFOR. It first analyzes sentence with a rule-based system to recognize target LUs and then identifies frames by using a statistical model. FEs are captured by another statistical model at the last step. Figure 4 is an example of semantic parsing.

For the sentence "Kate drove home in a stupor." two frames are identified from the sentence: BRINGING and FOREIGN_OR_DOMESTRIC_COUNTRY. Moreover, Frame Elements Agent and Theme are annotated in the frame BRINGING.

4.3. Sentence Semantic Similarity. Sentence semantic similarity is the kernel of the FSGM, which describes differences between sentences in semantic level by combining the FrameNet and word embedding. Lin et al. [38] propose a similarity measure for text by computing the similarity between two documents with respect to a feature. FSGM regards frame as the feature between sentences and takes following three cases into account.

If frames identified in both sentences are the same and the Frame Elements defined in the frames are also the same, the similarity between two sentences is 1, indicating in the sight of FrameNet that the semantics is the same. If the frame is different, it defined similarity as

$$\mathrm{Sim}\left(S_i, S_j\right) = \frac{\sum_{k=0}^{n} \theta_k \times \mathrm{Distance}\left(\mathrm{SF}_{ik}, \mathrm{SF}_{jk}\right)}{\sum_{k=0}^{n} \theta_k}, \quad (3)$$

where SF is word vector of the lexicon elements and θ is the coefficient of lexicon elements, which has 3 types. The first is that elements are under the same frame, the second is that elements are under the 8 frames mentioned above, and the third is all the other conditions. In this paper, θ is set by 1.2, 1.1, and 1.0. n is the smaller number in lexicon elements, set by $\min(\mathrm{num}(\mathrm{SF}_i), \mathrm{num}(\mathrm{SF}_j))$. Lexicon elements either are under frame or are not.

Word vectors are as follows:

$$\mathrm{SF} = \frac{\sum_{i=0}^{n} E_i}{n}, \quad (4)$$

where n is the number of elements in lexicon setting and E_i is the number of i's word vector.

$\mathrm{Distance}(\mathrm{SF}_i, \mathrm{SF}_j)$ is the Cosine distance:

$$\mathrm{Distance}\left(\mathrm{SF}_i, \mathrm{SF}_j\right) = \frac{\sum_{k=0}^{n} v_{ik} \times v_{jk}}{\sqrt{\sum_{k=0}^{n} v_{ik}^2} \sqrt{\sum_{k=0}^{n} v_{jk}^2}}, \quad (5)$$

where v_{ik} is SF_i's value in k dimension.

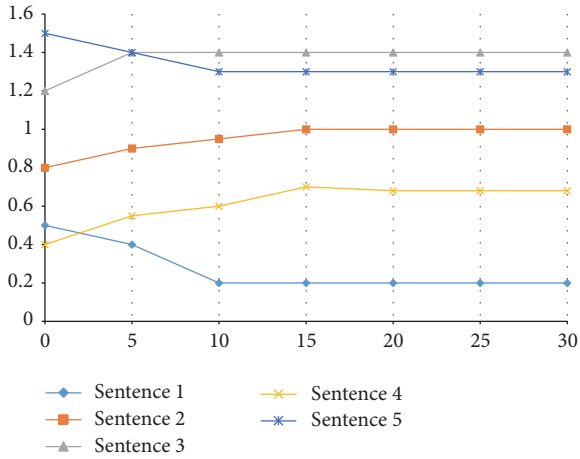

Figure 5: Example of convergence curves.

Word embedding generates a multidimension word vector for each word; [34] proved that adding up all the word vectors in one sentence and calculating the Cosine distance is an effective method to get the sentence similarity.

4.4. Construct Semantic Graph. After transforming sentences into semantic sentence representation and calculating of the semantic similarity between pairwise sentences, the sentence nodes need to be merged for variability and diversity. The threshold of the similarity is 1, which means FSGM only merges the same sentences.

FSGM builds a weighted undirected semantic graph $G = \{V, E\}$, where V is the vertex set and E is the edge set of the graph. Each sentence in documents is represented in frames and is built to be a graph vertex. If the semantic similarity between them is larger than some threshold, these representative vertexes in the semantic graph will be linked by edges and the edge weight is assigned by the value of the pairwise semantic similarity.

