The Improvements of Text Rank for Domain-Specific Key Phrase Extraction

Zhijuan Wang *, Yinghui Feng, Fuxian Li

College of Information Engineering, Minzu University of China, Minority Languages Branch, National Language Resource Monitoring & Research Center Beijing, 100081, China

***Abstract* — TextRank is a common method to extract keyphrases which are important for many tasks of Natural Language Processing. Although Term Frequency and Inverse Document Frequency (TFIDF) is used to calculate the node weight in improved TextRank method for Keyphrase extraction in previous work, it performs poorly when extracting domain-specific keyphrase. In this paper, we introduce Average Term Frequency(ATF) and Document Frequency(DF) to calculate the node weight. Further, we incorporate node weights and semantic relationship of two words into a new method-Hybrid TextRank for extracting Keyphrase from specific domain. When using these improvements approaches to extract keyphrases from a Tibetan religion domain, experiments demonstrate that hybrid TextRank is better than the others and its precisions of Top 10, 20, 30 words reach 90%, 85%, 80%.**

Keywords - TextRank; Keyphrase Extraction; Tibetan religion domain

I. INTRODUCTION

Keyphrases are phrases that capture the main topics discussed in a given document [1]. Keyphrase extraction is defined as the automatic selection of important, topical phrases from within the body of a document [1]. In this paper, we study the keyphrase extraction method not for one document but for specific domain.

Keyphrase extraction is important for many Nature Language Processing (NLP) tasks such as document summaries, document classification, information retrial, and so on. Its pole is to select important and topical phrases from a document or a domain corpus. However, the task is far from being solved: state-of-the-art performance on keyphrase extraction is still much lower than that on many core NLP tasks [2].

The similarity of keyphrase extraction from a document and a domain is that they can use similar algorithms. The difference lies in that domain-specific keyphrase extraction needs more domain related key information while document keyphrase extraction only concern about the topic of one document. The main task of this paper is to research how to improve domain-specific keyphrase extraction based on TextRank.

The domain-specific Keyphrase extraction includes three steps.

Firstly, construct a specified domain corpus.

Secondly, extract a list of words or phrases from the specific domain that serves as candidate keyphrases. Keyphrases should annotate the main meaning of a domain and they are usually nouns, adjectives, and verbs but they should not be meaningless words such as stop words. Therefore, in order to get a good candidate keyphrases, we should use stopword list to remove stop words and allow words with certain part of speech tags.

Thirdly, select keyphrases from candidate keyphrase list using supervised or unsupervised approaches which will be described below.

A. Supervised approaches

The main task for early supervised approaches is to train a classifier to determine whether a candidate phrase is a keyphrase or not. Some algorithms have been used to train this classifier: naive Bayes [3,4], decision trees [1,5], boosting [6], maximum entropy [7], support vector machines [8,9], and so on. Using these methods, we get a list of keyphrases only, where all keyphrases are equally important. But in fact, different Keyphrase has different importance (or weight) in identifying a domain.

Therefore, what we need is a ranking of keyphrases rather than a list of keyphrases. Jiang propose a ranking approach to keyphrase extraction [8], where the goal is to design a ranker to rank two candidate keyphrases. Using this idea, KEA shown significantly outperform [3,4], a popular supervised baseline that adopts the traditional supervised classification approach [10,11].

Besides, in order to get a better ranking of keyphrases, some features such as statistical features, structural features and syntactic features are often used when extracting keyphrase in supervised approaches.

B. Unsupervised approaches

Unsupervised approaches of keyphrase extraction can be categorized into four groups [2]: graph-based ranking, topic-based, simultaneous learning and language modeling.

Since the purpose of this paper is introducing our improved approaches of keyphrase extraction based on TextRank, which is one of graph-based ranking methods, graph-based ranking is discussed briefly here.

For graph-based ranking, a text or a domain corpus is represented as a graph in which each node presents one candidate keyphrase and each edge express the relationship between two related candidates. The weight of each edge is proportional to the syntactic and/or semantic relevance between the connected candidates. For every node, each of its edges is treated as a "vote" from the other node connected by the edge. The top-ranked candidates from the graph are selected as keyphrases for the input document.

