Knowledge-Based Systems 219 (2021) 106846

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

TripleRank: An unsupervised keyphrase extraction algorithm

Tuohang Li [a], Liang Hu [a], Hongtu Li [a], Chengyu Sun [a], Shuai Li [b], Ling Chi [a,*]

[a] College of Computer Science and Technology, Jilin University, Changchun, 130000, China
[b] College of Software Engineering Technology, Jilin University, Changchun, 130000, China

ARTICLE INFO

Article history:
Received 16 November 2020
Received in revised form 16 January 2021
Accepted 4 February 2021
Available online 19 February 2021

Keywords:
Keyphrase extraction
Keyphrase semantic diversity
Keyphrase coverage
Unsupervised approach

ABSTRACT

Automatic keyphrase extraction algorithms aim to identify words and phrases that contain the core information in documents. As online scholarly resources have become widespread in recent years, better keyphrase extraction techniques are required to improve search efficiency. We present two features, keyphrase semantic diversity and keyphrase coverage, to overcome limitations of existing methods for unsupervised keyphrase extraction. Keyphrase semantic diversity is the degree of semantic variety in the extraction result, which is introduced to avoid extracting synonym phrases that contain the same high-score candidate. Keyphrase coverage refers to candidates' representativeness of other words in documents. We propose an unsupervised keyphrase extraction method called TripleRank, which evaluates three features: word position (a sensitive feature for academic documents) and two innovative features mentioned above. The architecture of TripleRank includes three sub-models that score the three features and a summing model. Though involving multiple models, there is no typical iteration process in TripleRank; hence, the computational cost is relatively low. TripleRank has led the experiment results on four academic datasets compared to four state-of-the-art baseline models, which confirmed the influence of keyphrase semantic diversity and keyphrase coverage and proved the efficiency of our method.

1. Introduction

Cloud applications have facilitated access to academic materials for individuals in the recent decade. The top data provider, Google Scholar, listed tens of millions of papers in 2019. The surge of online scholarly information makes it challenging for academic users to find relevant academic documents. Hence, data providers investigate intelligent processing methods for academic papers. Keyphrases and streamline recapitulations of documents are considered efficient search tags because they provide a summary of documents or texts. Precise search tags can improve the efficiency of document retrieval tasks, especially for academic users. In addition, automatic keyphrase extraction saves time for users: they can search for the required documents without reading the content thoroughly. However, not all documents are annotated with proper keyphrases; thus, skimming and scanning is inefficient for documents that have no annotations. Therefore, information extraction, including keyphrase extraction, has become a topic of considerable research interest. Many outstanding keyphrase extraction algorithms have been developed in recent years. These algorithms are generally classified into supervised and unsupervised approaches.

For supervised approaches, the problem is regarded as a classification task [1]. An example of an early algorithm is the keyphrase extraction algorithm (KEA). In such approaches, the models are trained to distinguish whether a word is a keyphrase. A word is more likely to be classified as a keyphrase if it appears more frequently in the target document and relatively less frequently in general [2]. To train this classifier, different algorithms have been proposed [3], such as boosting [4], maximum entropy [5], and support vector machines (SVMs) [6]. Supervised keyphrase extraction methods are more accurate than unsupervised ones; nevertheless, they require enormous human labelling work and time to prepare large training datasets.

For unsupervised approaches, the tasks are regarded as ranking problems. Unsupervised approaches are commonly researched because of their low computational cost, although they are less precise than supervised approaches. The unsupervised approaches include graph-based ranking, topic-based, and simultaneous learning [3]; graph-based ranking is researched more than other methods. Inspired by PageRank [7], a web page ranking algorithm designed based on human interest using link structures of web pages, Mihalcea and Tarau [8] proposed an extraction algorithm called TextRank. The core idea of TextRank is to use words in documents (instead of web pages), build

* Corresponding author.
E-mail addresses: lith19@mails.jlu.edu.cn (T. Li), hul@jlu.edu.cn (L. Hu), lihongtu@jlu.edu.cn (H. Li), cysun20@mails.jlu.edu.cn (C. Sun), shuaili18@mails.jlu.edu.cn (S. Li), chiling@jlu.edu.cn (L. Chi).

https://doi.org/10.1016/j.knosys.2021.106846
0950-7051/

a network graph based on the adjacent relationship between words, and apply the PageRank algorithm to the network graph. However, TextRank neglected the influence of document topics and semantics, which reduced the overall precision. To solve this problem, Liu et al. built the topical PageRank (TPR) [9] using the latent Dirichlet allocation (LDA) model [10] that generates topic distributions to extract keyphrases more accurately. Liu et al. improved precision to a large extent. However, TPR is a complex model with numerous parameters, which makes it difficult to apply. In addition, Boudin et al. [11] proposed a model that encoded topical information with a multipartite graph: it represented documents as tightly connected sets of topic-related candidates, which further improved precision. Taking advantage of the specifics of scholarly documents, Florescu and Caragea [12] proposed PositionRank, an efficient keyphrase extraction algorithm for scholarly documents. PositionRank used the position information of words and achieved improvements of up to 29.09% compared with PageRank models. Hence, PositionRank provides satisfactory results and is computationally inexpensive because of the accessibility of position order. Moreover, simultaneous learning methods performed keyphrase extraction and text summarisation simultaneously, which is related to graph-based methods. Zha et al. [13] proposed a graph-based simultaneous learning method, and Wan et al. [14] extended this method by simultaneously using TextRank. Simultaneous learning methods and TextRank have a similar drawback: they do not consider topical information.

Numerous keyphrase extraction algorithms have obtained excellent results using different tools, including position information, graphs, and topic models. However, we identified the following problem, which frequently appeared in most models. Many models evaluate terms and phrases in scholarly documents based on a score determined by probability and weight, which is accurate in most cases. The extraction of phrases that contain more than one word is usually dominated by the weight and probability score of the words they contain, which is where potential inaccuracy lies. A single word with a high score or weight is more likely to be a part of several phrases. Thus, the model can predict multiple phrases involving the same high-score word as keyphrase results, ignoring their similar semantics.

For example, in one of our experiments on DUC datasets, as Fig. 1 shows, the red words in the context (document) are the author-input keyphrases, the red and bold words in the extraction results are correct keyphrases, and the extraction result by PositionRank is given. PositionRank [12] is an excellent method that substantially improved the extraction precision and extracted four out of six keyphrases correctly. However, we use this imperfection example to explain our idea. We found that the "PLO" appeared in four keyphrases out of six in the results of PositionRank, which is due to its high weight. Such phenomena could not only generate incorrect keyphrases but may also lead to the inaccuracy of output results (when many latent proper keyphrases that can highly represent the documents are left unpicked).

To address this problem, we developed two innovative features – keyphrase coverage and keyphrase semantic diversity – and proposed a model named TripleRank. The results of our method for the document from Fig. 1 is given in the "Supporting Evidence" section.

