Proceedings of the 2018 13th World
Congress on Intelligent Control and Automation
July 4-8, 2018, Changsha, China

# Research of Chinese Keyword Extraction Based on Weibo Information

Juntao Xue, Yunfeng Zhao*, Hao Guo, Kaiyu Li, and Xinshan Zhu

*Abstract*— **Weibo content has the characteristic of complex and changeable. And the traditional technology of keyword extraction has the limitation of ignoring the meaning of words. This paper combines different weights of speech to improve the TF-IDF formula and uses deep learning method to train text language model. Transforming the process of Weibo content into vector computation in vector space, the semantic similarity between words can be obtained by calculating the similarity between vectors. Then, the similarity and term frequency features of words in the sliding window are added to Textrank's iterative formula for edge weight optimization. By changing the number of keyword extraction and the size of sliding window, the optimal extraction effect of the algorithm is obtained. The algorithm proposed in this paper can solve the problems of the traditional Textrank algorithm, such as equal probability jumping of keyword extraction and the lack of meaning. It has the advantages of low time complexity and less corpus training.**

## I. Introduction

In recent years, with the rapid development of information society and the explosion of information in the Internet era, a large amount of data are constantly emerging in people's daily life. So the many emerging social media come in to being. In this context, it seems extremely time-consuming and laborious that users want to find some content of the specific topic from the massive Weibo. Although the commercial search engine can meet human needs, it has not yet fully reached the level of artificial intelligence. And if the excessive pursuit of the number of Weibo text, the accuracy of the search needs to be improved. While, if some keywords that represent thematic information are marked in the Weibo text, users can get what they want as much as possible by searching for them. Therefore, how to carry out Weibo text keyword extraction has become one of the hot research topics.

Since the first research by IBM's Luhn, the technology of keyword extraction has undergone more than 50 years of development [1]. Zhan Xuegang et al. calculated the weight of each word based on the weighted operators of term frequency statistics and grammatical analysis, and extracted the key words according to the weights [2]. Turney used supervised machine learning methods to extract keywords for the first time, and adjusted the parameters of the extractors according to the texts to maximize the performance of the algorithm and got the highest score as the key word [3]. Du Haizhou et al. proposed a Chinese text keyword extraction method based on contextual relations and TextRank algorithm to improve the recall and precision of the algorithm [4]. Li Yuepeng et al. applied the deep-learning keyword extraction algorithm to long texts to improve the accuracy of keyword extraction [5].

In summary, the technology of keyword extraction has made great progress. However, due to Weibo is complex and various, and also has a strong immediacy, interactivity, and dynamic, traditional algorithms of keyword extraction can not solve the problems well what is encountered in the research [6]. In addition, the Weibo posted by users on the Sina platform contains not only the text content but also a lot of accompanying information. Generally speaking, in the process of preprocessing the Weibo text, it is easy to lead to problems such as high text dimension and large text sparsity. However, the traditional algorithm of keyword extraction can not solve the above problem. For example, TF-IDF (Term Frequency-Inverse Document Frequency) simply calculates the term frequency but ignores the meaning of the word, and the vector space model has the disadvantages of high dimension and high sparseness [7].

In view of the research of extracting keywords from Weibo, this paper combines word vectors with traditional TF-IDF. Based on the classical Textrank algorithm, considering the weight of word vector and improving the weight of edge, and the keyword extraction based on the edge weight optimization Textrank algorithm is proposed. It can effectively solve the problem of the jumping with equal probability and the lack of meaning in the classic algorithm. By adjusting the number of keywords extracted and sliding the window size of parameters, the optimal performance of Textrank algorithm based on edge weight optimization is realized.

## II. Data Collection and Preprocessing

### A. Data Collection

Weibo is a social software, embedding the search engine, therefore, it is possible to crawl the content of Weibo through web crawler. Python is a widely used scripting language what has many powerful crawler libraries such as urllib and urllib2. In addition, the Scrapy web crawler is an open source crawler based on the Python language that allows users to crawl data within this framework [8].

