The Algorithm of Automatic Text Summarization Based on Network Representation Learning

Xinghao Song[1], Chunming Yang[1,3(✉)], Hui Zhang[2], and Xujian Zhao[1]

[1] School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, Sichuan, China
yangchunming@swust.edu.cn
[2] School of Science, Southwest University of Science and Technology, Mianyang 621020, Sichuan, China
[3] Sichuan Civil-Military Integration Institute, Mianyang 621010, Sichuan, China

Abstract. The graph models are an important method in automatic text summarization. However, there will be problems of vector sparseness and information redundancy in text map to graph. In this paper, we propose a graph clustering summarization algorithm based on network representation learning. The sentences graph was construed by TF-IDF, and controlled the number of edges by a threshold. The Node2Vec is used to embedding the graph, and the sentences were clustered by k-means. Finally, the Modularity is used to control the number of clusters, and generating a brief summary of the document. The experiments on the MultiLing 2013 show the proposed algorithm improves the F-Score in ROUGE-1 and ROUGE-2.

Keywords: Text summarization · Network representation learning
Graph clustering · Modularity

1 Introduction

Automatic text summarization is extracting a piece of concise text that can represent the important content of the original text from a large amount of text. It is an effective method to help people obtain main information quickly and efficiently. It saves users' time and resources, and solves the problem of information overload on the Internet. After the summarization was first proposed by Luhn [1] in 1958, it was used in grammatical, single and descriptive texts such as scientific literature and news. In recent years, Internet texts generated by users have the characteristics of large quantity, wide coverage, subjective and nonstandard language. These problems have challenged the traditional approaches based on statistics, topics. The graph-based methods abstract the semantic relationship between sentences as the vertex relationship of graph.

However, in the case of Internet texts with many fields and redundant contents, it will make the graph sparse and lead to a decline in the quality of summary. Network representation learning is an effective method for solving graph sparseness and

© Springer Nature Switzerland AG 2018
M. Zhang et al. (Eds.): NLPCC 2018, LNAI 11109, pp. 362–371, 2018.
https://doi.org/10.1007/978-3-319-99501-4_32

reducing computational complexity in recent years. The principle is to use deep learning to map high-dimensional networks to dense low-dimensional vectors, and then use the low-dimensional vectors for further calculation. We propose a graph based summarization algorithm based on the learning algorithm. The vertices of network which represent the text were mapped into low-dimensional dense vectors through representation learning, and then use graph clustering to select appropriate sentences as the summary.

The remained of the paper is organized as follows. In Sect. 2, we summarize the research related to summarization based on the graph model, and analyze the existing problems. In Sect. 3, we introduce the summarization algorithm based on the representation learning. In Sect. 4, we show the experimental data, methods, and results. In Sect. 5, we summarize the method of this article. The contribution of this paper is to propose using network representation to solve the sparseness problem of graph, and use graph clustering to solve the redundancy problem in summary.

2 Related Work

Automatic text summarization can be divided into two types: abstractive and extractive. Abstractive method generates summary by understanding the main content of the text, it uses the NLP technology to understand the text. And it requires that the main content of a text be understood like human, and use a writing technique similar to human to "write" the summary, therefore the abstractive method is very difficult. After several decades of development, no good results have been achieved. With the development of deep learning, some researchers have also begun to use neural networks to achieve certain progress in abstractive summarization. DRGN [2] uses recurrent neural networks (RNN) to make the summary, however, there is still the problem that the generated summary information is inconsistent with the original text, and there are meaningless contents when summarizing multiple sentences.

Extractive method uses sentences as the basic elements; through the various characteristics of sentences, the importance of each sentence is calculated and the appropriate sentence is selected as the summary of the text. The types of extractive method can be divided into methods based on statistics, topics, graph models and machine learning. Graph model algorithm has a wide range of applications in the extractive way, because the topological structure of the graph can reveal important information and relationships between elements in the text; it can reflect more information than other non-graph model methods, so the summarization based on graph model become a prominent current method in recent years.