Keyphrase coverage refers to the range of coverage or representativeness of a single word over other words. Keyphrases are the terms and phrases that should represent entire documents and summarise their contents. Therefore, it can also be considered that keyphrases summarise or cover more semantic content than other words in these documents. Keyphrase semantic diversity is the span of the semantic distribution of extracted keyphrases. The extraction results that were centralised to fewer domains had lower keyphrase semantic diversity. We maximised keyphrase semantic diversity and keyphrase coverage to avoid the problem mentioned above.

In this paper, we proposed TripleRank, which uses three features: keyphrase coverage, keyphrase semantic diversity, and position information. We assessed three features by scores and unified them with the output of the final score. To calculate the scores of three features, consulting the merits and demerits of existing works, three sub-models were constructed to evaluate features. We deem that the keyphrases covering more details in documents are more likely to represent other words, which is similar to the concept of similarity between word vectors. Thus, the sub-model for keyphrase coverage is based on similarity calculation and processing. In addition, for the sub-model evaluating keyphrase semantic diversity, we predict the potential topic distributions of every candidate. Position information is regarded as another feature, mainly because of specific properties of scholarly documents. There is no typical iterating process in our method. However, it unites the scores of three features evaluated by the corresponding algorithms to obtain the final score of our method.

Our major contributions can be summarised as follows: 1. We addressed a common problem in which several extracted phrases contained the same high-score word. 2. We proposed and analysed the reason and significance of two concepts to solve the problem: keyphrase coverage and keyphrase semantic diversity. 3. We proposed TripleRank, a method based on keyphrase coverage, keyphrase semantic diversity, and position information that has preferable precision compared with state-of-the-art models. 4. We proposed a computationally efficient model that is partly pre-trained, with no typical iteration process in the main body.

Our major contributions can be summarised as follows:

1. We addressed a common problem in which several extracted phrases contained the same high-score word.
2. We proposed and analysed the reason and significance of two concepts to solve the problem: keyphrase coverage and keyphrase semantic diversity.
3. We proposed TripleRank, a method based on keyphrase coverage, keyphrase semantic diversity, and position information that has preferable precision compared with state-of-the-art models.
4. We proposed a computationally efficient model that is partly pre-trained, with no typical iteration process in the main body.

2. Related work

In 1998, Brin and Page [3] proposed PageRank, a webpage ranking algorithm that scored webpages based on the importance of pages that link to them and ranked them by correlated assessment. Inspired by PageRank, graph-based ranking algorithms have been developed and improved recently. Such algorithms evaluate the importance of candidate keyphrases by the number of other words in the documents that they are related to. TextRank, SingleRank, and PositionRank are state-of-the-art approaches for keyphrase extraction based on graphs. TextRank uses undirected graphs and iterates with the PageRank algorithm to obtain and rank the scores of every word. The result is outstanding and outperforms any other previously proposed keyphrase extraction system.

Topical PageRank (TPR) is a significant method in which Liu considered the semantic topics of documents in a measure of candidates' scores. Compared with the ranking schemes based on the number of words related to candidates (such as TextRank), TPR is a two-stage process with a featured process to build topic

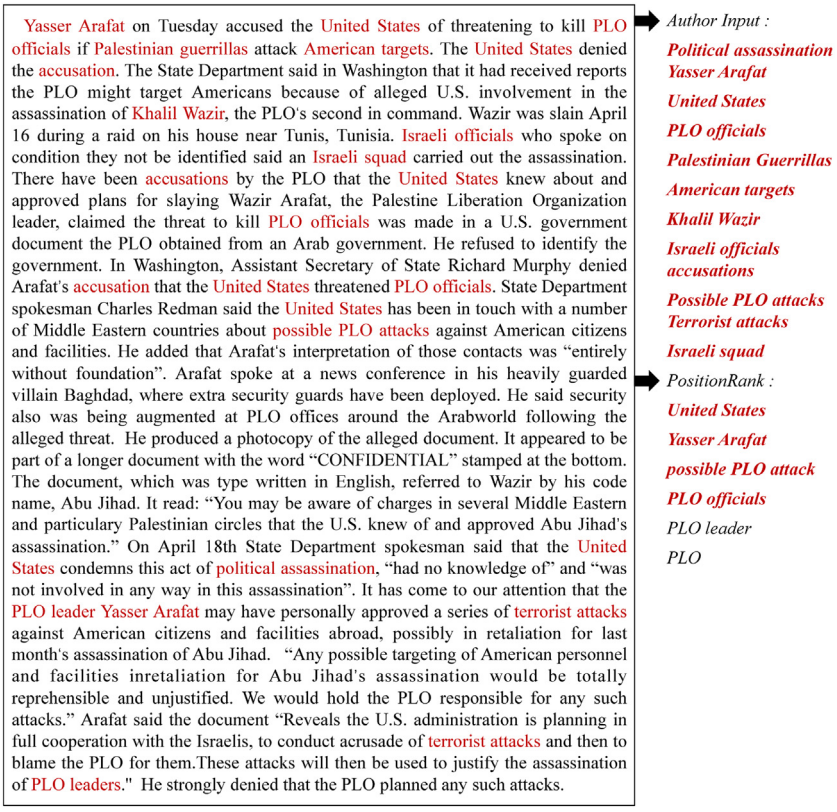


Fig. 1. Sociology document in the DUC dataset [15] and the extraction result of PositionRank.

interpreters. LDA [10] is a widely used topic model, which can generate the topic distribution of words and documents. LDA played a significant role in TPR. In the LDA process, documents are regarded as random mixtures of latent topics, and topics are characterised by a distribution over words [9]. Building topic interpreters is used to acquire topic distributions of words and documents using LDA. In each topic of a document, the topic-specific candidates are ranked by their importance scores related to the specified topic. The final keyphrase is based on the integrated ranking of the topic-specific rankings. Since Liu and his team first used the LDA model in keyphrase extraction, many methods involving LDA have been proposed. In 2015, Sterckx et al. [16] proposed an LDA-based method that improved TPR, computing a single PageRank for each text regardless of the number of topics, which enhanced the computing speed. Cho and Li [17] proposed a model using LDA to evaluate the importance of words in adjacent similar documents, thus, extracting the proper keyphrases. Teneva and Cheng [18] proposed the salience rank, which is also regarded as a modification of TPR that requires only one PageRank run to improve efficiency.

Bougouin et al. [19] researched topic information and proposed a concept similar to TripleRank. Their model, TopicRank, first built a graph of candidates according to topic relations and regarded candidates belonging to the same topic as equally important. In post-ranking heuristics, the most representative candidate in each topic was selected as a result. Bougouin et al. aimed to optimise topic coverage, which is an idea similar to our keyphrase semantic diversity. In another study, Bougouin et al. [11] proposed a multipartite rank, in which they combined ranking and post-ranking heuristics into a single operation and built a multipartite graph to connect sets of topic-related candidates. Their work further improved topic diversity and system conciseness.

