Graph-Based Text Summarization Using Modified TextRank

Conference Paper *in* Advances in Intelligent Systems and Computing · August 2018

DOI: 10.1007/978-981-13-0514-6_14

CITATIONS
51

READS
4,696

5 authors, including:

Chirantana Mallick
Indian Institute of Engineering Science and Technology, Shibpur
8 PUBLICATIONS **74** CITATIONS
SEE PROFILE

Ajit Kumar Das
Indian Institute of Technology Kharagpur
9 PUBLICATIONS **78** CITATIONS
SEE PROFILE

Madhurima Dutta
Indian Institute of Engineering Science and Technology, Shibpur
4 PUBLICATIONS **70** CITATIONS
SEE PROFILE

Asit Kumar Das
Indian Institute of Engineering Science and Technology, Shibpur
148 PUBLICATIONS **1,141** CITATIONS
SEE PROFILE

Graph-Based Text Summarization Using Modified TextRank

Chirantana Mallick, Ajit Kumar Das, Madhurima Dutta, Asit Kumar Das and Apurba Sarkar

Abstract Nowadays, the efficient access of enormous amounts of information has become more difficult due to the rapid growth of the Internet. To manage the vast information, we need efficient and effective methods and tools. In this paper, a graph-based text summarization method has been described which captures the aboutness of a text document. The method has been developed using modified TextRank computed based on the concept of PageRank defined for each page in the Web pages. The proposed method constructs a graph with sentences as the nodes and similarity between two sentences as the weight of the edge between them. Modified inverse sentence frequency-cosine similarity is used to give different weightage to different words in the sentence, whereas traditional cosine similarity treats the words equally. The graph is made sparse and partitioned into different clusters with the assumption that the sentences within a cluster are similar to each other and sentences of different cluster represent their dissimilarity. The performance evaluation of proposed summarization technique shows the effectiveness of the method.

Keywords Extractive summarization · Single-document source · Sentence index · Similarity graph · PageRank · ROUGE

C. Mallick (✉) · A. K. Das (✉) · M. Dutta · A. K. Das · A. Sarkar
Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Shibpur, Howrah, India
e-mail: chirantana9@gmail.com

A. K. Das
e-mail: writetoajit@yahoo.com

M. Dutta
e-mail: madhurima.pg2016@cs.iiests.ac.in

A. K. Das
e-mail: akdas@cs.iiests.ac.in

A. Sarkar
e-mail: as.besu@gmail.com

J. Nayak et al. (eds.), *Soft Computing in Data Analytics*,
Advances in Intelligent Systems and Computing 758,
https://doi.org/10.1007/978-981-13-0514-6_14

1 Introduction

Automatic text summarization reduces a text of a document in order to create a summary that retains the crux of the original text. The variety of applications like summaries of newspaper articles, book, magazine, resume, music, film, and speech increases the importance of text summarization methods.

In general, text summarization approaches can be categorized into two types: extractive summarization [21] and abstractive summarization [8]. Extractive summarization creates the summary from phrases or sentences in the source document(s) and abstractive summarization expresses the ideas in the source document using different words. Abstractive summarization is more efficient than extractive summarization as it generates humanlike summaries. Text summarization is also classified as indicative [7] and informative [17]. Indicative summary identifies the topics of the document which is mainly used for quick categorization and the informative summary elaborates these topics and includes conclusions, suggestions, and recommendations according to the reader's interest. Informative summarization technique is used for making generic [5] and query-oriented [20] summary. Generic summaries try to cover more information with preserving the generic topic of the given document and it provides author's view. Query-oriented summaries generate a query set that reflects user's interest. Background summary assumes reader's prior knowledge is poor and it is meant to be short and attention grabbing. Just-the-news summary [22] provides recent updated news according to users' interest. Single-document summary [10] takes sentences from the document itself, whereas multi-document summary [4] makes a summary by fusing sentences from different documents. Topic-oriented summarization [6] is made based on users' enthusiasm and the information extracted from the given document related to some specific topic. Centrality summarization [14] measures the centrality of a sentence in a given document. The centrality of a sentence is defined in terms of word centrality and the word centrality in a vector space is defined based on the centroid of the document cluster. Similarly, centroid-based summarization [16] is performed on centroid of sentences of the given document. The central sentences contain more words from the centroid of the cluster to measure how closely the sentence corresponds to the centroid of the cluster.