The earliest graph model work was proposed by Mani and Bloedorn [3], and the general graph model uses text elements such as sentences or words as the vertices of the graph, the relationships and information between the text elements represent the edges. Most of the graph-based methods are scoring graph vertices; they calculate the importance of the sentences in graph model, and get the summary by the sentence score. Such methods are generally called graph sorting algorithms, it has been proved that they are indeed effective in summarization.

LexRank [4] is a typical graph sorting algorithm, it constructs sentences graph by using the sentences in the text as vertices and using PageRank to score the vertices in the graph and obtain the most important sentences in the text. Ferreira [5, 6] incorporates the construction of multiple knowledge extension graph edges and takes into account the semantic relationships to construct sentence graphs, which makes the information richer and better results in more specialized text fields such as thematic information or linguistic knowledge. Giannakopoulos [7] uses words as the vertices of the graph model to extract the important components of the document using N-Gram graphs; it makes the entire process of the proposed summarization system provide rich information about the complex and contextual relationships between characters or words n-grams.

However, there are still the following problems in the above method:

1. Since the algorithms are based on vertex sorting, redundant content may appear in extracted summary sentences. For example, when there are two sentences in the text expressing the same content, the sorting algorithm will give the two sentences an approximate score, it will make two similar sentences in the summary.
2. The sparseness of graphs reduces the quality of summary. If the amount of text information is large, the graph model will be large and sparse, and the quality of the summarization will be affected to a large extent.

Based on the above problems, we propose a graph clustering automatic summarization algorithm based on network representation learning. This method constructs a sentence graph, and maps the graph vertex as a low-dimensional dense vector through the learning algorithm to solve the problems of graph sparseness and algorithm execution efficiency. Then we use the graph clustering to select the appropriate sentences as summaries, while integrating the group relations between sentences, we also solve the redundant information of the text.

3 Our Method

3.1 Graph Construction Through Sentence Similarity

First we need to construct a sentence graph, let $G = (V, E)$ be a graph, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes, $v_i = \{w_1, w_2, \ldots, w_n\}$ represents a sentence of text T, where w_i represents a word. And E is the set of edges, we need to add undirected edges between sentences by calculating the similarity between sentences. Before calculating the sentence similarity, we need to vectorize the sentences. Here we use the TF and IDF values of the words in the sentence to construct the vector of the sentence.

In information retrieval, TF-IDF [8] or TFIDF, short for *term frequency–inverse document frequency*, is a numerical statistic that is intended to reflect how important a

word is to a document in a collection or corpus. TF (*Term frequency*) is defined by the formula:

$$tf(w,T) = \frac{t_w}{n_w} \tag{1}$$

Where t_w is the number of occurrences of word w in text T, n_w is the total number of the words. And IDF *(Inverse document frequency)* is defined by the formula:

$$idf(w,s) = \log \frac{n_s}{1 + df(w,s)} \tag{2}$$

Where n_s is the total number of text sentences, $df(w,s)$ is the number of sentences s that contains the word w.

Then we use the bag-of-words to represent each sentence s as an n-dimensional vector $\boldsymbol{a} = (a_1, a_2, .., a_n)$, where n is the number of all words in the text T and $a_i = tf(w_i, T) \times idf(w_i, s)$. The similarity between two sentences is defined as the angle cosine of two sentence vectors:

$$tf - idf - cosine(x,y) = \frac{\sum_{w \in x,y} tf^2(w,T) idf(w,x) idf(w,y)}{\sqrt{\sum_{x_i \in x} (tf(x_i,T) idf(x_i,y))^2} \sqrt{\sum_{y_i \in y} (tf(y_i,T) idf(y_i,y))^2}} \tag{3}$$

Given the ε, for any two sentences s_i and s_j in the sentences set S; if $tf - idf - cosine(s_i, s_j) \geq \varepsilon$, we add an undirected edge e_{ij} between v_i and v_j to the graph G. The method we use to build sentence graph is presented in Algorithm 1.

