mathematics

MDPI

Article

A Graph-Based Keyword Extraction Method for Academic Literature Knowledge Graph Construction

Lin Zhang *, Yanan Li and Qinru Li

School of Maritime Economics and Management, Dalian Maritime University, Dalian 116026, China; jemmali@dlmu.edu.cn (Y.L.); liqinru@dlmu.edu.cn (Q.L.)
* Correspondence: zhanglin@dlmu.edu.cn

Abstract: In this paper, we construct an academic literature knowledge graph based on the relationship between documents to facilitate the storage and research of academic literature data. Keywords are an important type of node in the knowledge graph. To solve the problem that there are no keywords in some documents for several reasons in the process of knowledge graph construction, an improved keyword extraction algorithm called TP-CoGlo-TextRank is proposed by using word frequency, position, word co-occurrence frequency, and a word embedding model. By combining the word frequency and position in the document, the importance of words is distinguished. By introducing the GloVe word-embedding model, which brings the external knowledge of documents into the TextRank algorithm, and combining the internal word co-occurrence frequency in the documents, the word-adjacency relationship is transferred non-uniformly. Finally, the words with the highest scores are combined into phrases if they are adjacent in the original text. The validity of the TP-CoGlo-TextRank algorithm is verified by experiments. On this basis, the Neo4j graph database is used to store and display the academic literature knowledge graph, to provide data support for research tasks such as text clustering, automatic summarization, and question-answering systems.

Keywords: keyword extraction; TextRank; word embedding; text statistical features; academic literature knowledge graph

MSC: 68T50

Citation: Zhang, L.; Li, Y.; Li, Q. A Graph-Based Keyword Extraction Method for Academic Literature Knowledge Graph Construction. *Mathematics* **2024**, *12*, 1349. https://doi.org/10.3390/math12091349

Academic Editors: Ravil Muhamedyev and Evgeny Nikulchev

Received: 14 March 2024
Revised: 21 April 2024
Accepted: 23 April 2024
Published: 29 April 2024

1. Introduction

With the wide application and continuous development of Internet technology, the information contained in the academic literature available online has shown explosive growth. This has posed great challenges for researchers concerning their ability to quickly and accurately discover desired information from a large amount of material and to use the information to support and enrich their research.

To help users find accurate answers within the shortest possible time and improve user experience, Google launched the Knowledge Graph in 2012 [1]. Essentially, a knowledge graph is a semantic network composed of entities as nodes and relationships as edges representing the semantic relationships between entities [2]. A knowledge graph uses semantic retrieval methods to collect information from a variety of sources and provides answers with a complete body of knowledge to user's queries. As an intelligent and efficient means of knowledge organization, knowledge graphs have been widely used in fields such as question-answering, recommendation systems, medicine, and biology [3].

Therefore, to help researchers develop a comprehensive and clear understanding of the entire research field within the shortest possible time, so as to provide possibilities for more in-depth knowledge mining, this paper prepares to construct an academic literature knowledge graph based on the relationships between academic documents. Based on the context-rich heterogeneous graph generated via full-text publications by Liu et al. [4], this paper develops a revised graph schema [5] for an academic literature knowledge graph to

show the relationship between paper P_1 and P_2, as shown in Figure 1. It can be seen that keywords are a very important type of node in the academic literature graph schema.

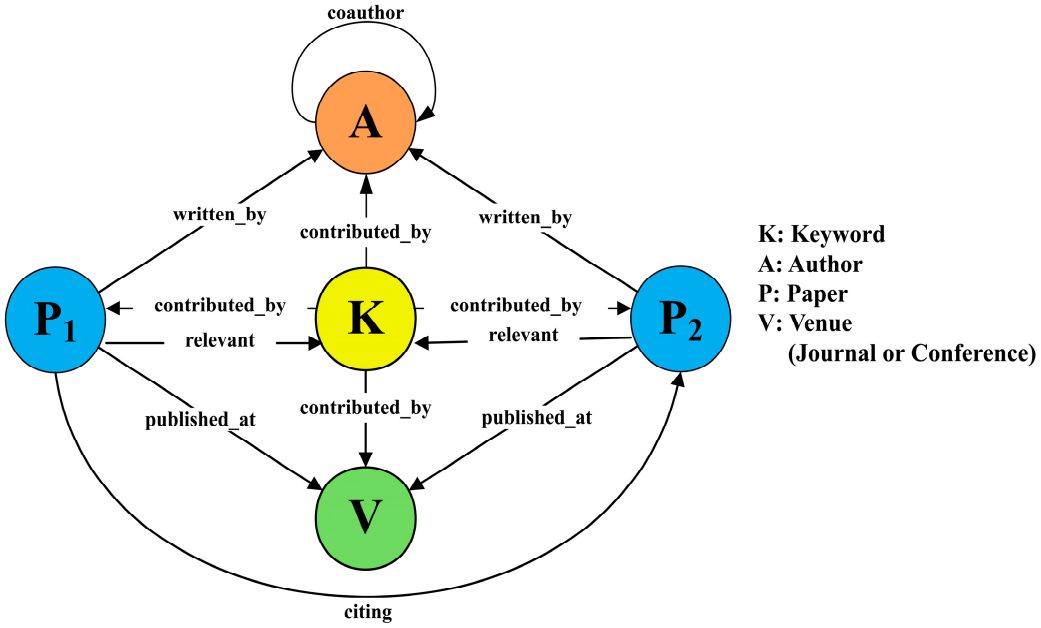

Figure 1. Graph schema for the academic literature knowledge graph. (Revised graph schema based on Liu et al. [4]).

Keywords provide a compact representation of documents and can be used as a substitute for documents in applications such as automatic summarization, classification, clustering, and sentiment analysis. At present, most journals or conferences require authors to provide keywords when submitting academic papers, but this part is often missing in papers published earlier. In addition, in the process of converting documents from PDF to plain text, the keywords of some documents may also be lost due to problems such as document encoding and character processing. Therefore, to construct a more comprehensive academic literature knowledge graph and provide high-quality data sources for the following studies, it is necessary to carry out keyword extraction for the documents without keywords.

TextRank [6] is a popular graph-based unsupervised algorithm that only uses the internal word structure information of a document to extract keywords. On the basis of the TextRank algorithm, this paper proposes a keyword extraction algorithm called TP-CoGlo-TextRank by using word frequency, position, word co-occurrence frequency, and the GloVe word embedding model. The main contributions of this paper are as follows: (1) We use word frequency and position to distinguish the difference in word importance, then combine the two parts of importance to form the overall importance of words; (2) We propose to use the internal and external word co-occurrence frequency information of documents to compute the influence transferred from the adjacent words and combine the two parts of influences to form the overall transition probability matrix of the candidate keywords.

The rest of the paper is organized as follows. Section 2 provides an overview of related studies. Section 3 presents our methodology and design in detail. Section 4 describes the experiments and discusses the results. Section 5 constructs the academic literature knowledge graph. Section 6 concludes the paper.

2. Related Work

2.1. Automatic Keyword Extraction

Over the past fifty years, scholars have studied keyword extraction from multiple perspectives, including statistics, automatic term indexing, information retrieval, natural language processing (NLP), and the emerging neural paradigm [7]. Keyword extraction algorithms are mainly categorized as supervised algorithms and unsupervised algorithms [8]. Compared with supervised algorithms, which are costly, time-consuming, and overfitting, unsupervised algorithms are more widely used. At present, the commonly used unsupervised keyword extraction algorithms can be broadly classified into three categories: statistical feature-based algorithms, graph-based algorithms, and topic model-based algorithms.

Statistical feature-based keyword extraction algorithms usually determine text keywords by quantifying statistical features such as part-of-speech, term frequency (TF), term frequency-inverse document frequency (TF-IDF), word position, and word length. Wang et al. [9] proposed a keyword extraction model, SRP-TF-IDF, for the field of scientific research by introducing a weight balance algorithm. SRP-TF-IDF combines the TF-IDF values of words with the semantic similarity between words and information on the scientific research project, such as the project title, research fields, and subproject titles, to obtain the weights of words. Rathi et al. [10] proposed a unigram keyword extraction method using features such as relative entropy, variance, and displacement of terms in a document. The proposed method is competitive compared with TF and YAKE! [11,12], which uses five text statistical features, such as word casting, word position, word frequency, word relatedness to context, and word different sentences, to extract keywords from individual documents. Among the three features, variance and relative entropy are dominant in weighting terms. Lu et al. [13] proposed an incremental TF-IDF keyword feature extraction method, ITFIDF-LP, which takes word position and part of speech into account.

Graph-based keyword extraction algorithms regard text words as nodes, perform the importance ranking of nodes according to their relationship, and select the most important words as keywords. Goz et al. [14] proposed a keyword extraction method, MGRank, based on weighted complete multigraphs. MGRank eliminates the sliding window size parameter, allows for the establishment of multiple relationships between candidate keywords, and weighs edges according to the positional distance of candidate keywords, thereby representing the text document as a multigraph-based graph-of-word structure. Jain et al. [15] proposed the keyword extraction method F-GAKE based on fuzzy graph connectivity measures to solve the problem that many graph-based methods take co-occurrence as an edge weight and neglect the semantic relationship between words. Compared with the state-of-the-art methods, F-GAKE has higher precision and recall in keyword extraction from short messages. Yan et al. [16] proposed a graph-based method incorporating with clustering algorithm for keyword extraction, which takes the impact of sentences on word importance into consideration and measures the importance of words by building three graphs—namely a word-to-word graph, a sentence-to-sentence graph, and a sentence-to-word graph—and selects keywords combined with the centroids of the K-means algorithm. The proposed method outperforms TF-IDF and TextRank on datasets of Hulth2003 and DUC2001. Abimbola et al. [17] proposed a graph-based noun-centric keyword extraction method, which integrates the TextRank algorithm, noun phrase identifier, AP algorithm, K-means algorithm, and cosine similarity to extract keywords. Precision, recall, and the F_1-measure are all improved by more than 5% compared to the method proposed by Yan et al. [16].

A document is usually composed of several topics, and a topic is composed of a group of words. The topic model-based keyword extraction algorithms extract the words with the best topic coverage from the document as keywords by analyzing the document topic distributions and the word topic distributions. Liu et al. [18] proposed Topical PageRank (TPR) for keyword extraction by integrating the LDA topic model and the PageRank algorithm. By calculating the PageRank value of words on different topics and combining with the topic distribution of the document, the TPR method obtains the final ranking of

words and extracts the top-ranked words as keywords. Compared with TF-IDF, PageRank, and LDA, TPR performs best in keyword extraction. Based on TPR [18], Teneva et al. [19] proposed the Salience Rank (SR) method, which balances the topic specificity and corpus specificity of words by introducing word salience. The SR method extracts comparable or better keywords than TPR; also, it runs PageRank only once when ranking words, so it is more efficient than TPR. Lu et al. [13] proposed a subject word feature extraction method, LDA-SLP, based on LDA, Word2Vec-based similarity, word position, and the part of speech.