The proposed work is developed on extractive summarization technique. Extractive summarization contains three distinguished independent tasks: intermediate representation of the document, scoring sentences based on different parameters, strategies for selection of summary sentences.

A. Intermediate Representation: For any simplest summarizer, intermediate representation of the text to be summarized is done to identify the important content. It is an essential step. In topic representation approaches, the original text document is converted into an intermediate representation. Intermediate representation [14] includes frequency, TF-IDF [19] approach, and topic-word approach [6]. The topic representation of topic-word approaches consists of a simple list of words and their corresponding weights where higher weighted words are considered as indicative topic words. The topic signature words for latent semantic analysis are decided

based on the patterns of word co-occurrence and the weights of each pattern. In graph models [2], such as LexRank [3], the entire input text document is represented as a graph based on the interrelated sentences. In Bayesian topic models, the document is represented as a combination of topics where each topic is presented by means of word probabilities (weights) for that topic. In indicator representation approaches, the input text document is represented as a list of indicators of such as sentence length, presence of certain phrases pertaining to the theme of the documet, location in the document.

B. Score Sentences: After converting the input text document into intermediate representation, score (or weight) is assigned to each sentence of the given document to identify the important sentences. The weight of each sentence is determined by examining the different parameters used in the method. Machine learning techniques are most commonly used to determine indicator weights. In LexRank, the score of sentence is determined from the graphical representation [12] of the text.

C. Selecting Summary Sentences: This is the final and last step to construct a relevant summary. Generally, it selects the best combination of sentences that gives the important information of the original text and the desired summary is the combination of top *n* important sentence. The optimal and best collection of sentences is selected to maximize overall importance, minimize redundancy, or maximize coherence in global selection procedure.

The summarization approach of Salton et al. [18] uses degree scores to extract the most important sentences of a document. Moens et al. [13] partition a text document into different topical regions using cosine similarity between the sentences. Zha [23] uses mutual reinforcement principal and a bigraph from the set of terms to the set of sentences to reduce a solution for the singular vectors of the transition matrix of the bipartite graph. Mihalcea and Tarau (2004) [3] proposed a centrality algorithm on weighted graphs for single-document summarization by computing eigenvectors. In our proposed method, to determine the similarity between two sentences, inverse sentence frequency modified cosine (isf-modified-cosine) similarity is measured to provide weights of the edges in the graph. In this paper, a graph-based method is used to extract the most important sentences from the given text, sufficient to provide the overall idea about the text document. Here, the overall centrality of a sentence is computed based on its similarity to other sentences. This approach mainly determines the similarity between the sentences based on the modified PageRank algorithm [12], named here as modified TextRank algorithm. The remaining part of the paper is organized as follows: Detailed procedure of the proposed method is described in Sect. 2. In this section, we have explained the process of extraction of the text document as input, the preprocessing of the raw input and summarization technique using TextRank. Subsequently, entire evaluation of the obtained summary using ROUGE method and experimental results is provided in Sect. 3. Finally, Sect. 4 holds future work and the final conclusion of the paper.

2 Proposed Methodology

With this paper, we have tried to present a modified graph-based approach on achieving extractive summary of a text document. Calculating similarity between pair of sentences, instead of the regular TF-IDF, isf-weight-based cosine similarity is used that provides the interesting results when tested with news articles. Figure 1 represents the system architecture of the overall proposed methodology.