Based on the weighted undirected graph, every sentence is evaluated by applying the PageRank-like algorithm. Equation (6) shows how to calculate the weight of each vertex in graph,

$$\text{weight}\left(V_i\right) = (1-d) + d \times \sum_{V_j \in \ln(V_i)} \frac{\text{Sim}\left(S_i, S_j\right) \times \text{weight}\left(V_j\right)}{\sum_{V_k \in \text{Out}(V_j)} \text{Sim}\left(S_j, S_k\right)}, \quad (6)$$

where weight(V_i) represents the weight of V_i which also represents the sentence S_i, d represents the damping factor, usually set to 0.85, $\ln(V_i)$ is the set of vertexes which is connected to V_i, since the graph is undirected, $\text{Out}(V_i) = \ln(V_i)$, and $\text{Sim}(S_i, S_j)$ is the semantic similarity between S_i and S_j. The convergence threshold η is set to 10^{-5}. In actual calculation, an initial value is given for V_i and then updated by (6). Experiments show that (6) usually converges in 20–30 iterations in a sentence semantic graph.

Figure 5 shows the weight change of five sentences in a document with a given initial value C. The abscissa represents the number of iterations; the ordinate represents the sentence weight of current state. In the undirected weighted graph, vertex weight converges very quickly. After a few iterations, the difference between the vertex weight values is far less than η.

With sentence-level semantic graph, it can obtain the weight of each sentence in the document. After calculating the similarity, it gathers the sentences if the values are larger than the threshold α and generate a new node. Given the new node its weight is based on the number of combining sentences. Then, if the similarity values are larger than the threshold β, we link these two sentences. After that, it generates a weighted graph. The convergence results in graph ranking model are irrelevant with the initial weight vector. After iteration, it finally gets a FSGM.

4.5. Sentence Semantic Weight. Word embedding can represent word senses in word embedding vectors. As the dimension of word embedding vectors is consistent between words, words in a sentences can sum their word embedding vectors up to represent the sentence. Also sentences in a document can sum their vectors to represent the document. So the semantic representation of document can be word embedding vectors.

The vectors of the document contain all semantic information. The sentence semantic weights should be the amount of information that the vectors of sentence contained. So the weight is calculated by the Cosine distance between the document vectors and the sentence vectors.

4.6. Sentence Selection. It should consider both the semantic weight and the weight in the text structure when generating summary. FSGM combined the semantic weight and the relation weight; the formula is as follows:

$$W = \mu W_s + (1-\mu) W_g, \quad (7)$$

where μ is the coefficient of semantic weight, $(1-\mu)$ is coefficient of relation weight, W is the final weight, W_s is the semantic weight for sentence node, and W_g is the relation weight for sentence node.

5. Experiments

In this section, it introduces the data set, the evaluation metric, and the result of text summarization using proposed FSGM.

5.1. Setup. Firstly, to compare our methods, it uses several document summarization baseline systems. The definitions of these systems are as follows.

Lead. For each topic, it would return the leading sentences of every document.

Random. For each topic, it would select sentences randomly.

GraphRank. Rank sentences by graph-based algorithms using traditional bag-of-word.

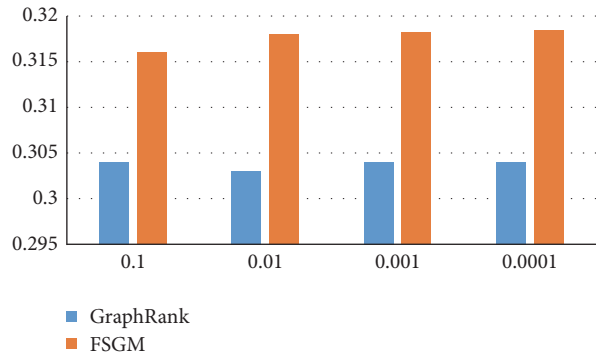

Figure 6: Sensitivity of ROUGE-1 with respect to threshold η.

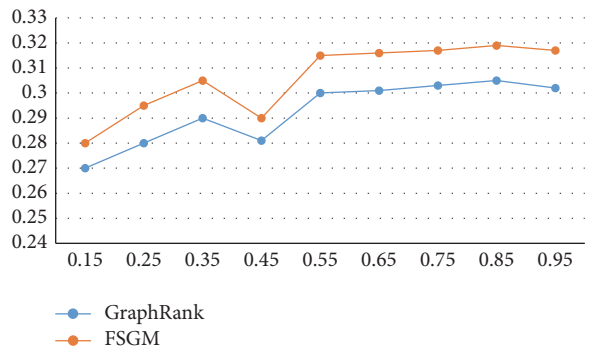

Figure 7: Sensitivity of ROUGE-1 with respect to damping factor d.

