Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 107 (2017) 248 – 255

Procedia Computer Science

International Congress of Information and Communication Technology (ICICT 2017)

A Graph-based Approach of Automatic Keyphrase Extraction

Yan Ying[a,*], Tan Qingping[a], Xie Qinzheng[a], Zeng Ping[a], Li Panpan[a]

[a]*College of Computer, National University of Defense Technology, Changsha 410073, China*
** Corresponding authors: hlyanying6666@126.com Tel.:+86 15616156298*

Abstract

Existing graph-based ranking techniques for keyphrase extraction only consider the connections between words in a document, ignoring the impact of the sentence. Motivated by the fact that a word must be important if it appears in many important sentences, we propose to take full advantage of the reinforcement between words and sentences by melting three kinds of relationships between them. Moreover, a document is grouped with many topics. The extracted keyphrases should be synthetic in the sense that they should deal with all the main topics in a document. Inspired by this, we take topic model into consider. Experimental results show that our approach performs better than state-of-the-art keyphrase extraction method on two datasets under three evaluation metrics.

Keywords: Keyphrase extraction; graph-based; cluster; topic ;

1. Introduction

Keywords give a high level summarization of a document, which is vital for many areas of natural language processing, including document categorization, clustering and classification[1]. Before the emergence of the technology of automatic keyphrase extraction, the task is usually conducted by human which is very time-consuming. Moreover, the scale of Information is becoming larger owing to the development of the Internet and it is inefficient for professional human indexers to note documents with keyphrases manually. How to automatically extract the exact keyphrase of a given document becomes an important research problem and many approaches have generally appeared.

The task of keyphrase extraction usually conducts in two steps[2]: (1) extracting a bunch of words serving as candidate keyphrases and (2) determining the correct keyphrases using unsupervised or supervised approaches.

In the unsupervised approach, graph-based ranking methods perform the best[3]. These methods construct a word graph based on word co-occurrences within the document firstly and then ranking the words according to their scores. As a result, the top ranked words are the key words we want. However, this method just maintains a single score of a word without considering the impact of sentence which is composed of words. Motivated by the work of Wan[4], we propose to extend the word graph to three graphs, namely word to word graph, sentence to sentence graph and sentence to word graph. However, Wan's method has a disadvantage: the same as TextRank, it does not

Peer-review under responsibility of the scientific committee of the 7th International Congress of Information and Communication Technology
doi:10.1016/j.procs.2017.03.087

guarantee that the extracted keyphrases will deal with all main topics[2]. Motivated by the work by Liu[5] who proposes a clustering-based approach that groups semantically similar candidates making use of Wikipedia and co-occurrence-based statistics. We come up with the solution to avoid the weakness of Wan's model and propose to adopt a topic clustering algorithm on the W-W graph to generate the topic clusters, and then ensure that keyphrases are selected from every main topic cluster. Therefore, we can extract keyphrases related to the document and cover document's major topics at the same time. The new method is an improvement of Wan's work applied to keyphrase extraction.

In the experiments we find that our method outperforms other baseline methods under the three standard evaluation metrics on two authoritative data sets. Moreover, in this paper we just concentrate on the keyphrase extraction task but our model takes consider in the sentence, so we can get the document summary at the same time and the summary formed must have higher accuracy compared with other methods which conduct single document summary task.

2. Related Work

Some researchers[6] took keyphrase extraction for a classification problem and learned models for classification based on training data. These supervised methods require manually annotated training dataset, which costs lots of time. Because our approach is unsupervised, we focus merely on unsupervised techniques in this section. The most basic unsupervised method for keyphrase extraction is TFIDF[7]. TFIDF selects candidate keyphrases merely based on their statistical frequencies without considering semantic similarity between words. The method has a problem that it ignores the words which has low frequency but can be regarded as the keywords.