Florescu and Caragea [12] proposed an idea that used the position information to extract keyphrases in scholarly documents. PositionRank is an unsupervised, graph-based model that incorporates position information into a biased PageRank algorithm. The occurrence position of candidates is calculated as the weight in a biased PageRank algorithm. PositionRank was shown to be more efficient than other methods.

Embedding algorithms – including Word2Vec [20], Doc2Vec, GloVe, and BERT – also contributed to keyphrase extraction as common approaches to semantic representation. Word2Vec is an efficient predictive model consisting mainly of neural networks for generating distributed vector representations of words. Compared with one-hot encoding (the basic word embedding algorithm with sparse discrete outputs), Word2Vec provides lower-dimension vector representations. Key2Vec [21] is a phrase-embedding algorithm developed for keyphrase extraction. With trained phrase embeddings, thematic weights are assigned to candidate phrases, and thematic-weighted PageRank is applied to rank candidates. EmbedRank [22] is another embedding-based keyphrase extraction algorithm that uses Doc2Vec [23] (or another method), which embeds candidate phrases and documents into the same vector space and calculates their cosine similarity for ranking. Moreover, the reference vector algorithm (RVA) [24] uses locally trained GloVe and calculates the reference vector of words in titles and abstracts of documents. By calculating the cosine similarity of the candidate vectors and the reference vector, the closer candidates are extracted.

3. Proposed model

The architecture of TripleRank consists of three sub-models, with scoring features including keyphrase coverage, keyphrase semantic diversity, and position information.

As Fig. 2 shows, three features are scored separately and subsequently combined by mathematical methods (including the

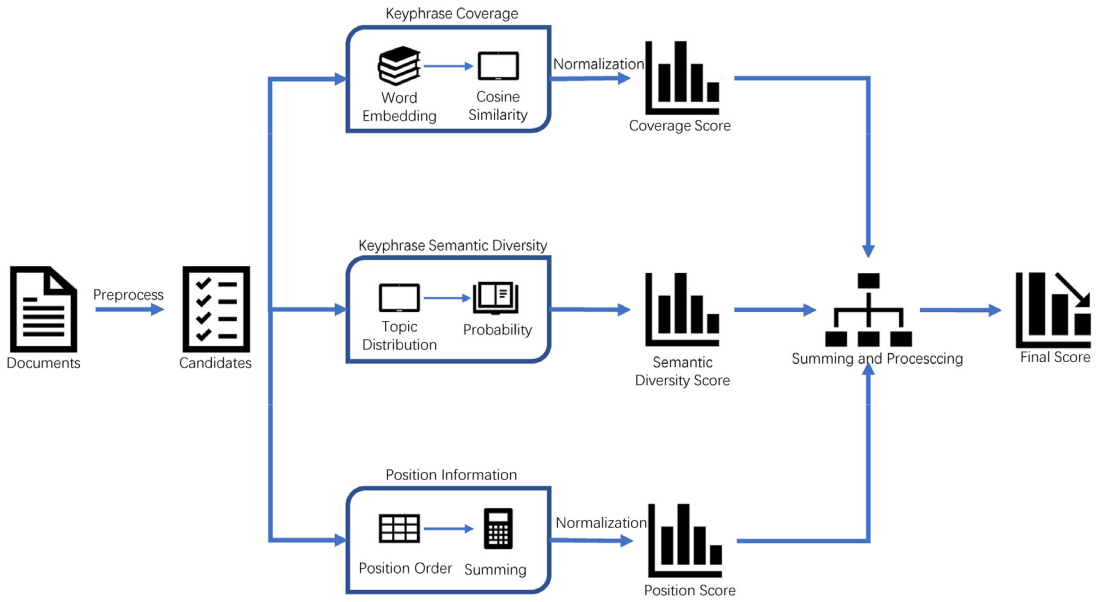


Fig. 2. Architecture of TripleRank, where the input is a document and the output is a candidates' ranking.

weighted sum and the weighted harmonic sum) to output a comprehensive ranking score.

The keyphrase coverage score evaluates the breadth of content in documents that the extracted keyphrases can express. We found that the similarity between candidates and other words in documents was effective for scoring keyphrase coverage because higher similarity indicated a better representative. To acquire this similarity, we embedded words into word vectors, which possessed the semantic information for adjacent words that are usually associated. Hence, we used word vectors to assess the proximity of candidates to other words, thus, assessing their coverage level. The keyphrase semantic diversity score enhanced the precision of keyphrases by involving more topics to avoid the circumstances in which phrases involved the same highly weighted word from the rank list. We used LDA, a topic model, to evaluate the keyphrase semantic diversity. Although there are adequate and feasible topic models and solutions for the keyphrase semantic diversity, they are relatively more complex. As a well-researched model with wide applicability, the LDA model can be the best to improve the performance of our model. The keyphrase semantic score is calculated by the probability of the topic to which a candidate belongs. With the topic distribution output by LDA, we can elevate the topical influence in the extraction results to avoid semantically similar phrases. The position information score is obtained by position order, which is inspired by PositionRank; however, we did not use the process of PageRank and only kept the position weight. The final score is calculated by combining the normalised scores of the three features.

3.1. Keyphrase coverage

In our model, keyphrase coverage was evaluated using Word2Vec. Word2Vec is a word-embedding algorithm that provides distributed representations of words in a vector space. Proposed by Mikolov et al. the model is a simplified neural network with an input layer, hidden layer, and output layer. Generally, Word2Vec has two approaches: continuous bag-of-words (CBOW) and skip-gram [16]. The training input of the CBOW model is the word vector of the context words of a specific word, and the output is the word vector of this specific word. In contrast, skip-gram is reverse to CBOW: the input is a word vector of a specific word, and the output is a context word vector of the specific word. CBOW is more efficient for small databases, while skip-gram performs better in large corpora. Hence, skip-gram was selected for our experiments.

Our objective is to predict the context word in the training process, assuming a specific vocabulary in the training corpus. The input word is called the target word, and the context words are the words around the target word, whose predicted probabilities are the output [25]. In the skip-gram model architecture, the input layer is a one-hot encoded word vector of the target word. One-hot encoding represents a V (according to vocabulary size) dimension vector $\{x_1, \ldots, x_V\}$ with only unit x_k as 1, while the other units are 0. The hidden layer size is N, and it appears as an N-dimensional vector $\{h_1, \ldots, h_n\}$. The output layer is a $C \times V$ dimension $\{y_{11}, \ldots, y_{CV}\}$, where C stands for the number of words in the context. Units in adjacent layers are fully connected, and the weights between the input layer and the hidden layer are described by matrix $W_{V\times N}$, while the weights between the hidden layer and the output layer are $W'_{N\times V}$.

