Extracting Keyphrases from Chinese News Articles Using TextRank and Query Log Knowledge *

Weiming Liang[a], Chang-Ning Huang[b], Mu Li[b], and Bao-Liang Lu[a,c]

[a] Center for Brain-Like Computing and Machine Intelligence
Department of Computer Science and Engineering
Shanghai Jiao Tong University
800 Dong Chuan Road, Shanghai 200240, China
bllu@sjtu.edu.cn
[b] Microsoft Research Asia
49 Zhichun Road, Haidian District, Beijing 100080, China
{v-cnh, muli}@microsoft.com
[c] MOE-Microsoft Key Laboratory for Intelligent Computing and Intelligent Systems
Shanghai Jiao Tong University
800 Dong Chuan Road, Shanghai 200240, China

Abstract. Keyphrases extracted from articles are beneficial in helping people boost browsing speed, but unfortunately keyphrases are rarely available for news articles due to the high expense of labor and time for manual annotation. This paper proposes a practical approach to extracting keyphrases for Chinese news articles using the TextRank and query log knowledge. Previous work is word based, while our approach uses phrase as its basic element. We generate phrases by employing several statistical criteria with the huge amount of queries as a training corpus. We use TextRank, a graph-based learning algorithm, for extracting keyphrases from Chinese news articles. In addition, two instructive features, lengths and positions of phrases, are incorporated into the TextRank model. Experimental results demonstrate that our methods improve the performance significantly.

Keywords: keyphrase extraction, Chinese News Articles, TextRank, query log.

1 Introduction

Keyphrase extraction algorithms can be classified into supervised or unsupervised. When a large training corpus is available, a supervised learning algorithm is a possible solution. Statistical information, such as frequencies, positions and lengths of phrases, was proved to be important features to supervised algorithm. GenEx (Turney, 1999) used a hybrid genetic algorithm to evaluate the parameters of these features. These parameters were used to assign each phrase a score and the ones with higher scores were output. Frank *et al.* (1999) also developed a simple but efficient keyphrase extraction system based on naive Bayes training algorithm. Supervised training algorithms are limited to a specific field because of high dependency on the training corpus. Many works use academic papers as training datasets because authors are always required to provide keyphrases for their papers. But news articles, which are the most widely used application of keyphrase extraction, lack training corpora.

* W.Liang and B.L.Lu are supported by the National Natural Science Foundation of China (Grant No. 60773090 and Grant No. 90820018), the National Basic Research Program of China (Grant No. 2009CB320901), and the National High-Tech Research Program of China (Grant No. 2008AA02Z315). This work was done while the first author visited Microsoft Research Asia.

For a great number of unlabelled news articles, an unsupervised algorithm is a good choice. Mihalcea *et al.* (2004) introduced a PageRank-based (Brin and Page, 1998) model called TextRank to extract keyphrases. But the TextRank does not make full use of the statistical information, such as the length and the position of the phrase. In this paper, we propose a method which incorporates various features into the TextRank which can improve the F-measure by 8.37%.

Yang *et al.* (2008) also developed a PageRank-based algorithm which used average term frequency and proportional document frequency to extract Chinese keywords instead of keyphrases. A single word often fails to convey a certain meaning. For example, "金融/危机" (financial crisis) is a hot topic nowadays. However, neither "金融" (finance) nor "危机" (crisis) individually can express the meaning of the whole phrase "金融危机" (financial crisis). The TextRank generates keyphrases by combining the adjacent keyword sequences after the PageRank iteration, while we use a more efficient method which generates phrases before the PageRank iteration. Furthermore, query logs are used to help determine the phrase boundary. Experimental results show that our phrase generation method improves the F-measure by 10.37%.

We propose an efficient and practical algorithm to extract keyphrases from Chinese news articles. First, we apply several statistical criteria to generate phrases. Query logs are used to help determine the phrase boundary. Then, we propose a method which can integrate various features into the TextRank. Our test set covers various areas of news. All the news are manually assigned keyphrases by a third party. Experimental results show that our algorithm can improve the performance significantly.

We introduce the framework of our algorithm in section 2. The phrase generation method is described in section 3. The features are introduced in section 4. The experimental results are shown in section 5. Finally, we give our conclusion and future work in section 6.