With the development and application of deep learning technology in the field of NLP, some scholars have begun to apply deep learning models to keyword extraction tasks in recent years, to better capture the semantic information of keywords by automatically learning the deep semantic features of text. Sarracén et al. [20] proposed a keyword extraction method for offensive tweets, which uses the weights learned by the multi-head self-attention mechanism of BERT to generate a word graph and then extracts keywords according to eigenvector centrality. Duan et al. [21] proposed OILog, an online incremental log keyword extraction method based on deep LSTM. OILog can accurately capture high-frequency log keywords and new log keywords generated by the system to help assist the operations staff in effectively maintaining and monitoring the system. Based on the LSTM model and combined with contextual information on the target word, Zhang et al. [22] proposed a target center-based LSTM (TC-LSTM) model with a self-attention mechanism, which has good performance in keyword extraction.

To improve the extraction effect of keywords, scholars usually use a combination of multiple methods—one of which is the main method supplemented by others—to extract keywords, such as the studies mentioned above by Lu et al. [13], Jain et al. [15], Abimobola et al. [17], Liu et al. [18], and Teneva et al. [19]. Deep learning-based methods have become important means of keyword extraction because of their high accuracy, adaptability, and generalization performance. However, deep learning models are more dependent on the high memory and computing power of terminal devices than other models because they need large-scale datasets to train parameters, and the more parameters there are, the higher the computational complexity and time complexity. Graph-based keyword extraction methods are widely used and effective, so this paper conducts an in-depth study on the TextRank algorithm, a graph-based method.

2.2. Automatic Keyword Extraction Based on TextRank

TextRank [6] is a classic graph-based ranking algorithm inspired by Google's PageRank [23]. It splits up the document into words, represented by nodes, and constructs edges between nodes based on word co-occurrence relationships. Thus, the document words are converted into a graph $G = <V, E>$, where V is a set of word nodes and E is a set of edges connecting the nodes. The word co-occurrence relationship refers to the co-occurrence of words within a sliding window of a given size in the same document. The importance of a node in the word graph is determined by the importance of other nodes pointing to it—that is, the importance of a word is related to its co-occurring words within the sliding window.

The formula of TextRank is shown in Equation (1), where $S(V_i)$ represents the importance score of the word node V_i, N represents the total number of words in the document, $In(V_i)$ represents the set of nodes pointing to V_i, $Out(V_j)$ represents the set of nodes pointed to by V_j, and d is the damping factor, which is usually set to 0.85.

$$S(V_i) = \frac{1-d}{N} + d \times \sum_{V_j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j) \quad (1)$$

The TextRank algorithm has some major advantages, such as simplicity and high efficiency. However, it treats all co-occurring words of the target word as equally important and uniformly assigns the importance score of the target word to its co-occurring words, resulting in poorer keyword extraction. Scholars have conducted extensive and in-depth studies on this problem and put forward many improved methods.

Liu et al. [24] proposed a TextRank_Revised algorithm by taking the conditional probability of co-occurring words in the same sentence as the weight of the edge between nodes. TextRank_Revised achieves good keyword extraction performance on both labeled and unlabeled samples. Gu et al. [25] combined the LDA topic model and TextRank algorithm for keyword extraction. LDA is used to obtain the topic influence of words in the document, which is incorporated into the construction of the transition probability matrix to realize the non-uniform transfer of word importance according to topic influences and adjacency relationships. Experiments show that the combination of LDA and TextRank can extract better keywords than LDA and TextRank alone. Xia [26] proposed a weighted TextRank algorithm based on word vector clustering. The improved method takes the distance between words and cluster centroids as the clustering influence between words and combines the coverage influence and position influence of words to construct the transition probability matrix. Experiments show that the improved method can extract keywords effectively. When using the TextRank algorithm to extract key information from forestry text, Chen et al. [27] take the comprehensive weight of words obtained based on features such as word length, word span, and title as the initial weight of nodes, and the cosine similarity between nodes as the edge weight. Xiong et al. [28] proposed a semantic clustering news keyword extraction algorithm, SCTR, based on TextRank. SCTR uses word vectors generated by the BERT model to carry out *k*-means clustering and constructs the TextRank weight transition probability according to the clustering results. Based on TextRank, Guo et al. [29] optimize the state transition probability matrix by using multiple features, such as BERT semantic similarity features, word position features, and word coverage features. Qiu et al. [30] improved TextRank by using the membership of words to each document obtained by the tolerance rough set theory as the initial word weight and the fuzzy membership of words to each word tolerance class to optimize the transfer probability between nodes.

The above studies mainly focus on optimizing the transition probabilities between words by combining other models or text features. Based on these studies, this paper proposes the TP-CoGlo-TextRank algorithm for keyword extraction by using statistical features-based approaches and graph-based approaches. Specifically, in the iterative process of the TP-CoGlo-TextRank algorithm, words are weighted by text statistical features, such as word frequency and position, and ranked by the graph-based TextRank algorithm. In addition, a word embedding model is introduced to add external knowledge vectors from a corpus to TextRank, and word co-occurrence frequency and the word embedding model are combined to realize the non-uniform transfer of word scores to the co-occurring words.

3. TP-CoGlo-TextRank Keyword Extraction Algorithm

3.1. Word Frequency

Term frequency (TF) refers to the number of occurrences of words in a document, which is an indicator for measuring word importance in the document from a global perspective. It is generally believed that the more occurrences of a word there are in a document, the higher the word's relevance to the document topic and the more likely it is to become a keyword [31].

The traditional TextRank algorithm is limited to taking co-occurrence word pairs in the sliding window, which can only consider the local adjacency relationship between words—that is, the extracted keywords can only reflect the local, not the global, information of the document. Therefore, to measure the importance of words in the word graph from a global perspective, this paper introduces word frequency into the traditional TextRank algorithm.

Considering the influence of text length on word frequency, when computing word frequency, this paper standardizes the number of occurrences of words in a document using the document length (i.e., the total number of words in the document), as shown in

Equation (2), where *count*(*i*) represents the number of occurrences of word *i* in the document and *doc_length* represents the length of the document.

$$tf(i) = \frac{count(i)}{doc\_length} \tag{2}$$

3.2. Word Position Importance

The title of an academic document is a one-sentence summary of the core contents of the document. Compared with the words appearing in other parts of the document, the effective words (such as nouns, verbs, and adjectives) appearing in the title often better express the research topics of the document and are more likely to become keywords. Therefore, when using TextRank to extract keywords from documents, the title words in the word graph should be given higher weights [32]. This paper uses Equation (3) to measure the position importance of words.

$$pos(i) = \begin{cases} 1, & \text{word } i \text{ in the title.} \\ \beta, & \text{word } i \text{ in the abstract. } 0 < \beta \leq 1 \end{cases} \tag{3}$$

When word *i* appears in the title, its weight is 1. When word *i* appears in the abstract, its weight is β, $0 < \beta \leq 1$. The value of β will be determined by experiments in Section 4.3.2.

3.3. Word Co-Occurrence Frequency

It can be seen from Equation (1) that the traditional TextRank algorithm uses the word co-occurrence relationship within the document and assigns the importance score of the word uniformly to its co-occurring words when measuring word importance, ignoring the differences between these words. For word V_i, in general, words with more co-occurrences with V_i in the sliding window of a given size are more correlated with V_i than words with less co-occurrences with V_i. In other words, words with more co-occurrences with V_i should be assigned higher scores by V_i. Therefore, to better assign the importance scores of words to their co-occurring words, this paper improves the transition probability of the TextRank algorithm by using the word co-occurrence frequency within the document.

The formula used to calculate the co-occurrence frequency of words V_j and V_i within the sliding window in the document is shown in Equation (4), where $Out(V_i)$ represents the set of words co-occurring with V_i, and *co-occurrences*(V_i, V_j) represents the number of co-occurrences of V_i and V_j.

$$Co(V_i, V_j) = \frac{co\text{-}occurrences(V_i, V_j)}{\sum_{V_k \in Out(V_i)} co\text{-}occurrences(V_i, V_k)} \tag{4}$$

3.4. GloVe Word Embedding

GloVe (Global Vectors for Word Representation) [33] is a distributed word representation model that combines the advantages of prior statistics extracted from the global word co-occurrence matrix and the local context windows. It overcomes the problems related to the high computational complexity of the LSA model computing singular value decomposition (SVD) for large matrices and the drawbacks of the Word2Vec model, which does not make full use of the global corpus.

GloVe is an unsupervised learning algorithm for obtaining word vector representations. It first creates a word-context co-occurrence matrix based on a given corpus. Each element in the matrix represents how often two words co-occur within a context window of a specific size. Then it constructs a loss function that is optimized by regression. GloVe essentially learns word vectors by performing dimensionality reduction on the co-occurrence matrix.

As mentioned in Section 3.3, the traditional TextRank algorithm uses the co-occurrence relationship between words within the document to assign word importance scores, without considering the differences in the similarities between the word and all its co-occurring

words. For word V_j, in general, words with higher similarity to V_j within the sliding window of a given size are more correlated with V_j and should be assigned higher scores by V_j. Considering that the GloVe model is a distributed representation method of words based on global word-word co-occurrence statistics from a corpus, to better measure word importance, this paper uses external knowledge of documents, that is, the GloVe model, to optimize the transition probabilities between words when using the traditional TextRank algorithm.

The similarity of the words V_i and V_j is calculated using cosine similarity and GloVe word embedding, as shown in Equation (5), where $\vec{V_i}$ represents the word vector of V_i based on the GloVe model and $\left\|\vec{V_i}\right\|$ represents the magnitude of vector $\vec{V_i}$.

$$sim(V_i, V_j)' = \frac{\vec{V_i} \cdot \vec{V_j}}{\left\|\vec{V_i}\right\| \cdot \left\|\vec{V_j}\right\|} \tag{5}$$

The cosine similarity takes values between -1 and 1. Considering that the transition probability is non-negative and unbounded, to assign word importance scores based on similarity, we normalize the cosine similarity between words V_i and V_j to [0, 1] using Equation (6), where $\min_{V_p, V_q \in D} sim(V_p, V_q)'$ and $\max_{V_p, V_q \in D} sim(V_p, V_q)'$ represent the minimum and maximum cosine similarity between words in the dataset, respectively. Based on the value range of cosine similarity, the minimum similarity between words is set to -1 and the maximum similarity is set to 1, that is, $\min_{V_p, V_q \in D} sim(V_p, V_q)' = -1$, $\max_{V_p, V_q \in D} sim(V_p, V_q)' = 1$. Therefore, Equation (6) can be transformed into Equation (7).

$$sim(V_i, V_j) = \frac{sim(V_i, V_j)' - \min_{V_p, V_q \in D} sim(V_p, V_q)'}{\max_{V_p, V_q \in D} sim(V_p, V_q)' - \min_{V_p, V_q \in D} sim(V_p, V_q)'} \tag{6}$$

$$sim(V_i, V_j) = \frac{sim(V_i, V_j)' + 1}{2} \tag{7}$$

On this basis, the proportion of the similarity between words V_j and V_i in the total similarities between V_i and all of its co-occurring words can be obtained, as shown in Equation (8), where $sim(V_i, V_j)$ represents the similarity between V_i and V_j obtained by Equation (7), and the meaning of $Out(V_i)$ is given in Equation (4).

$$Glo(V_i, V_j) = \frac{sim(V_i, V_j)}{\sum_{V_k \in Out(V_i)} sim(V_i, V_k)} \tag{8}$$