When a serial of words $w_1, w_2, \ldots, w_t$ is trained, the skip-gram model aims to maximise the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log p\left(w_{t+j}|w_t\right) \tag{1}$$

The posterior distributions of words are outputted through a softmax function on the output layer:

$$p\left(w_O|w_I\right)=\frac{exp\left(v'_{w_O}{}^{T}v_{w_I}\right)}{\sum_{w=1}^{W}exp\left(v'_{w}{}^{T}v_{w_I}\right)} \tag{2}$$

There are C posterior distributions on the output layer computed by sharing the same weight matrix panel, which represents the prediction to C context words. To maximise (1), the representation of $\log p\left(w_{t+j}|w_t\right)$, expression (2) should be maximised. However, the basic skip-gram formulation by softmax may require excessive computational resources for large deltas $\log p\left(w_O|w_I\right)$. A hierarchical softmax method is used [26].

The keyphrase coverage is evaluated by the similarity between word vectors, and we use the cosine similarity. The smaller angle indicates the higher similarity between words. The keyphrase

coverage score is calculated by the sum of the similarities between all pairs of candidates:

$$Coverage\left(\vec{w}_i\right) = \sum_{j=1}^{v} Similarity\left(\vec{w}_i, \vec{w_{j\neg i}}\right) \tag{3}$$

where *Coverage* is the coverage score, *Similarity* is the cosine similarity of two word vectors, and $\vec{w_{j\neg i}}$ is the word vectors of words in the document except w_i.

3.2. Position information

Position information is another important feature used by our algorithm, which is assessed by part of the PositionRank model. Proposed by Florescu and Caragea, PositionRank is an unsupervised, graph-based model that considers the positions of words. PositionRank utilises the specifics of scholarly documents, where the important words appear earlier in the passage. The algorithm is divided into three stages:

(1) Word-level graph construction
(2) Position-biased PageRank
(3) Candidate phrase formation [12]

The whole model is a biased PageRank with the weight calculated by position information. In our model, similarly, we calculated candidates' appearance orders as scores while PositionRank utilised them as biases. However, as one of three features, to keep the conciseness and economy of the entire model, we did not employ any iteration process or graph construction. Hence, the position information score is calculated by the method below. To examine our simplification is resultful, PositionRank has experimented as one of the baseline models, which is also because it is one of the best approaches proposed recently. P is calculated as a sum of all the occurrences of a word in d:

The position information score is calculated as: Let d be a document for keyphrase extraction that contains words w_i. The weight factor p_{w_i} is calculated as a sum of the reciprocals of all the occurred position orders:

$$p_{w_i} = \frac{1}{position\ 1} + \frac{1}{position\ 2} + \cdots + \frac{1}{position\ \alpha} \tag{4}$$

Where *position* α is the α_{th} occurred order of w_i.

Consider the following specific example. If a word appears at the fourth, sixth, and eighth positions in the target document, the weight is computed as

$$\frac{1}{4} + \frac{1}{6} + \frac{1}{8} = 0.54 \tag{5}$$

Accordingly, we obtain the position information score of our model. Hence, we propose

$$Position\left(w_i\right) = p_{w_i} \tag{6}$$

where $Position\left(w_i\right)$ is the position information score of candidates w_i.

3.3. Keyphrase semantic diversity

Keyphrase semantic diversity is evaluated by LDA, a generative probabilistic model for collections of discrete data with a three-level Bayesian model [10]. The LDA model can generate the topic distribution of the document and words. In our model, the LDA model describes one of the three significant features, keyphrase semantic diversity. We aimed to maximise the topic diversity and eliminate the different forms of phrases with similar semantics from the extraction results. Therefore, the topic probabilities of words and phrases are scored and ranked in the final score.

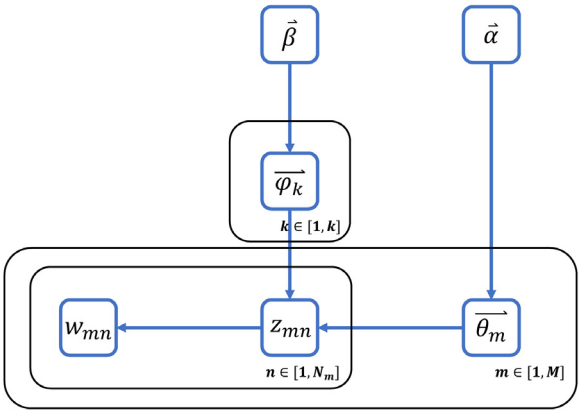


Fig. 3. Bayesian Network of LDA [10].

The LDA model generates words by sampling processes with plates, topic plates, document plates, and word plates. Assume that corpus W includes V words, M documents with N_m words, and K topics. The process above can be described by the Dirichlet-multinomial unigram model, which is based on the following theory:

$$Posterior = \frac{likelihood \cdot prior}{evidence} \tag{7}$$

Fig. 3 shows the network of LDA, where $\vec{\alpha}$ and $\vec{\beta}$ are hyperparameters, $\vec{\theta_m}$ is the topic mixture proportion for document m, $\vec{\varphi_k}$ is the mixture component of topic k, z_{mn} is the mixture indicator that chooses the topic for n_{th} word in the document m, and w_{mn} is the term indicator for the n_{th} word in the document m.

The network is a generative process that is constructed by two Dirichlet-multinomial unigram models [27]:

$$\vec{\alpha} \rightarrow \vec{\theta}_m \sim Dir\left(\vec{\alpha}\right) \rightarrow Mult\left(\vec{\theta}_m\right) \tag{8}$$

$$\vec{\beta} \rightarrow \vec{\varphi}_K \rightarrow Dir\left(\vec{\beta}\right) \rightarrow Mult\left(\varphi_{\vec{Z_{mn}}}\right) \tag{9}$$

where symbol $\rightarrow$ is a sampling process. The joint distribution can be factored. Gibbs sampling is a special case of Markov chain Monte Carlo (MCMC) simulation [28], which is used as an approach for approximate inference in LDA. After using Gibbs sampling, the result is as follows:

$$p\left(z_i = k|z\neg i, w\right) \propto \frac{n_{m,\neg i}^{(k)} + \alpha_k}{\sum_{k=1}^{K}\left(n_{m,\neg i}^{(t)} + \alpha_k\right)} \cdot \frac{n_{k,\neg i}^{(t)} + \beta_t}{\sum_{t=1}^{V}\left(n_{k,\neg i}^{(t)} + \beta_t\right)} \tag{10}$$

where $i = (m, n)$ corresponds to n_{th} word in the document m, t is a term for w_{mn}, and $n_m^{(k)}$ and $n_k^{(t)}$ are count statistics.