The basic process of crawling content from Sina Weibo consists of several steps, as follows:

1. Set the initial URL (Uniform Resource Locator). (Http://login.sina.com.cn/)

2. Simulate the login process of Sina Weibo, and set the Weibo account and password in the program.

3. Wait for the program to automatically jump, and simulation the process of user inputs keyword.

This work was supported in the Opening Project of State Key Laboratory of Information Security under Grant 2017-MS-11 and the Seed Foundation of Tianjin University under Grant 1705.

Juntao Xue is with the School of Electrical and information Engineering, Tianjin University, Tianjin 300072, China (e-mail: xuejt@tju.edu.cn).

Yunfeng Zhao*(corresponding author) is with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (phone 15222580162, e-mail: 2016203147@tju.edu.cn).

Hao Guo is with the School of Electrical and information Engineering, Tianjin University, Tianjin 300072, China (e-mail: 476820191@qq.com.)

978-1-5386-7346-1/18/$31.00 © 

4. Automatically crawl the contents of this page, including Weibo users, Weibo content, and the number of following and follower.

5. Automatically jump to the next page to continue crawling data after getting this page content.

6. Save the obtained content to the local text file.

### B. Data Preprocessing

In general, the data is divided into valid data and invalid data. Invalid data is not the subject of our concern and may affect the outcome of the experiment. The format of the Weibo is complicated and contains many invalid data. Therefore, it is necessary to preprocess the crawled data. Weibo data preprocessing involves the process of Chinese word segmentation, part of speech tagging, stop word rejection and so on.

English text can be distinguished by the space between words. While there is no space between Chinese words. In order to ensure that the algorithm used in this paper can effectively extract keywords and emotion word information, it is necessary to cut the Weibo sentence into word fragments, which is the process of Chinese word segmentation in Weibo sentences.

In this paper, the Chinese word segmentation is based on the precise pattern of the jieba based on Python, which can cut the sentence most accurately and is suitable for text analysis [9].

According to experience, the keywords are mainly concentrated in the real words such as verbs, nouns, and adjectives, a total of about 92.1%. Non-related information and more concentrated in prepositions, conjunctions and other parts of the word. For Weibo text, the part of speech of keywords mainly are verbs, nouns, adjectives [10]. Therefore, it is necessary to tag the part of speech of Weibo words.

In addition, stop words such as "of" and "the" account for a large part of text words, which may affect the outcome of the experiment. Therefore, it is necessary to filter punctuation marks and stop words, which can help to improve the performance of the algorithm [11].

However, unlike the traditional text structure, Weibo text contains a lot of noises and a variety of online vocabulary. Therefore, only the Chinese word segmentation, part of speech tagging and stop words rejection are not enough, and it is necessary to make special preprocessing for Weibo.

Sina Weibo data is mainly composed of the user ID, nickname, release URL and time, Weibo content, likes number, following number and repost number and other components. This paper only takes the content of the Weibo as the research object. The content of the Weibo includes the content of the text, the subject (marked by #), the object of interest (marked by @), the punctuation mark, the expression, the useless symbol and the picture. Among them, the text content is the main research object, other elements can be ignored. Therefore, non-Weibo text elements can be filtered as noises.

The process of Weibo data preprocessing is as follows:

1. Remove the number of reposting, publishing time and other non-text elements of Weibo to get the Weibo text content.

2. Use regular expressions to match the # symbols in the Weibo text to get the filtering content of the Weibo.

3. Use the jieba based on Python to make Chinese word segmentation on Weibo content.

4. Remove the punctuation, the useless symbols and other stop words on the text after Chinese word segmentation.

5. Tag the part of speech, filtering out the auxiliary words, prepositions and other semantic meaningless words, focusing on the choice of part of the word as a noun, adjective and verb words.

## III. SYSTEM MODELING

### A. System Block Diagram

This paper combines word vectors with traditional TF-IDF. Based on the classic Textrank algorithm, considering the weight of word vector and improving the weight of edge, and the algorithm of keyword extraction based on edge weight optimization Textrank is proposed. System block diagram shown in Fig.1. Finally, the optimal performance of Textrank algorithm based on edge weight optimization is achieved by adjusting the number of keywords extracted and the size of the sliding window.