2.1 Preprocessing

Several approaches exist to extract information from the Web. Structured data is preferably extracted through an API over Web screaping. Unfortunately not all Web sites provide an API because they do not want the users to extract data in structured way. We can perform Web scrapping to transform unstructured data in HTML or XML format into structured data (database or spreadsheet). In this paper, we have used Beautiful Soup [1] for scrapping a Web page. Beautiful Soup is a Python [15] library for extracting body of HTML and XML files. It also defines a class for auto detecting the encoding of an HTML or XML document, and converting it to Unicode. After extracting structured data, first we perform removal of stop words from the raw data. Next, we tokenize the given text into sentences and assign sentence index according to the input sequences of the sentences in the document. Finally, tokenize each sentence into a collection of words which is required for graphical representation.

2.2 TextRank-Based Summarization

In the proposed method, we have considered that the sentences of the document are equivalent to Web pages in the PageRank system [12]. The probability of going from sentence *A* to sentence *B* is equal to the similarity between the pair of sentences. The modified TextRank proposed in this paper uses the intuition behind the PageRank

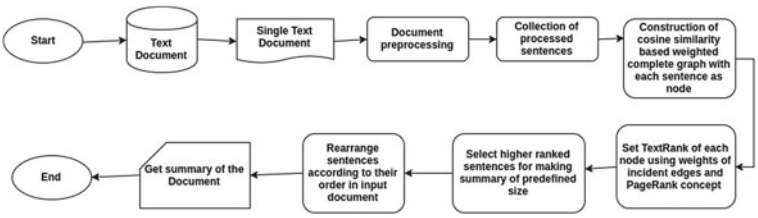

Fig. 1 System architecture

algorithm to rank sentences using which we can select the most important sentences from the input text document. In PageRank, important Web pages are linked with other important Web pages. Similarly, in our approach we assume that the important sentences are linked (similar) to other important sentences of the input document. In this model, at first to identify the sentences, we have transferred the input document D into sentences $\{s_1, s_2, \ldots, s_n\}$ using NLTK package, i.e., $D = \{s_1, s_2, \ldots, s_n\}$. We have assigned index for each of the sentence according to the input sequence of the sentences in the document. Index values are significantly used for proper ordering of the summarized sentences to make the summary meaningful. In the next step, each sentence is tokenized into a set of words. To define similarity among the sentences, we represent each sentence of the given text document as a word vector. Next, a complete weighted graph $G = (V, E, W)$ of the input document is built, where V is the set of vertices, each of which corresponds to a sentence represented by the word vector. E is the set of edges between every pair of vertices. W is the set of weights assign to the edges of the graph G. The weight w associated to edge $(u, v) \in E$ is assigned using following logic:

(i) Suppose $S(u) = \{w_u^1, w_u^2, \ldots, w_u^s\}$ is the sentence corresponds to the vertex u and $S(v) = \{w_v^1, w_v^2, \ldots, w_v^t\}$ is the sentence corresponds to the vertex v. Here, we have considered term frequency (*tf*) and inverse sentence frequency (*isf*) for each word in the sentence. For an example, $tf_w(S(u))$ is the term frequency of word w in $S(u)$ which gives the number of occurrences of the word w in the sentence. Similarly, $isf_w(D)$ is the inverse sentence frequency of the word w in the input document D which is defined by Eq. (1).

$$isf(w, D) = \log \frac{\|D\|}{\|\{S \in D : w \in S\}\|} \tag{1}$$

where, $\|\{S \in D : w \in S\}\|$ is the number of sentences in which word w appears, and $\|D\|$ is the number of sentences present in the input document.