5.2. Data Set. In the experiments, it chooses DUC 2004 dataset for text summarization. It is the collections of newspaper documents from TREC. It has the original summaries. Besides, the datasets are public. Therefore, lots of researchers select DUC 2004 to study text summarization. For each document, NIST human assessors wrote two summaries: one is 200 words and the other is 400 words. And for each task, there are two summarizer models. There are 5 tasks in DUC 2004; each task has 15 clusters. It has chosen 15 clusters randomly from the 75 clusters.

5.3. Evaluation Metrics. ROUGE is a performance evaluation method widely applied by DUC. It chooses this method to measure our FSGM. It measures the summary quality by counting the overlaps between a set of reference summaries and candidate summary. After that, there are several kinds of automatic evaluation methods, such as ROUGE-N, ROUGE-W, ROUGE-L, and ROUGE-SU. Equation (8) shows the compute step for ROUGE-N which is a n-gram recall:

$$\text{ROUGE-}N = \frac{\sum_{S\in\{\text{RefSum}\}} \sum_{n\text{-gram}\in S} \text{Count}_{\text{match}}(n\text{-gram})}{\sum_{S\in\{\text{RefSum}\}} \sum_{n\text{-gram}\in S} \text{Count}(n\text{-gram})}, \quad (8)$$

where n is the length of the n-gram and Ref is the reference summaries. In a candidate summary and the reference summary, $\text{Count}_{\text{match}}(n\text{-gram})$ is the max of n-gram, and in the reference summaries, the number of n-gram is $\text{Count}(n\text{-gram})$. The longest common subsequence (LCS) has been used in ROUGE-L statistics; while ROUGE-W is based on weighted LCS, ROUGE-SU is on the basis of skip-bigram plus unigram. All these four evaluation methods in ROUGE can form three scores, which are recall, precision, and F-measure.

It uses the ROUGE toolkit 1.5.5 for evaluation. It calculates these scores by our FSGM method and compared with three other systems, (Lead, Random, and GraphRank). It uses the scores of ROUGE-1, ROUGE-2, and ROUGE-L.

5.4. Result and Discussion. For graph-based text summarization method, the value of threshold η and damping factor d will affect the performance of the graph-based algorithm. It analyzes the sensitivity of both parameters for two graph-based systems involved in our experiment including GraphRank and FSGM. The threshold η is varied in the set $\{0.1, 0.01, 0.001, 0.0001\}$ and damping factor d is varied in the range between 0.05 and 0.95 with an interval of 0.1. From Figure 6, it can be inferred that the choice of threshold η in the set $\{0.1, 0.01, 0.001, 0.0001\}$ is quite insensitive to both systems. And the experiments show that when η is set to 0.0001, both systems gave the best results. Figure 7 shows that the results are not sensitive to the damping factor d in the range between 0.65 and 0.95 of both systems.

$d = 0.85$ is chosen for further experiments as it gave the best results.

In order to investigate the effects of performance by different parameters, it compares the default setting of several parameters with designed parameters.

Table 1: Performance comparison of FSGM with word embedding and without word embedding.

Method	FSGM (with word embedding, default settings)	FSGM (without word embedding, default setting)
ROUGE-1	0.325	0.315
ROUGE-2	0.052	0.049
ROUGE-L	0.298	0.293

Table 1 illustrates the performance comparison of FSGM with and without word embedding. The table clearly shows that word embedding can better utilize the semantic information to calculate similarity between sentences. It should mention that although word embedding can finitely improve the performance of generating summary, the option of adapting word embedding or not should be considered thoroughly.

Different settings of coefficients of lexical elements in the semantic similarity are compared in Table 2. The coefficient is a vector of the three elements. Each element is the weight of the related collections of lexical elements. It can be inferred from the table that the default setting achieved the best performance. The default setting assigns reasonable weights to three types of lexical elements collections. The θ_3 assigns all types of lexical elements with the weight of one, but it mixes different types of collections without consideration.

Table 2: Performance comparison of FSGM with different coefficients of lexical elements.

Parameter	θ_{default} (1.2, 1.1, 1.0)	θ_1 (2, 1.5, 1.0)	θ_2 (1.5, 1.3, 1.0)	θ_3 (1.0, 1.0, 1.0)	θ_4 (0.5, 0.8, 1.0)
ROUGE-1	0.325	0.315	0.322	0.310	0.298
ROUGE-2	0.052	0.049	0.049	0.047	0.042
ROUGE-L	0.298	0.305	0.303	0.293	0.289

Table 3: Performance comparison of FSGM with different thresholds of combining sentences.