3.5. TP-CoGlo-TextRank Algorithm

Based on the traditional TextRank algorithm, the TP-CoGlo-TextRank keyword extraction algorithm is proposed by combining word frequency, position, word co-occurrence frequency, and a word embedding model, as shown in Equation (9).

$$S(V_i) = w_t(V_i) \times \left[\frac{(1-d)}{N} + d \times \sum_{V_j \in In(V_i)} w_e(V_j, V_i) S(V_j) \right] \tag{9}$$

In Equation (9), $w_t(V_i)$ represents the overall weighting coefficient of the word V_i, as shown in Equation (10). $tf(V_i)$ and $pos(V_i)$ represent the word frequency and position importance of the word V_i, as shown in Equations (2) and (3), respectively. λ is the proportion of the position importance in the overall weighting coefficient of the word V_i,

$0 \leq \lambda \leq 1$. $w_e(V_j, V_i)$ represents the transition probability from V_j to V_i, which is optimized by utilizing the internal word co-occurrence frequency information and the external word embedding model, as shown in Equation (11), where $Co(V_j, V_i)$ and $Glo(V_j, V_i)$ are shown in Equations (4) and (8), respectively. γ represents the proportion of the co-occurrence frequency in the total transition probability from V_j to V_i, $0 \leq \gamma \leq 1$.

$$w_t(V_i) = \lambda pos(V_i) + (1 - \lambda) tf(V_i) \tag{10}$$

$$w_e(V_j, V_i) = \gamma Co(V_j, V_i) + (1 - \gamma) Glo(V_j, V_i) \tag{11}$$

The TP-CoGlo-TextRank algorithm is language-independent. When using this algorithm for keyword extraction, we only need to train the GloVe model or select a publicly available pre-trained GloVe model for the language used in the text dataset.

The procedure of the TP-CoGlo-TextRank algorithm is shown in Algorithm 1.

Algorithm 1 TP-CoGlo-TextRank Algorithm

Input: *document*, *GloVe.model*, sliding window size *sw*, damping factor d, iteration threshold θ, *max_iterations*, the number of words to be extracted k
Output: keywords extracted from *document*

#(**Step 1**) Preprocessing
document = lowercase(*document*)
sentences = split up *document* into sentences
words = split up *document* into words
for each *word* in *words* **do**
is_stopword (*word*), is_singleword (*word*), is_specialsymbol (*word*), postagging (*word*)
end for
#(**Step 2**) Text features extraction
for each *word* in *words* **do**
compute term frequenty *tf*(*word*) by Equation (2)
compute position importance *pos*(*word*) by Equation (3)
compute the overall weighting coefficient w_t(*word*) by Equation (10)
end for
#(**Step 3**) Candidate keyword score
compute word co-occurrence frequency matrix M within a given sliding window size *sw*
construct the word graph $G = <V, E>$
for each (V_i, V_j) in M **do**
compute word co-occurrence frequency-based transition probability $Co(V_i, V_j)$ by Equation (4)
compute similarity-based transition probability $Glo(V_i, V_j)$ by Equation (8)
get the final transition probability $w_e(V_i, V_j)$ by Equation (11)
end for
for each *word* in *words* **do**
compute *word* score by Equation (9)
end for
#(**Step 4**) Ranking
sort *words* by descending scores
initial_keywords = top k words with the highest scores
#(**Step 5**) Combining adjacent keywords into phrases
for each *sentence* in *sentences* **do**
if two or more keywords adjacent in *sentence*
combine them into a phrase as a new keyword
end if
end for

Zhang et al. [34] evaluated the influence of the sliding window size on the performance of the TextRank algorithm in keyword extraction by extracting the top (k = 5, 10, 15) most important words as keywords on the Hulth2003 and Krapivin2009 datasets; they concluded

that the TextRank algorithm had the best performance when the sliding window size was 3. Therefore, when using the TP-CoGlo-TextRank algorithm to extract keywords from documents, this paper will follow the research results of Zhang et al. [34] and set the sliding window size to 3.

4. Experiment and Discussion

4.1. Experimental Dataset

Considering that the number of keywords in a document is generally between three and five, we selected 10,000 documents published after 2010 with a title, abstract, and three to five keywords from the DBLP-Citation-network V14 [35] available from the AMiner website at https://www.aminer.cn/billboard/citation (accessed on 19 February 2024) as the experimental dataset. We used a five-fold cross-validation method to divide the dataset into five equal parts and used four of them as the training set and the other one as the test set. The training set was used to determine the values of the model parameters β, γ, and λ, while the test set was used to verify the effectiveness of the TP-CoGlo-TextRank algorithm in keyword extraction.

Table 1 shows the number of documents with three to five keywords in the experimental dataset, with an average of 4.2044 keywords. Each keyword in the experimental dataset was composed of at least one word and at most nine different words, with an average of 2.1071 different words. Table 2 shows the number of keywords composed of one to nine different words. It can be seen that most keywords are composed of one to four words. Table 3 shows the minimum and maximum total number of different words in all keywords of a document in the experimental dataset, as well as the minimum and maximum average number of different words in each keyword. Therefore, in the following experiments, according to the average number of keywords in each document (i.e., 4.2044) and the number of different words in each keyword (at least one word, with an average of 2.1071 words), this paper will extract four to nine words with the highest scores from the abstract of each document, and compare them with the keywords provided by the document itself, to verify the validity of the features such as word frequency, position, co-occurrence frequency, and similarity (see Section 4.3.1); to determine the values of the model parameters (see Section 4.3.2); and to verify the effectiveness of the proposed algorithm (see Section 4.3.3).

Table 1. The number of documents with three to five keywords in the experimental dataset.

Number of Keywords	Number of Documents
3	2153
4	3650
5	4197

Table 2. The number of keywords composed of one to nine words in the experimental dataset.

Number of Words	Number of Keywords
1	9303
2	22,489
3	7725
4	1882
5	467
6	132
7	33
8	9
9	4

Table 3. The number of different words in all keywords or each keyword in a document.

Minimum Total Number of Different Words in All Keywords in a Document	Maximum Total Number of Different Words in All Keywords in a Document	Minimum Average Number of Different Words in Each Keyword in a Document	Maximum Average Number of Different Words in Each Keyword in a Document
3	22	1	5.5

This paper uses Stanford CoreNLP [36], a natural language processing toolkit from Stanford University, to segment the abstract of each document. The segmentation results are filtered by stop words, single words, and special symbols such as @ and $, as well as part-of-speech tagging. Only verbs, nouns, and adjectives are reserved as candidate keywords. After preprocessing, the experimental dataset had an average of 82.1598 words per document.

4.2. Measurement for Evaluation

Precision (P), recall (R), and the F_1-measure (F_1) are employed to evaluate the performance of the proposed TP-CoGlo-TextRank algorithm. Precision quantifies the proportion of the number of keywords correctly extracted by the algorithm to the total number of keywords extracted by the algorithm, as shown in Equation (12). Recall measures the proportion of the number of keywords correctly extracted by the algorithm to the number of keywords provided by the document, as shown in Equation (13). The meanings of TP, FP, and FN are shown in Table 4. The values of precision and recall are between 0 and 1, and the closer they are to 1, the better the algorithm performance is. F_1 is calculated as the harmonic mean of precision and recall, as shown in Equation (14).

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$F_1 = \frac{2 \times P \times R}{P + R} \tag{14}$$

Table 4. Contingency table of keywords extracted by the algorithm and keywords provided by the document.

	Keywords Provided by the Document	Not Keywords Provided by the Document
Keywords Extracted by the Algorithm	*TP*	*FP*
Not Keywords Extracted by the Algorithm	*FN*	*TN*

When counting TP, FP, and FN, the traditional method is to perfectly match the keywords extracted by the algorithm with the keywords provided by the document. Considering that the keywords provided by the document are not necessarily the words in the abstract, which may be synonyms or near-synonyms, to better measure the performance of the keyword extraction algorithm, this paper uses the GloVe model to judge whether a keyword is correctly extracted by setting a similarity threshold. That is, as long as the similarity between the extracted keyword and the document keyword exceeds this threshold, it is considered that the algorithm has correctly extracted the keyword.

For a dataset of n documents, this paper first calculates the precision and recall of the keyword extraction algorithm for each document and then obtains the AP (Average Precision, see Equation (15)) and AR (Average Recall, see Equation (16)) by averaging the precision and recall of n documents, respectively. Based on AP and AR, AF_1 (Average F_1) is

expressed by Equation (17). In this paper, AP, AR, and AF_1 are employed to evaluate the performance of the keyword extraction algorithm for a dataset of n documents.

$$AP = \frac{1}{n}\sum_{i=1}^{n} P_i \tag{15}$$

$$AR = \frac{1}{n}\sum_{i=1}^{n} R_i \tag{16}$$

$$AF_1 = \frac{2 \times AP \times AR}{AP + AR} \tag{17}$$

4.3. Results and Analysis

4.3.1. Validity Verification of Word Frequency, Position, Co-Occurrence Frequency, and Similarity

In the process of extracting keywords based on the TextRank algorithm, this paper introduces the word frequency and position in the document to distinguish the importance of words and assigns the score of the word non-uniformly to its co-occurring words based on their co-occurrence frequency and similarity. This section verifies the validity of the above four features based on the AF_1 metric.

(1) Validity verification of word frequency

TF-TextRank: Based on the TextRank algorithm, word frequency is introduced to weight the words in the word graph, as shown in Equation (18), where $tf(V_i)$ is shown in Equation (2).

$$S(V_i) = tf(V_i) \times \left[\frac{1-d}{N} + d \times \sum_{V_j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)\right] \tag{18}$$

The AF_1 comparison of TextRank and TF-TextRank in extracting keywords from the test document set is shown in Figure 2. TF-TextRank is 0.47% higher than TextRank on average in terms of AF_1, indicating that word frequency has a certain influence on measuring word importance.

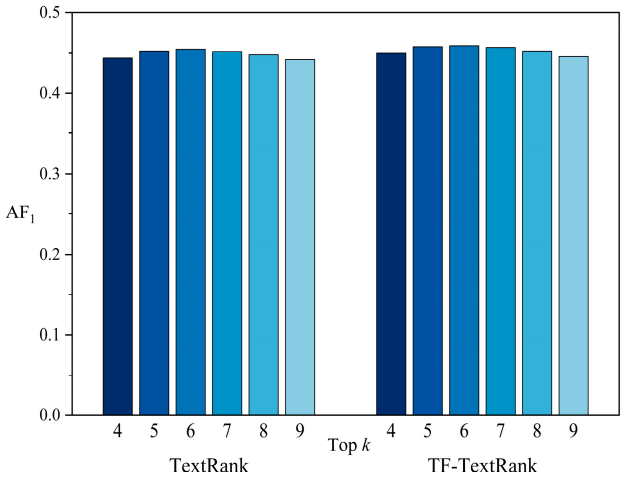

Figure 2. AF_1 comparison of TextRank and TF-TextRank.

As for why TF-TextRank is not significantly better than TextRank, we believe that it is mainly because words with higher frequency in a document are more likely to co-occur with other words, and there are more edges connected to them in the word graph. That is, the influence of word frequency on word importance has been reflected in the TextRank

algorithm, so when word frequency is used again to measure the importance of words, the performance is not particularly outstanding.