Starting from TextRank[3], graph-based ranking methods perform the best for keyphrase extraction task[8]. TextRank first build a word graph according to a given document in which the connections between words depict their semantic connections, which are often computed by the word co-occurrences in the document. After that, we can get the score of each word to get the ranking candidate keyphrase by executing PageRank[9] on the graph. In TextRank, a low-frequency word will get high score due to its neighbors with high frequency which can not happen in TFIDF. However, TextRank tend to construct a word graph merely based on word co-occurrences as a proximity of the semantic correlations between words. The correlations between words are often set to be the co-occurrence count inside a sliding window with maximum W words in the word sequence and this will bring in much noise due to the fact that semantically unrelated words may be connected and it splits up the sentence to pieces which ignores the fact that a word is important if it appears in important sentences. That is to say, the important words and important sentences can be extracted at the same time and this two processes can be both reinforced compared with single task. Zha[10] invents a method for simultaneous keyphrase extraction and text summarization by utilizing the various sentence-to-word relationships. Motivated by this, Wan[4] adds two hypothesizes into Zha's work. They intend to consider all the three kinds of relationships between sentences and words (word-word relation, sentence-sentence relation and word-sentence relation). The score of a word (sentence) is determined by both the score of related words and the score of related words. Given a document, the article constructs three graphs: word to word graph, word to sentence graph and sentence to sentence graph. The connections between nodes is respectively the similarities of words, sentences and word with sentence. Then construct the iteration formulas and get the ranking of the words and sentences. The top ranked words and sentences can be regarded as keyphrase and summary.

As mentioned in above section, the obtained keyphrases are not likely to deal with all the topics. To solve this problem, we propose to adopt a topic clustering algorithm on the graph. The method leverages clustering techniques on the graph to obtain several clusters and then choose the candidate keyphrase close to the centroid of each cluster. Clustering is a vital unsupervised learning problem which assigns objects to different groups. After clustering, similar objects are grouped into the same cluster and different objects will be divided into different clusters.

The rest of this paper is structured as follows: Section 3 will introduce the details of the construction of three graphs. Section 4 presents the iteration formulas which is used to obtain the scores of each word and sentence. Section 5 describes the clustering method. Section 6 conducts the experiments and discusses the details of evaluation results. We summarize our paper in Section 7.

3. Graph Building

When coming up with a new document, we split up the document to sentences and words. The obtained words are single characters other than phrases. Stop words will be then removed from the words connection and there does not exist identical words in the connection. We do not token the document which annotates the document using part-of-speech (POS) tags which most researchers use to promote the precision because the other words without noun words have effect on sentence score or word score as well.

Each sentence and word represents a point in the graph. The picture below depicts the three graphs we intend to build:

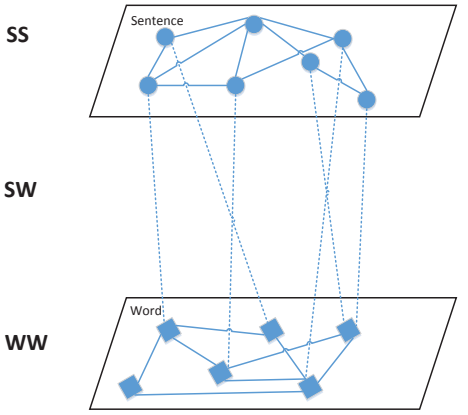

Fig. 1. The graph we will build according to a specific document. The circle represents a sentence and the square represents a word. The edge between two nodes are undirected and each has a corresponding weight which is computed below.

There exists an edge between every sentence (word) which means every sentence (word) is connected with each other. However, the sentences are not connected with all words due to the fact that every sentence has limited words and the words connection is composed of all the diverse words in the document. There must be the situation that a certain word does not appear in a sentence which means they should not be connected.

3.1. Sentence to Sentence graph (SS-Graph)

Every sentence is composed of several words so we can obtain a word set of a long sentence. If we want to obtain the similarity between two sentences, we need to transfer the sentence to mathematic expression and after that computation will proceed. In this paper, we represent a sentence as a vector and then compute the cosine of two vectors as the similarity between two sentences.

Given a sentence S_1, the word set of it is $T_1\{w_1, w_2, ..., w_n\}$. Given a sentence S_2, the word set of it is $T_2\{w_1', w_2', ..., w_m'\}$. So we can get the union set of S_1 and S_2: $T\{w_1, w_2, w_3, ..., w_N\}$. Obviously $N \le n+m$ because sentence can obtain the same words but word set can not. For example, there are two sentences: "I came from China.", "I came from Canada.". The word sets corresponding to the two sentences are $T_1\{I,\ came, from, China\}$ and $T_1\{I,\ came, from, Canada\}$. The union set T is $T\{I,\ came, from, China, Canada\}$. After that, we can obtain two vectors corresponding to the two sentences and the elements of the vector is the word frequency in S_1 and S_2. The number of the elements in set T is the dimensionality of the vector and the sequence of the frequency values is the same as the words listed in T. Obviously, the word occurs in sentence S_1 but not in sentence S_2 have zero frequency in corresponding position in S_2's vector.