TextRank is one of the most well-known graph-based approaches to keyphrase extraction. We will use its improved approaches to extract keyphrase from a domain corpus.

The plan of the paper is as follows. In the next section, we describe domain corpus and experiment. In Section 3 we introduce TextRank. In section 4 we present improved approaches based on TextRank in detail. Experiment and results of improved approaches are also given in this section.

Finally, Section 5 concludes the paper.

II. CORPUS AND EXPERIMENT

A. Corpus

Domain keywords serve as a highly condensed summary for a domain, and they can be used as labels for a domain and be used in text classification, information retrial and so on.

Here, Gelugpa, a subdomain of Tibetan religion, is selected. Gelugpa is one of the most important sects of Tibetan Buddhism. But there is little research on Gelugpa keyphrase. Therefore, our evaluation use improved methods to extract keyphrases from Gelugpa corpus. In our experiment, 50 documents and 26504 words are used.

B. Experiment Design

Fig. 1 shows the flow chat of extracting domain keyphrases.

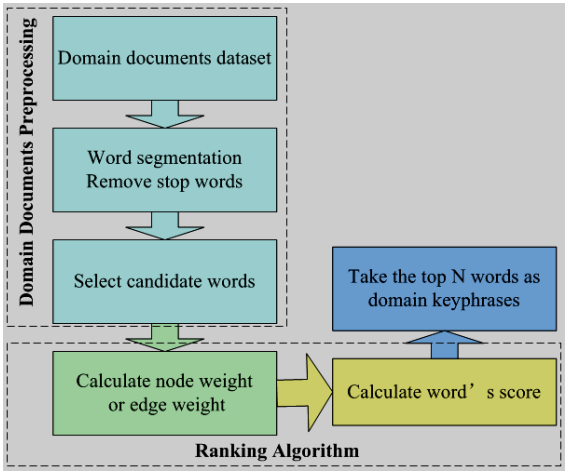

Fig.1 The Flow Chat of Extracting Keyphrases from a Domain Corpus

The first step is domain documents preprocessing including preparing the domain documents dataset, word segmentation, removing stop words and selecting candidate words. The free word segment tool ICTCLAS Segmenter (http://ictclas.org) is used here.

The second step is calculating node weight or edge weight and word's score using ranking algorithm on the prepared dataset.

Finally, take the top N words as domain keyphrases.

Here, the precisions of Top N keyphrases are used to evaluate the improved methods based on TextRank.

III. RELATED WORK

TextRank is a well-known method to extract keyphrases. It is inspired from PageRank and proposed in 2004 [12].

The formula of TextRank is shown in formula (1).

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_j)} w_{jk}} WS(V_j) \quad (1)$$

Where $WS(V_i)$ is the score of node V_i.

d is the damping factor that can be set from 0 to 1, which represents the probability of jumping from a given node to another random node in the graph. The value of d is usually set to 0.85.

w_{ji} is the weight of the edge from the previous node V_j to the current node V_i. $In(V_i)$ is the set of nodes that point to it (predecessors). $Out(V_j)$ is the set of nodes that node V_i points to (successors). $\sum_{v_k \in Out(V_j)} w_{jk}$ is the summation of all edge weights in the previous node V_j.

w_{ji} is defined as the frequency that V_j and V_i appear in a window of maximum L words in the associated text, where L□[2,10] [13].

IV. THE IMPROVEMENT APPROVACHES OF TEXTRANK

From the formula of TextRank, we can see:

(1) TextRank only takes edge weights into account. Node weights are also very important for node scores.

(2) The edge weight of two words is only counted by the co-occurrence of two words in a window. The semantic relationship of two words does not been taken into account.

Therefore, the TextRank can be improved in considering node weights and semantic relationships between words.

A. Improving TextRank based on the Node Weight

1) Improved TextRank Algorithm

For TextRank, the score of V_i only takes edge weights into account. In fact, the weight of V_i, called node weight, is also important for calculating the score of Vi. Bellaachia and Al-Dhelaan proposed the NE-Rank(Node weight and Edge weight) in 2012. It can be written in formula (2).

$$WS(V_i) = (1-d) * W(V_i) + d * W(V_i) * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_j)} w_{jk}} WS(V_j) \quad (2)$$

Where $W(V_i)$ is the weight of current node(V_i).

