IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 May 2022, accepted 19 June 2022, date of publication 6 July 2022, date of current version 12 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3188861

Multifeature Fusion Keyword Extraction Algorithm Based on TextRank

WENMING GUO[1,2,3], ZIHAO WANG[1,2], AND FANG HAN[3], (Member, IEEE)
[1]School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education, Beijing 100876, China
[3]Xinjiang Institute of Engineering, Ürümqi 830023, China

Corresponding author: Zihao Wang (wangzihao2020@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62162060.

ABSTRACT Keyword extraction is the predecessor of many tasks, and its results directly affect search, recommendation, classification, and other tasks. In this study, we take Chinese text as the research object and propose a multi-feature fusion keyword extraction algorithm combined with BERT semantics and K-Truss graph(BSKT). The BSKT algorithm is based on the TextRank algorithm, which combines BERT semantic features, K-Truss features, and other features. First, the BSKT algorithm obtains the word vectors from the BERT pretraining model to calculate the semantic difference, which is used to optimize the iterative process of the TextRank word graph. Then, the BSKT algorithm obtains its K-Truss graph by decomposing the TextRank word graph and obtains the truss level feature of the word. Finally, by combining the word IDF and truss level features, the BSKT algorithm scores the words to extract keywords. Experimental results show that the BSKT algorithm achieves better performance than the latest keyword extraction algorithm SCTR in the task of extracting 1-10 keywords. Furthermore, the increment in F1 increased by 11.2% when the BSKT algorithm was used to extract three keywords from the Sensor dataset.

INDEX TERMS Keyword extraction, BERT word vector, K-Truss graph, TextRank, semantics.

I. INTRODUCTION

Keyword is a high-level summary of a text or an article, that can reflect the core content and highlight the theme of the text or article. Keyword extraction technology currently has important applications in information retrieval and natural language processing, such as automatic summarization, text classification, and sentiment analysis. With the continuous development of society and the internet, an increasing number of texts are being generated. Manually extracting keywords is time-consuming, laborious and largely affected by subjective factors. Therefore, it is necessary to automatically extract keywords from texts.

Keyword extraction technology was first proposed by Luhn [1] in 1958. Keyword extraction technology can be divided into supervised extraction and unsupervised extraction according to whether there is supervision in the keyword extraction process. The supervised keyword extraction algorithm requires a large number of labeled corpora to train the model, such as the BERT-based multitask keyword extraction method proposed by Sun S [2]. Ding T [3] proposed a keyword extraction approach that combines BERT and CRF. The method creatively uses distributed computing to save the overall computing time. Kong S [4] proposed a hybrid supervised method for Chinese keyword extraction. The method uses the BERT_EXT model to extract key sentences and then assigns different weights to different sentences. Finally, the keywords can be extracted. The unsupervised keyword extraction method uses statistical knowledge and graph theory knowledge to discover important words in the text as keywords. These methods only need ~~a~~ part of the test dataset to verify the results. Unsupervised keyword extraction methods are easy to transfer and apply to new domains or languages. Therefore, unsupervised keyword extraction methods that do not require a training set are currently the mainstream in the field of keyword extraction. These methods include the TF-IDF method [5], keyword extraction based on relative word frequency [6], and the TextRank method [7]. At the

The associate editor coordinating the review of this manuscript and approving it for publication was Ali Kashif Bashir.

same time, Chinese keyword extraction and English keyword extraction are different. It is unnecessary to separate words because the words in English text have been distinguished by separators. Thus, only the phrase problem needs to be considered in English. However, the Chinese keyword extraction task needs to segment the text first because that the result of keyword extraction can be directly affected by the word segmentation.

In this paper, we propose a keyword extraction algorithm combined with BERT semantics and K-Truss graph (BSKT). The BSKT algorithm can be applied to Chinese text and transferred to different fields easily, and does not require a large corpus. Based on TextRank, the BSKT algorithm optimizes the state transfer matrix by introducing BERT word vectors in the iterative process to calculate the semantic variability of words. The BSKT algorithm finds the K-Truss graph of the TextRank word graph to obtain the truss level feature of words. At the same time, the BSKT algorithm introduces other features to jointly construct the keyword scoring function to improve the original TextRank method and enhance the keyword extraction effect.

II. RELATED WORKS

The TextRank algorithm [7] has been a popular approach in the field of keyword extraction since it was proposed in 2004. In the keyword extraction task, the TextRank algorithm uses a sliding window to obtain the words' cooccurrence relationship in the text. Then, the relationship can be used to construct a word graph. In the graph, the nodes indicate words. The edges of the graph indicate that there is a cooccurrences relationship between two words. Additionally, the weights of the edges indicate the number of cooccurrence within a specified size window. Using the chain idea in the PageRank algorithm [8], the word graph iterates multiple times to finally obtain the scores of each node in the graph. The words with the top node scores are the keywords in the text.