After we obtain the specific vector of two sentences, then we can compute the cosine similarity to measure the

similarity of the two sentences. Here we construct an undirected graph G_{ss} to represent the document and the nodes of it are sentences in the document. Each edge connecting two nodes (sentences) has a corresponding weight which is decided by the similarity of two nodes (sentences). If a document has m sentences, we use a matrix $U_{m\times m}$ to save the weights. Therefore, U_{ij} represents the similarity of sentence S_i and sentence S_j. To make the obtained number scale at a reasonable level. We normalize matrix $U_{m\times m}$ to $\tilde{U}_{m\times m}$ which means the sum of each line is 1.

3.2. Word-to-Word graph (WW-graph)

Given the word set $W=\{w_i \mid 1\le i\le n\}$ of a document, the similarity between two words can be estimated through various methods such as semantic similarity, co-occurrence in a window W. Different from other most of the methods used in keyphrase extraction, we use word embeddings to measure the similarity of two words. Word embeddings are dense and real-valued vectors and they can encode both syntactic and semantic information. Compared with co-occurrence in a window W approach, this computation method take consideration of semantic factors and avoid some noise because of the undefined variable W to some extent. However, training word embeddings is a time-consuming and computationally expensive task. In this paper, we utilize one pre-trained publicly available embeddings: SENNA[11, 12]. SENNA is trained over Wikipedia including 130000 words for about 2 months. Cosine Similarity is one of the commonly used approaches to measure the semantic relatedness or similarity between two nodes in a vector space. The formulation is as follows:

$$sim(w_1, w_2) = \frac{\vec{w}_1 \bullet \vec{w}_2}{|\vec{w}_1| \bullet |\vec{w}_2|} \tag{1}$$

After that, we can build an undirected graph called G_{ww} to describe the word-word graph. The points in the graph represents one word and the weight associated with the edge is computed by word embeddings. Here we use a matrix V_{n*n} to describe the graph and each element of the matrix is the similarity of two nodes. That is $sim(w_i, w_j) = V_{ij}(i, j = 1...n)$ and if $i = j$, $V_{ij} = 0$ .Then normalize V to $\tilde{V}$.

3.3. Sentence-to-Word graph (SW-graph)

We can build an undirected graph G_{sw} based on G_{ss} and G_{ww} which connects the nodes in G_{ss} and G_{ww}. There exists an edge between two points if the word is contained in the sentence. Given the word set $W=\{w_j \mid 1\le j\le n\}$ and the sentence set $S=\{s_i \mid 1\le i\le m\}$, the weight between two nodes is:

$$\frac{wf_{w_j} \times isf_{w_j}}{\sum_{w\in s_i} wf_w \times isf_w} \tag{2}$$

w_j is a specific word in s_i and isf_w, wf_w are respectively the inverse sentence frequency and the word frequency in the sentence. They are defined as follows:

$$wf_{w_j} = \frac{count\ of\ w_j\ in\ s_i}{count\ of\ words\ in\ s_i} \tag{3}$$

$$isf_{w_j} = \log\left(\frac{total\ number\ of\ sentences}{number\ of\ sentences\ containing\ w_j + 1}\right) \tag{4}$$

Similar to the upper methods, we use a matrix $W_{m\times n}$ to save the weights of the edges and normalize it to $\tilde{W}_{m\times n}$. Moreover, we normalize the transpose of W to $\hat{W}$.

4. Graph Ranking Algorithm

Zha[10] introduces the first graph-based approach for simultaneous keyphrase extraction and summarization, inspired by an important observation: important words appear in important sentences and a sentence is important if it contains important words. Wan[4] add two assumptions based on Zha's work:

Assumption1: A word must be important if it is connected to other important words and a sentence must be important if it is connected to other important sentences.

Assumption2: A word must be important if it appears in many important sentences and a sentence must be important if it contains important words.