Term frequency and document frequency are often used to compute the node weight.

TFIDF is used to compute the node weight [14]. It can be written in (3).

$$W(V_i)_{TFIDF} = TF(V_i) * \log_2 \frac{N}{DF(V_i)} \quad (3)$$

The method combing formula (2) and (3) is referred as NERank_tfidf. Although this method performs well on extracting keyphrases from a document, it performs poorly when we want to extract keyphrases from document set (for example from a domain corpus) without giving a higher weight for DF (document frequency). Therefore, in this situation, ATF (Average Term Frequency) and DF (Document Frequency) are used. Then we can calculate $W(V_i)$ it according to formula (4).

$$W(V_i) = ATF(V_i) * DF(V_i) \quad (4)$$

The method combing formula (2) and (4) is referred as NERank_atfdf.

In formula (4), ATF and DF are equally important. But, in fact, for domain-specific keyphrases, the DF of a word is more important than its ATF. So we prefer to use formula (5) to calculate the node's weight.

$$W(V_i) = \sqrt{A\,TF(V_i)} * e^{DF(V_i)} \quad (5)$$

The method combining formula (2) and (5) is referred as NERank_sqrt(atf)exp(df).

2) Evaluation

In our experiment, we use method TextRank, NERank_tfidf, NERank_atfdf and NE-Rank_sqrt(atf)exp(df) to extract keyphrases-fromGelugpa corpus separately.

Fig. 2 demonstrates the comparison in precision of these four methods. We can clearly see:

(1) The precisions of two improved methods using ATF and DF (NERank_atfdf and NERank_sqrt(atf)exp(df)) have higher precision than TextRank's.

(2) NERank_tfidf performs poorly in extracting keyphrase from domain.

(3) NERank_sqrt(atf)*exp(df) has the best result among the four methods.

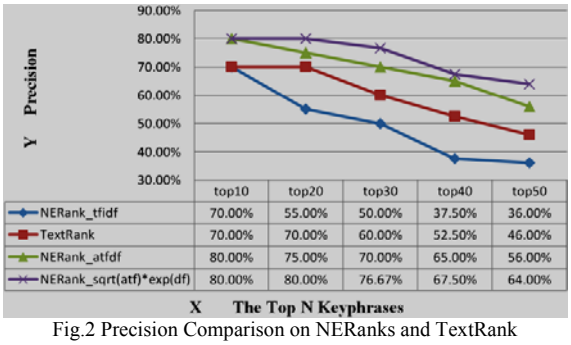

Fig.2 Precision Comparison on NERanks and TextRank

B. Improving TextRank based on the Edge Weight

For TextRank, edge weight reflects the relationship of two words, which can be gotten with the local information and global information of words. The relationship of two words that is gotten from the content of single document is called local words relationship. While the relationship of two words that is gotten from the contents of a document set is called global words relationship.

1) Local Words Relationship

Co-occurrence is a common method to calculate the local relationship of two words. TextRank uses co-occurrence times of two words (V_i,V_j) in a window(L) to calculate the edge weight of two words (W_{ij}). W_{ij} can be calculated by formula (6).

$$W_{ij} = n_L(W_i, W_j) \quad (6)$$

Where $n_L(W_i,W_j)$ is the co-occurrence times in a window L□[2,10] [15]. The formula (6) is referred as local words relationship.

In TextRank, the semantic relationships of two words are not considered. However , from our experiment, it indicates that the edge weight of two words which have higher semantic similarity is bigger than the edge weight of words with lower semantic similarity.

2) Global Words Relationship

Global words relationship is also called word semantic similarity, which can be gotten by analyzing the contents of a document set. It reflects the semantic similarity of two words [16,17]. The global words relationship can be calculated by using mutual information, Google Distance and Latent Semantic Model [18]. It can also be calculated by Hownet [19], which is denoted as $S_G(W_i, W_j)$.

Global words relationship shows general semantic relationship [20]. Therefore, it can be used to calculate the edge weight when we extract keyphrases from a domain corpus.