To improve the TextRank algorithm, Li W [9] proposed an extraction method that cites an external knowledge base for the lack of information embedded in short texts. The method represents the semantic information of a word by utilizing the concept distribution of Wikipedia entries. Then the similarity between words can be measured. This method overcomes the shortcomings of the lack of information in short texts. However, it is poorly effective for new words and words that are not included on Wikipedia. Huang Z [10] constructed a prior knowledge network by using public dictionary data and improved the scoring function in the TextRank network by using some prior relations in this network. Kazemi A [11] optimized the state transfer matrix by randomly resetting the weights during iterations of the TextRank graph, which improved the effect of the keyword extraction. Li Y [12] proposed a keyword extraction algorithm based on Word2vec. The algorithm uses the Word2vec word vector model to learn the semantic relationships of words from Chinese Wikipedia and obtain word vectors. Then the nearest word to the cluster center can be selected as keywords in the algorithm by performing K-means clustering on the text to be extracted. Xia T [13] proposed a Word2vec word vector weighted TextRank keyword extraction algorithm. The algorithm uses Word2vec word vectors to calculate the semantic discrepancy. Then, the semantic discrepancy optimizes the state transition matrix. Tixier [14] proposed an unsupervised keyword extraction method based on graph simplicity that performed a K-Truss decomposition of the graph to find its dense subgraphs and used the nodes in the K-Truss graph as keywords. However, because the K-Truss graph consists of at least 3 nodes, the method may lead to extracting too many keywords. Yao [15] combined the TF-IDF algorithm with the TextRank algorithm and filtered twice the extraction results. The results obtained were improved compared to both prior methods. Xu Z [16] proposed a novel approach to extract keywords from text that combines the features of word frequency and association. The experimental results show that the precision rate, recall rate and F1 of the approach are all better than those of TextRank and TF-IDF. Campos R [17] proposed a lightweight unsupervised keyword extraction algorithm YAKE, which does not rely on any external features and uses only internal features of the text, such as position and semantic relevance to extract keywords.

In summary, the TextRank method can be improved in two ways. The first aspect starts from the text itself. Using graph theory knowledge and mining the close relationships in the word graph can discover the important words. The second aspect is the introduction of external knowledge. Introducing external knowledge to optimize its algorithm iteration process or filtering twice the results can improve the effectiveness of the algorithm. For the first aspect, the K-Truss graph can adequately express the relationship of words to each other as an extremely dense subgraph of an undirected graph. For the second aspect, the BERT [18] model uses a bidirectional transformer encoder to learn the semantic relationship between any two positions in the text. Meanwhile, the pretraining model can obtain the word vector of any word and does not depend on the dataset. Therefore, we propose a keyword extraction algorithm combined with BERT semantics and K-Truss graph (BSKT). The BSKT algorithm is based on the TextRank algorithm, which incorporates BERT semantic features, K-Truss features and other features. The effectiveness of the BSKT algorithm in the keyword extraction task can be verified by experiments on the Sensor dataset, Sohu news dataset and Southern Weekend news dataset.

III. APPROACH

A. GENERAL FRAMEWORK OF ALGORITHM

The BSKT algorithm shown in Figure 1, can be organized into five parts. We need to preprocess the text data in the first part. In the second part, we construct a word graph model based on the candidate words obtained by preprocessing. In the third part, we obtain the representation of the BERT word vector according to the candidate word in the word graph model and calculate its semantic similarity. In the fourth part, we process the word graph constructed to obtain its K-Truss subgraph

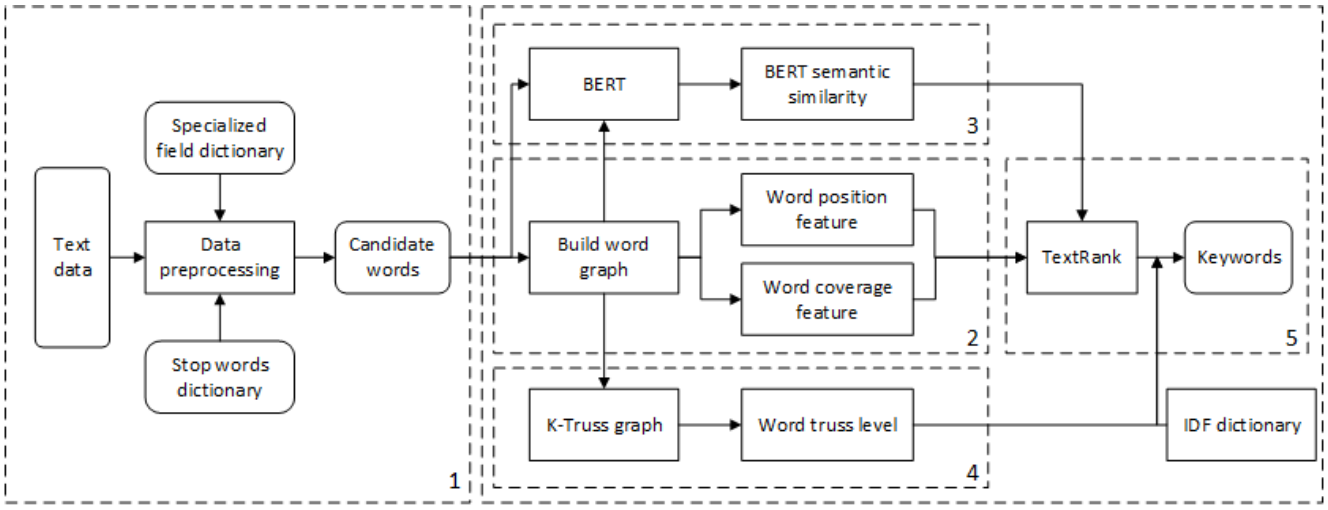

FIGURE 1. Framework of the BSKT algorithm.

and the words' truss level features. Finally, we sort the words according to the existing features to obtain the keywords of the text.

B. TEXT PREPROCESSING

The text preprocessing stage is indispensable for most NLP tasks. The results of data preprocessing directly affect the subsequent keyword extraction tasks. The task of the Chinese text preprocessing stage of the BSKT algorithm is to clean the meaningless symbols and words in the text. Then, the text can be separated into words. Text preprocessing can obtain the keywords' candidate words, which are directly used to construct the word graph. The steps of Chinese text preprocessing are as follows.

a) Building domain dictionaries: Compared with English text, the complexity of Chinese text is that there exist many Chinese phrases and multiple polysemy words. Additionally, there are no natural separators in Chinese text. Therefore, we need a tool or method to segment the text.