We use two column vectors $\vec{w}_{n\times 1}$ and $\vec{s}_{m\times 1}$ to represent the importance of the words and the sentences in one document. Therefore, given the assumptions, we can get the mathematical forms:

$$\begin{aligned} s_i &\propto \sum_j \tilde{U}_{ji} s_j \qquad & \mathrm{w}_j &\propto \sum_i \tilde{V}_i w \\ s_i &\propto \sum_j \tilde{W}_{ij} w_j \qquad & w_j &\propto \sum_i \hat{W}_{j\,i} s \end{aligned} \tag{5}$$

According to the above equations, we can get the iterative forms as follows

$$\begin{aligned} s_i &= \alpha \sum_{j=1}^{m} \tilde{U}_{ji} s_j + \beta \sum_{j=1}^{n} \tilde{W}_{ij} w_j \\ w_j &= \alpha \sum_{i=1}^{n} \tilde{V}_{ij} w_i + \beta \sum_{j=1}^{m} \hat{W}_{ji} s_i \end{aligned} \tag{6}$$

Where α and β are the relative contributions of the sentences and the words to the importance of a specific sentence or a word and we have $\alpha+\beta=1$.

5. Term Clustering

In this paper, we concentrate on the task of keyphrase extraction other than document summarization so we just need to apply clustering techniques on the word graph. That does not mean we do not take consideration of the effect of sentence to words because only when the iteration converges and the ranking of the words are obtained, the clustering method will be applied to the graph. After clustering, we can get several clusters and each cluster contains a bunch of words which are similar to a certain topic. Then we can select the words near the centroid of each cluster as keyphrases according to the importance we just obtained in above sections.

Here we use the widely used clustering algorithms K-means[13] to cluster the candidate keyphrase based on the word graph we build. The number of clusters is decided by the length of the document and we will explore the preference value through experiments.

6. Experiments

6.1. Datasets

We carry on our experiment on two standard datasets to evaluate the performance of our method. One dataset is built by Hulth2003[14] and this dataset contains 1460 abstracts of research articles. Each abstract has two kinds of manually labeled keyphrases, one controls the limit the other does not.
Another dataset is created by Luis Marujo who annotates the articles collected from the Web with a ranked list of key phrases. The corpus contains 900 articles and each article has a list of keyphrases. We call this dataset 500N in this paper.

6.2. Evaluation metrics

For evaluation, we use the commonly used metrics used by other researchers. That is precision/recall/F-measure. Following are their definitions,

$$precision = \frac{c_{correct}}{c_{extract}} \tag{7}$$

$$recall = \frac{c_{correct}}{c_{standard}} \tag{8}$$

$$F-measure = \frac{2pr}{p+r} \tag{9}$$

Where $c_{correct}$ is the number of correctly matched words, $c_{extract}$ is the number of extracted words, $c_{standard}$ is the number of assigned words.

6.3. Evaluation results

We here suppose that the contributions of sentences and words are equal which means that the parameter α and β are 0.5 each. The proposed method will be compared with several typical keyphrase extraction: **TF-IDF, TextRank.**

Here we intend to introduce the main knowledge of the three models:

TF-IDF The method estimates the scores of words according to two values:

$$TfIdf_{t,d} = \frac{N_t}{N_d} \times \lg \frac{D}{D_t} \tag{10}$$

N_t is the frequency of the word in the document and N_d is the number of words in the document. D is the number of the document and D_t is the number of the document which contains the specific word.

TextRank The method takes the document as a graph and the edge between two nodes depicts the semantic connection of two words. After that applying PageRank on the graph and get the ranking scores of each word. TextRank holds the view that the importance of a word $WS(v_i)$ is decided by the importance of the words connecting to it:

$$WS(v_i) = (1-d) + d \sum_{v_j \in Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \in Adj(v_j)} w_{jk}} WS(v_j) \tag{11}$$

w_{ij} is the similarity of node v_i and v_j and its property is decided by the co-occurrences in a window. $Adj(v_i)$ is the nodes which connect with v_i , d is the damping factor and often set to be 0.85.

Table 1 and Table 2 give the results.We compare our method with two baselines: TF-IDF, TextRank. Both of the basic methods fail to consider the effect of sentences to the importance of words. Moreover, the keyphrase they extract do not cover all the topics corresponding to the document.

Finally, we display the best results below. The number of clusters is set to be 2 on Hulth2003 datasets and 10 on DUC2001 datasets because the former dataset is composed of short abstracts which have less words. We extract 20% words of each cluster to be keyphrases.

Table 1: Comparing results on Hulth2003 datasets.