The training process is relatively brief. We randomly assign topic indicators z to every word w in the corpus. Subsequently, we sample the topic using (10) until Gibbs sampling converges. The topic–word co-occurrence matrix is the LDA model result. The output is a dictionary of words and a topic probability distribution. The top probability topic of a word is regarded as the topic of the word.

The keyphrase semantic diversity score is the possibility of the topic that a candidate belonged to:

$$Diversity\left(w_i\right) = p\left(z_i|w_i\right) \tag{11}$$

where w_i and z_i stand for a candidate and the topic that w_i belonged to respectively.

3.4. Ranking scheme

Our keyphrase extraction algorithm is an unsupervised algorithm that takes advantage of the implicit features of words. In contrast to previous studies, our algorithm voids conventional iterations (LDA model uses iterations in the training process, but our algorithm barely uses the universal training result). The combination of features and ranking (see Fig. 4) is based on hyperparameters and weighted averaging. Before the combination, the position score and keyphrase coverage score need to be normalised.

$$C_i = \frac{Coverage\,(w_i)}{\sum_{i=1}^{v} Coverage\,(w_i)} \tag{12}$$

$$P_i = \frac{Position\,(w_i)}{\sum_{i=1}^{v} Position\,(w_i)} \tag{13}$$

where w_i is a candidate, C_i is the normalised keyphrase coverage score, v is the number of candidates, and P_i is the normalised position score. The keyphrase semantic diversity score is formed in probability; hence, a normalisation process is not required.

The combining process is mainly divided into two parts:

(a) A weighted sum between the keyphrase semantic diversity score and keyphrase coverage score.
(b) Weighted harmonic mean between (a) and the position score.

For (a), the keyphrase coverage score and the keyphrase semantic diversity score are aggregated to D_i. We consider keyphrase coverage and the keyphrase semantic diversity analogue as a function that mainly assists in optimising precision. The weighted sum is normally applied when summing units of different importance. Through repeated attempts, according to the best result, we decided to weight keyphrase coverage and keyphrase semantic diversity to 0.3 and 0.7. Subsequently, we applied an optimising factor on the basis of topic importance, which aimed at increasing the influence of topical features, to the summing result, and we normalised it.

$$D_i = \frac{(0.3 \cdot C_i + 0.7 \cdot Diversity\,(w_i)) \cdot \frac{N_{z_i}}{v}}{\sum_{i=1}^{v} (0.3 \cdot C_i + 0.7 \cdot Diversity\,(w_i)) \cdot \frac{N_{z_i}}{v}} \tag{14}$$

where N_{z_i} is the number of words in the same topic, v is the number of candidates, and D_i is the normalised ranking score with keyphrase coverage and keyphrase semantic diversity aggregated.

For (b), we already combined keyphrase coverage and keyphrase semantic diversity, and the next aggregating process is realised by the weighted harmonic mean. In contrast to the arithmetic mean, the weighted harmonic mean is susceptible to the influence of the upper and lower values, in which the lower value is more influential. Because we only have two variables, once D_i or P_i is excessively low, the synthesis is degraded. Hence, by engaging the harmonic mean to branch scores, we are supposed to obtain a more integrated final score. The hyperparameters, weights of D_i and P_i, are set to 0.8 and 0.2, respectively, according to the fact that D_i contains two features and to our repeated experiments. Hence, we propose the following final score:

$$S_i = \frac{0.16 \cdot D_i \cdot P_i}{0.8 \cdot D_i + 0.2 \cdot P_i} \tag{15}$$

where S_i, D_i, and P_i represent the final score of a candidate w_i, normalised ranking score with keyphrase coverage and keyphrase semantic diversity, and normalised position score, respectively.

We formed candidate phrases according to their continuous appearance in the documents. Only phrases ended by a noun length of up to 3 are regarded as candidate phrases. The phrase score is calculated as the sum of the single words' final score. The complete algorithm is shown in Algorithm 1.

Algorithm 1: ***TripleRank***

Input: Document or passage $(w_1, ..., w_i, ..., w_n)$, $i \in [1, n]$
Value k for extraction numbers
Output: k for extraction numbers
Begin
Initialization
for *each candidate* w_i **do**
  initialize word vector $\vec{w}_i$ Eq.(1) and Eq.(2)
  Calculate keyphrase coverage score $Coverage(w_i)$ by Eq.(3)
  Calculate position score $Position(w_i)$ by Eq.(5) and Eq.(6)
  Calculate topic distributionp$(z_i\,|w_i)$ by Eq.(10)
  Calculate keyphrase semantic diversity score $Diversity(w_i)$ by Eq.(11)
endfor
Scoring
for *each* $Position(w_i)$ *and* $Coverage(w_i)$ **do**
  calculate C_i by Eq.(12)
  calculate P_i by Eq.(13)
  for *each* C_i *and* P_i **do**
    calculate D_i by Eq.(15)
  endfor
endfor
Ranking
for *each candidate* w_i **do**
  Form phrase and calculate phrase score
  Rank the phrase scores from
  $(Keyphrase(1)..., Keyphrase(j), ..., Keyphrase(n), i \in [1, n]$
  while $j \leq k$ **do**
    Return $Keyphrase(j)$
  end
endfor

4. Experiments and results

4.1. Experiment datasets

Our experiments used four datasets: Knowledge Discovery and Data Mining (KDD), World Wide Web Conference (WWW), Inspec, and Document Understanding Conference (DUC) [15]. KDD and WWW [29] are compiled from the CiteSeerX digital library and include research papers from the two ACM conferences by Gollapalli and Caragea. Inspec [30] has been a dataset used in science and technology literature since 1898. The statistics of the datasets are shown in Table 2, together with an experimental discovery.

4.2. Evaluation metrics

The evaluation metrics used in the experiments include precision(Pre), recall(Re), and F1-score(F1). Precision, recall, and F1-score represents:

$$P = \frac{c_{correct}}{c_{extract}}, R = \frac{c_{correct}}{c_{standard}}, F1 = \frac{2PR}{P+R} \tag{16}$$

where $c_{correct}$ is the number of correct extracted keyphrases, $c_{extract}$ is the number of total extracted keyphrases, and $c_{standarad}$ is the number of keyphrases given by the authors or annotators. The three evaluating metrics are involved in our experiments to provide an objective result. Among the three standards, $F1 - Score$ is regarded as the comprehensive metric in our experiments. In addition, the evaluation results are compared with four baseline methods.