2 TextRank

The TextRank (Mihalcea *et al.*, 2004) model is the framework. It is based on the PageRank (Brin and Page, 1998). Brin and Page developed the PageRank model to generate a probability distribution over pages in a network. A page is more "important" than others if it has a higher probability. The matrix form of the PageRank model is

$$\vec{p} = [\frac{1-d}{n}U + dM^T]\vec{p} = X\vec{p}, \tag{1}$$

where $\vec{p}$ is a probability vector of pages; d is a damping factor which always set to 0.85; $U_{n\times n}$ is a matrix with all the elements equal to 1; M is a weight matrix of links. We can find the stationary distribution by a simple iterative algorithm.

In the TextRank, "phrase" is "page", and the semantic similarity between "phrases" is the "link" between "pages". The TextRank builds transition matrix M from co-occurrence information. A slide window is set first. If v_i and v_j appear in the same slide window, they are counted as co-occurrence $m_{ij}^{'} = co(v_i, v_j)$, where $m_{ij}^{'}$ is the initial weight of the edge $v_i \rightarrow v_j$, and $co(v_i, v_j)$ is the co-occurrence times of v_i and v_j. The transition matrix $M = [m_{ij}]_{n\times n}$ is given by

$$m_{ij} = m_{ij}^{'} / \sum_{j=1}^{n} m_{ij}^{'}. \tag{2}$$

3 Phrase Generation

Chinese is different from English because there is no explicit separator between words in a sentence. Segmentation converts a sequence of continuous Chinese characters into a sequence of

delimited words. It is a necessary step for most Chinese information processing systems including keyphrase extraction. S-MSRSeg (Gao *et al.*, 2003) is used as our segmentation tool[1].

One crucial step to generate phrases from words is to determine the phrase boundary. In TextRank, only single words take part in the PageRank iteration. After a small number of candidate keywords have been extracted, the sequences of adjacent keywords are merged into keyphrases. One disadvantage of this method is that not all parts of the keyphrase can always be extracted correctly as keywords. Any loss of the adjacent keywords will cause the failure of keyphrase generation. In this section, we introduce three statistical criteria to generate phrases before the PageRank iteration.

A straightforward criterion is frequency. For an article A, a word sequence $w_1..w_k$ should be a phrase if

$$\begin{cases} fre(w_1..w_k, A) \geq TH_{lfre} \\ w_i \notin ST, i = 1, 2, ..., k \end{cases}, \tag{3}$$

where $fre(w_1..w_k, A)$ is the frequency of $w_1..w_k$ in article A, TH_{lfre} is the threshold, ST is the stop word collection. We call this criterion "local frequency" because the frequency is based on the article from which we extract keyphrases. The advantage of this criterion is robust and efficient, especially for domain specific articles.

Boundary entropy (BE) (Zhao and Kit, 2008) measures the boundary by entropy.

$$BE(w_1..w_k) = -\sum_{w \in C} p(w|w_1..w_k) \log p(w|w_1..w_k), \tag{4}$$

where w is a Chinese character, $p(w|w_1..w_k)$ is the probability of $w_1..w_k$ adjacent to w. As w can be right or left to $w_1..w_k$, two types of BE named BE_r and BE_l can be defined.

Feng *et al.* (2004) and Zhao and Kit (2008) used Access Variety (AV) to measure the boundary of Chinese words as

$$AV(w_1..w_k) = \log RL_{av}(w_1..w_k), \tag{5}$$

where $RL_{av}(w_1..w_k)$ is the number of the distinct Chinese characters which adjacent to $w_1..w_k$. Also, we can define left access variety (AV_l) and right access variety (AV_r).