Parameter	α_{default} (1)	α_1 (0.95)	α_2 (0.90)	α_3 (0.85)	α_4 (0.5)
ROUGE-1	0.325	0.320	0.320	0.317	0.198
ROUGE-2	0.052	0.049	0.052	0.046	0.019
ROUGE-L	0.298	0.300	0.295	0.293	0.135

Table 4: Performance comparison of FSGM with different thresholds of estimating sentence relationship.

Parameter	β_{default} (0.5)	β_1 (0.9)	β_2 (0.7)	β_3 (0.3)	β_4 (0.1)
ROUGE-1	0.325	0.167	0.240	0.214	0.156
ROUGE-2	0.052	0.019	0.032	0.024	0.013
ROUGE-L	0.298	0.126	0.145	0.156	0.113

Table 5: Performance comparison of FSGM with different coefficients of estimating semantic weight.

Parameter	μ_{default} (0.3)	μ_1 (0.9)	μ_2 (0.7)	μ_3 (0.5)	μ_4 (0.1)
ROUGE-1	0.325	0.298	0.320	0.307	0.306
ROUGE-2	0.052	0.043	0.051	0.049	0.045
ROUGE-L	0.298	0.270	0.295	0.293	0.273

Table 3 is the performance comparison of FSGM with different thresholds of combining sentences. When the threshold is set by 1, the performance is the best. When it comes lower, all parameters are worse. And when it comes to 0.5, which means that it combines two sentences when they are only half same, ROUGE parameters make no sense. The more similar the two sentences are when combined, the better their performance is.

Table 4 is the performance comparison of FSGM with different thresholds of estimating sentence relationship. The best performance appears on 0.5. This line almost obeys the normal distribution, while the sentences should not be too similar nor too dissimilar. When β is small, there are little edges; when β is too big, nearly all lines link between nodes.

Table 5 is the performance comparison of FSGM with different coefficients of estimating semantic weight. This parameter has less influence than others. The best setting is $\mu = 0.3$.

Moreover, this paper analyzes and compares the FSGM performance on the DUC2004 with several baseline systems. Table 6 presents the results that are achieved by Lead, Random, GraphRank, and FSGM. The table clearly shows the proposed FSGM outperformed other systems. It can be inferred from the results that text summarization gets better performance using sentence-level semantic information.

It should notice that Lead's performance outperformed Random in every ROUGE measure, and its results of measure ROUGE-2 and ROUGE-L even approach graph-based system GraphRank. Lead method only chooses the first sentence of the document as the abstract, while the result is similar to GraphRank. It means that the location of sentences must be considered when generating abstract. Although Lead method is simple, its performance is instable, because it greatly depends on the context.

Unlike Lead, GraphRank analyzes the context based on relations of words. It builds a graph model based on sentence similarity and analyzes their occurrence relations. So the performance is much better than baseline. But it is not enough to consider the sentence-level semantic similarity. It only focuses on the relationship between sentences, ignoring sentence itself as basic morpheme. On such basis, FSGM considers the semantic similarity between sentences, so it gets the best performance. Meanwhile, as it considers the relationship between sentences, the performance is very stable.

While there are multiple choices of graph ranking algorithm, it choses PageRank-like algorithm as the default ranking algorithm of FSGM. It choses another famous graph ranking algorithm HITS for performance comparison, which is also widely applied in measuring the importance of graph vertexes. Both HITS and PageRank-like algorithm applied in the FSGM achieve the best results. It can be inferred from Table 6 that taking sentence-level semantic information into consideration can improve the performance of general graph ranking algorithm. The reason why PageRank-like algorithm does better than HITS in the experiment may be because that the former is topic independent while the latter is topic related. The FSGM based on HITS may be more suitable in query-based tasks.

6. Conclusion

In this paper, it reviews the common methods of text summarization and proposes a Semantic Graph Model using FrameNet called FSGM. Besides the basic functions, it particularly takes sentence meaning and words order into consideration, and therefore it can discover the semantic relations between sentences. This method mainly optimizes the sentences nodes by combining similar sentences using word embedding. Also, giving the sentences its weight and optimizing the PageRank can make the model more rigorous. The results show that FSGM is more effective from the understanding of sentence-level semantic.