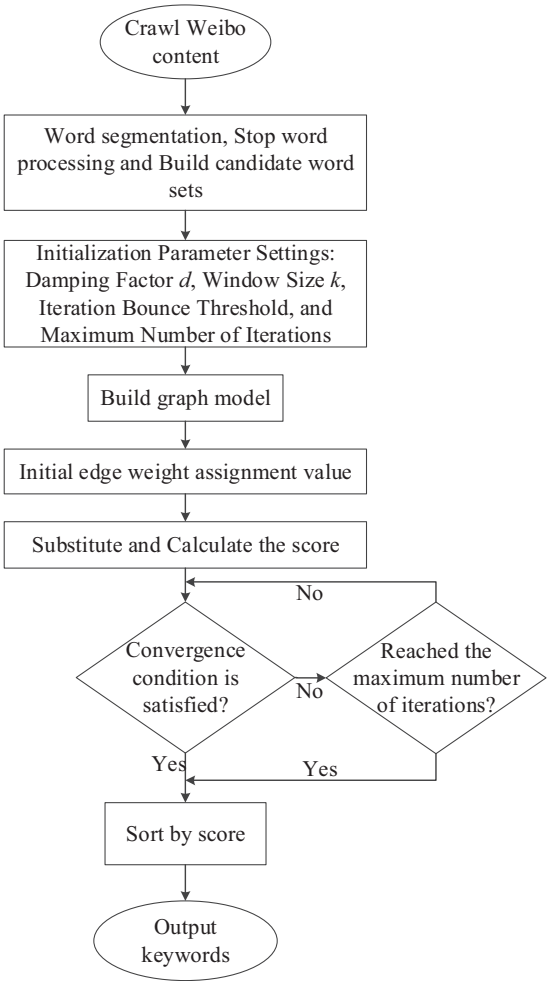


Fig.1. System Block Diagram.

### B. Textrank Algorithm

The Textrank algorithm forms the graph model according to the content of the Weibo. The graph includes the points and the edges of the connection points. The terms form the points, and the terms similarity form the edge in the graph model.

Afterwards, using the Textrank algorithm to iterate until it reach convergence, the first n keywords can be got [12]. The general process is as follows:

1. Identify the terms of the Weibo content and add them to the points in the graph model.

2. Identify the terms similarities in Weibo content and add them to the edges in the graph model.

3. Iterate until it reach convergence according to Textrank formula.

4. Sort the word scores obtained by convergence to get the final keywords.

The graph model for the Textrank algorithm is represented as $G = (V, E)$, where $V$ represents the set of all points and $E$ represents the set of all edges. The score of the node $V_i$ is given by (1).

$$K(V_i) = (1-d) * \sum_{j \in in(V_i)} \frac{1}{out(V_j)} K(V_j) \tag{1}$$

Where: $in(V_i)$ represents the set that all nodes points to $V_i$, $out(V_j)$ represents the set that all nodes directed by $V_j$, $d$ represents the damping coefficient, generally taken as 0.85, which represents the probability of a jump from a point to another point in the graph model.

When using the Textrank algorithm to build the graph model, it needs to assume the initial weight of each point, which is assumed to be 1 here. After that, any points in the graph are iterated until them reach convergence, and the change rate of the vertices generally is less than a certain threshold. Finally, get the score of each vertex.

### C. Edge Weight Optimization

The weight of edge based on the Textrank graph model refers to the weight between two words. The problem of edge weight optimization is how to assign the scores of candidate words during the iterative process. In the paper, the edge weight is calculated by the word vector and the text features of Weibo.