3) SemanticRank

SemanticRank [20] is proposed on the basis of TextRank. This method combined local words relationship and global words relationship. It is expressed in formula (7).

$$e(W_i, W_j) = n_L(W_i, W_j)((1-\varepsilon)S_G(W_i, W_j) + \varepsilon) \quad (7)$$

Here, ε is a weight that can be set from 0 to 1. It controls the proportion of global words relations in this formula. When $\varepsilon \neq 1$, the local words relationship and global words relationship are used. When ε is equal to 1, SemanticRank becomes the normal TextRank.

4.2.4 Evaluation

We use the SemanticRank and TextRank to extract keyphrase from Gelugpa corpus. Here ε is set to 0.8 and $S_G(W_i, W_j)$ is calculated with Hownet.

Fig. 3 shows the precision comparison of these two methods. We can see that the precision of SemanticRank is better than TextRank.

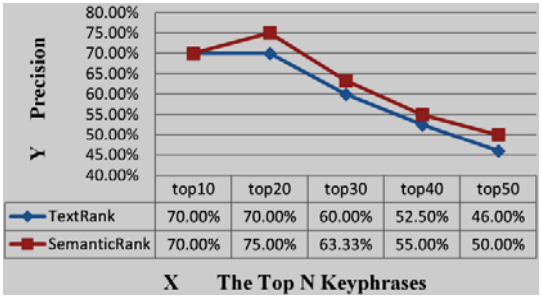

Fig.3 Precision Comparison on SemanticRank and TextRank

C. Improving TextRank based on the Node Weight and Edge Weight

1) Hybrid TextRank

From above, we can see TextRank can be improved by using node weight and edge weight independently.

This inspires us to think naturally that can we get better result if we combine node weight and edge weight? Experiment confirmed our supposition. We call this new method as Hybrid TextRank. This method is expressed in formula (8).

$$WS(V_i)=(1-d)*W(V_i)+d*W(V_i)*\sum_{V_j\in In(V_i)}\frac{e(W_j,W_i)}{\sum_{v_k\in Out(V_j)}e(W_j,W_k)}WS(V_j) \quad (8)$$

Where node weight $W(V_i)$ is calculated with $\sqrt{ATF(V_i)}*e^{DF(V_i)}$.

The edge weight is calculated using local word relationship and global words relationship, $n_L(W_i,W_j)((1-\varepsilon)S_G(W_i,W_j)+\varepsilon)$. Here, $\varepsilon=0.8$ and **Error! Reference source not found.** calculated with Hownet.

$n_L(W_i,W_j)$ is the co-occurrence times in a window L=5.

2) Evaluation

We use the Hybrid TextRank, SemanticRank, NERank_sqrt(arf)*exp(df) and TextRank to extract keyphrases from Gelugpa corpus separately. Fig. 4 shows the precision comparison of the four methods. We can clearly see that the precision of the hybrid TextRank is better than others

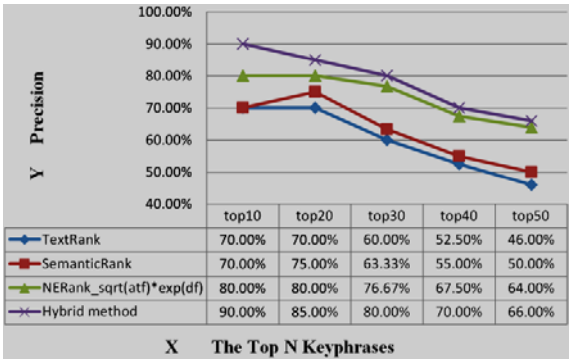

Fig.4 Precision Comparison on Hybrid TextRank, SemanticRank ,NERank_sqrt(atf)exp(df) and TextRank

V. CONCLUSIONS

Domain keyphrase extraction is important for many nature language processing tasks. Using node weight and edge weight, we analyzed the improvement of TextRank. Experiment indicates that node weight that is calculated by ATF and DF can improve the precision of TextRank markedly. Then combining node weight and edge weight method, Hybrid TextRank, is proposed. Experiments show that the hybrid TextRank has the highest precision when we extract Top 10, 20, 30, 40, 50 words from a domain corpus.