For a specific field, it is necessary to build a domain dictionary to improve the accuracy of the word segmentation algorithm, which is verified in follow-up experiments. The domain dictionary used in these experiments is the keywords of the articles marked in the dataset. The current common dictionary format is [phrase, frequency, POS].

b) Deactivating stop words: The common stop words include intonation words, special symbols, and auxiliary words. These stop words need to be removed during keyword extraction. A Stop word dictionary can be built according to specific scenarios. We can also use publicly available stop word dictionaries such as Baidu and HIT. The BSKT algorithm uses the Baidu stop word dictionary for text preprocessing.

c) Chinese word splitting: Hanlp, LTP, THULAC, PKUSEG, NLTK, and Jieba are common open source word segmentation tools. All these word splitting tools can split Chinese words, POS annotation and other common text processing functions. These tools allow users to load their own domain dictionaries and stop word dictionaries. In this paper, we use Jieba to facilitate comparison with other Chinese keyword extraction methods. Word segmentation belongs to the data preprocessing process, and aims to obtain high-quality candidate words. Although word segmentation is not the core content of the algorithm, it has an extremely important impact on the results of the algorithm. To obtain candidate words and word graphs accurately and efficiently, any language without separators, not only Chinese, needs word segmentation.

C. TEXTRANK AND WORD GRAPH

We first need to find the nodes and edges in the graph to construct a word graph for a piece of text. The piece of text is the processed document by the method of Section 3.B. We do this by treating candidate words as nodes, the relationship between two nodes as edges, and constructing a word graph. In the word graph, the relationship between two nodes can be confirmed by using a sliding window model. The window size is typically 2-10 [7]. It can be considered that there is a co-occurrence relationship between two words if the two words appear in a window at the same time. In this paper, we use a dictionary to store graph relations in the experimental process. The dictionary structure is shown in Formula (1), where w_{ij} is the weight value of the edge.

$$(\text{Key} : v_i, \text{Value} : [(v_i, v_j, w_{ij}), (v_i, v_k, w_{ik}) \ldots\ldots]) \quad (1)$$

The score of each candidate word can be calculated according to the graph linkage relationship between words in the word graph. Formula (2) presents the calculation method, where d is the smoothing factor less than 1 to ensure convergence of iterations and $In(v_i)$ is the set of all neighboring

points of node v_i.

$$T_{v_i} = (1-d) + d * \sum_{v_j \in In(v_i)} \frac{w_{ji}}{\sum_{v_k \in In(v_j)} w_{jk}} T_{v_j} \quad (2)$$

D. BERT WORD VECTOR

It is generally accepted that computers cannot directly understand the semantic relationships between Chinese words. Understanding the semantic relationships of words requires inputting the words into the computer as word vectors. The vector representation of Chinese characters can be obtained by mapping them to a high-dimensional space through a neural network model. The common methods include BERT [18] and Word2vec [19]. Word2vec can obtain a vector representation of words in a specified corpus by using a shallow neural network to train the corpus data. Word2vec is suitable for the professional domain because it has fewer corpora than BERT. However, Word2vec depends on the corpus quality so that the model cannot obtain unregistered word vectors. BERT is a fine-tunable neural network model based on Transformer and self-attentive mechanisms released by Google in 2018. BERT is characterized by many parameters and a rich training corpus. BERT has the capacity to learn a number of semantic relations in text and is fine-tuned for various NLP tasks. Meanwhile, BERT takes a single word as the training unit. Thus, BERT makes it possible to obtain the vector representation of almost all Chinese words under the training of super large scale samples, which solves the problem of unregistered words in Word2vec. In this paper, we use Chinese_L-12_H-768_A-12, which is a Chinese pretraining model consisting of 12 layers of transformer and 12 self-attentive heads.

In the experiment, we use the BERT-Serving tool to obtain the candidate words' corresponding 768-dimensional word vectors and then store them. The input unit of the BERT model is a word, which is the candidate word in the word graph. Then, we use the cosine distance among the word vectors to obtain their semantic relationships in the text and optimize the state transfer matrix during the iterative process of TextRank.

E. K-TRUSS GRAPH

The K-Truss graph is a dense subgraph structure proposed by Cohen J in 2008 [20] that is derived from the concept of a cluster. A K-Truss graph is an extremely large subgraph of an undirected weighted simple graph G. In a K-Truss graph, every edge belongs to at least K-2 triangles. From the property of the K-Truss graph, every edge in the graph has at least K-2 strong relations. Thus, K-Truss graphs are smaller and tighter than K-Core graphs. K-Core graphs mainly concern the degree of nodes while K-Truss graphs focus more on the closeness of the relationships between nodes [14].

An edge E's truss level is K if it belongs to a K-Truss graph rather than a (K+1)-Truss graph. An edge is proven to be located in more triangles if the truss level is higher. Correspondingly, the relationship between nodes v_i and v_j at both ends of the edge is closer. Figure 2 shows an undirected graph G containing 10 nodes. The graph G is a 2-Truss graph with a truss level of 2 for each node. The subgraph composed of nodes [1, 2, 3, 4, 5, 6, 7, 9] is a 3-Truss graph with a truss level of 3 for each node. The subgraph composed of nodes [2, 3, 6, 7, 9] is both a 4-Truss and a 5-Truss graph, so that the truss levels of each node are 4 and 5.

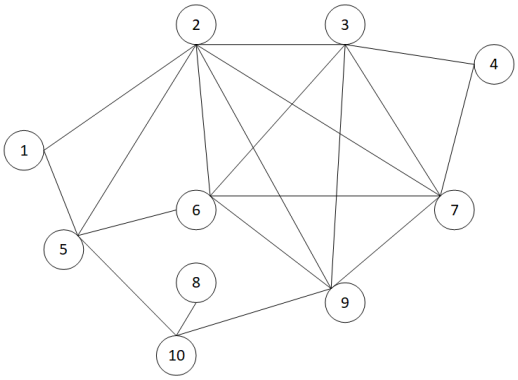

FIGURE 2. Undirected graph G.

$$K_{v_i} = max\left(\text{level}_{v_i \to v_j}\right), \quad v_i \to v_j \in E \quad (3)$$

The K-Truss diagram with different truss levels from 3 to n can be obtained by decomposing a word graph level by level.