System	Precision	Recall	F-measure
TF - IDF	32.8	33.0	31.4
TextRank	39.7	40.1	37.3
Our Method	**43.0**	**40.2**	**39.6**

Table 2: Comparing results on 500N datasets

System	Precision	Recall	F-measure
TF - IDF	43.8	43.8	41.1
TextRank	48.6	**50.0**	47.7
Our Method	**48.7**	49.8	**47.8**

Seen from the table, our method outperforms the baseline approaches significantly. Seen from table 2, although the recall of TextRank is better than the results of ours but our F-measure and Precision are better than it. Because we fail to get the number of keyphrase extracted from each article by TextRank, there exists a probability that they may extract more words than we do and this can improve the value of recall.

7. Conclusions and Future work

In this paper we propose a graph-based method incorporating with clustering algorithm for keyphrase extraction. Experiments show that our method outperforms other baseline methods on two datasets. Compared with the old work conducted by Wan, we apply the clustering algorithm on the word graph and obtain an improved result which can cover the main topics the former fails to do.

Next some other works can be conducted on this task:

1. The clustering method can be modified due to the appearance of many clustering methods which outperforms K-Means such as Affinity Propagation, hierarchical clustering method.
2. The method only takes consider in a single document next we can make full use of corpus which has a bunch of documents similar to the specific document.

8. Acknowledgements

This work was supported by Scientific Research Fund of Hunan Provincial Education Department (No. 15A007)

9. References

1. C.D. Manning and H. Schutze. Foundations of statistical natural language processing. MIT Press. (2000)
2. Kazi Saidul Hasan and Vincent Ng.. Automatic Keyphrase Extraction: A survey of the State of the Art. (2015)
3. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. In Proceedings of EMNLP(2004)

4. Xiaojun Wan and Jianguo Xiao. CollabRank: Towards a collaborative approach to single-document keyphrase extraction. In Proceedings of the 22nd International Conference on Computational Linguistics, pages 969–976.(2008a)
5. Liu, Z., Li, P., Zheng, Y., Sun, M.: Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing 257–266 (2009)
6. Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. Domain-specific keyphrase extraction. In Proceedings of 16th International Joint Conference on Artificial Intelligence, pages 668–673(1999)
7. Gerard Salton and Christopher Buckley. Termweighting approaches in automatic text retrieval. Information Processing and Management, 24(5):513–523)(1988)
8. Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 366–376.(2010)
9. Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine.Computer Networks, 30(1–7):107–117(1998)
10. Hongyuan Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In Proceedingsof 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 113–120(2002)
11. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. Kuksa. Natural Language Processing (Almost) from Scratch, Journal of Machine Learning Research (JMLR),(2011)
12. R. Collobert. Deep Learning for Efficient Discriminative Parsing, in International Conference on Artificial Intelligence and Statistics (AISTATS), (2011)
13. Hartigan J A, Wong M A. Algorithm AS 136: A K-Means Clustering Algorithm[J]. Applied Statistics, 28(1):100-108.(1979)
14. Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In Proceedings of EMNLP, pages 216–223(2003)
15. Xiaojun Wan, Jianguo Xiao.Single Document Keyphrase Extraction Using Neighborhood Knowledge. Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence(2008)
16. Luis Marujo, Anatole Gershman, Jaime Carbonell, Robert Frederking, João P. Neto, Supervised Topical Key Phrase Extraction of News Stories using Crowdsourcing, Light Filtering and Co-reference Normalization, Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC '12), Istanbul, Turkey, May 2012
17. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. Kuksa. Natural Language Processing (Almost) from Scratch, Journal of Machine Learning Research (JMLR), (2011)
18. Brendan J J. Frey and Delbert Dueck. 2007. Clustering by passing messages between data points. Science.(2007)
19. Matsuo, Y., Ishizuka, M.: Keyword extraction from a single document using word co-occurrence statistical information. International Journal on Artificial Intelligence Tools 157–169 (2004)
20. Grineva, M., Grinev, M., Lizorkin, D.: Extracting key terms from noisy and multitheme documents. In: Proceedings of the 18th International Conference on World Wide Web, pp. 661–670. ACM (2009)
21. Liu, Z., Chen, X., Zheng, Y., Sun, M.: Automatic keyphrase extraction by bridging vocabulary gap. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learningpp. 135–144 (2011)
22. Collobert R, Weston J, Bottou, et al. Natural Language Processing (Almost) from Scratch[J]. Journal of Machine Learning Research, 12(1):2493-2537.(2011)