Algorithm 1 Build Sentence Graph
Input: Text T, Threshold ε
Output: Sentence Graph $G = (V, E)$

```
Build G with n nodes, n is the sentences number of T
for i = 0 to n do
    for j = i + 1 to n do
        if tf − idf − cosine(s_i, s_j) ≥ ε
            add e_ij to E
    end for
end for
```

In the construction of graph G, the choice of the threshold ε will affect the sparseness of the graph. If ε is too large, it will cause the graph G to be too sparse and there are too many isolated points. If ε is too small, the graph G will be too dense to affect the result of the last cluster. Figure 1 shows the results of a graph constructed using 10 sentences as examples.

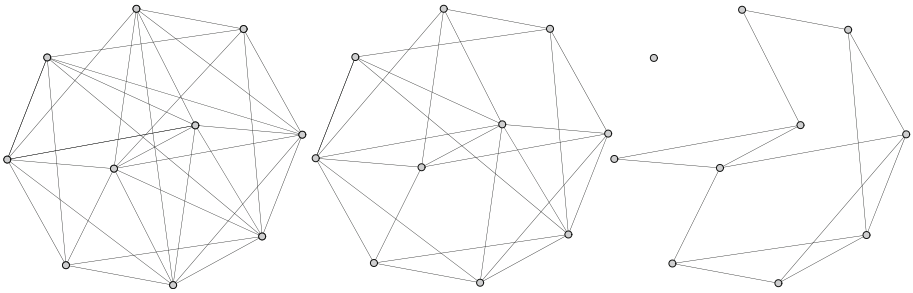

Fig. 1. Different ε sentence diagrams (from left to right ε values is 0.05, 0.1, 0.15)

3.2 Automatic Text Summarization via NRL

Network representation learning (NRL) algorithm refers to the algorithm for learning vector representation of each vertex from the network data, and the "network" refers to information networks such as social networks, web linked networks, and logistics networks. These networks are usually represented by data structure of "graph". Although graphs can be represented by adjacency matrices, adjacency matrices have great problems in computational efficiency, and the vast majority of adjacency matrices are 0, and the data is very sparse. This kind of data sparsity makes the application of fast and effective statistical learning methods difficult. Therefore, researchers turn to learning low dimensional dense vector representations for vertices in the network. Formally, the goal of NRL is to learn a real vector $a_v \in R^d$ for each vertex $v \in V$, in which the dimension d of the vector is far smaller than the total number of vertices $|V|$.

In 2013, the famous "word2vec" [9] used probabilistic neural networks to map words in natural language into low dimensional vector space, and then Perozzi *et al.* proposed *DeepWalk* [10] on the basis of the Skip-Gram model of "word2vec", for the first time the technology of deep learning was introduced into the NRL field. *DeepWalk* generalizes the idea of the Skip-Gram model that utilizes word context in sentences to learn latent representations of words, to the learning of latent vertex representations in networks, by making an analogy between natural language sentence and short random walk sequence. Given a random walk sequence with length L, $\{v_1, v_2, \ldots, v_L\}$, following Skip-Gram, *DeepWalk* learns the representation of vertex v_i by using it to predict its context vertices, which is achieved by minimize the optimization problem $-logPr(\{v_{i-w}, \ldots, v_{i+w}\} \backslash v_i | \Phi(v_i))$, where $\{v_{i-w}, \ldots, v_{i+w}\} \backslash v_i$ are the context vertices of vertex v_i within w window size. Subsequent work *node2vec* [11] and *LINE* [12] have further improved *DeepWalk*.

In order to remove redundant information in the document, we cluster the sentence vectors, clustering can group together similar sentences. When we select a summary sentence, we only need to select representative sentences from each cluster to compose the summary of the text; the clustering algorithm here we choose the k-means. One problem of the k-means algorithm is the choice of the number of clusters. In order to get better clustering results and solve the problem, we paper introduces "Modularity" to obtain the optimal number of clusters.