The nodes in the highlevel K-Truss graph are all very closely related. The highest truss level of the nodes can be calculated by Formula (3). Taking Figure 3 as an example, the maximum truss level corresponding to nodes [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] in Figure G is [3, 5, 5, 3, 3, 5, 5, 2, 5, 2].

The result of the K-Truss method is multiple levels of K-truss subgraphs of a graph. The K-Truss features of words can be obtained through these subgraphs of different levels. However, there are still many words in the top-level subgraph. For example, there are often more than 10 words in the top-level subgraph of the sensor dataset. This number cannot meet the required number of general keyword extraction tasks (1-5). We cannot select keywords from many candidates because their K-Truss levels are the same. Therefore, in the keyword extraction task, the K-Truss method needs to be used in combination with other methods.

F. WORD FEATURES

a) $P_{cover}\left(v_i, v_j\right)$: TextRank initial node transfer probability

The probability of node v_i transferring to all its neighboring nodes in the original TextRank method is equal. In other words, each neighboring node has an equal chance to contribute to node v_i. The formula to calculate this transfer probability is shown in Formula (4), where $P_{cover}\left(v_i, v_j\right)$ denotes the transfer probability of node v_i to v_j,and deg (v_i) denotes the degree of node v_i. The iterative process of the TextRank algorithm is a hidden Markov process. In the hidden Markov process, each word has a probability of being transferred to

other connected words.

$$P_{cover}\left(v_i, v_j\right) = \begin{cases} \dfrac{1}{\deg\left(v_i\right)}, & \left(v_i \to v_j\right) \in E \\ 0, & \text{other} \end{cases} \tag{4}$$

b) $P_{Bs}\left(v_i, v_j\right)$: transfer probability of node v_i to v_j under the influence of BERT semantics

In an article, the keywords in the general text are usually words that can express different meanings rather than words with similar semantics. Thus, the words that are more semantically different from v_i in the neighboring nodes of v_i should be given a higher transfer probability. In this paper, we represent the semantic distance of words by using the cosine distance between vectors similarity(v_i, v_j) in Formula (5). In Formula (5), A_k is the component of v_i's BERT word vector, and B_k is the component of v_j's BERT word vector. m is the dimension of the BERT word vector, and $k \in [1, m]$. The transfer probability $P_{Bs}\left(v_i, v_j\right)$ of nodes v_i to v_j based on BERT semantics is shown by Formula (6).

$$similarity\left(v_i, v_j\right) = \frac{\sum_1^m A_k B_k}{\sqrt{\sum_1^m A_k^2}\sqrt{\sum_1^m B_k^2}}, \quad k \in [1, m] \tag{5}$$

$$P_{Bs}\left(v_i, v_j\right) = 1 - \text{similarity}(v_i, v_j) \tag{6}$$

c) $P_{loc}\left(v_i, v_j\right)$: transfer probability of nodes v_i to v_j under the influence of location

A node's location weight can be defined as the importance of the location where the word represented by the node appears in the text. We assume that the word appearing in the title has a greater chance of becoming a keyword. The word is given ~~a~~ more weight [12]. The words in other locations have decreasing weights according to their locations in the text. The location weight of node v_i is calculated by Formula (7). In the Formula (7), i is the position of the word in the text and $i \in [0, n]$. The closer v_i is to the beginning of the text, the smaller i and the larger L_{v_i}. In the Formula (7), a is a constant parameter and a > 1. The word location affects the probability of node v_i transferring to $v_j P_{loc}(v_i, v_j)$, as shown by Formula (8) because a word may appear in multiple locations in a text.

$$L_{v_i} = \begin{cases} 10, & \text{if } v_i \text{ in title} \\ \dfrac{1}{(i+a)}, & \text{if } v_i \text{ not in title} \end{cases} \tag{7}$$

$$P_{loc}(v_i, v_j) = \frac{L_{v_j}}{\sum_{v_k \in In(v_i)} L_{v_k}} \tag{8}$$

The final transfer probability $P(v_i, v_j)$ from node v_i to v_j can be calculated from the above multiple influences, which is expressed by Formula (9),where $\alpha = 0.33$, $\beta = 0.34$, and $\gamma = 3$.

$$P\left(v_i, v_j\right) = \left(\alpha P_{cover} + \beta P_{loc} + \gamma P_{Bs}\right) * \frac{w_{ji}}{\sum\limits_{v_k \in In\left(v_j\right)} w_{jk}} \tag{9}$$

We can determine that the iterative process of node scores is a Markov process from TextRank. The final node scores are only related to the state transfer matrix between nodes rather than the initial values of nodes. The final state transfer matrix of the nodes in the optimized word graph is shown in Formula (10), where P_{ij} denotes the probability of node v_i transferring to v_j. n is the number of nodes.

$$M_f = \begin{bmatrix} P_{11} & \cdots & P_{n1} \\ \vdots & \ddots & \vdots \\ P_{1n} & \cdots & P_{nn} \end{bmatrix} \tag{10}$$

$$B_i = (1 - d) + d * B_{i-1} * M_f \tag{11}$$

The iterative calculation process in the improved TextRank algorithm is shown by Formula (11), where B_i represents the score matrix of all nodes after the ith iteration. The iteration stops when the convergence error between the results of two consecutive iterations is small enough. B_0 is a row vector of length n. The initial score for each node is $1/n$. The parameter d is the smoothing factor, which is generally set to 0.85 to ensure the equation's stability and to converge at the iteration. After the iterations converge, the elements in matrix B_i are the final scores of the words. These scores are TBS_{v_i}. of the words.