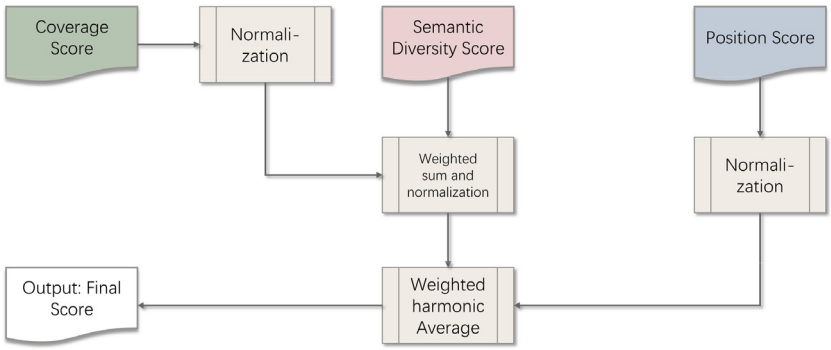


Fig. 4. Process of calculating the final score from feature scores.

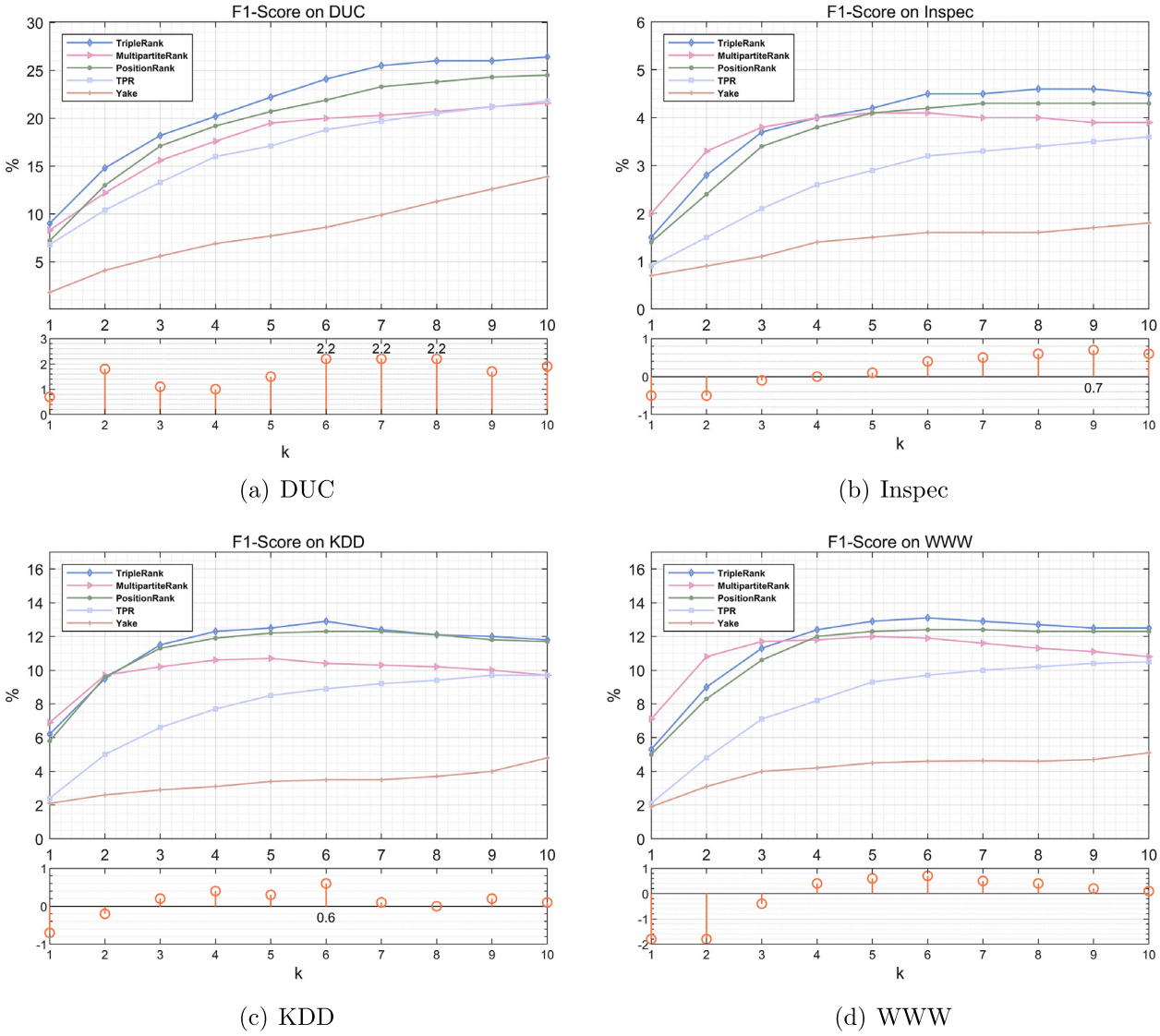


Fig. 5. F1-Score curves for different models.

4.3. Overall performance

To examine the performance of TripleRank, we selected four strong baseline unsupervised methods. We mainly compare our algorithm with MultipartiteRank and PositionRank, state-of-the-art competitive baseline methods. In addition, we used a similar feature, position information, as in PositionRank, and MultipartiteRank used a similar concept to keyphrase coverage; hence, we compared TripleRank to them. Considering other ordinary methods, we chose TPR and Yake, which are relatively inferior in our experiments. The experiment project and baseline method setup are based on PKE [31] .