"Modularity" [13] is one measure of the structure of networks or graphs. It was designed to measure the strength of division of a network into modules (also called groups, clusters or communities). Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules. Given a community division C, the modularity is defined as:

$$Q = \frac{1}{2m}\sum_{ij}\left(\boldsymbol{A}_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j) \tag{4}$$

Where $\boldsymbol{A}$ is the adjacency matrix, k_i is the degree of vertex v_i, $\delta(C_i, C_j)$ is defined as:

$$\delta(C_i, C_j) = \begin{cases} 1 & C_i = C_j \\ 0 & C_i \neq C_j \end{cases} \tag{5}$$

For different k of k-means, we select the clustering result with the largest modularity as the final sentence clustering result, and then select the vertex sentence with the largest degree in each cluster as the candidate summary sentence. The details of the method we used to generate automatic summarization via NRL is presented in Algorithm 2, here the NRL algorithm we used is the *node2vec*.

Algorithm 2 Auto Summarization via NRL
Input: Sentence graph G, Text D, dimension d
Output: Sentence of text summarization $s_1, \ldots, s_n$
$X \leftarrow \text{node2vec}(G, d)$
$C_{best} \leftarrow k - means(X, 2)$
$Q_{max} \leftarrow Q(C)$
for $i = 3$ to n **do**
  $C \leftarrow kmeans(G, i)$
  if $Q(C) > Q_{max}$
    $C_{best} \leftarrow C$
    $Q_{max} \leftarrow Q(C)$
end for
$s_i \leftarrow maxdegree(C_i)$

4 Experiment

4.1 Datasets and Evaluation Method

We used the MultiLing 2013 dataset in our experiment; it contains a collection of texts in 40 languages, each text collection contains 30 single text files, and there are about 100–300 sentences in each single text. In the experiment, we select the data set of the English language as the experimental data set to generate a summary for each single text.

For evaluation, we used the automatic summary evaluation metric, ROUGE [14]. ROUGE is a recall-based metric for fixed-length summaries which is based on n-gram co-occurrence. It reports separate scores for 1, 2, 3, and 4-gram matching between the model summaries and the summary to be evaluated. In the experiment, we mainly used ROUGE-1 and ROUGE-2 evaluation standards. And the ROUGE-N is defined by the formula:

$$ROUGE - N = \frac{\sum_{S \in ref} \sum_{gram_n \in S} count_{match}(gram_n)}{\sum_{S \in ref} \sum_{gram_n \in S} count(gram_n)} \tag{6}$$

Where n stands for the length of the n-gram, $gram_n$ and $count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries.

4.2 Results

We evaluate the performance of our method against the following automatic text summarization algorithms:

- Text-Rank [15]: A graph-based ranking model using PageRank for text processing and text summarization.
- LSA [16]: A summarization methods based on LSA, which measure a content similarity between an original document and its summary.

The number of summary sentences used for Text-Rank and LSA is 10; for our method we set $\varepsilon = 0.14$ for Algorithm 1 and $d = 20$ for Algorithm 2. Table 1 lists the average results of 30 documents ROUGE-1, and Table 2 lists the average results of 30 documents ROUGE-2.

Table 1. Comparison of the results of the three algorithms ROUGE-1 ($\varepsilon = 0.14, d = 20$)

Method	Avg_Recall	Avg_Precision	Avg_F-Score
Text-Ran	0.5676	0.3268	0.4019
LSA	0.4386	0.3515	0.3876
Our	0.4446	0.4445	0.4332

Table 2. Comparison of the results of the three algorithms ROUGE-2 ($\varepsilon = 0.14, d = 20$)

Method	Avg_Recall	Avg_Precision	Avg_F-Score
Text-Ran	0.1214	0.1305	0.1256
LSA	0.1026	0.0839	0.0916
Our	0.1355	0.1372	0.1326

From the evaluation results on the MultiLing 2013 single text dataset; we can see that our method is better than the text-rank and LSA on the F-score of ROUGE-1 and

ROUGE-2. And the results of text-rank and our algorithm are all better than the LSA algorithm which based on semantic analysis. This shows that the graph model is indeed better than the traditional statistical method in summarization the algorithm. It can also be seen that the graph model represented by the learning algorithm has improved on the summary result.