Experiments also demonstrate that the extraction precision of hybrid TextRank can reach 90%, 85%, 80% for top 10, 20, 30 words when extracting keyphrases from a Tibetan religion domain. This method can be extended to extract keyphrases from other domains.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflicts of interest.

ACKNOWLEDGMENT

The research was sponsored by Key Program of National Natural Science Foundation of China (No. 61331013), Projects of The Chinese Language Committee (No. WT125-46 and WT125-11) and Graduate Student Project of Minority Languages Branch, National Language Resource Monitoring & Research Center (No.CML15A02).

REFERENCES

[1] Peter Turney. "Learning algorithms for keyphrase extraction. Information Retrieval", *Information Retrieval*, 2(4), 2002, pp. 303–336.

[2] Kazi Saidul Hasan and Vincent Ng. "Automatic Keyphrase Extraction: A Survey of the State of the Art". In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014, pp. 1262–1273.

[3] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. "KEA: Practical automatic keyphrase extraction". In Proceedings of the 4th ACM Conference on Digital Libraries, 1999, pp. 254–255.

[4] 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, 1999, pp. 668–673.

[5] Peter Turney. "Learning to extract keyphrase from text". National Research Council Canada, Institute for Information Technology, Technical Report ERB-1057, 1999.

[6] Anette Hulth, Jussi Karlgren, Anna Jonsson, Henrik-Bostr¨om, and Lars Asker. "Automatic keyword extraction using domain knowledge". In Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Text Processing, 2001, pp. 472–482.

[7] Wen-Tau Yih, Joshua Goodman, and Vitor R. Carvalho. "Finding advertising keywords on web pages". In Proceedings of the 15th International conference on World Wide Web, 2006, pp. 213–222.

[8] Xin Jiang, Yunhua Hu, and Hang Li. "A ranking approach to keyphrase extraction". In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 756–757.

[9] Patrice Lopez and Laurent Romary. "HUMB: Automatic key term extraction from scientific articles in GROBID". In Proceedings of the 5th International Workshop on Semantic Evaluation, 2010, pp. 248–251.
[10] Min Song, Il-Yeol Song, and Xiaohua Hu. "KPSpotter: A flexible information gain-based keyphrase extraction system". In Proceedings of the5th ACM International Workshop on Web Information and Data Management, 2003, pp. 50–53.
[11] Daniel Kelleher and Saturnino Luz. "Automatic hypertext keyphrase detection". In Proceedings of the 19th International Joint Conference on Artificial Intelligence, 2005, pp. 1608–1609.
[12] Rada Mihalcea and Paul Tarau. "TextRank: Bringing order into texts". In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004, pp. 404–411.
[13] Kazi Saidul Hasan and Vincent Ng. "Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art". In Proceedings of the 23rd International Conference on Computational Linguistics: Posters, 2010, pp. 365–373.
[14] Bellaachia A, Al-Dhelaan M. "NE-Rank: A Novel Graph-Based Keyphrase Extraction in Twitter". *IEEE/WIC/ACM International Conferences on Web Intelligence & Intelligent Agent Technology*, 2012, pp. 372-379.
[15] Hasan K S, Ng V. "Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art". Proceedings of the 23rd International Conference on Computational Linguistics: Posters. *Association for Computational Linguistics*, 2010, pp. 365-373.
[16] Liu Z, Li P, Zheng Y, et al. "Clustering to Find Exemplar Terms for Keyphrase Extraction". Proceedings of EMNLP, 2009, pp. 257–266.
[17] Grineva M, Grinev M, Lizorkin D. "Extracting Key Terms from Noisy and Multitheme Documents". Proceedings of WWW, 2009, pp. 661–670.
[18] Cilibrasi R, et al. "The google similarity distance". *IEEE Transactions on knowledge and data engineering*, 2007, pp. 370–383.
[19] Qun Liu and Sujian Li. "Word Similarity Computing Based on Hownet". *Computational Linguistics and Chinese Language Processing*, 7(2), 2002, pp. 59-76.
[20] Xiance Si. "Content based Recommendation and Analysis of Social Tags". Tsinghua University, 2010.