(ii) Now, the isf-modified-cosine similarity is used to measure the similarity between every pair of sentences and it is used as weight w of edge (u, v) using Eq. (2).

$$w(u, v) = \frac{\Sigma_{\forall x \in S(u) \cup S(v)} tf_x(S(u)) tf_x(S(v)) (isf_x(D))^2}{\sqrt{\Sigma_{\forall y \in S(u)} (tf_y(S(u)) isf_y(D))^2} \times \sqrt{\Sigma_{\forall z \in S(v)} (tf_z(S(v)) isf_z(D))^2}} \tag{2}$$

Traditional cosine similarity only takes care of the term frequency of the corresponding words in the text document, and in case of cosine similarity, the dimension should be same for all sentence vectors, whereas isf-modified-cosine similarity takes care of the different level of importance for the corresponding words in the sentences and also considers different length of the sentence in the document.

Thus, we get a complete weighted graph which is made sparse by removing the edges having weight less than the threshold (t), set as the average weight of all the edges in the graph. This graph represents the similarity graph of the input document. For weight assignment, score is assigned to every sentence corresponding to every

node of the graph G. In this proposed method, we initialise the TextRank score of each node of the graph by the average weight of the edges incident to it for giving importance to the weights associated with the edges(E), whereas in traditional PageRank the initial PageRank value of each node is set to $\frac{1}{T}$, where T is the total number of nodes in the graph. Next, the TextRank score is updated via modified TextRank as defined in Eq. (3).

$$TR(S(u)) = \frac{d}{T} + (1-d) * \Sigma_{v \in adj(u)} \frac{TR(S(v))}{deg(S(v))} \tag{3}$$

where $TR(S(u))$ is the text rank of sentence S corresponds to the node u, $TR(S(v))$ is the text rank of the sentence S corresponds to the node v such as $v \in adj(u)$, $deg(S(v))$ is the degree of the node v corresponding to the sentence S, T is the total number of nodes present in the graph, and d is a "damping factor." Here, we have considered the value of d as 0.15. Finally for summarization, we have selected top n scored sentences and rearranged those n sentences according to the sentence index which we have assigned at first step. The n output sentences construct the summary of our proposed model. Using the pseudocode of the proposed Algorithm (2.1), we can extract important sentences from a given input document.

Algorithm 2.1: MODIFIEDTEXTRANKSUMMARY(D, n)

Input: Extracted text from the target document D and size of summary = n (say).
Output: Summary of the input document.
1. Encode D into D' using UTF-8 format;
2. D' is tokenized into individual sentences (S_i) using the NLTK library;
3. Initialize *index_value* to zero for the sentence index;
4. **for each** $S_i \in D'$ in order of appearance in encoded D **do**
   4.1. Remove the stop words from the sentence;
   4.2. Increase the *index_value* by 1;
   4.3. Assign the *index_value* to the sentence;
5. **for each** processed tokenized sentence $S_i \in D'$ **do**
   Take word count vector $v_i = \{w_1, w_2, \ldots, w_{t_i}\}$;
6. Build a graph $G = (V, E, W)$ where V = set of vertices corresponding to sentences represented by word count vector, E = set of edges and W = set of weights associated with each edge $(u, v) \in E$ computed by Eq. (2);
7. Remove edges with weight less than the average weight considering all edges of G;
8. $\forall v \in V$ Initialize TextRank by the average weight of the edges incident to it;
9. Modify TextRank of every node $v \in V$ using Eq. (3) based on the initial TextRank;
10. Arrange the vertices of the graph in descending order of their TextRank score;
11. Take first n vertices from the sorted list;
12. Put in summary the sentences associated to the selected n vertices in order of the *index_value*;
return (*summary*)

3 Experimental Result

Since early 2000s, Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [9] is widely used for performance evaluation of summarization techniques. Some of the ROUGE methods, i.e., ROUGE_N, ROUGE_L, ROUGE_W and ROUGES_U are

Table 1 Rouge values of the proposed method and traditional TF-IDF method with respect to ground truth