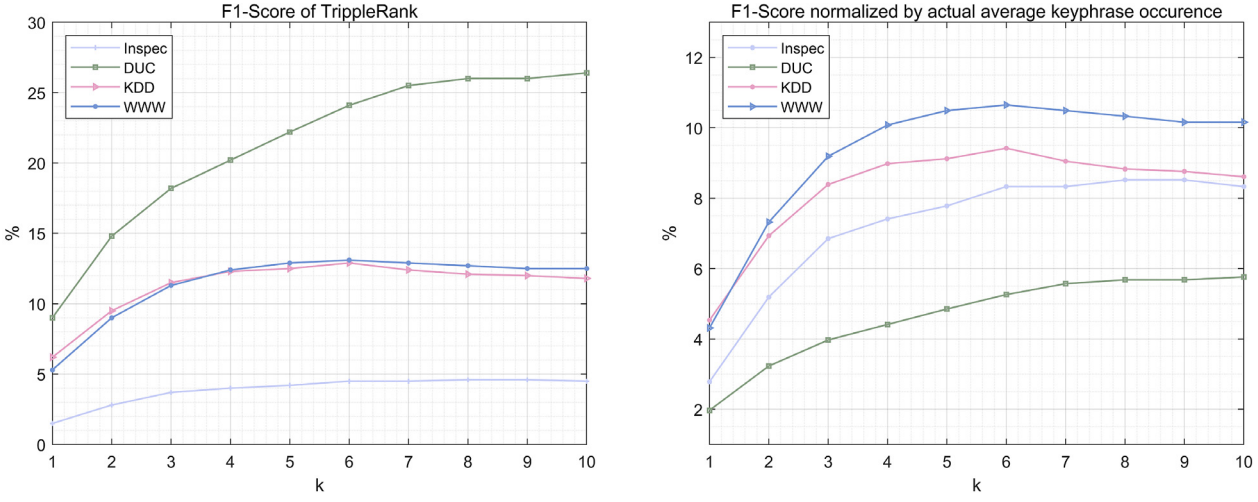
Yasser Arafat on Tuesday accused the United States of threatening to kill PLO officials if Palestinian guerrillas attack American targets. The United States denied the accusation. The State Department said in Washington that it had received reports the PLO might target Americans because of alleged U.S. involvement in the assassination of Khalil Wazir, the PLO's second in command. Wazir was slain April 16 during a raid on his house near Tunis, Tunisia. Israeli officials who spoke on condition they not be identified said an Israeli squad carried out the assassination. There have been accusations by the PLO that the United States knew about and approved plans for slaying Wazir Arafat, the Palestine Liberation Organization leader, claimed the threat to kill PLO officials was made in a U.S. government document the PLO obtained from an Arab government. He refused to identify the government. In Washington, Assistant Secretary of State Richard Murphy denied Arafat's accusation that the United States threatened PLO officials. State Department spokesman Charles Redman said the United States has been in touch with a number of Middle Eastern countries about possible PLO attacks against American citizens and facilities. He added that Arafat's interpretation of those contacts was "entirely without foundation". Arafat spoke at a news conference in his heavily guarded villain Baghdad, where extra security guards have been deployed. He said security also was being augmented at PLO offices around the Arabworld following the alleged threat. He produced a photocopy of the alleged document. It appeared to be part of a longer document with the word "CONFIDENTIAL" stamped at the bottom. The document, which was type written in English, referred to Wazir by his code name, Abu Jihad. It read: "You may be aware of charges in several Middle Eastern and particulary Palestinian circles that the U.S. knew of and approved Abu Jihad's assassination." On April 18th State Department spokesman said that the United States condemns this act of political assassination, "had no knowledge of" and "was not involved in any way in this assassination". It has come to our attention that the PLO leader Yasser Arafat may have personally approved a series of terrorist attacks against American citizens and facilities abroad, possibly in retaliation for last month's assassination of Abu Jihad. "Any possible targeting of American personnel and facilities inretaliation for Abu Jihad's assassination would be totally reprehensible and unjustified. We would hold the PLO responsible for any such attacks." Arafat said the document "Reveals the U.S. administration is planning in full cooperation with the Israelis, to conduct acrusade of terrorist attacks and then to blame the PLO for them.These attacks will then be used to justify the assassination of PLO leaders." He strongly denied that the PLO planned any such attacks.

Author Input :
Political assassination
Yasser Arafat
United States
PLO officials
Palestinian Guerrillas
American targets
Khalil Wazir
Israeli officials
accusations
Possible PLO attacks
Terrorist attacks
Israeli squad

PositionRank :
United States
Yasser Arafat
possible PLO attack
PLO officials
PLO leader
PLO

TripleRank:
United States
Yasser Arafat
PLO officials
State department spokesman
Possible PLO attack
American target

Fig. 6. Sociology document in DUC dataset [15] and the extraction result of PositionRank and TripleRank.



(a) F1-Score of our method that uses different datasets

(b) F1-Score of our method that uses different datasets normalized by average keyphrase actual occurrence.

Fig. 7. F1-Score curves by datasets.

As Table 1 shows, the experimental results of our method compared with three baseline methods are shown in terms of four metrics: precision, recall, F-1 score, and MRR for top k = 4, 6, 8, and 10 predicted keyphrases. The value of k is selected from 4 to 10 for the reason that our model aimed to reduce synonym phrases' occurrences. When predicting too few keyphrases, our enhancement may be not practical. TripleRank outperformed the four baseline methods. As competitive baseline methods, MultipartiteRank and PositionRank tied the result or showed an advantage in some metrics; nevertheless, none of them is the comprehensive metric F1-score. In the full version of the results shown in Appendix A, when k is valued from 1 to 3, our result shows less superiority. This conforms to our prediction that the way TripleRank improved performance is based on more prediction attempts. In addition, this phenomenon indirectly demonstrates that the promotion of TripleRank compared to former works is based on solving the problem of synonym phrases by the one high score word.

Table 1
Performances of TripleRank and three baseline models.

	@k=4			@k=6			@k=8			@k=10		
DUC												
	Pre	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*
TripleRank	**32.9**	**14.8**	**20.2**	**30.1**	**20.5**	**24.1**	**27.6**	**24.9**	**26**	**25.8**	**27.7**	**26.4**
MultipartiteRank	28.4	12.9	17.6	24.9	17	20	21.9	19.9	20.7	20.5	23.2	21.6
PositionRank	31.1	14	19.2	27.4	18.6	21.9	25.3	22.9	23.8	23.4	26.2	24.5
TPR	25.8	11.7	16	23.5	15.9	18.8	21.8	19.6	20.5	20.8	23.2	21.8
Yake	11.2	5	6.9	10.9	7.3	8.6	12.1	10.8	11.3	13.3	14.9	13.9
INSPEC												
	@k=4			@k=6			@k=8			@k=10		
	Pre	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*
TripleRank	**4.2**	**4.5**	**4**	**3.9**	**6.2**	**4.5**	**3.6**	**7.4**	**4.6**	**3.4**	**8.2**	**4.5**
MultipartiteRank	**4.2**	4.3	**4**	3.6	5.4	4.1	3.1	6.3	4	2.9	7.1	3.9
PositionRank	4	4.2	3.8	3.7	5.7	4.2	3.4	6.9	4.3	3.2	7.8	4.3
TPR	2.6	3	2.6	2.8	4.5	3.2	2.7	5.5	3.4	2.6	6.5	3.6
Yake	1.4	1.6	1.4	1.3	2.2	1.6	1.3	2.9	1.6	1.3	3.5	1.8
KDD												
	@k=4			@k=6			@k=8			@k=10		
	Pre	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*
TripleRank	**12.9**	**12.8**	**12.3**	**11.5**	**16.4**	**12.9**	**9.9**	**18.3**	**12.1**	**9.2**	**19.8**	**11.8**
MultipartiteRank	11.1	11	10.6	9.3	13.1	10.4	8.2	15.3	10.2	7.4	16.8	9.7
PositionRank	12.5	12.4	11.9	10.9	15.7	12.3	9.8	18.2	12.1	9	19.9	11.7
TPR	8	7.9	7.7	7.9	11.1	8.9	7.6	13.9	9.4	7.4	16.4	9.7
Yake	3.2	3.3	3.1	2.9	4.5	3.5	2.8	5.7	3.7	3.5	8.8	4.8
WWW												
	@k=4			@k=6			@k=8			@k=10		
	Pre	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*	*Pre*	*Re*	*F1*
TripleRank	**13.7**	**12.2**	**12.4**	**12.2**	**15.8**	**13.1**	**10.8**	**18.1**	**12.7**	**10.1**	19.6	**12.5**
MultipartiteRank	13.3	11.4	11.8	11.3	14	11.9	9.8	15.7	11.3	8.7	17.1	10.8
PositionRank	13.3	11.9	12	11.6	15	12.4	10.5	17.4	12.3	9.9	**19.7**	12.3
TPR	9.1	8.1	8.2	9.1	11.7	9.7	8.6	15.7	10.4	8.5	16.7	10.5
Yake	4.6	4.1	4.2	4.2	5.7	4.6	3.7	6.8	4.6	3.9	8.6	5.1

The results are shown as percentages, and the best results are shown in bold and red.
Pre stands for precision, *Re* stands for recall and *F1* stands for F1-Score.