(2) Validity verification of word position

Pos-TextRank: Based on the TextRank algorithm, word position is introduced to weight the words in the word graph, as shown in Equation (19), where $pos(V_i)$ is shown in Equation (3) and β is set to 0.4, which is determined in Section 4.3.2.

$$S(V_i) = pos(V_i) \times \left[\frac{1-d}{N} + d \times \sum_{V_j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j) \right] \tag{19}$$

The AF_1 comparison of TextRank and Pos-TextRank in extracting keywords from the test document set is shown in Figure 3. Pos-TextRank is 8.30% higher than TextRank on average in terms of AF_1, indicating that word position has a great influence on measuring word importance and can significantly improve the performance of TextRank in extracting keywords.

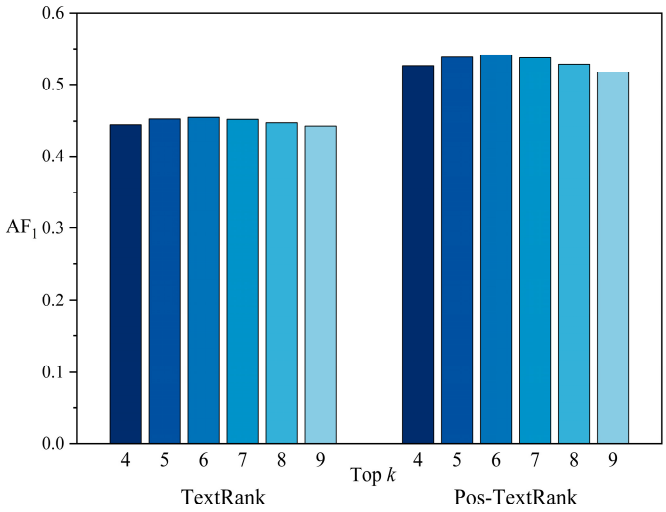

Figure 3. AF_1 comparison of TextRank and Pos-TextRank.

(3) Validity verification of word co-occurrence frequency

Co-occur-TextRank: Based on the TextRank algorithm, the co-occurrence frequency between words is used as the probability of transitioning from one word to another, as shown in Equation (20), where $Co(V_j, V_i)$ is shown in Equation (4).

$$S(V_i) = \frac{(1-d)}{N} + d \times \sum_{V_j \in In(V_i)} Co(V_j, V_i) S(V_j) \tag{20}$$

The AF_1 comparison of TextRank and Co-occur-TextRank in extracting keywords from the test document set is shown in Figure 4. Co-occur-TextRank is slightly better than TextRank, with an average increase of 0.64% in AF_1 value, indicating that assigning the importance score of the target word to its co-occurring words based on word co-occurrence frequency can highlight the importance of topic-related words to a certain extent.

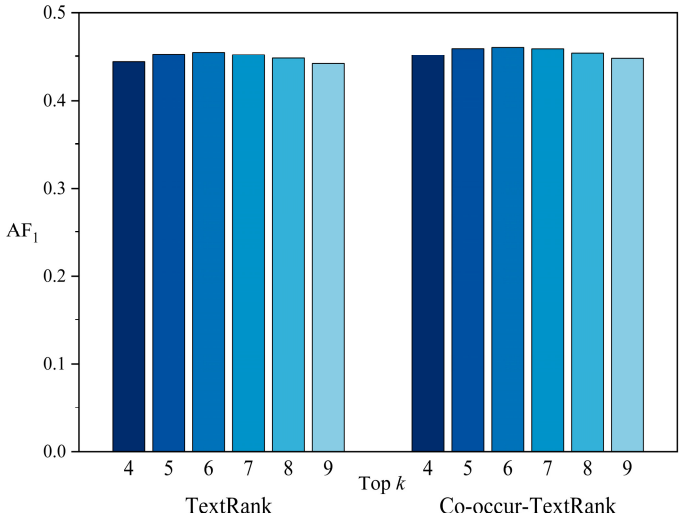

Figure 4. AF_1 comparison of TextRank and Co-occur-TextRank.

As for why Co-occur-TextRank is not significantly better than TextRank, we believe that it is mainly because, in this paper, keywords are extracted from the document abstract, which is usually short and concise, so the co-occurrences of words in the abstract cannot well reflect their co-occurrences in the full text of the document. When the sliding window size is set to 3, the mean, median, and mode of the number of co-occurrences between words in the 10,000 experimental documents are 1.1294, 1, and 1, respectively, which means the number of co-occurrences between most words is 1. This makes the transition probability from word V_j to V_i obtained by Equation (4) similar to that obtained according to the adjacency of V_j (i.e., $\frac{1}{|Out(V_j)|}$), meaning that Co-occur-TextRank is not significantly superior to TextRank.

(4) Validity verification of the similarity between words

CoGlo-TextRank: Based on the Co-occur-TextRank algorithm, GloVe-based similarity is used to transfer word importance score to the co-occurring words, as shown in Equation (21), where ω represents the proportion of the co-occurrence frequency of V_j and V_i in the transition probability from V_j to V_i, and $Co(V_j, V_i)$ and $Glo(V_j, V_i)$ are shown in Equation (4) and Equation (8), respectively.

$$S(V_i) = \frac{(1-d)}{N} + d \times \sum_{V_j \in In(V_i)} \left(\omega Co(V_j, V_i) + (1-\omega) Glo(V_j, V_i)\right) S(V_j) \quad (21)$$

To determine the value of ω, this paper tests the changes in AF_1 when using the CoGlo-TextRank algorithm to extract four to nine words with the highest scores from each document in the training set as keywords, with ω varying from 0 to 1 in steps of 0.1, as shown in Figure 5. It can be seen that AF_1 is optimal at ω = 0.8 when k = 5, 6, 7, 8, or 9 and the third-best at ω = 0.8 when k = 4. Therefore, when using the CoGlo-TextRank algorithm to extract keywords from documents, ω is set to 0.8.

The AF_1 comparison of Co-occur-TextRank and CoGlo-TextRank in extracting keywords from the test document set is shown in Figure 6. It can be seen that there is no significant difference between CoGlo-TextRank and Co-occur-TextRank in terms of the AF_1 value. CoGlo-TextRank is 0.03% higher than Co-occur-TextRank on average, indicating that the CoGlo-TextRank algorithm, with the introduction of the external knowledge of documents, has achieved a relatively small improvement in keyword extraction.

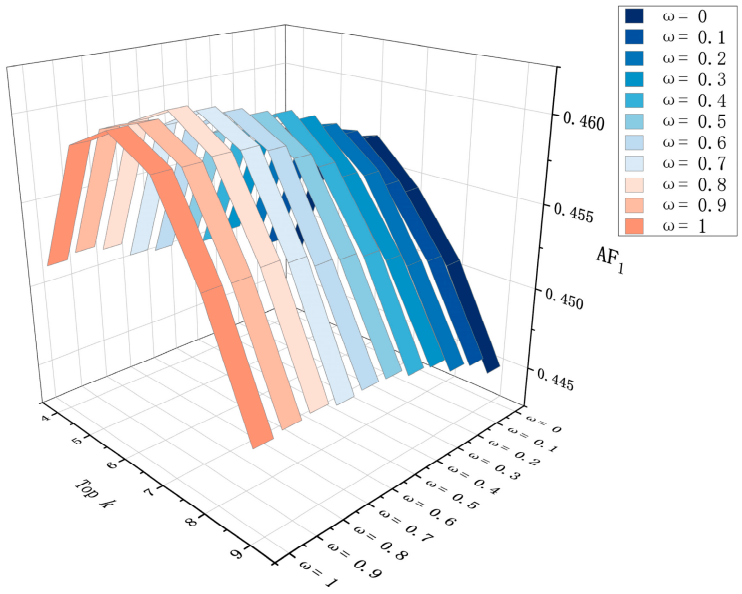

Figure 5. Changes in AF_1 with ω.

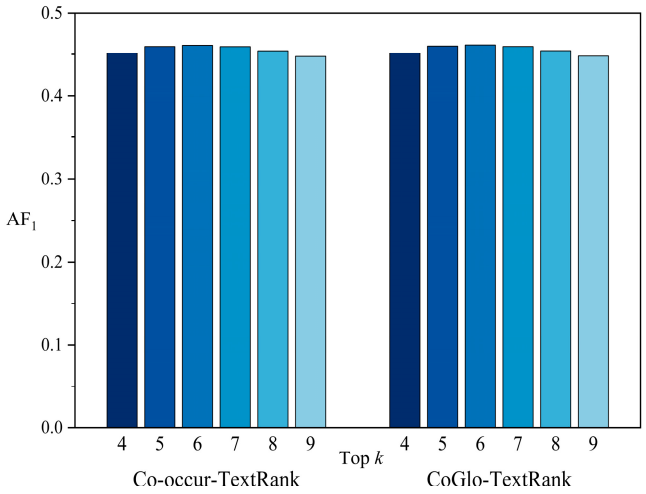

Figure 6. AF_1 comparison of Co-occur-TextRank and CoGlo-TextRank.

As for why CoGlo-TextRank is not significantly better than Co-occur-TextRank, we believe that it is mainly because when assigning the importance score of a word to its co-occurring words according to the similarity in the iterative process, to ensure the similarity is not less than 0, this paper standardizes it by the min-max normalization method, which will lead to fewer similarity differences between words. For example, V_1 has two co-occurring words, V_2 and V_3, in the sliding window, and the similarities between V_1 and V_2, and V_1 and V_3 according to Equation (5) are $sim(V_1, V_2)' = 0.05$ and $sim(V_1, V_3)' = 0.5$. The two similarities are normalized by Equation (7) as $sim(V_1, V_2) = 0.525$ and $sim(V_1, V_3) = 0.75$. It can be seen that the similarity ratio of V_1 and V_2 increases from 9.09% (0.05/(0.05 + 0.5) = 9.09%) to 41.18% (0.525/(0.525 + 0.75) = 41.18%), making the similarity difference between V_1/V_2 and V_1/V_3 decrease, which leads to no significant improvement in the CoGlo-TextRank algorithm in keyword extraction performance with the introduction of word similarity.

4.3.2. Determination of Model Parameters

Since AF_1 takes AP and AR into account, we mainly use the AF_1 metric to determine the values of the parameters β, γ and λ in this paper.

(1) Determination of the parameter β of word position importance

According to Equation (3), the position importance of the abstract words appearing in the title is 1, and that of other abstract words is β ($0 < \beta \leq 1$). To determine the value of β, this paper tests the changes in AF_1 when using the Pos-TextRank algorithm (as shown in Equation (19)) to extract four to nine words with the highest scores from each training document as keywords and combining the adjacent keywords in the original text into phrases, with β varying from 0 to 1 in steps of 0.1, as shown in Figure 7.