Method	Rouge values of the proposed method						Rouge values of traditional TF-IDF method					
Length	10%			15%			10%			15%		
Metric	Recall	Precision	f-score	Recall	Precision	f-score	Recall	Precision	f-score	Recall	Precision	f-score
Rouge-1	1.00000	0.72632	0.84147	0.68807	0.78947	0.73529	0.77215	0.64211	0.70115	0.72895	0.72895	0.72895
Rouge-2	0.98529	0.71277	0.82716	0.62963	0.72340	0.67327	0.67949	0.56383	0.60624	0.60963	0.60963	0.60963
Rouge-L	1.00000	0.72632	0.84147	0.59813	0.68817	0.64000	0.75494	0.63258	0.68956	0.74682	0.74682	0.74682
Length	20%			25%			20%			25%		
Metric	Recall	Precision	f-score	Recall	Precision	f-score	Recall	Precision	f-score	Recall	Precision	f-score
Rouge-1	0.68807	1.00000	0.73529	0.55232	1.00000	0.70561	0.62114	1.00000	0.70529	0.43379	1.00000	0.60510
Rouge-2	0.62963	0.95819	0.67327	0.55232	0.96809	0.58333	0.61232	0.94872	0.70459	0.41743	0.90590	0.58333
Rouge-L	0.63303	1.00000	0.67647	0.55232	1.00000	0.70561	0.62140	1.00000	0.66236	0.40092	1.00000	0.60510

commonly used to measure the performance. ROUGE_N is the N-gram ($N \geq 1$) recollection between a system summary and human-generated or reference summaries. It is used to estimate the fluency of summaries. ROUGE_N is computed using Eq. (4). Here, the value of N is considered as 1 and 2. ROUGE_1 and ROUGE_2 denote the overlap of 1-gram and bi-grams between the system and sample summaries, respectively.

$$Rouge\_N = \frac{\Sigma_{s \epsilon refsum} \Sigma_{gram_n \epsilon S} count_{match}(gram_n)}{\Sigma_{s \epsilon refsum} \Sigma_{gram_n \epsilon S} count(gram_n)} \tag{4}$$

Rouge_L is used to identify the longest co-occurring in sequence n-grams automatically. Suppose we assume that A is the set of sentences of the reference summary and B is the set of sentences of the candidate summary represented by the sequence of words and LCS-based F score (F_{lcs}) indicates the similarity between A (of length m) and B (of length n) according to Eqs. (5), (6), (7).

$$R_{lcs} = \frac{LCS(A, B)}{n} \tag{5}$$

$$P_{lcs} = \frac{LCS(A, B)}{m} \tag{6}$$

$$F_{lcs} = \frac{(1 + \beta^2) R_{lcs} P_{lcs}}{(R_{lcs} + \beta^2 P_{lcs})} \tag{7}$$

where LCS (A, B) denotes the length of the LCS of A and B and $\beta = P_{lcs}/R_{lcs}$. We have used the news articles generated from the BBC News Feed. The results of our evaluation and traditional TF-IDF are given below in Table 1. The unbiased ground truth or the reference summaries of the news articles are obtained from our fellow peers and research scholars having different field expertise. We have considered 10%, 15%, 20%, and 25% sentences of the whole document as the summary size and the performance of the summaries are evaluated using Eqs. (5), (6), and (7) as listed in Table 1. The values demonstrate the effectiveness of the proposed method in compared to the traditional TF-IDF.

4 Conclusion and Future Work

In this paper, we have discussed various methods of abstractive summarization as well as extractive summarization and different other types of summarization. We can get better view of important sentences by constructing the similarity graph of sentences using isf-modified-cosine similarity in comparison to the centroid approach. We also have tried to make use of more of the information from the graph representation of the input document and got better results in many cases. The results of applying these methods on extractive summarization are proving to be effective. Our proposed algorithm does not take care of anaphora [11] resolution problem. We can

include it to obtain more informative summary. It may be possible that more than one similar type of sentences with high score is selected for the summary. To eliminate this kind of duplication, we may consider any kind of clustering algorithm and merge it with our proposed graph-based algorithm. In this proposed method, we have considered the weight of the graph only in the initialization of the TextRank score for every sentences correspond to the vertices of the graph. It may be possible to improve our result by considering the weights in the modification of the TextRank defined in Eq. (3). As an extension of the paper, we will compare our method with many more existing effective text summarization methods with different performance measurement metrics.