4.3 Parameter Sensitivity

There are two important parameters ε and *d* in our method, we analyzed the F-score of ROUGE-1 and ROUGE-2 under different ε-values and dimensions *d* on the data set, the result is shown in Figs. 2, 3 and 4:

Experiments show that when ε is less than 0.14, the F value of ROUGE-1 and ROUGE-2 increases with the increase of ε. When ε is greater than 0.14, the summary results begin to decline. This is consistent with our previous analysis that sentences are too dense or sparse will affect the effectiveness of the summary, and map the sentence to low dimensional vector will result in better quality of text summary.

From Figs. 3 and Fig. 4, the effect of the dimension on the Rouge value can be seen that when the value of the dimension d is between 20 and 30, the result of the summary is the best, and the overall trend of the summary results is decreasing with the increase of the dimension. This shows that embedding a sentence into a low-dimensional vector space not only improves the efficiency of the algorithm but also improves the quality of the summarization.

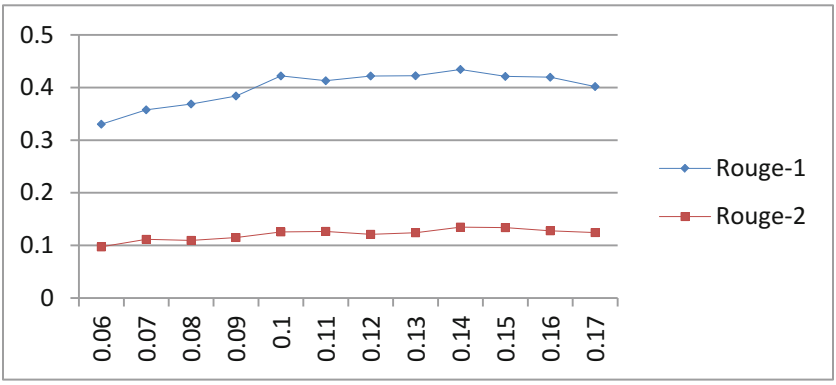

Fig. 2. Rouge with different ε values (d = 20)

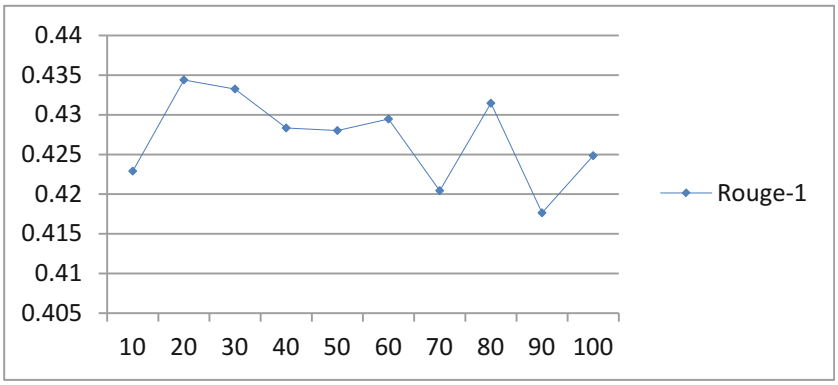

Fig. 3. Rouge-1 values for different dimensions d ($\varepsilon = 0.14$)