Above all, if it can take more semantic information into account, it may probably get a better result. In the future work, it prepares to build a multiple-layer model to further show the

Table 6: Performance comparison on DUC2004 using ROUGE evaluation methods.

Parameter	Lead	Random	GraphRank	FSGM (PageRank-like)	FSGM (HITS-like)
ROUGE-1	0.292	0.290	0.304	0.325	0.310
ROUGE-2	0.043	0.041	0.041	0.052	0.045
ROUGE-*L*	0.271	0.264	0.265	0.298	0.280

accuracy rate of application in text summarization. And in this paper, it only applies FSGM to a test corpus. Nowadays, text from social media is the main resource. And there will be more serious problems about the credibility. It will research on the social media content in the future.

Disclosure

This paper is based on the authors' paper "Text Summarization Using Sentence-Level Semantic Graph Model" from 2016 4th IEEE International Conference on Cloud Computing and Intelligence Systems.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this article.

Acknowledgments

This work is supported by Basic Research of the Ministry of Science and Technology, China (2013FY114000).

References

[1] K. S. Jones, "Automatic summarising: factors and directions," in *Advances in Automatic Text Summarization*, I. Mani and M. Maybury, Eds., pp. 1–12, MIT Press, Cambridge, Mass, USA, 1999.

[2] K. M. Svore, L. Vanderwende, and C. J. C. Burges, "Enhancing single-document summarization by combining RankNet and third-party sources," in *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '07)*, pp. 448–457, June 2007.

[3] T. Hirao, Y. Sasaki, and H. Isozaki, "An extrinsic evaluation for question-biased text summarization on QA tasks," in *Proceedings of the NAACL Workshop on Automatic Summarization*, 2001.

[4] X. Wan and J. Zhang, "CTSUM: extracting more certain summaries for news articles," in *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '14)*, pp. 787–796, Queensland, Australia, July 2014.

[5] B. Chen, S.-H. Lin, Y.-M. Chang, and J.-W. Liu, "Extractive speech summarization using evaluation metric-related training criteria," *Information Processing & Management*, vol. 49, no. 1, pp. 1–12, 2013.

[6] D. R. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," *Computational Linguistics*, vol. 28, no. 4, pp. 399–408, 2002.

[7] E. Baralis, L. Cagliero, N. Mahoto, and A. Fiori, "Graph Sum: discovering correlations among multiple terms for graph-based summarization," *Information Sciences*, vol. 249, no. 16, pp. 96–109, 2013.

[8] X. Li, S. Zhu, H. Xie et al., "Document summarization via self-present sentence relevance model," in *Database Systems for Advanced Applications*, pp. 309–323, Springer, Berlin, Germany, 2013.

[9] P. Goyal, L. Behera, and T. M. McGinnity, "A context-based word indexing model for document summarization," *IEEE Transactions on Knowledge & Data Engineering*, vol. 25, no. 8, pp. 1693–1705, 2013.

[10] D. R. Radev, H. Jing, M. Styś, and D. Tam, "Centroid-based summarization of multiple documents," *Information Processing & Management*, vol. 40, no. 6, pp. 919–938, 2004.

[11] D. Wang, T. Li, S. Zhu, and C. Ding, "Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 307–314, ACM, July 2008.

[12] J.-P. Mei and L. Chen, "SumCR: a new subtopic-based extractive approach for text summarization," *Knowledge & Information Systems*, vol. 31, no. 3, pp. 527–545, 2012.

[13] Y. Gong and X. Liu, "Generic text summarization using relevance measure and latent semantic analysis," in *Proceedings of the 24th ACM Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '01)*, pp. 19–25, New Orleans, La, USA, September 2001.

[14] D. Wang, S. Zhu, T. Li et al., "Comparative document summarization via discriminative sentence selection," *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 3, pp. 1963–1966, 2009.

[15] R. He, J. Tang, P. Gong, Q. Hu, and B. Wang, "Multi-document summarization via group sparse learning," *Information Sciences*, vol. 349-350, pp. 12–24, 2016.

[16] T. Li, "A general model for clustering binary data," in *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 188–197, August 2005.

[17] S. Park, J.-H. Lee, D.-H. Kim, and C.-M. Ahn, "Multi-document summarization based on cluster using non-negative matrix factorization," in *SOFSEM 2007: Theory and Practice of Computer Science: 33rd Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 20–26, 2007. Proceedings*, vol. 4362 of *Lecture Notes in Computer Science*, pp. 761–770, Springer, Berlin, Germany, 2007.