Both "BE" and "AV" need a lot of samples to estimate the phrase boundary. We use query logs which contain more than 400 million queries as our samples. One of the advantages of using query logs is that the contents of queries are always what people are interested in. It can help us generate the phrases in conformity with the reading habits of customers. We assume that each query has a reasonable left boundary. Only the queries that start with word sequence $w_1..w_k$ will be counted. In this way, the left boundary of $w_1..w_k$ is reasonable. So we just need to measure the right boundary. Table 1 shows part of the query log. For example, if we need to determine the boundary of the word sequence ".../爆发/金融/危机/..." (the outbreak of the financial crisis). First, we calculate the right "BE" value and "AV" value of "爆发金融" (outbreak finance). Because no query starts with "爆发金融", it is not a qualified phrase. Then, we calculate the right "BE" value and "AV" value of "金融危机" (financial crisis). We search all queries start with it. Its right "BE" value is 1.73 and its right "AV" value is 2.60. We consider it as a qualified phrase. We can accelerate the process by sorting the queries first.

For boundary entropy, word sequence $w_1..w_k$ should be a phrase if

$$\begin{cases} BE_r(w_1..w_k, Q) \geq TH_{be} \\ w_i \notin ST, i = 1, 2, ..., k \end{cases}, \tag{6}$$

[1] http://research.microsoft.com/en-us/downloads/7a2bb7ee-35e6-40d7-a3f1-0b743a56b424/default.aspx

Table 1: Part of query log

queries	frequencies
...	
金融危局	1
金融危机	537
金融危机　个人理财	1
金融危机　中国银行	1
金融危机　老百姓　关系	1
...	
金融危机，保险	6
金融危机，保险，中英人寿	1
金融历史	3
...	

where $BE_r(w_1..w_k, Q)$ is the right boundary entropy of $w_1..w_k$ which is estimated on query log set Q; TH_{be} is the threshold.

In a similar way, for access variety, $w_1..w_k$ should be a phrase if

$$\begin{cases} AV_r(w_1..w_k, Q) \geq TH_{av} \\ w_i \notin ST, i = 1, 2, ..., k \end{cases}, \tag{7}$$

where $AV_r(w_1..w_k, Q)$ is the right access variety of $w_1..w_k$ which is estimated on query log set Q; TH_{av} is the threshold.

Our generation algorithm can also recover the loss in segmentation process. For example, "房利美"(Federal National Mortgage Association) is an American company. But the segmentation tool outputs '房'(house) / '利'(profit) / '美'(beautiful), because it is out of vocabulary (OOV). However it has a high "BE" value in the query log. Our phrase generation algorithm can correct this kind of mistake.

4 Features

In the TextRank, the transition matrix M is only determined by the co-occurrence of words. However, wealth statistic information such as length of phrase and position of phrase are not applied. In this section, we proposed a method which can incorporate "length" and "position" into the model.

4.1 Length of Phrase

Generally, long phrases are more informative than short ones. Most valuable phrases have the length between four characters and six characters. So phrases with different lengths have different weights. The "length" feature is used by the following modification

$$\forall(i, j), m'_{ij} \leftarrow m'_{ij} \times s_l(v_j), \tag{8}$$

where m'_{ij} is the weight of the edge $v_i \rightarrow v_j$; $s_l(v_j)$ is the length weight of v_j. In this way, the importance of v_j is affected by its length weight. If $s_l(v_j)$ is greater than 1, the links from all other nodes point to v_j will increase.

4.2 Position of Phrase

The title and the first paragraph are usually summary of an article. Most of the keyphrases tend to appear in the title and the first paragraph. So the phrases which appear in the title and the first paragraph have bigger weights than others. The "position" feature is used by the following modification

$$\forall(i, j), m'_{ij} \leftarrow m'_{ij} \times s_p(v_j), \tag{9}$$

where $s_p(v_j)$ is the position weight of v_j.

5 Experiments

5.1 Dataset

Four categories of news, IT, Telecom, Electrical, Internet, are crawled from Sina[2]. We randomly choose 100 news articles from each category as test data (400 articles in all). All test data are manually assigned keyphrases by two independent annotators from a third party. The keyphrase set assigned by the annotator is called *truth set*. Detail is shown in Table 2.

Table 2: Truth set detail. Truth set 1 and truth set 2 are two independent sets of keyphrases assigned by third party.

	Ave. # of KPs per art.	Ave. # of cha. per KP
Truth Set 1	6.17	4.1
Truth Set 2	6.93	5.4
Average	6.55	4.75

We test the agreement between two truth sets, treating truth set 1 as standard and truth set 2 as predicted. The result is shown in Table 3. It seems that there exists a large difference between the two truth sets. So, Chinese keyphrase extraction is a difficult task.