d) K_{v_i}: truss level of node v_i

Node v_i may appear in multiple K-Truss subgraphs. In different subgraphs, the truss level of nodes v_i is also different. We need to find the highest truss level of the node. As shown in Formula (3), E denotes the edge containing v_i in graph G and $\text{level}_{v_i \to v_j}$ is the truss level of edge $E(v_i \to v_j)$. K_{v_i} is the highest truss level of node v_i. As the node truss level becomes higher, the node becomes more important, and the dense subgraph becomes smaller.

e) IDF_{v_i}: inverse document frequency of node v_i

IDF_{v_i} denotes the inverse document frequency of node v_i. The IDF value of node v_i can be obtained by using the existing corpus, which can be calculated in Formula (12). |D| denotes the total number of documents in the corpus, d_j denotes the jth document in the corpus, and Q_{v_i,d_j} denotes whether document d_j contains word v_i. The larger the IDF value, the fewer times the word appears, and the word may be more important in the corpus.

$$\text{IDF}_{v_i} = \lg \frac{|\text{D}|}{\left|\sum_{j=0}^{|\text{D}|} Q_{v_i,d_j}\right| + 1}, \quad Q_{v_i,d_j} = 1 \; if \; v_i \; in \; d_j \tag{12}$$

f) Score_{v_i}: final score of node v_i

The final score of node v_i consists of three parts: TBS_{v_i} of node v_i, node truss level K_{v_i} of node v_i, and inverse document frequency IDF_{v_i} of node v_i. The final score of node v_i is derived from Formula (13). The formula provides a secondary weighting for the candidate words obtained by the improved TextRank. The node truss feature filters out more important words. The inverse document frequency of the node filters out nonimportant words in the corpus. Finally, the effect of the algorithm makes further improvements.

$$\text{Score}_{v_i} = \text{TBS}_{v_i} * K_{v_i} * \text{IDF}_{v_i} \tag{13}$$

IV. EXPERIMENTS

A. DATASET

The keyword extraction algorithm uses three public datasets to ensure the objectivity and reproducibility of the results.

The Sensor dataset [3] has approximately 1,000 labeled articles. Each document in the dataset is professionally annotated and double-checked, which eliminates meaningless data such as image links and special symbols in the text. The data include article title, article body, and tag keywords. The average length of these articles is 1020. There are 3 keywords in each article. The dataset has 3,000 labeled keywords. The Sensor dataset has the highest quality among the three datasets.

The Sohu news dataset [3] has approximately 1,000 labeled articles. The average length of these articles is 860. There are 2-4 keywords in each article. The dataset has 2,700 labeled keywords.

The South Weekend news dataset [16] has approximately 1,500 labeled articles. The average length of these articles is 2,500. There are 3-4 keywords in each article. The dataset has 5,400 labeled keywords.

The BERT model is Chinese_L-12_H-768_A-12, which is a Chinese pretraining model. The obtainable vector dimension of the BERT model is 768.

B. IMPLEMENTATION DETAILS

The evaluation indices used in this experiment are Precision(P), Recall(R), and F1. These calculation methods are shown in Formulas (14), (15), and (16), respectively.

$$\text{Precision} = \left| \frac{Q_m \bigcap Q_p}{Q_p} \right| \tag{14}$$

$$\text{Recall} = \left| \frac{Q_m \bigcap Q_p}{Q_m} \right| \tag{15}$$

$$\text{F1} = \frac{2PR}{P+R} \tag{16}$$

Q_m indicates a set of marked keywords in the datasets, and Q_p indicates a set of extracted keywords in the experiment. F1 is used to evaluate both accuracy and recall as a whole.

1) EXPERIMENT 1

Comparing six different keyword extraction algorithms on three datasets

Experiment 1 chose to extract N = 1-10 keywords for comparative analysis of the results. Although there are only 3 standard keywords for each article in the dataset, we still extract 1-10 keywords to evaluate the recall rate of the algorithm.

M1: TextRank without loading domain dictionaries.

M2: TextRank loaded with domain dictionaries.

M3: SCTR algorithm [21].

M4: BSKT algorithm without IDF feature and K-truss feature.

M5: BSKT algorithm without K-truss feature.

M6: BSKT algorithm.

In the M3 method, the word graph iteration process uses K-means clustering to obtain the words in the cluster centers and uses them as partial features. In the M4 method, the word graph iteration process uses the BERT word vector to calculate the semantic similarity of two words, and then uses this as a partial feature. The M3 method is the latest unsupervised Chinese keyword extraction algorithm used in the process of writing this paper. Compared with the M3 method, the BSKT algorithm has been effectively verified. The comparison of the five algorithms (M1, M2, M4, M5, and M6) belongs to the ablation experiment, which verifies that the improvement method proposed in this paper is effective.

The experimental design ideas refer to [22]. In this paper, all formulas and algorithm implementations are implemented in Python. The text preprocessing uses the word segmentation tool Jieba, which can load the Baidu stop word dictionary and the dataset specialized domain dictionary. The word separation process screens [n, nz, ns, v, vd] to ensure the consistency of the word separation results. The number of iterations in TextRank is 10 or convergence, where the convergence error is 0.000001, parameter d = 0.85 and the sliding window size is 8. In the M3 method, clusters of k-means are 3, and other parameter settings are the same as in [19]. In the M4, M5, and M6 methods, parameters α, β, and γ are the same as in the third part. Compared with the M6 method, the M4 method does not use IDF features and K-truss features. Meanwhile, the M5 method does not apply the K-truss feature either. The results of Experiment 1 are shown in Tables 1-3. In addition, due to the high quality of the sensor dataset, we select the experimental results of the sensor dataset to draw a discounted statistical graph (Figures 3-5).