This paper argues that only the similarity of words does not adequately express the importance of words. Generally, it is considered that if two words appear less frequently in the text, even if the two words are highly similar, it can not be said that the two words are important. Conversely, if two words are of high importance and at least one of them appears frequently, the probability that they are candidates is high. For example, in a Weibo text, "cate" and "foodie" both appear only once, and although they are highly similar, they can not be regarded as candidate keywords. However, if "cate" appears many times and "foodie" appears only once, it is still considered "cate" and "foodie" as candidate keywords because of their high similarity. Therefore, if two words are more likely to be candidate keywords, they will be given a higher edge weight. The edge weight is expressed as (2):

$$w_{ij} = \alpha * freq(w_i, w_j) + (1-\alpha) * vec(w_i, w_j) \tag{2}$$

$$freq(w_i, w_j) = TF\text{-}IDF(w_i) * TF\text{-}IDF(w_j). \tag{3}$$

Where: $\alpha$ represents the trade-off coefficient between the two features, $w_i$ represents $i$ words, $freq(w_i, w_j)$ represents the common frequency of the two words, and $vec(w_i, w_j)$ represents the word similarity between the two word vectors.

*1) TF-IDF:*

Term Frequency (TF) is the frequency at which words appear in the text, reflecting the importance of words on the text. Inverse Document Frequency (IDF) reflects the measure of the importance of words in sentences. The TF of a word in a Weibo can be calculated by the formula (4). The IDF of a word can be calculated by the formula (5). In order to avoid the case that the denominator is 0, it plus 1. Combining the TF and IDF to form TF-IDF, as shown in formula (6). It reflects the importance of some words in the Weibo sentence collection to the text and the weight factor often extracted as keywords [13]. In this paper, it is assumed that the frequently occurring words are more likely to be keywords and should be given more weight.

$$TF(w, d_i) = \frac{count(w, d_i)}{count(w, d)} = \frac{count(w, d_i)}{\sum_{k=1}^{d} count(w, d_k)} \tag{4}$$

$$IDF(w) = \log\left(\frac{d}{docs(d)+1}\right) \tag{5}$$

$$TF\text{-}IDF(w, d_i) = TF(w, d_i) * IDF(w) \tag{6}$$

Where: $count(w, d_i)$ indicates the number of occurrences of word $w$ in Weibo sentence $d_i$, $count(w, d)$ indicates the number of occurrences of word in Weibo subset $d$, and $docs(w, d)$ indicates the number of sentences in which word $w$ appears in $d$.

The part-of-speech characteristic indicates the part of speech tagging information of a word, that is, one word has part of speech such as noun, verb or adjective. The keywords part of speech are mostly adjectives, nouns, and verbs[14]. Studies have shown that nouns are significantly more important than other parts of speech, thus it should be given higher scores. The part-of-speech score $s(w)$ of a word is given by (7).

$$s(w) = \begin{cases} 1.5 & p(w) = n \\ 1 & p(w) = v \\ 0.8 & p(w) = a \\ 0.5 & \text{other} \end{cases} \tag{7}$$

Where: $p(w)$ means part of speech, $n$ means noun, $v$ means verb, and $a$ means adjective.

Some words can be used as verbs, but also as nouns. For this kind of conversion word, the basic TF-IDF formula is improved in this paper. By calculating the term frequency separately for the same words of different parts of speech, the improved TF-IDF formula is shown in (8):

$$TF\text{-}IDF(w, d_i) = \frac{\sum_{m=1}^{mc} freq(w_m) * s(w_m)}{count(w, d)} * IDF(w). \tag{8}$$

Where, $freq(w_m)$ represents the number of times that word $w$ appears in sentence $d$, and $mc$ represents the total number of conversion words of $w$.

Semantic features is the meaning of the text. Text can be meaningful only if it is given the meaning of a word. The semantics are great helpful for understanding the purpose of Weibo text, therefore, it is essential to consider the semantic features of Weibo text.

*2) Word similarity:*

The problem of natural language processing is how to translate it into the computer-recognizable problem, which requires the digitization of these abstract literal symbols. Word vector is a way of digitizing words. The word vector based on depth learning can solve the problem of expression method based on the bag-of-words model to a certain extent and provide the basis for calculating the similarity between words. The idea of word vector is: through the existing corpus to train the language model, and to map each word into a fixed-length vector. In this way, the formed vectors form a vector space, and each vector can be regarded as a point in the vector space. The measure of "distance" is introduced which can determine the semantic similarity of two terms according to the distance between vectors [15].