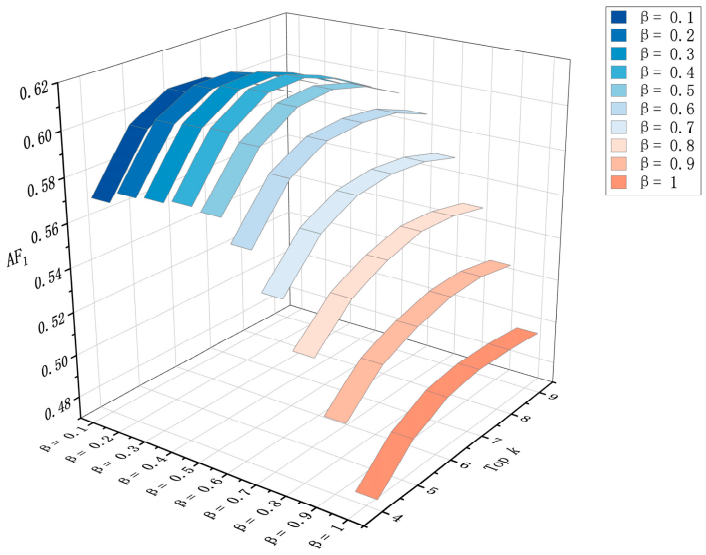

Figure 7. Changes in AF_1 with β.

The k in Figure 7 represents the number of words extracted from each training document. As can be seen from Figure 7, the variation trend of AF_1 with β is basically the same when extracting the top four to nine words with the highest scores as keywords—that is, with the increase in β, AF_1 slightly increases, reaches the maximum value, and then decreases. When k = 4, 5, 6, 7, 8, or 9, AF_1 is optimal at β = 0.4. Therefore, when using the TP-CoGlo-TextRank algorithm to extract keywords from documents, the position importance parameter β is set to 0.4.

When β = 1, the importance of the title word and the abstract word is the same—that is, the position importance of the word is not taken into account. In this case, the Pos-TextRank algorithm is the TextRank algorithm. It can be seen from Figure 7 that the Pos-TextRank algorithm at $0 < \beta < 1$ has better performance in extracting keywords from the training documents than at β = 1. As described in Section 4.3.1, this also shows that word position can help improve the performance of the TextRank algorithm in extracting keywords.

(2) Determination of the parameter γ in the transition probability and the parameter λ in the overall weighting

In Equation (10), λ ($0 \leq \lambda \leq 1$) represents the proportion of the position importance in the overall weighting of word V_i, and in Equation (11), γ ($0 \leq \gamma \leq 1$) represents the proportion of the co-occurrence frequency in the total transition probability from word V_j to word V_i. To determine the values of λ and γ, this paper tests the changes in AF_1 when using the TP-CoGlo-TextRank algorithm to extract four to nine words with the highest scores from each document in the training set as keywords and combining the adjacent keywords in the original text into phrases, with λ and γ respectively varying from 0 to 1 in steps of 0.1, as shown in Figure 8.

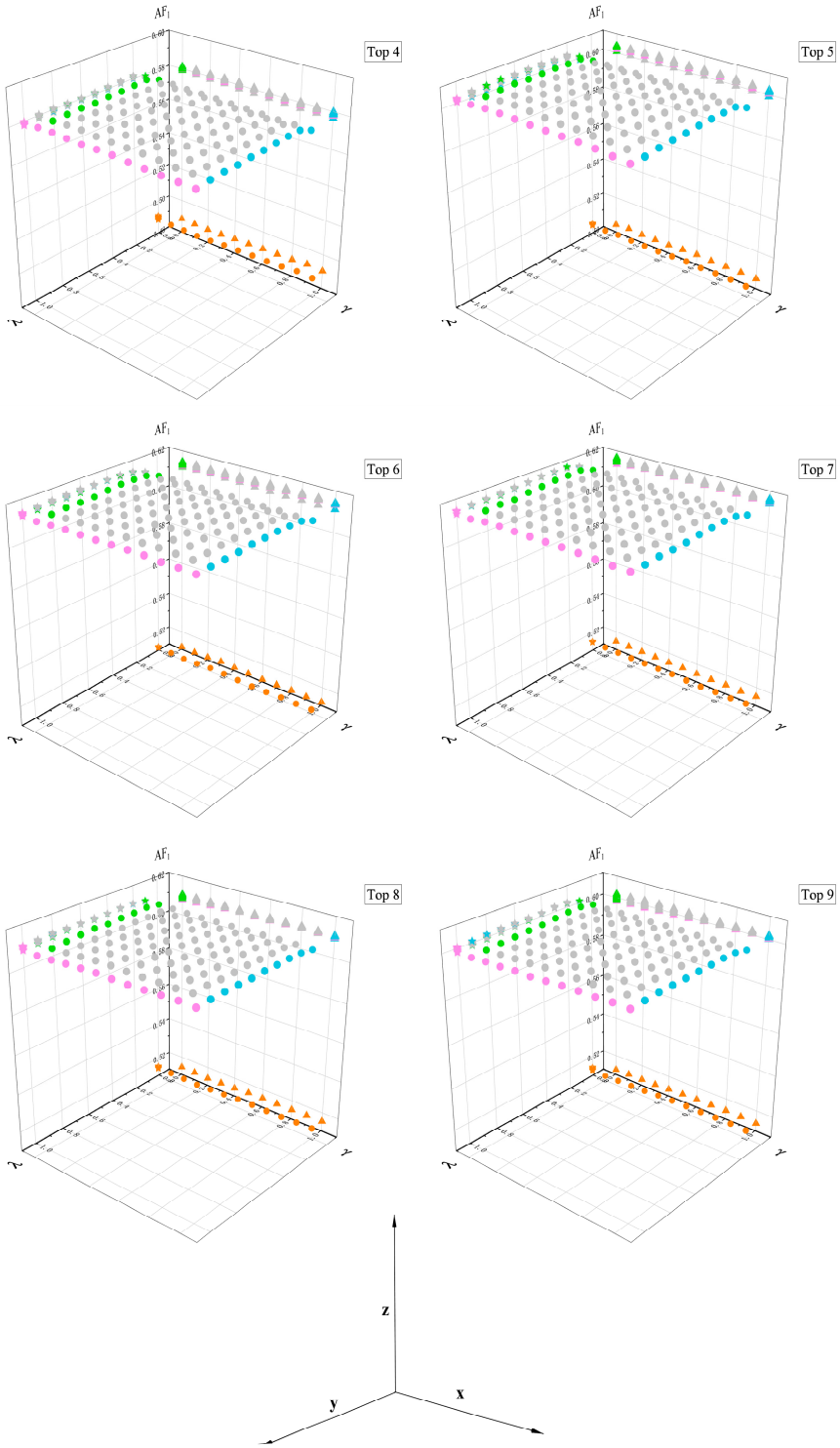

Figure 8. Changes in AF_1 with γ and λ.

The three-dimensional coordinate system is shown in the lower part of Figure 8, where the x-axis refers to γ, the y-axis refers to λ, and the z-axis refers to AF_1. In Figure 8, the dots represent the AF_1 values of the TP-CoGlo-TextRank algorithm as λ and γ change, the triangles on the zx-plane are the projections of the AF_1 values as λ changes, and the stars on the zy-plane are the projections of the AF_1 values as γ changes. The orange, pink, green, blue, and gray dots represent the AF_1 value at $\lambda = 0$, $\lambda = 1$, $\gamma = 0$, $\gamma = 1$, $0 < \lambda < 1$ and $0 < \gamma < 1$, respectively. Correspondingly, the orange, pink, green, blue, and gray triangles on the zx-plane represent the projections of the AF_1 values at $\lambda = 0$, $\lambda = 1$, $0 < \lambda < 1$ and

$\gamma = 0$, $0 < \lambda < 1$ and $\gamma = 1$, $0 < \lambda < 1$ and $0 < \gamma < 1$, respectively; the orange, pink, green, blue, and gray stars on the zy-plane represent the projections of the AF_1 values at $0 \leq \gamma \leq 1$ and $\lambda = 0$, $0 \leq \gamma \leq 1$ and $\lambda = 1$, $\gamma = 0$ and $0 < \lambda < 1$, $\gamma = 1$ and $0 < \lambda < 1$, $0 < \gamma < 1$ and $0 < \lambda < 1$,respectively. It can be seen that when $\lambda = 0$ (i.e., only word frequency is considered regardless of word position), the AF_1 values (orange dots) are projected at the lowest level (orange triangles), far below the projections (grey or pink triangles) at which the position importance is considered. When $\lambda = 1$ (i.e., only position importance is considered regardless of word frequency), the projections (pink triangles) of the AF_1 values (pink dots) are mostly at the lower-middle level compared with other projections (the grey triangles) at $0 < \lambda < 1$. This illustrates that both word position and word frequency can help distinguish the importance of words, as described in Section 4.3.1. It can also be seen that when $\gamma = 0$ (i.e., only similarity-based transition probability is considered), the AF_1 values (green dots) are mostly projected at the middle or upper-middle level (green stars) compared with other projections (grey stars) at $0 < \gamma < 1$. When $\gamma = 1$ (i.e., only co-occurrence frequency-based transition probability is considered), the projections (blue stars) of the AF_1 values (blue dots) are mostly at the middle level. This illustrates that the word importance score can be better assigned to its co-occurring words according to their co-occurrence frequency and similarity ratio, as described in Section 4.3.1. When $\lambda = 0.3$ and $\gamma = 0.2$, the TP-CoGlo-TextRank algorithm with adjacent keywords combined into phrases has the best average AF_1 value in extracting four to nine words from the training documents. Therefore, considering the influence of γ and λ on the performance and stability of the TP-CoGlo-TextRank algorithm, this paper sets $\lambda = 0.3$ and $\gamma = 0.2$.

4.3.3. Comparative Experiment

To verify the effectiveness of the proposed TP-CoGlo-TextRank algorithm in keyword extraction, six different algorithms are compared in the experiment.

(1) M1: the TF-IDF method. The TF score is calculated by Equation (2) and the smoothed IDF is calculated by Equation (22), where N represents the size of the document set and $df(w)$ represents the document frequency of word w.

$$idf(w) = \log \frac{N+1}{df(w)+1} + 1 \tag{22}$$

(2) M2: the LDA topic model-based method. In addition to the 10,000 documents in the experimental dataset, another 38,365 documents are selected from the DBLP-Citation-network V14 dataset, and a total of 48,365 documents are used to train the LDA model. The trained LDA model is then used to compute the influence of words in each test document, and words with the highest word influence are extracted as keywords.
(3) M3: the TextRank algorithm [6].
(4) M4: the modified TextRank algorithm proposed in [32], which uses word frequency, position, and word co-occurrence relationship to compute the transition probability matrix. The optimal weights of the three parts of transition probabilities are 0, 0.9, and 0.1, respectively.
(5) M5: the modified TextRank algorithm proposed in [37], which uses word similarities and co-occurrence relationship, both of which have a weight of 0.5, to compute the transition probability matrix, and takes the sum of similarities between the word and all its co-occurring words as the initial value of the word. The same word embedding model and word similarity calculation method as M6 are used here.
(6) M6: the TP-CoGlo-TextRank algorithm proposed in this paper. The glove.42B.300d [31] word embedding model, trained on 42 billion tokens from Common Crawl (http://commoncrawl.org, accessed on 1 March 2024), is used to obtain the word vectors.

The above six algorithms are used to extract four to nine words from each document in the test set as keywords, and AP, AR, and AF_1 are employed to evaluate their performance.