2) EXPERIMENT 2

Verifying the effect of sliding window size on the BSKT algorithm on Sensor dataset

Experiment 2 uses the BSKT algorithm proposed in this paper. It is useful to examine the effect of the sliding window size on the results by adjusting the size of the window in the TextRank composition process. In this experiment, the BSKT algorithm extracts three keywords. The number of iterations is 10, d = 0.85 and the sliding window size is 3-10. Parameters α, β, and γ are set as in Experiment 1. The results of Experiment 2 are shown in Figure 6.

C. RESULTS

It can be concluded that the addition of domain dictionaries has a greater impact on the keyword extraction results by comparing the M1 method and the M2 method in three tables. The three experimental results of the M2 method are better than those of M1 method in all indicators. This is because the dictionaries greatly improve the quality of candidate words in the data preprocessing stage. This conclusion validates the theory in Section 3.B.a. Comparing the three experimental results of the M4 method and the M2 method proves the effectiveness of the BERT semantic feature and word position feature proposed in this paper. Comparing the

TABLE 1. Six algorithm comparison for sensor dataset.

Method	N=1			N=3			N=5			N=7			Average(N=1-10)		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
M1	0.286	0.096	0.143	0.172	0.173	0.173	0.126	0.212	0.158	0.101	0.238	0.142	0.141	0.205	0.149
M2	0.541	0.181	0.272	0.340	0.342	0.341	0.254	0.426	0.318	0.203	0.478	0.285	0.277	0.408	0.294
M3	0.455	0.152	0.228	0.290	0.292	0.291	0.303	0.508	0.380	0.267	0.628	0.375	0.292	0.481	0.329
M4	0.574	0.192	0.288	0.356	0.358	0.357	0.265	0.445	0.332	0.211	0.497	0.297	0.291	0.417	0.309
M5	0.627	0.210	0.315	0.391	0.393	0.392	0.273	0.462	0.346	0.213	0.500	0.299	0.308	0.440	0.322
M6	0.640	0.214	0.321	0.402	0.404	0.403	0.292	0.489	0.366	0.228	0.534	0.319	0.322	0.464	0.339

TABLE 2. Six algorithm comparison for Sohu news dataset.

Method	N=1			N=3			N=5			N=7			Average(N=1-10)		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
M1	0.268	0.098	0.143	0.178	0.195	0.186	0.144	0.264	0.186	0.119	0.305	0.171	0.153	0.256	0.171
M2	0.309	0.113	0.165	0.216	0.238	0.226	0.165	0.302	0.213	0.138	0.355	0.199	0.178	0.293	0.199
M3	0.240	0.087	0.128	0.199	0.218	0.208	0.206	0.382	0.267	0.167	0.433	0.241	0.185	0.343	0.241
M4	0.336	0.123	0.180	0.218	0.240	0.228	0.170	0.311	0.220	0.137	0.352	0.197	0.181	0.297	0.201
M5	0.374	0.136	0.200	0.246	0.270	0.257	0.180	0.329	0.232	0.144	0.370	0.207	0.197	0.319	0.217
M6	0.406	0.152	0.221	0.272	0.302	0.286	0.202	0.374	0.262	0.155	0.402	0.223	0.218	0.352	0.269

TABLE 3. Six algorithm comparison for South Weekend news dataset.

Method	N=1			N=3			N=5			N=7			Average(N=1-10)		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
M1	0.316	0.089	0.139	0.203	0.171	0.186	0.151	0.213	0.177	0.122	0.242	0.163	0.165	0.205	0.163
M2	0.474	0.133	0.208	0.317	0.268	0.291	0.238	0.335	0.278	0.195	0.385	0.259	0.257	0.315	0.257
M3	0.362	0.102	0.159	0.226	0.191	0.207	0.224	0.316	0.262	0.197	0.389	0.261	0.221	0.304	0.231
M4	0.506	0.142	0.222	0.325	0.275	0.298	0.220	0.346	0.287	0.199	0.394	0.265	0.267	0.335	0.265
M5	0.569	0.160	0.250	0.359	0.303	0.329	0.268	0.379	0.314	0.217	0.428	0.288	0.295	0.365	0.291
M6	0.572	0.161	0.251	0.361	0.305	0.331	0.271	0.382	0.317	0.218	0.431	0.290	0.297	0.369	0.293

three experimental results of the M5 method and the M4 method proves the effectiveness of the IDF feature. Additionally, the three experimental results of the M6 method and the M5 method are compared, which proving the effectiveness of the K-Truss feature proposed in this paper in the BSKT algorithm.

Table 1 shows that when extracting 3 keywords, the M6 method improves the F1 by 11.2% compared with the M3 method. Table 2 shows that when extracting 3 keywords, the M6 method improves the F1 by 2.7% compared with the M3 method. Table 3 shows that when extracting 3 keywords, the M6 method improves the F1 by 12.4% compared with the M3 method. Table 1 and Table 2 show that the recall and F1 of the M3 method are significantly better than those of the M6 method when the number of extracted keywords >3. This is because the cluster center (candidate word) obtained by k-means clustering using the BERT word vector always belongs to the keyword range of the article under certain conditions. The BERT word vector clustering works well. This is the main reason for the increase in the recall rate of the algorithm, which leads to an increase in F1. However, Table 3 shows that the M3 method is less effective on the South Weekend news dataset because the longer articles in the dataset lead to poor clustering results. In general, Table 1 shows that on the average results of extracting 1-10 keywords, the M6 method has a 3% increase in precision, 2% decrease in recall, and 1% increase in F1 compared to the M3 method. Table 2 shows that on the average results of extracting 1-10 keywords, the M6 method has a 3.3% increase in precision, 0.9% increase in recall, and 2.8% increase in F1 compared to the M3 method. Table 3 shows that on the average results of extracting 1-10