Table 3: Agreement between truth set 1 and truth set 2

Precision	Recall	F-measure
59.2	64.8	63.0

5.2 Evaluation measure

We use "precision", "recall" and "F-measure" as experiment criteria. First, we calculate the "precision" and "recall" of every article. Then, we calculate the average "precision" and "recall" of all articles. Then, "F-measure" is calculate as follows

$$F - measure = \frac{2 \times Precision_{ave} \times Recall_{ave}}{Precision_{ave} + Recall_{ave}}. \tag{10}$$

5.3 Experimental Results

First, we evaluate the performance of the features: "length of phrase (L)" and "position of phrase (P)". The TextRank is used as our baseline, in which no feature are used. For a better comparison, all the experiments use our phrase generation method, and all phrase generation criteria are used, the length of the slide window is 7, the number of the output keyphrases is 8. The weights of the lengths and the positions are shown as

$$s_{l2} : s_{l3} : s_{l4-6} : s_{l7-} = 3 : 8 : 10 : 8 \tag{11}$$

$$s_{title} : s_{fisrt} : s_{other} = 5 : 3 : 2 \tag{12}$$

where s_{li} is the weight of the phrases with the length i, s_{l4-6} is the weight of phrases with length between 4 and 6, s_{l7-} is the weight of phrases with the length equal or greater than 7, s_{title}, s_{fisrt}, s_{other} are the weights of the title, the first paragraph, and the other words separately. All parameters are tuned on a small development set.

Figure 1 shows the detail. "length of phrase (L)" performs better than "position of phrase (P)". Recall that we assign the highest weights to phrases with length between 4 and 6. In fact, in two truth sets, the proportions of the keyphrases with length between 4 and 6 are 45% and

[2] http://www.sina.com.cn

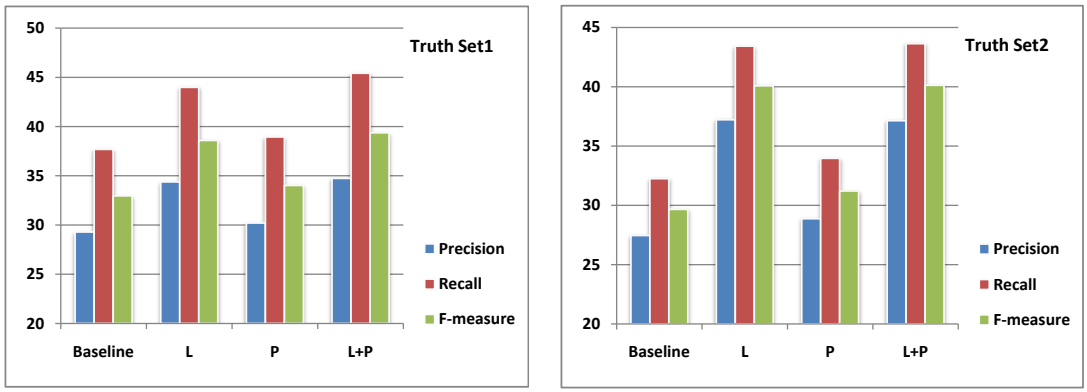

Figure 1: Performance of features

50% respectively. Therefore "length of phrase" is an important feature. "Position of phrase" is also effect, although the improvement is slight. We also evaluate the combination of two features ("L+P"). The performance is slightly improved. Compared with the baseline, "L+P" improves the F-measure by 6.42% and 10.32% on two truth sets respectively (8.37% on average).

The length of the slide window is important. It decides the structure of graph G. A longer slide window leads to a more complex graph. We adjust the length from 3 to 10. In this experiment, we use our phrase generation method, and all phrase generation criteria are used, "L+P" position are used and the number of the output keyphrases is 8. Result on truth set1 is shown in Table 4. We can see that the performance is slightly influenced by different lengths of the slide window.