Figure 9 shows the comparison of the six algorithms when the extracted keywords that are adjacent words in the original text are not combined into phrases.

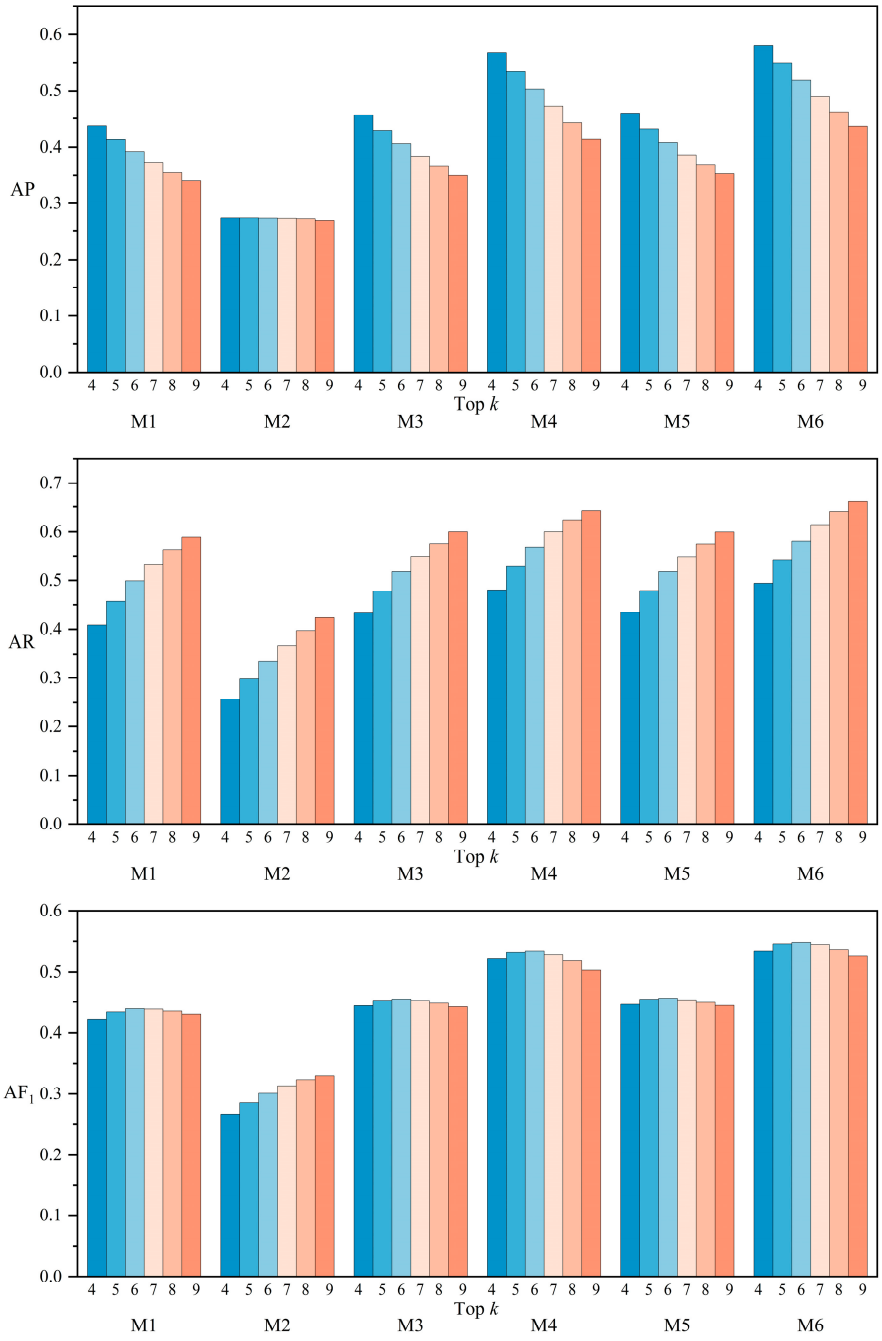

Figure 9. Comparison of the six algorithms with adjacent keywords uncombined.

The following points can be determined from Figure 9.

(1) The M2 algorithm has the worst performance in keyword extraction. We believe this is mainly because the M2 algorithm is closely related to the training effect of the LDA model. Although the 48,365 documents used to train the LDA model are all computer-related, their topic distribution is relatively scattered. As shown in Figure 10, the optimal number of topics is 15, and the highest topic coherence score is 0.5250, indicating that the semantic coherence of the words in the topics is weak, resulting in the keywords extracted by the M2 algorithm not being able to express the document topics well.

(2) As the number of extracted keywords k increases, the AP of the six algorithms gradually decreases, indicating that with the increase in k, the number of correctly extracted keywords increases slightly. AR gradually increases, mainly because the number of keywords provided by the document itself is fixed, but as k increases, the number of correctly extracted keywords increases, although by a smaller amount. With the increase in k, except for the M2 algorithm, the AF_1 of the other five algorithms increases first and then decreases gradually after reaching a peak. As for the M2 algorithm, its AF_1 reaches a peak at k = 16 and then shows a downward trend. The reason for the late peak of the M2 algorithm is that the topic coherence of the trained LDA model is not high, which leads to the fact that when a small number of words are extracted as keywords, the extracted words cannot express the document topics well. However, as k increases, more high-quality words are extracted, making the M2 algorithm gradually reach its peak.

(3) Graph-based algorithms M3, M4, M5, and M6 are better than the statistical feature-based algorithm M1 in extracting keywords. The average AP values increased by 1.40%, 10.42%, 1.65%, and 12.14%, respectively. The average AR values increased by 1.73%, 6.60%, 1.74%, and 8.09%, respectively. The average AF_1 values increased by 1.57%, 8.95%, 1.74%, and 10.61%, respectively. This shows that graph-based keyword extraction algorithms can better measure the importance of words by using the relationship between words.

(4) Compared with the other five baseline algorithms, the M6 algorithm has a significant improvement in AP, AR, and AF_1. The average AP values increased by 12.14%, 23.34%, 10.74%, 1.72%, and 10.49%, respectively. The average AR values increased by 8.09%, 24.31%, 6.36%, 1.48%, and 6.34%, respectively. The average AF_1 values increased by 10.61%, 23.65%, 9.04%, 1.66%, and 8.87%, respectively. This shows that the M6 algorithm has good performance in keyword extraction when adjacent keywords in the original text are not combined into phrases.

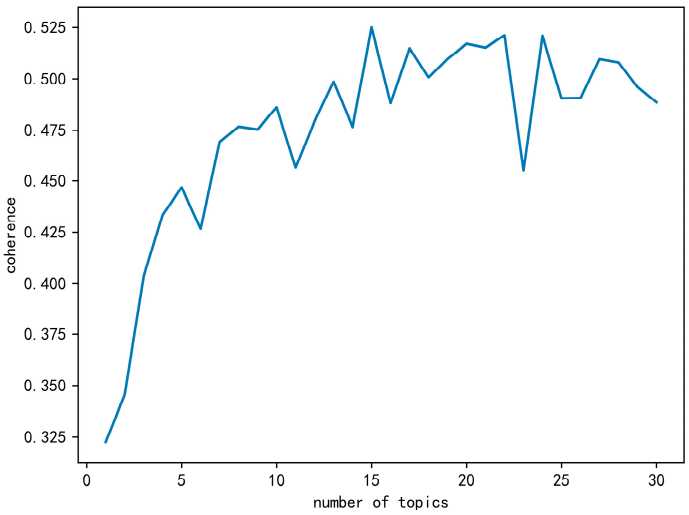

Figure 10. Variation of coherence with the number of topics.

Figure 11 shows the comparison of the six algorithms when the extracted keywords that are adjacent in the original text are combined into phrases. It can be seen that the M6 algorithm is still superior to the other five algorithms in terms of AP, AR, and AF_1.

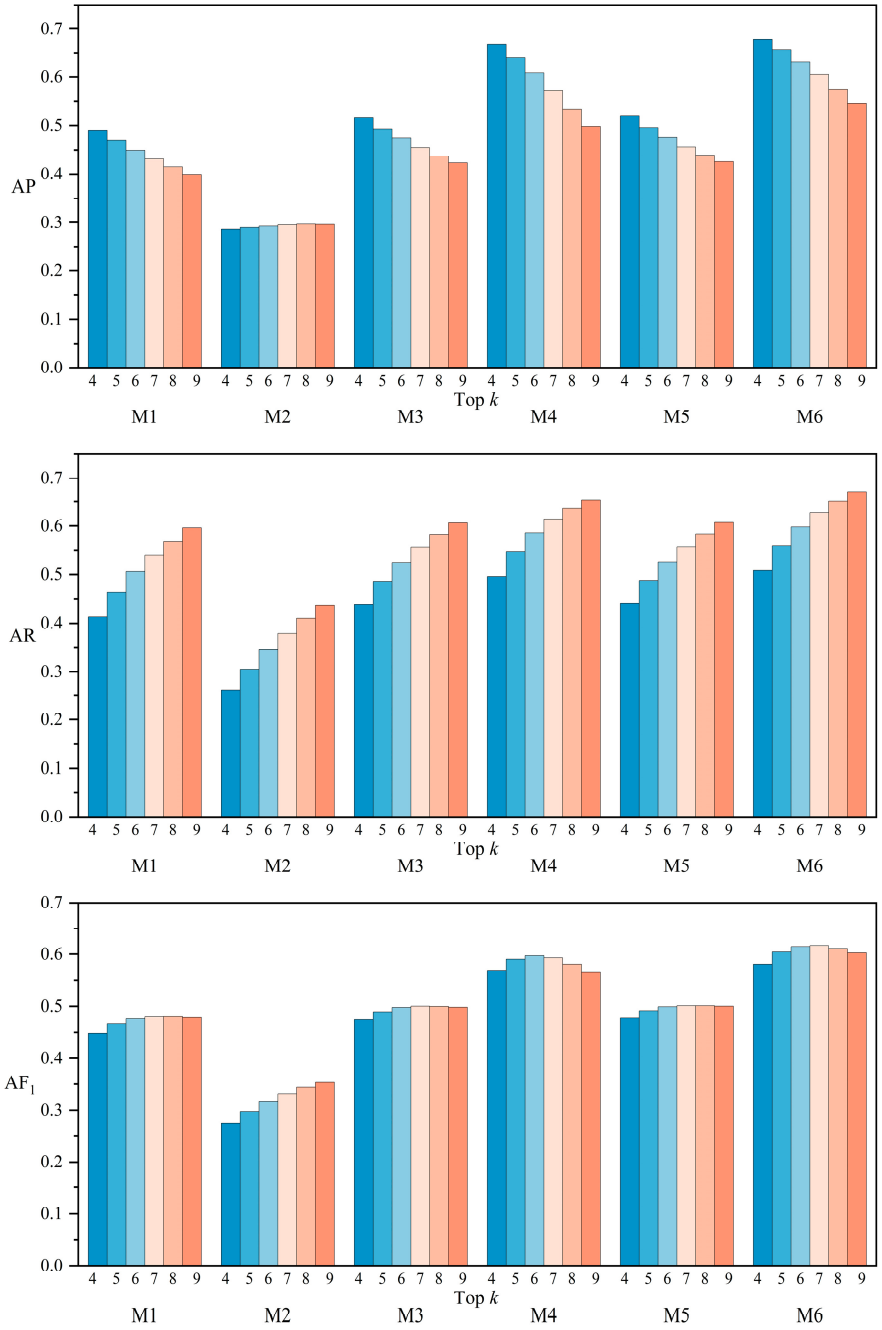

Figure 11. Comparison of the six algorithms with adjacent keywords combined.