In this paper, $vec(w_i,w_j)$ is calculate by the Word2vec, which can form the similarity matrix between words and words. It can effectively solve the problems of the initial edge weight of classic Textrank and the probability transfer matrix of words. Using Word2vec train Weibo text, the word similarity matrix also can be get. Specifically, the Weibo texts are trained by the Skip-gram model, and the K-dimension is extended in each word between word vectors, then the relationship between each word and other words is got by calculating the similarity between vectors, As shown in (9).

$$vec(w_i, w_j) = \frac{w_i * w_j}{\|w_i\| * \|w_j\|} \tag{9}$$

In summary, the simultaneous (8) (9) into the edge weight (2), get(10):

$$w_{ij} = \alpha * TF\text{-}IDF(w_i) * TF\text{-}IDF(w_j) + (1-\alpha) * \frac{w_i * w_j}{\|w_i\| * \|w_j\|} \tag{10}$$

### *D. EW-Textrank Algorithm (Textrank Algorithm Based on Edge Weight Optimization)*

When the classic Textrank algorithm is used for keyword extraction, the side weight of the graph model can not be fully considered. And there is the problem of jumping with equal probability and the word meaning is ignored [16]. Specifically, the classic Textrank algorithm calculates the number of co-occurrences of words and words according to the window size of the graph model, and then calculates the edge weights. And the classic algorithm pays more attention to the words within the sliding window size. For the two keywords with fewer co-occurrences, the classic algorithm can not extract two keywords outside the window size well. In addition, the classical algorithm also only considers that one word can jump equally probabilistically to the next and neglects the probability of appearance of the word itself. However, in fact, the probability of occurrence of each word is different, and the deviation from the idea of such equality jump may appear. Therefore, in this paper the edge weight of the classic Textrank algorithm is improved to solve the problems encountered in keyword extraction.

In this paper, the semantics of the word itself is combined with the text characteristics, and taking into account the problem of existing lexical, edge weight and jumping with equal probability. Using (10) to improve (1), the (11) is obtained.

$$K(V_i) = (1-d) + d * \sum_{V_j \in in(V_i)} \frac{w_{ji}}{\sum_{V_k \in out(V_i)} w_{jk}} K(V_j) \tag{11}$$

Where: $in(V_i)$ represents the set that all nodes points to $V_i$, $out(V_j)$ represents the set that all nodes directed by $V_j$, $d$ represents the damping coefficient, generally taken as 0.85. $w_{ij}$ represents the edge weight between node $V_i$ and node $V_j$.

In simple terms, the EW-Textrank algorithm improves word score distribution. The word vectors and text features are introduced to take full account of the word meanings. The algorithm considers word itself to have probability of occurrence, makes word jump more reasonable and more in line with Weibo, and improves the efficiency of keyword extraction. Among them, the word vector is mainly calculated using Google's Word2vec tool, and the text features are mainly TF-IDF and part of speech.

The specific steps for keyword extraction using edge-weighted Textrank algorithm are as follows:

1. Split the collected Weibo text according to the sentence to get $T = [S_1, S_2, S_3 \ldots S_m]$.

2. For each sentence $S_i \in T$, perform Chinese preprocessing such as Chinese word segmentation, part-of-speech tagging, and stop-word filtering to obtain the term to be input, that is $S_i = [t_{i,1}, t_{i,2}, \ldots, t_{i,n}]$, where $t_{i,j}, j \in [1,n]$ is a candidate keyword.

3. Use Word2vec training text to get $vec(w_i,w_j)$, at the same time, statistics $TF\text{-}IDF(w_i)$, $TF\text{-}IDF(w_j)$.