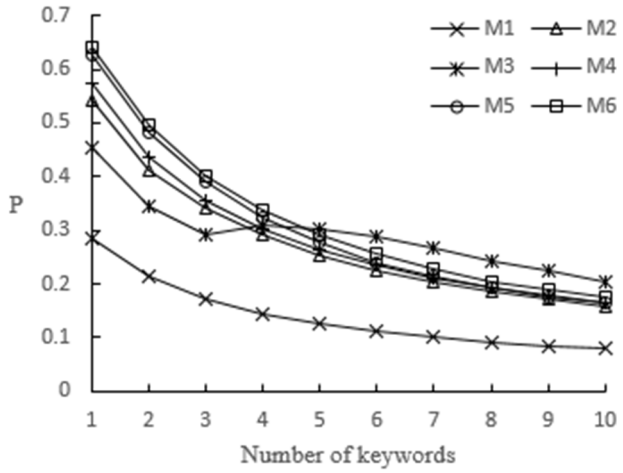

FIGURE 3. Precision on sensor dataset.

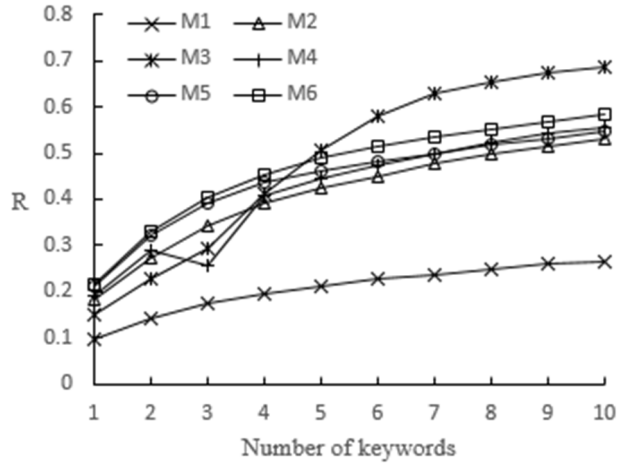

FIGURE 4. Recall on sensor dataset.

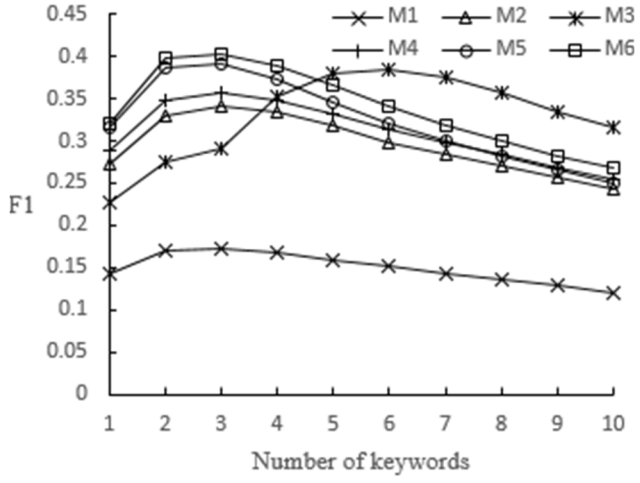

FIGURE 5. F1 on sensor dataset.

keywords, the M6 method has a 7.6% increase in precision, 6.5% increase in recall, and 6.2% increase in F1 compared to the M3 method. In conclusion, the M6 method BSKT works better than the M3 method SCTR.

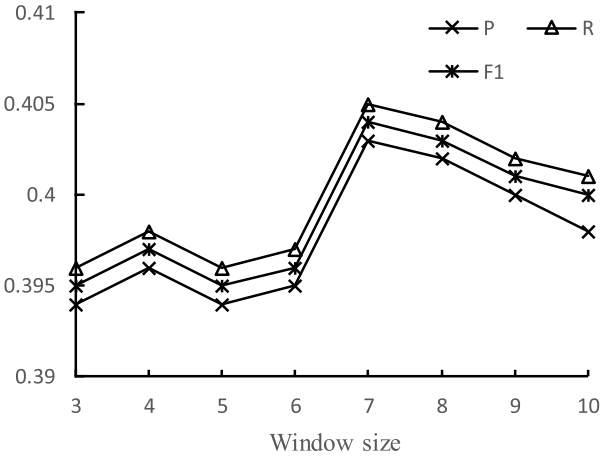

FIGURE 6. Effect of sliding window size on sensor dataset.

Figure 6 shows that the size of the sliding window has a great influence on the keyword extraction results during the construction of the word graph by TextRank. Figure 6 shows the keyword extraction effect is best when the window size is 7 in this experiment. As the sliding window increases, the overall keyword extraction effect shows an augmentation first and then a loss.

V. CONCLUSION AND FUTURE WORK

Chinese keyword extraction is an important technique to rapidly obtain document topics and core contents. For many tasks in the field of NLP, Chinese keyword extraction is the primary task and of great significance. In this paper, we proposed a keyword extraction algorithm combined with BERT semantics and the K-Truss graph algorithm (BSKT), which incorporates BERT semantic features, K-Truss graph features, word location features, and word inverse document frequency features. The experimental results demonstrated that this method significantly outperforms the other five methods mentioned in the paper in terms of the keyword extraction effect. BERT semantic features, K-Truss graph features, word location features, and IDF features proposed in the BSKT algorithm all play a positive role in keyword extraction. And in the keyword extraction task, the fused feature is better than any single feature. At the same time, the BSKT algorithm requires only a few specialized domain corpora in the data preprocessing process and the word inverse document frequency calculating process. Thus, the BSKT algorithm is easier to transfer to other domains for keyword extraction and is of high practical value.