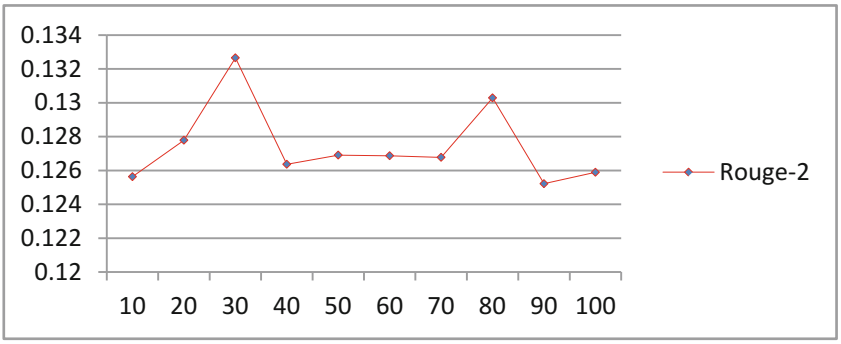

Fig. 4. Rouge-2 values for different dimensions d ($\varepsilon = 0.14$)

5 Discussion and Conclusion

In this paper, we use network representation learning to propose a new summarization algorithm based on graph model. This method is slightly better than the other two methods on the MultiLing 2013 document set. The NRL has attracted much attention since 2014. The current research is still at the initial stage and there is no representation learning method for the sentence graphs of the text. This paper only studies the clustering-based summarization methods after expressing the vector representation of the sentences through learning. In the future work, there are many works worthy of further research and exploration in the study of sentence graph and sentence vector based summarization algorithms.

Acknowledgement. This work is supported by the Ministry of education of Humanities and Social Science project (17YJCZH260), the Next Generation Internet Technology Innovation Project (NGII20170901), the Fund of Fundamental Sichuan Civil-military Integration (JMRHH01, 18sxb017, 18sxb028).

References

1. Luhn, H.P.: The automatic creation of literature abstracts. IBM Corp. (1958)
2. Li, P., Lam, W., Bing, L., et al.: Deep Recurrent Generative Decoder for Abstractive Text Summarization. arXiv preprint arXiv:1708.00625 (2017)
3. Mani, I., Bloedorn, E.: Multi-document summarization by graph search and matching. In: Proceedings of AAAI 1997, pp. 622–628 (1997)
4. Erkan, G., Radev, D.R.: LexRank: graph-based lexical centrality as salience in text summarization. J. Qiqihar Jr. Teach. Coll. **2011**, 22 (2004)
5. Ferreira, R., Freitas, F., Cabral, L.D.S., et al.: A four dimension graph model for automatic text summarization. In: IEEE/WIC/ACM International Joint Conferences on Web Intelligence, pp. 389–396. IEEE (2013)
6. Ferreira, R., Lins, R.D., Freitas, F., et al.: A new sentence similarity method based on a three-layer sentence representation. In: ACM Symposium on Document Engineering, pp. 25–34. ACM (2014)
7. Giannakopoulos, G., Karkaletsis, V., Vouros, G.: Testing the use of N-gram graphs in summarization sub-tasks. In: Proceedings of Text Analysis Conference, TAC 2008, pp. 158–167 (2008)
8. Salton, G., Fox, E.A., Wu, H.: Extended boolean information retrieval. Cornell University (1982)
9. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
10. Perozzi, B., Alrfou, R., Skiena, S.: DeepWalk: online learning of social representations, pp. 701–710 (2014)
11. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: KDD, p. 855 (2016)
12. Tang, J., Qu, M., Wang, M., et al.: LINE: large-scale information network embedding, vol. 2, pp. 1067–1077 (2015)
13. Newman, M.E.: Fast algorithm for detecting community structure in networks. Phys. Rev. E Stat. Nonlinear Soft Matter Phys. **69**(6 Pt 2), 066133 (2003)
14. Lin, C.: ROUGE: a package for automatic evaluation of summaries. In: ACL, pp. 74–81 (2004)
15. Mihalcea, R., Tarau, P.: TextRank: bringing order into texts. UNT Scholarly Works, pp. 404–411 (2004)
16. Steinberger, J., Jezek, K.: Using latent semantic analysis in text summarization and summary evaluation. In: International Conference ISIM, pp. 93–100 (2004)