4. Build Textrank diagram model.

5. Calculate the score of the node set and iterate according to (10) and (11) until them reach the convergence.

6. Sort each node of the node set, the top score of the first n words as the final keywords.

Fig.2 shows a part of the Textrank graph model with edge weights. There are three vertices in the figure, namely $V_i$, $V_j$ and $V_k$, the edge weight between any two points can be obtained from (10). Then, the Textrank score of node $V_j$ can be calculated from (12).

$$K(V_j) = (1-d) + d * \frac{w_{kj}}{w_{kj} + w_{ki}} K(V_k) \tag{12}$$

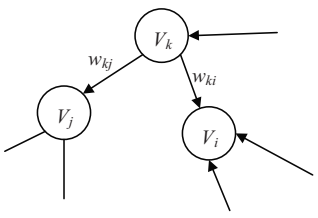


Fig.2. Textrank Diagram Model with Edge Weight.

## IV. EXPERIMENTAL RESULTS

### A. Evaluation Index

At present, there is no unified evaluation index for Weibo keyword extraction, which can only be verified by hand and has some subjectivity. Because the choice of experimental evaluation index often plays an important role in the evaluation of experimental results, it is necessary to fully consider the result of keyword extraction. Combining with the evaluation index of information retrieval system, the accuracy $p$, the recall rate $r$, $f$ value and the time complexity of building the model are selected as indicator of the evaluation algorithm.

Similar to the information retrieval system, the keyword extraction result can be divided into four parts:

1. The number of keywords which is extracted by the keyword extraction algorithm ($a$)

2 The number of non-keywords which is extracted by the keyword extraction algorithm ($b$)

3. The number of keywords which is not extracted by the keyword extraction algorithm ($c$)

4. The number of non-keywords which is not extracted by the keyword extraction algorithm ($d$)

The accuracy $p$ and the recall $r$ are both measures of the ability of the algorithm to extract the keywords. And, $f$ value is to evaluate the overall performance of an indicator. The specific definitions are shown in (13), (14) and (15).

$$p = \frac{a}{a+b} \tag{13}$$

$$r = \frac{a}{a+c} \tag{14}$$

$$f = \frac{2*p*r}{p+r} \tag{15}$$

### B. Results Analysis

Experiments based on Weibo keyword extraction mainly focus on comparing and analyzing the keywords extracted manually with the keywords extracted from the experimental results. Some experiments are carried out, including the number of keywords on the performance of the three algorithms experiment, and the algorithm proposed the performance experiments involved in optimization in this paper.

Experiment one is the number of keywords on the performance of the three algorithms experiment. In this paper, the keyword extraction experiments of three algorithms are compared and analyzed by taking the number of extracted keywords as variables. When the number of extracted keywords were selected 3, 5, 7 or 9, the corresponding performance of the three algorithms are shown in Table I.

From the Table I, it can get the keyword extraction algorithm proposed in this paper is better than the first two algorithms in accuracy $p$, recall $r$ and $f$ value. When the number of keywords is small, the performance of TF-IDF keyword extraction algorithm is also well. But with the increase of the number of keywords, the performance of TF-IDF keyword extraction algorithm is obviously not as well as the latter two algorithms. The TF-IDF keyword extraction algorithm can take full account of the text features such as term frequency, part of speech, which can extract the surface information of the Weibo text, and mostly high-frequency vocabulary. However, as the textual information increases, this method is inadequate in terms of meaning and can not extract low-frequency words that represent thematic information. The EW-Textrank proposed in this paper can not only take into account the text features, such as part of speech, term frequency, but also combine the meaning of the word. The proposed algorithm combines the advantages of the first two algorithms, therefore, it performs better in terms of keyword extraction. For example, when extracting keywords of Weibo, the algorithm in this paper can extract keywords that are not extracted by the other two algorithms: "Great Wall", "Disneyland" and "Guangxi". For such kind of keywords that appear only a few times but are closely related to the topic of Weibo, the algorithm proposed in this paper can include these keywords as much as possible and break through the limitations of the traditional algorithms in keyword extraction.