Figure 12 shows the AF_1 comparison of the six algorithms when adjacent keywords in the original text are not combined or combined into a phrase as one keyword. Adding "Uncombined" and "Combined" after the algorithm name refers to whether adjacent keywords in the original text are combined or not. It can be seen that the AF_1 value of each algorithm with adjacent keywords combined is higher than that without combining adjacent keywords. Specifically, the M1 (Combined) algorithm has an average improvement of 3.83% compared with the M1 (Uncombined) algorithm; the M2 (Combined) algorithm has an average improvement of 1.67% compared with the M2 (Uncombined) algorithm; the M3 (Combined) algorithm has an average improvement of 4.46% compared with the M3 (Uncombined) algorithm; the M4 (Combined) algorithm has an average improvement of 6.00% compared with the M4 (Uncombined) algorithm; the M5 (Combined) algorithm has an average improvement of 4.46% compared with the M5 (Uncombined) algorithm; and the

M6 (Combined) algorithm has an average improvement of 6.54% compared with the M6 (Uncombined) algorithm. This is mainly because for each test document, after combining adjacent keywords into a phrase as one keyword, the number of keywords extracted by the algorithm decreases greatly, while the number of correctly extracted keywords changes little. Therefore, the precision of keyword extraction of each algorithm for most test documents is significantly improved, and so is the AP. Since the number of keywords provided by the document itself is unchanged, each algorithm has a relatively small increase in recall for keyword extraction from most test documents, and correspondingly, there is a small increase in AR. As AF_1 is a combination of AP and AR, the AF_1 of each algorithm increases significantly after combining adjacent keywords into a phrase as one keyword. It can also be seen from Figure 12 that the AF_1 value of the M6 (Combined) algorithm reaches its peak at $k = 7$ (i.e., extracting seven words with the highest scores from each test document).

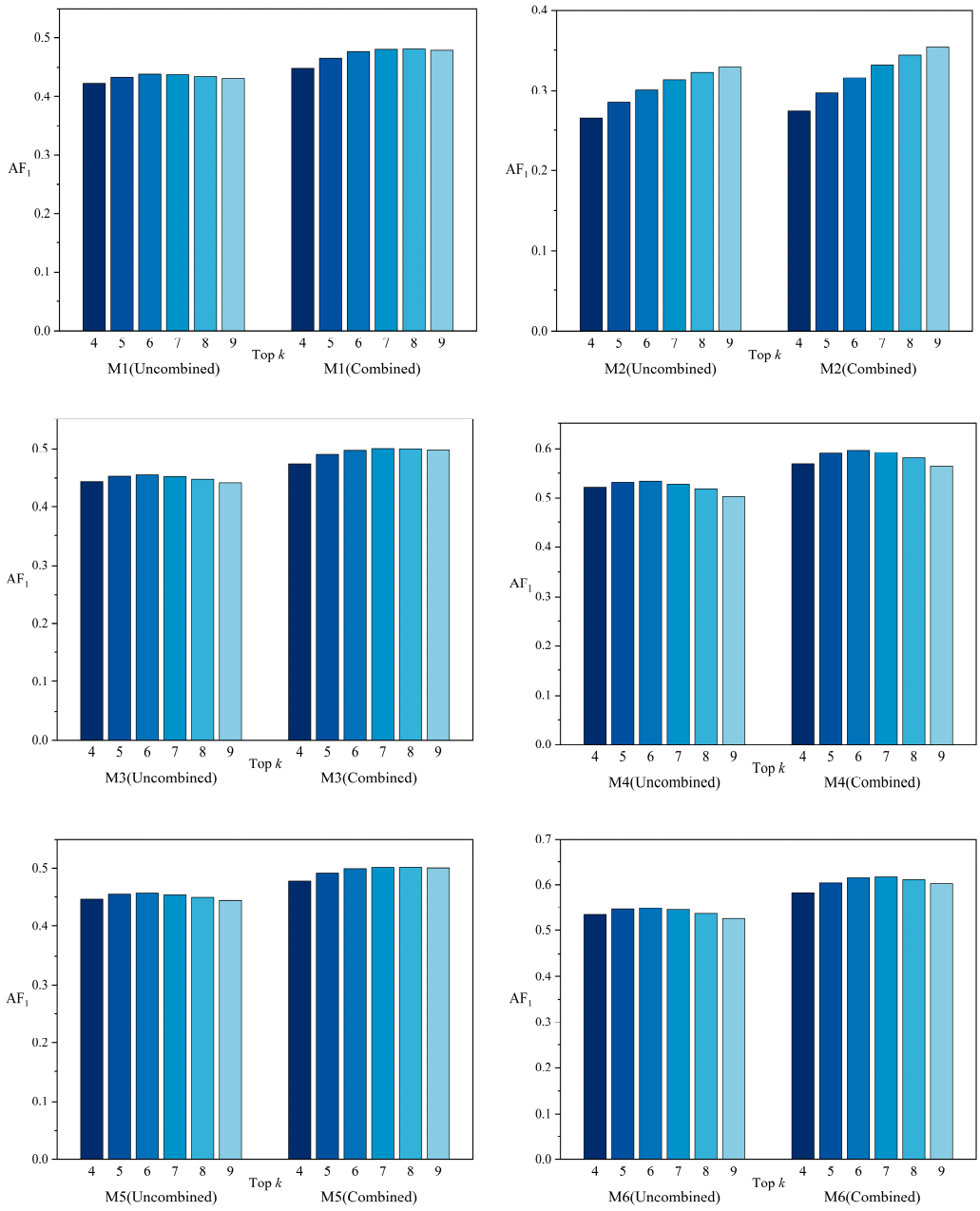

Figure 12. AF_1 comparison of the six algorithms with adjacent keywords uncombined and combined.

Table 5 compares the effects of the six algorithms when adjacent keywords are not combined or combined into a phrase as one keyword at $k = 7$. It can be seen that when extracting seven words with the highest scores from each test document, the M6 (TP-CoGlo-TextRank) algorithm proposed in this paper has the best performance compared with the five baseline algorithms (M1 to M5), regardless of whether adjacent keywords in the original text are combined. When adjacent keywords are not combined into one keyword, the AF_1 value of the M6 algorithm is 10.65%, 23.16%, 9.25%, 1.62%, and 9.13% higher than that of the M1 to M5 algorithms, respectively. When adjacent keywords are combined into one keyword, the AF_1 value of the M6 algorithm is 13.65%, 28.45%, 11.58%, 2.34%, and 11.50% higher than that of the M1 to M5 algorithms, respectively. Therefore, based on the TextRank algorithm, combining word frequency and position to measure the importance of words, and using word co-occurrence frequency and the GloVe model to achieve the non-uniform transfer of word scores, can obtain keywords that are more consistent with the document topic.

Table 5. Comparison of the six algorithms while extracting top seven most important words.

Algorithms	Top 7 (Uncombined)			Top 7 (Combined)		
	***AP*%**	***AR*%**	***AF*$_1$%**	***AP*%**	***AR*%**	***AF*$_1$%**
M1	37.21	53.32	43.83	43.16	54.03	47.99
M2	27.32	36.69	31.32	29.52	37.90	33.19
M3	38.44	54.93	45.23	45.50	55.65	50.06
M4	47.23	60.01	52.86	57.29	61.45	59.30
M5	38.66	54.86	45.35	45.60	55.67	50.14
M6	48.95	61.42	54.48	60.54	62.79	61.64

Therefore, when constructing the academic literature knowledge graph, this paper will use the TP-CoGlo-TextRank algorithm to extract keywords from documents without keywords, select the top seven words with the highest scores as keywords, and combine the keywords that are adjacent words in the original text into phrases. The combined final results are used as the keywords of the document.

5. Academic Literature Knowledge Graph Construction

In this paper, the DBLP-Citation-network V14 [35] dataset, which contains 5,259,858 documents and 36,630,661 citation relationships, is used to construct the academic literature knowledge graph. Each document in the DBLP-Citation-network V14 dataset is associated with a title, authors, a venue, a doc_type, an abstract, keywords, and references. In the process of constructing the academic literature knowledge graph based on the relationships between documents shown in Figure 1, this paper uses the TP-CoGlo-TextRank algorithm proposed in Section 3.5 to extract keywords for the documents that do not have keywords for some reason. On this basis, the academic literature knowledge graph is constructed based on the Neo4j graph database to provide data support for subsequent studies, such as automatic summarization, text classification, text clustering, and question-answering systems.

Part of the academic literature knowledge graph constructed by Neo4j is shown in Figure 13. Each type of entity node in this knowledge graph corresponds to a color, where the blue node represents the Paper node, the green node represents the Author node, the orange node represents the Venue node, and the yellow node represents the Keyword node. Figure 14 is an example of nodes and relationships in the academic literature knowledge graph, in which the coauthor relationship indicates co-authorship, written_by indicates that a paper is written by an author, published_at indicates that a paper is published in a certain journal (or conference), contributed_by indicates that a paper, an author, or a journal (or conference) contributes to a certain keyword, *relevant* indicates that a paper is related to a keyword, and citing indicates that a paper cites another paper [4].

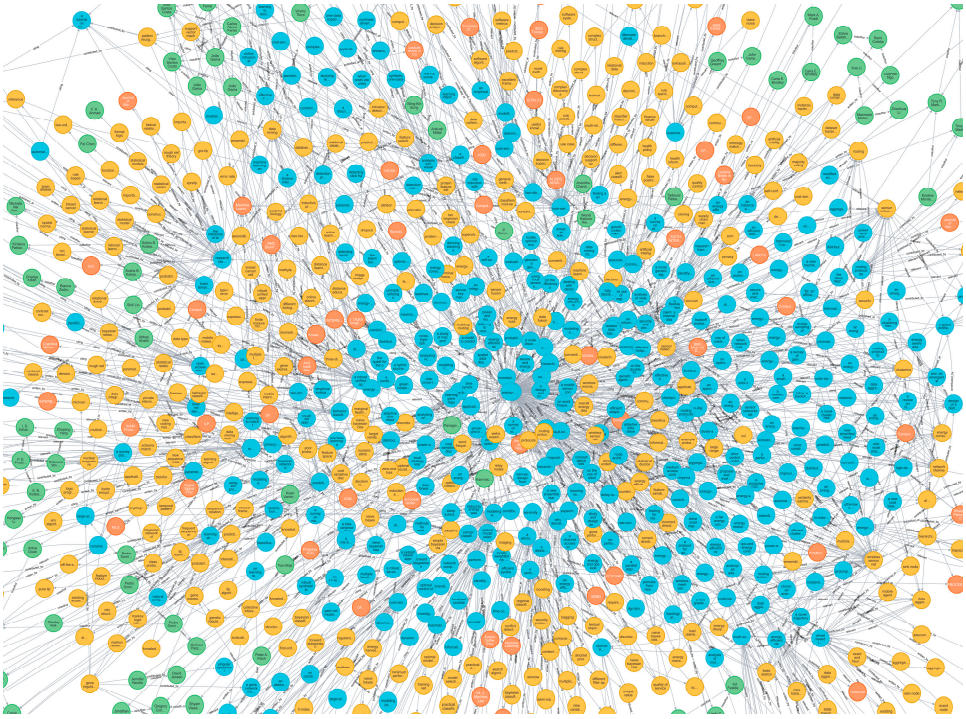

Figure 13. Part of the academic literature knowledge graph.