Table 4: Performance of the different length of the slide window on truth set 1

Length	Precision	Recall	F-measure	Length	Precision	Recall	F-measure
3	0.3478	0.4554	0.3922	7	0.3482	0.4552	0.3923
4	0.3475	0.4546	0.3916	8	0.3486	0.4562	0.3930
5	0.3470	0.4531	0.3908	9	0.3474	0.4547	0.3917
6	0.3492	0.4567	0.3936	10	0.3471	0.4546	0.3914

Figure 2 shows the performance of the phrase generation. The results are the average F-measure of two truth sets. "All", "Len>=4", "Len<=3" are the results of, respectively, all keyphrases, the keyphrases with length equal or greater than 4, the keyphrases with length less than 4. The experimental setup is, $TH_{lfre} = 3$, $TH_{be} = 0.50$, $TH_{av} = 0.69$. All experiments use "L+P" feature, and length of the slide window is 7, number of the output keyphrases is 8.

In the first group "All", "non phrase generation" performs the worst. It is not surprising because nearly half of the keyphrases need to be generated from words. "Adj-post" (the phrase generation method used in the TextRank) performs worse than all pre-generation methods ("BE","AV","Local Fre"). As we discussed before, in "Adj-post" any loss of the keywords may cause the failure of keyphrase generation. Among pre-generation methods, "Local Fre" achieves the best performance. "Boundary Entropy (BE)" and "Access variety (AV)" are evenly matched. We also combine "BE" and "AV", their performance is nearly the same. Both boundary entropy and access variety measure the boundary by uncertainty at the end of word sequence. The major difference between "BE" and "AV" is that the former focuses on the probability distribution of successors and the latter focuses on the number of distinct successors. Actually, if the successors after a word sequence have a uniform distribution, "AV" can be casted to "BE". The last method, the combination of "BE", "AV" and "Local Fre" gets the best performance (39.75%). Based on "Local Fre", the query logs help us improve the F-measure by 1.94%. For example, in an financial news, word "房/利/美(Fannie Mae)" appear only twice. Due to its low frequency, "Local Fre" fails to merge

"房(house)", "利(profit)", "美(beauty)" into the whole phrase. But in our query logs, its "AV" value is 1.69. The phrase can be generated successfully. Compared with "Adj-Post", it improves the F-measure by 10.37%.

Comparing "Len>=4" with "Len<=3", we can see that the improvement for long phrases (length$\geq$ 4) is obvious. It proves that our phrase generation strategy is efficient.

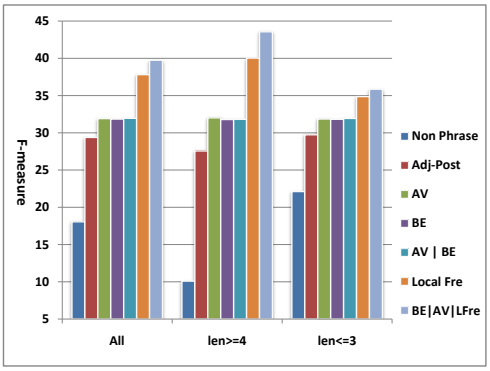

Figure 2: Performance of phrase generation

We also test the influence of output keyphrase number to the performance. The result is shown in Figure 3. There is a trade-off between precision and recall. As the keyphrases number increase, the corresponding recall increases, but precision decreases. When our system extracts more keyphrases, the probability of hitting standard keyphrases is higher, then, recall is higher. While we extract more keyphrases, the noise increases too, so, precision decreases accordingly. When the number of output keyphrases is 7, F-measure achieves the highest F-score on both truth sets. In our system, we choose 8 as the output keyphrases number because we tend to support more candidates to the readers.