In future work, we will introduce the vector clustering feature of words to the BSKT algorithm. We can also bring the K-Truss feature to the TextRank iterative process. We can obtain preferable results of keyword extraction by taking these measures. In addition, the deep learning method to extract keywords is the focus of our future work.

REFERENCES

[1] H. P. Luhn, "The automatic creation of literature abstracts," *IBM J. Res. Develop.*, vol. 2, no. 2, pp. 159–165, Apr. 1958.

[2] S. Sun, C. Xiong, Z. Liu, Z. Liu, and J. Bao, "Joint keyphrase chunking and salience ranking with BERT," 2020, *arXiv:2004.13639*.

[3] T. Ding, W. Yang, F. Wei, C. Ding, P. Kang, and W. Bu, "Chinese keyword extraction model with distributed computing," *Comput. Elect. Eng.*, vol. 97, Jan. 2022, Art. no. 107639.

[4] S. Kong, P. Zhu, Q. Yang, and Z. Wei, "HSCKE: A hybrid supervised method for Chinese keywords extraction," in *Proc. 3rd Int. Conf. Algorithms, Comput. Artif. Intell.*, Dec. 2020, pp. 1–7.

[5] T. Joachims and B. Str, *A Probabilistic Analysis of the Rocchio Algorithm With TFIDF for Text Categorization*. Cham, Switzerland: Springer, 1997.

[6] K. S. Jones, "Index term weighting," *Inf. Storage Retr.*, vol. 9, no. 11, pp. 619–633, Nov. 1973.

[7] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proc. Conf. EMNLP, ACL*, 2004, pp. 404–411.

[8] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford Digit. Libraries Work. Paper, Stanford, CA, USA, Tech. Rep., 1998. [Online]. Available: http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf, doi: 10.1007/978-3-319-08789-4_10.

[9] W. Li and J. Zhao, "TextRank algorithm by exploiting Wikipedia for short text keywords extraction," in *Proc. 3rd Int. Conf. Inf. Sci. Control Eng. (ICISCE)*, Jul. 2016, pp. 683–686.

[10] Z. Huang and Z. Xie, "A patent keywords extraction method using TextRank model with prior public knowledge," *Complex Intell. Syst.*, vol. 8, no. 1, pp. 1–12, Feb. 2022.

[11] A. Kazemi, V. Pérez-Rosas, and R. Mihalcea, "Biased TextRank: Unsupervised graph-based content extraction," 2020, *arXiv:2011.01026*.

[12] Y. Li, J. Cui, and J. Ji, "A keyword extraction algorithm based on Word2vec," *e-Sci. Technol. Appl.*, vol. 6, pp. 54–59, Jun. 2015.

[13] T. Xia, "Extracting keywords with modified TextRank model," *Data Anal. Knowl. Discovery*, vol. 1, no. 2, pp. 28–34, 2017.

[14] A. Tixier, F. Malliaros, and M. Vazirgiannis, "A graph degeneracy-based approach to keyword extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1860–1870.

[15] L. Yao, Z. Pengzhou, and Z. Chi, "Research on news keyword extraction technology based on TF-IDF and TextRank," in *Proc. IEEE/ACIS 18th Int. Conf. Comput. Inf. Sci. (ICIS)*, Jun. 2019, pp. 452–455.

[16] Z. Xu and J. Zhang, "Extracting keywords from texts based on word frequency and association features," *Proc. Comput. Sci.*, vol. 187, pp. 77–82, Jan. 2021.

[17] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "YAKE! Keyword extraction from single documents using multiple local features," *Inf. Sci.*, vol. 509, pp. 257–289, Jan. 2020.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[19] Y. Goldberg and O. Levy, "word2vec explained: Deriving Mikolov's negative-sampling word-embedding method," 2014, *arXiv:1402.3722*.

[20] J. Cohen, "Trusses: Cohesive subgraphs for social network analysis," Nat. Secur. Agency, Fort Meade, MD, USA, Tech. Rep. 16(3.1), 2008. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.505.7006&rep=rep1&type=pdf

[21] A. Xiong, D. Liu, H. Tian, Z. Liu, P. Yu, and M. Kadoch, "News keyword extraction algorithm based on semantic clustering and word graph model," *Tsinghua Sci. Technol.*, vol. 26, no. 6, pp. 886–893, Dec. 2021.

[22] M. Zhang, X. Li, S. Yue, and L. Yang, "An empirical study of TextRank for keyword extraction," *IEEE Access*, vol. 8, pp. 178849–178858, 2020.

WENMING GUO received the B.S. degree in applied mathematics and software engineering from the Beijing Institute of Technology, Beijing, China, in 1989, and the M.E. degree in system engineering from the North University of China, Taiyuan, China, in 1992. He joined the Beijing University of Posts and Telecommunications, in 2002, and is currently a Professor with the School of Computer Science. He is the author of seven books, more than 60 articles, and more than 20 inventions. His research interests include computer vision, big data application, and intelligent information processing.

ZIHAO WANG received the B.E. degree in computer science and technology from Beijing Information Science and Technology University, Beijing, China, in 2019. He is currently pursuing the master's degree in software engineering with the Beijing University of Posts and Telecommunications, Beijing. His research interest includes natural language processing.

FANG HAN (Member, IEEE) received the bachelor's degree in computer science and technology from Xinjiang Normal University, Ürümqi, China, in 2001, and the master's degree in computer technology from Xinjiang University, Ürümqi, in 2004. Since 2013, she has been an Associate Professor with the Information Engineering College, Xinjiang Institute of Engineering. She is the author of three books, more than 20 articles, and four inventions. Her research interests include computer networks, big data application, and the Internet of Things engineering.