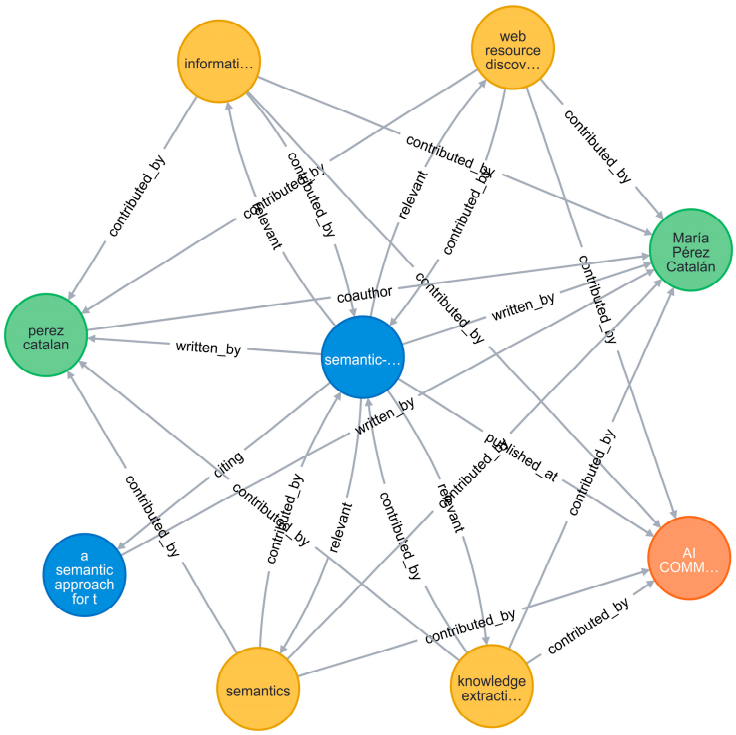

Figure 14. An example of nodes and relationships in the academic literature knowledge graph.

6. Conclusions

Keywords are a very important type of node in academic literature knowledge graphs. In the process of constructing the academic literature knowledge graph, this paper improved the classic TextRank keyword extraction algorithm by using text features such as word frequency, position, word co-occurrence frequency, and the GloVe model and proposed the TP-CoGlo-TextRank algorithm to extract keywords from the documents without keywords. On this basis, entity and relationship extraction was carried out based on the

DBLP-Citation-network V14 dataset, and the graph database Neo4j was used to store and display the academic literature knowledge graph.

A shortcoming of this paper is that keywords are mostly nouns or nominal phrases, but in this paper, nouns, verbs, and adjectives are taken as candidate keywords, so it is inevitable that verbal and adjectival keywords will be selected, which affects the quality of the keywords extracted. Although keywords that are adjacent words in the original text are combined in this paper, sometimes the combined result is not a nominal phrase. In the following study, we will focus on solving this problem. In addition, based on the constructed academic literature knowledge graph, we will also carry out studies on text clustering, automatic text summarization, and the academic influence analysis of papers, journals (or conferences), and authors in the future.

Author Contributions: Conceptualization, L.Z.; methodology, L.Z.; software, L.Z.; validation, L.Z.; formal analysis, L.Z. and Y.L.; data curation, Y.L. and Q.L.; writing—original draft preparation, Y.L. and Q.L.; writing—review and editing, L.Z.; visualization, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Social Science Planning Fund Project of Liaoning Province (No. L18CTQ004), China Postdoctoral Science Foundation (No. 2015M571292), and the Fundamental Research Funds for the Central Universities (No. 3132023297).

Data Availability Statement: The data used to support the results of this study are publicly available and referenced in the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Singhal, A. Introducing the Knowledge Graph: Things, Not Strings. 2012. Available online: https://www.blog.google/products/search/introducing-knowledge-graph-things-not/ (accessed on 14 February 2024).
2. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge graph embedding by translating on hyperplanes. In Proceedings of the 28th AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 1112–1119.
3. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Yu, P.S. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef] [PubMed]
4. Liu, X.; Yu, Y.; Guo, C.; Sun, Y. Meta-path-based ranking with pseudo relevance feedback on heterogeneous graph for citation recommendation. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, Shanghai, China, 3–7 November 2014; pp. 121–130.
5. Jiang, Z.; Yin, Y.; Gao, L.; Lu, Y.; Liu, X. Cross-language citation recommendation via hierarchical representation learning on heterogeneous graph. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 635–644.
6. Mihalcea, R.; Tarau, P. TextRank: Bringing order into texts. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 404–411.
7. Nomoto, T. Keyword extraction: A modern perspective. *SN Comput. Sci.* **2022**, *4*, 92. [CrossRef] [PubMed]
8. Hasan, K.S.; Ng, V. Automatic keyphrase extraction: A survey of the state of the art. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; pp. 1262–1273.
9. Wang, Z.; Wang, D.; Li, Q. Keyword extraction from scientific research projects based on SRP-TF-IDF. *Chin. J. Electron.* **2021**, *30*, 652–657.
10. Rathi, R.N.; Mustafi, A. Designing an efficient unigram keyword detector for documents using relative entropy. *Multimed. Tools Appl.* **2022**, *81*, 37747–37761. [CrossRef]
11. Campos, R.; Mangaravite, V.; Pasquali, A.; Jorge, A.M.; Nunes, C.; Jatowt, A. A text feature based automatic keyword extraction method for single documents. In Proceedings of the 40th European Conference on Information Retrieval, Grenoble, France, 26–29 March 2018; pp. 684–691.
12. Campos, R.; Mangaravite, V.; Pasquali, A.; Jorge, A.; Nunes, C.; Jatowt, A. Yake! keyword extraction from single documents using multiple local features. *Inf. Sci.* **2020**, *509*, 257–289. [CrossRef]
13. Lu, X.; Zhou, X.; Wang, W.; Lio, P.; Hui, P. Domain-oriented topic discovery based on features extraction and topic clustering. *IEEE Access* **2022**, *8*, 93648–93662. [CrossRef]
14. Goz, F.; Mutlu, A. MGRank: A keyword extraction system based on multigraph GoW model and novel edge weighting procedure. *Knowl.-Based Syst.* **2022**, *251*, 109292. [CrossRef]
15. Jain, M.; Bhalla, G.; Jain, A.; Sharma, S. Automatic keyword extraction for localized tweets using fuzzy graph connectivity measures. *Multimed. Tools Appl.* **2022**, *81*, 42932–42956. [CrossRef]

16. Yan, Y.; Tan, Q.; Xie, Q.; Zeng, P.; Li, P. A graph-based approach of automatic keyphrase extraction. *Procedia Comput. Sci.* **2017**, *107*, 248–255.
17. Abimbola, R.O.; Awoyelu, I.O.; Hunsu, F.O.; Akinyemi, B.O.; Aderounmu, G.A. A noun-centric keyphrase extraction model: Graph-based approach. *JAIT* **2022**, *13*, 578–589. [CrossRef]
18. Liu, Z.; Huang, W.; Zheng, Y.; Sun, M. Automatic keyphrase extraction via topic decomposition. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, MA, USA, 9–11 October 2010; pp. 366–376.
19. Teneva, N.; Cheng, W. Salience Rank: Efficient keyphrase extraction with topic modeling. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 530–535.
20. Sarracén, G.L.D.I.P.; Rosso, P. Offensive keyword extraction based on the attention mechanism of BERT and the eigenvector centrality using a graph representation. *Pers. Ubiquitous Comput.* **2021**, *27*, 45–57. [CrossRef]
21. Duan, X.; Ying, S.; Chen, H.; Yuan, W.; Yin, X. OILog: An online incremental log keyword extraction approach based on MDP-LSTM neural network. *Inf. Syst.* **2021**, *95*, 101618. [CrossRef]
22. Zhang, Y.; Tuo, M.; Yin, Q.; Qi, L.; Wang, X.; Liu, T. Keywords extraction with deep neural network model. *Neurocomputing* **2020**, *383*, 113–121. [CrossRef]
23. Page, L.; Brin, S.; Motwani, R.; Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web*; Technical Report; Stanford Infolab: Stanford, CA, USA, 1998.
24. Liu, L.; Yu, H.; Fei, N.; Chen, C. Key-word extracting algorithm from single text based on TextRank. *Appl. Res. Comput.* **2018**, *35*, 705–710.
25. Gu, Y.; Xia, T. Study on keyword extraction with LDA and TextRank combination. *Data Anal. Knowl. Discov.* **2014**, *Z1*, 41–47.
26. Xia, T. Extracting keywords with modified TextRank model. *Data Anal. Knowl. Discov.* **2017**, *1*, 28–34.
27. Chen, Z.; Li, Y.; Xu, F.; Feng, G.; Shi, D.; Cui, X. Key information extraction of forestry text based on TextRank and clusters filtering. *Trans. Chin. Soc. Agric. Mach.* **2020**, *51*, 207–214+172.
28. Xiong, A.; Liu, D.; Tian, H.; Liu, Z.; Yu, P.; Kadoch, M. News keyword extraction algorithm based on semantic clustering and word graph model. *Tsinghua Sci. Technol.* **2021**, *26*, 886–893. [CrossRef]
29. Guo, W.; Wang, Z.; Han, F. Multifeature fusion keyword extraction algorithm based on TextRank. *IEEE Access* **2022**, *10*, 71805–71813. [CrossRef]
30. Qiu, D.; Zheng, Q. Improving TextRank algorithm for automatic keyword extraction with tolerance rough set. *Int. J. Fuzzy Syst.* **2022**, *24*, 1332–1342. [CrossRef]
31. Matsuo, Y.; Ishizuka, M. Keyword extraction from a single document using word co-occurrence statistical information. *Int. J. Artif. Intell. Tools* **2004**, *13*, 157–169. [CrossRef]
32. Xia, T. Study on keyword extraction using word position weighted TextRank. *Data Anal. Knowl. Discov.* **2013**, *29*, 30–34.
33. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
34. Zhang, M.; Li, X.; Yue, S.; Yang, L. An empirical study of TextRank for keyword extraction. *IEEE Access* **2020**, *8*, 178849–178858. [CrossRef]
35. Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; Su, Z. ArnetMiner: Extraction and mining of academic social networks. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 990–998.
36. Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; McClosky, D. The Stanford CoreNLP natural language processing toolkit. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MD, USA, 22–27 June 2014; pp. 55–60.
37. Ning, J.; Liu, J. Using Word2vec with TextRank to extract keywords. *Data Anal. Knowl. Discov.* **2016**, *32*, 20–27.