References

1. Beautifulsoup documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (2017). Accessed 30 Nov 2017
2. Dutta, S., Ghatak, S., Roy, M., Ghosh, S., Das, A.K.: A graph based clustering technique for tweet summarization. In: 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), pp. 1–6. IEEE (2015)
3. Erkan, G., Radev, D.R.: Lexrank: graph-based lexical centrality as salience in text summarization. J. Artif. Intell. Res. **22**, 457–479 (2004)
4. Goldstein, J., Mittal, V., Carbonell, J., Kantrowitz, M.: Multi-document summarization by sentence extraction. In: Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic Summarization, vol. 4, pp. 40–48. Association for Computational Linguistics (2000)
5. Gong, Y., Liu, X.: 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, pp. 19–25. ACM (2001)
6. Harabagiu, S., Lacatusu, F.: Topic themes for multi-document summarization. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 202–209. ACM (2005)
7. Kan, M.-Y., McKeown, K.R., Klavans, J.L.: Applying natural language generation to indicative summarization. In: Proceedings of the 8th European workshop on Natural Language Generation, vol. 8, pp. 1–9. Association for Computational Linguistics (2001)
8. Khan, A., Salim, N.: A review on abstractive summarization methods. J. Theor. Appl. Inf. Technol. **59**(1), 64–72 (2014)
9. Lin, C.-Y.: Rouge: a package for automatic evaluation of summaries. In: Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, Barcelona, Spain, vol. 8 (2004)
10. Litvak, M., Last, M.: Graph-based keyword extraction for single-document summarization. In: Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization, pp. 17–24. Association for Computational Linguistics (2008)
11. Loaiciga Sanchez, S.: Pronominal anaphora and verbal tenses in machine translation. Ph.D. thesis, University of Geneva (2017)
12. Mihalcea, R.: Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, pp. 20. Association for Computational Linguistics (2004)
13. Moens, M.-F., Uyttendaele, C., Dumortier, J.: Abstracting of legal cases: the potential of clustering based on the selection of representative objects. J. Assoc. Inf. Sci. Technol. **50**(2), 151 (1999)
14. Nenkova, A., McKeown, K.: A survey of text summarization techniques. Mining Text Data, pp. 43–76 (2012)

15. Python 2.7.14 documentation. https://docs.python.org/2/index.html (2017). Accessed 30 Nov 2017
16. Radev, D.R., Jing, H., Styś, M., Tam, D.: Centroid-based summarization of multiple documents. Inf. Process. Manag. **40**(6), 919–938 (2004)
17. Saggion, H., Lapalme, G.: Generating indicative-informative summaries with sumum. Comput. linguist. **28**(4), 497–526 (2002)
18. Salton, G., Singhal, A., Mitra, M., Buckley, C.: Automatic text structuring and summarization. Inf. Process. Manag. **33**(2), 193–207 (1997)
19. Seki, Y.: Sentence extraction by tf/idf and position weighting from newspaper articles (2002)
20. Tang, J., Yao, L., Chen, D.: Multi-topic based query-oriented summarization. In: Proceedings of the 2009 SIAM International Conference on Data Mining, pp. 1148–1159. SIAM (2009)
21. Wong, K.-F., Wu, M., Li, W.: Extractive summarization using supervised and semi-supervised learning. In: Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1, pp. 985–992. Association for Computational Linguistics (2008)
22. Yeh, J.-Y., Ke, H.-R., Yang, W.-P., Meng, I.-H.: Text summarization using a trainable summarizer and latent semantic analysis. Inf. process. Manag. **41**(1), 75–95 (2005)
23. Zha, H.: Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 113–120. ACM (2002)