TABLE I. THE INFLUENCE OF THE NUMBER OF KEYWORDS ON THE PERFORMANCE OF THREE ALGORITHMS

| Algorithm | | TF-IDF | Textrank | EW-Textrank |
|---|---|---|---|---|
| key=3 | *p* | 0.53 | 0.6 | 0.66 |
| | *r* | 0.16 | 0.18 | 0.25 |
| | *f* | 0.35 | 0.27 | 0.36 |
| key=5 | *p* | 0.5 | 0.62 | 0.68 |
| | *r* | 0.25 | 0.31 | 0.34 |
| | *f* | 0.33 | 0.41 | 0.45 |
| key=7 | *p* | 0.52 | 0.6 | 0.66 |
| | *r* | 0.37 | 0.18 | 0.25 |
| | *f* | 0.43 | 0.27 | 0.36 |
| key=9 | *p* | 0.57 | 0.62 | 0.68 |
| | *r* | 0.51 | 0.31 | 0.34 |
| | *f* | 0.54 | 0.41 | 0.45 |

The second experiment is to compare the time complexity of algorithm two and algorithm three. Comparing with the traditional Textrank algorithm, the proposed algorithm can effectively reduce the time complexity of the algorithm. Here simply uses run time to represent the time complexity and regards sliding window size as a variable, when the sliding window size to take 2, 3, 5, 7 and 9, the specific experimental results shown in Fig.3.

The Fig.3 shows that the algorithm proposed in this paper can effectively reduce the time complexity of Textrank algorithm, and thus improve the efficiency of keyword extraction. The time complexity of Textrank algorithm is mainly reflected in the process of computing iteration. While the algorithm proposed in this paper improves the traditional edge weight and solves the problem of jumping with equal

probability of words. So it can effectively reduce the computational iteration process and accelerate the convergence of the model. Therefore, the proposed algorithm can effectively reduce the time complexity of keyword extraction.

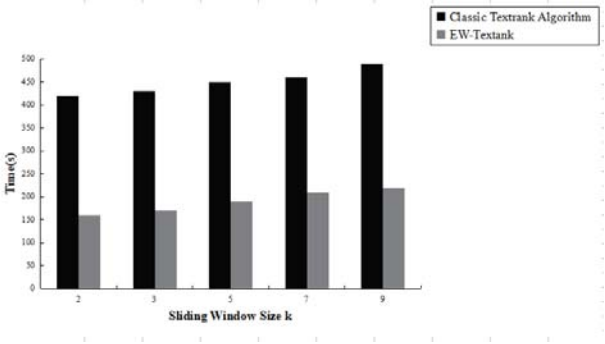


Fig.3. Algorithm two and Algorithm three time complexity comparison.

The third experiment is about the performance improvement of the proposed algorithm, mainly the adjustment of the sliding window size in the parameters. The other parameters in the algorithm are fixed. These parameters include the maximum number of iterations which is set to 200, the iterative threshold which is set to 0.001, the number of keywords extracted by the algorithm which is set to 7, and the damping factor *d* which is set to 0.85. When the sliding window size were selected 3, 5, 7 or 9 respectively, the performance of the keyword extraction algorithm based on edge weight optimization is shown in Table II.

TABLE II. THE INFLUENCE OF EXTRACT THE WINDOW SIZE ON ALGORITHM THREE

| **Sliding Window Size *k*** | ***p*** | ***r*** | ***f*** |
|---|---|---|---|
| 3 | 0.80 | 0.56 | 0.66 |
| 5 | 0.81 | 0.57 | 0.67 |
| 7 | 0.78 | 0.55 | 0.65 |
| 9 | 0.78 | 0.55 | 0.65 |

For the proposed algorithm, the change of sliding window size will have some impact on the experimental results. The sliding window is used to obtain the relationship between words and words. The larger the sliding window, the stronger the relationship between the representative words and the words. With the enhancement of the relationship between words, the optimized performance of edge weight is more and more obvious. However, if the relationship between words is increased excessively, the accuracy of keyword extraction will decrease. It can be seen from the experimental results that as the size of the sliding window increases, the performance of the algorithm initially increases and then decreases. When the window size is equal to 5, the performance of the proposed algorithm reaches its maximum, that is, the algorithm achieves the best performance. At this point, the best accuracy, recall and value were 0.81, 0.57, 0.67.