[18] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*, pp. 269–274, San Francisco, Calif, USA, August 2001.

[19] R. Katragadda, P. Pingali, and V. Varma, "Sentence position revisited: a robust light-weight update summarization 'Baseline' algorithm," in *Proceedings of the 3rd International Workshop Cross Lingual Information Access: Addressing the Information Need of Multilingual Societies (CLIAWS3 '09)*, pp. 46–52, Boulder, Colo, USA, June 2009.

[20] C.-Y. Lin and E. Hovy, "Identifying topics by position," in *Proceedings of the 5th conference on Applied Natural Language Processing*, pp. 283–290, Washington, DC, USA, April 1997.

[21] C.-Y. Lin and E. Hovy, "The automated acquisition of topic signatures for text summarization," in *Proceedings of the 18th Conference on Computational Linguistics (COLING '00)*, pp. 495–501, Saarbrücken, Germany, August 2000.

[22] R. Barzilay and M. Elhadad, "Using lexical chains for text summarization," in *Proceedings of the ACL Workshop Intelligent Scalable Text Summarization*, pp. 10–17, Madrid, Spain, July 1997.

[23] M. H. Haggag, "Semantic text summarization based on syntactic patterns," *International Journal of Information Retrieval Research*, vol. 3, no. 4, pp. 18–34, 2013.

[24] Y. Ko and J. Seo, "An effective sentence-extraction technique using contextual information and statistical approaches for text summarization," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1366–1371, 2008.

[25] E. Canhasi and I. Kononenko, "Multi-document summarization via Archetypal Analysis of the content-graph joint model," *Knowledge & Information Systems*, vol. 41, no. 3, pp. 821–842, 2014.

[26] F. Wei, W. Li, Q. Lu, and Y. He, "A document-sensitive graph model for multi-document summarization," *Knowledge and Information Systems*, vol. 22, no. 2, pp. 245–259, 2010.

[27] P. Goyal, L. Behera, and T. M. McGinnity, "A context-based word indexing model for document summarization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 8, pp. 1693–1705, 2013.

[28] S. Harabagiu and F. Lacatusu, "Topic themes for multi-document summarization," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*, pp. 202–209, August 2005.

[29] R. Mihalcea, "Graph-based ranking algorithms for sentence extraction, applied to text summarization," in *Proceedings of the ACL on Interactive Poster and Demonstration Sessions*, p. 20, Association for Computational Linguistics, 2004.

[30] R. Ferreira, F. Freitas, L. De Souza Cabral et al., "A four dimension graph model for automatic text summarization," in *Proceedings of the IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies (IAT '13)*, vol. 1, pp. 389–396, November 2013.

[31] R. Ferreira, R. D. Lins, F. Freitas, S. J. Simske, and M. Riss, "A new sentence similarity assessment measure based on a three-layer sentence representation," in *Proceedings of the ACM Symposium on Document Engineering (DocEng '14)*, pp. 25–34, Fort Collins, Colo, USA, September 2014.

[32] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. 6, pp. 1137–1155, 2003.

[33] G. E. Hinton, "Learning distributed representations of concepts," in *Proceedings of the 8th Annual Conference of the Cognitive Science Society*, vol. 1, pp. 1–12, Amherst, Mass, USA, 1986.

[34] M. J. Kusner, Y. Sun, N. I. Kolkin et al., "From word embeddings to document distances," in *Proceedings of the 32nd International Conference on Machine Learning (ICML '15)*, pp. 957–966, Lille, France, 2015.

[35] K. Sarkar, "Automatic single document text summarization using key concepts in documents," *Journal of Information Processing Systems*, vol. 9, no. 4, pp. 602–620, 2013.

[36] Y. Sankarasubramaniam, K. Ramanathan, and S. Ghosh, "Text summarization using Wikipedia," *Information Processing & Management*, vol. 50, no. 3, pp. 443–461, 2014.

[37] J.-U. Heu, I. Qasim, and D.-H. Lee, "FoDoSu: multi-document summarization exploiting semantic analysis based on social Folksonomy," *Information Processing & Management*, vol. 51, no. 1, pp. 212–225, 2015.

[38] Y.-S. Lin, J.-Y. Jiang, and S.-J. Lee, "A similarity measure for text classification and clustering," *IEEE Transactions on Knowledge & Data Engineering*, vol. 26, no. 7, pp. 1575–1590, 2014.