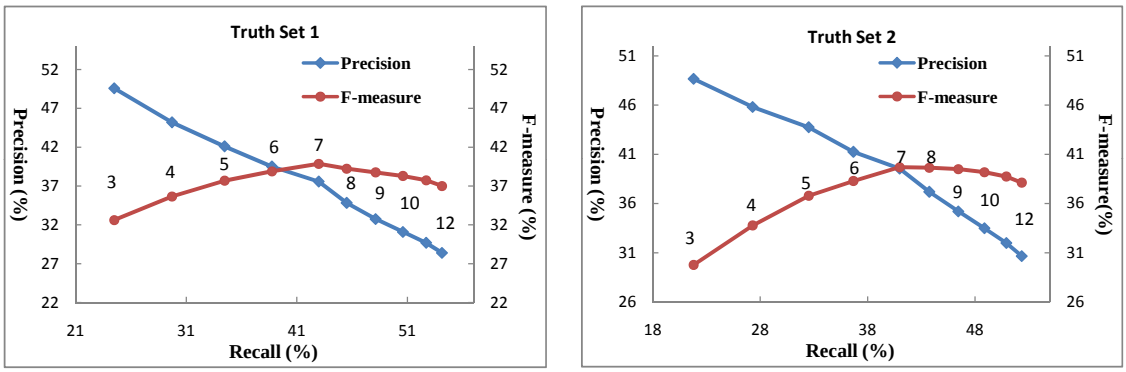

Figure 3: Performance of output keyphrase number

Sometimes, a customer may want a certain number of keyphrases for an article. For instance, if a customer needs 10 keyphrases, but our system outputs 8 keyphrases, the customer has to extract 2 extra keyphrases himself. Figure 3 shows the performance of the customize keyphrases number. The extraction system knows the exactly number of manually assigned keyphrase. "+k" means that the system outputs k extra keyphrases based on the number of the manually assigned keyphrases. "0" means that the system outputs exactly customize number keyphrases. Figure 3 shows that if we support one or two extra keyphrases to the customer, the performance is better.

6 Conclusion

In this paper, we propose a practical approach to extracting keyphrases from Chinese news articles. We use query logs as the phrase generation training corpus. And we integrate two statistical

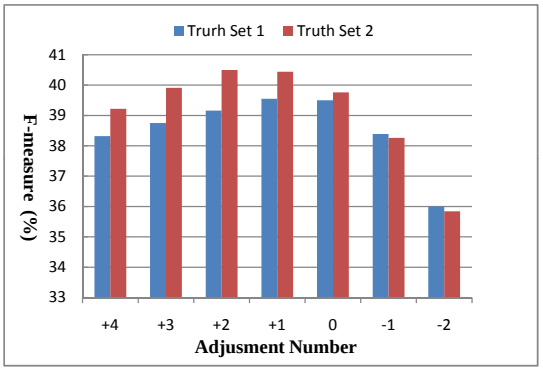

Figure 4: Performance of the output keyphrase number 2

features, "length of phrase" and "position of phrase", into the TextRank. The experimental results are encouraging. The keyphrases output by our system are more informative and readable than keywords. We are going to publish our test set as the official data set for Chinese keyphrase extraction.

References

Brin, S. and L. Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7), pp.107-117.

Feng, H., K. Chen, X. Deng and W. Zheng. 2004. Accessor variety criteria for Chinese word extraction. *Computational Linguistics*, 30(1), pp.75-93. MIT Press.

Frank, E., G.W. Paynter, I.H. Witten, C. Gutwin,and C.G. Nevill-Manning. 1999. Domain-Specific Keyphrase Extraction. *JCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 668-673. Morgan Kaufmann Publishers Inc.

Gao, J., M. Li and C.-N. Huang. 2003. Improved source-channel models for Chinese word segmentation. *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp.272-279. Association for Computational Linguistics.

Mihalcea, R., P. Tarau, D. Lin, and D. Wu. 2004. TextRank: Bringing Order into Texts. *Proceedings of EMNLP 2004*, 404-411.

Turney, P.D. 1999. Learning to Extract Keyphrases from Text *National Research Council, Institute for Information Technology, Technical Report ERB-1057*.

Yang, J., D. Ji, D. Cai, X. Lin and Y. Bai. 2008. Keyword Extraction Multi-Document Based on Joint Weight (in Chinese). *Journal of Chinese Information Processing (in Chinese)*, 22(06), pp.75-79.

Zhao, H. and C. Kit. 2008. Exploiting Unlabeled Text with Different Unsupervised Segmentation Criteria for Chinese Word Segmentation. *The 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2008)*.

Zhao, H. and C. Kit. 2008. Unsupervised Segmentation Helps Supervised Learning of Character Tagging for Word Segmentation and Named Entity Recognition. *Sixth SIGHAN Workshop on Chinese Language Processing*, pp.106-111. Association for Computational Linguistics.