## V. CONCLUSION AND OUTLOOK

For the classical textrank algorithm which can not solve the problems of jumping with equal probability and lacking of semantic meaning, this paper introduces the word vector to train language model in this paper. In feature selection, the word vector and term frequency are merged, and the edge weight of the classical Textrank algorithm is improved, then the Textrank algorithm based on edge weight optimization is proposed. In the experimental results of the keyword extraction of this paper, the efficiency of different keyword extraction algorithms is compared by the evaluation index of the accuracy *p*, the recall *r* and the *f* value. Finally, comparing and analyzing the performance of the three algorithms in keyword extraction, it can get that the performance of the proposed algorithm is superior to the first two algorithms, which can improve the accuracy *p*, recall *r* and *f* value and effectively reduce time complexity.

## REFERENCES


[1] Zuo Xiaofei, "Keyword Extraction Based on Complex Networks" Master. dissertation, Xidian University, Xi'an, 2013.

[2] Zhan Xuegang, Wu Qiang, "Key words extraction algorithm based on TF statistics and syntax analysis," *Computer Applications and Software*, vol. 31, Jan. 2014, pp. 47-49.

[3] Turney P D, "Learning to Extract Keyphrases from Text" presented at the National Research Council of Canada, Dec. 8 2002, DBLP, 2002

[4] DU Haizhou, CHENG Zhengbo, ZHONG Konglu, "Keyword Extraction Method Based on Context and TextRank Algorithm," *Journal of Shanghai University of Electric Power*, vol. 33, Dec. 2017, pp. 607-612.

[5] Li Yuepeng, Jin Cui, Ji Junchuan. "A Keyword Extraction Algorithm Based on Word2vec," *e-Science Technology & Application,* vol. 4, Jun. 2015, pp. 54-59.

[6] Fan Dongping, "An Empirical Research on the Influence of Micro-blog on Tourist Decision-making," Master. dissertation, Hubei University, Hubei, 2013.

[7] WEIMING LIANG, "Chinese Keyphrases Extraction Technique" Master. dissertation, Shanghai Jiao Tong University, Shanghai, 2010.

[8] Qian Cheng, Yang Xiaolan, Zhu Fuxi, "Network crawler technology based on Python," *Heilongjiang Science and Technology Information*, vol. 36, Dec. 2016, pp. 273-273.

[9] Liu Xinliang, Yan Shanshan, "Based on python Chinese word segmentation and its application," *Computer and Information Technology*, vol. 11, Nov. 2008, pp. 89-92.

[10] Zhang Jin. "A Method of Intelligence Key Words Extraction Based on Improved TF-IDF," *Journal of Intelligence*, vol. 33, Apr. 2014 pp. 153-155.

[11] LIAN Jie, ZHOU Xin, CAO Wei, "Study on data mining of Sina Weibo," *Journal of Tsinghua University (Science and Technology)*, vol. 51, Oct. 2011, pp. 1300-1305.

[12] Niu Ping, "TF-IDF and Rules Based Automatic Extraction of · Chinese Keywords," Master. dissertation, Dalian University of Technology, Dalian, 2015.

[13] SHI Yongbin, YU Qingsong, "Key Words Extraction Algorithm Based on Chi-square value of Co-occurrence word," *Computer Engineering*, vol. 42, Jun. 2016, pp. 191-195.

[14] LIU Jiabin, CHEN Chao, SHAO Zhengrong, "Automatic extraction of key phrases from scientific articles based on machine learning method," *Computer Engineering and Application*, vol. 14, May. 2007, pp. 170-172.

[15] Mikolov T, Chen K, Corrado G, et al. "Efficient Estimation of Word Representations in Vector Space," *Computer Science*, 2013.

[16] Chen Wanzheng, "Research on the Optimization of Textrank Keyword Extraction Algorithm and SOM Text CLUSTERING Model," Master. dissertation, Guangxi University, Guangxi, 2016.