Table 2
Actual occurrence times of keyphrases in datasets.

Occurrences	0	1	2	3	4	5	6	7	>7	***Documents***	***Average***
DUC	3	9	25	42	50	41	39	22	23	254	4.58
Inspec	1211	555	180	47	7	0	0	0	0	2000	0.54
KDD	183	244	230	77	16	7	0	0	0	757	1.37
WWW	375	472	331	114	25	8	3	1	0	1329	1.23

Fig. 5 shows the F1-Score graphs for the four datasets, where the upper graph is the performance of four models on the basis of k and the lower graph is the gap between TripleRank and the best-performing baseline models. According to Fig. 5(a)(b), the superiority of TripleRank is more evident in DUC and Inspec. On the other hand, in the KDD and WWW datasets, the result is more approaching, but TripleRank also generally leads. However, when predicting 8 to 10 keyphrases, the curve trend is approximate to PositionRank's curve. In other words, our method performs better when k varies from 4 to 8. This phenomenon could be explained by the fact that when k increases, the influence of synonyms gradually dwindled because the error tolerance increased.

In general, although the superiority floated along with the value of k, we proved that TripleRank outperforms other baseline models comprehensively in the experiment.

4.4. Supporting evidence

Fig. 1 presents the extraction result of PositionRank on a DUC document. For comparison, the extraction result of our method is added in Fig. 6.

The result shows that TripleRank predicted five correct keyphrases out of six attempts, while PositionRank made 4. However, the results of a single document are limited. We focused on using this example to illustrate the difference and enhancement of our method compared with the other methods. In our result, "PLO" and "state" appeared twice as high weighted words, whereas most of the appearances are correct. Those of PositionRank, as a comparison, included "PLO" in four phrases. The keyphrases extracted by TripleRank are more decentralised in semantics, which attributes to the proper use of keyphrase semantic diversity and keyphrase coverage.

4.5. Effects of datasets

We observed that the performance of TripleRank fluctuates in different datasets. The F1-score gap between Inspec and DUC when k = 10 is up to 10.3%. Inspec is the experimental dataset that our method performs exceedingly excellent. We were afraid that the huge gap might have resulted from the experiment fault. The consolation is that the other four baseline methods also showed a similar gap in results, which shifted our attention to the influence of datasets.

We stated the actual occurrence times of author given keyphrases in original documents and found that the author given keyphrases are less likely to appear in documents. The detailed statistics are shown in Table 2.

The average occurrences of author-given keyphrases are significantly less than KDD and WWW. Having similar actual keyphrase occurrences, KDD and WWW showed analogical curves

Table A.1
Performances of TripleRank and baseline models.

								DUC							
	Pre@1	Re@1	F1@1	Pre@2	Re@2	F1@2	Pre@3	Re@3	F1@3	Pre@4	Re@4	F1@4	Pre@5	Re@5	F1@5
TripleRank	44.1	5	9	40	9.2	14.8	36.2	12.3	18.2	32.9	14.8	20.2	31.1	17.5	22.2
MultiPartiteRank	40.6	4.6	8.3	33.3	7.5	12.2	30.7	10.5	15.6	28.4	12.9	17.6	27.2	15.5	19.5
PositionRank	35.8	4	7.2	35.4	8	13	34	11.5	17.1	31.1	14	19.2	29	16.3	20.7
TPR	32.7	3.8	6.8	28	6.5	10.4	26.1	9	13.3	25.8	11.7	16	23.9	13.5	17.1
Yake	9.4	1	1.8	11.4	2.5	4.1	11.2	3.8	5.6	11.2	5	6.9	10.9	6	7.7
	Pre@6	Re@6	F1@6	Pre@7	Re@7	F1@7	Pre@8	Re@8	F1@8	Pre@9	Re@9	F1@9	Pre@10	Re@10	F1@10
TripleRank	30.1	20.5	24.1	29.1	23.1	25.5	27.6	24.9	26	26.2	26.3	26	25.8	27.7	26.4
MultiPartiteRank	24.9	17	20	23.2	18.4	20.3	21.9	19.9	20.7	21.2	21.6	21.2	20.5	23.2	21.6
PositionRank	27.4	18.6	21.9	26.5	21.1	23.3	25.3	22.9	23.8	24.4	24.8	24.3	23.4	26.2	24.5
TPR	23.5	15.9	18.8	22.6	17.8	19.7	21.8	19.6	20.5	21.3	21.6	21.2	20.8	23.2	25.9
Yake	10.9	7.3	8.6	11.4	8.9	9.9	12.1	10.8	11.3	12.6	12.8	12.6	13.3	14.9	13.9
								Inspec							
	Pre@1	Re@1	F1@1	Pre@2	Re@2	F1@2	Pre@3	Re@3	F1@3	Pre@4	Re@4	F1@4	Pre@5	Re@5	F1@5
TripleRank	3.8	1	1.5	4.4	2.4	2.8	4.4	3.6	3.7	4.2	4.5	4	4	5.3	4.2
MultiPartiteRank	4.9	1.3	2	5.1	2.7	3.3	4.6	3.6	3.8	4.2	4.3	4	3.9	4.9	4.1
PositionRank	3.4	1	1.4	3.7	2.1	2.4	4.1	3.3	3.4	4	4.2	3.8	3.9	5.1	4.1
TPR	1.9	0.6	0.9	2.2	1.3	1.5	2.5	2.2	2.1	2.6	3	2.6	2.7	3.7	2.9
Yake	1.7	0.5	0.7	1.3	0.8	0.9	1.3	1.1	1.1	1.4	1.6	1.4	1.4	2	1.5
	Pre@6	Re@6	F1@6	Pre@7	Re@7	F1@7	Pre@8	Re@8	F1@8	Pre@9	Re@9	F1@9	Pre@10	Re@10	F1@10
TripleRank	3.9	6.2	4.5	3.7	6.7	4.5	3.6	7.4	4.6	3.5	7.9	4.6	3.4	8.2	4.5
MultiPartiteRank	3.6	5.4	4.1	3.4	5.9	4	3.1	6.3	4	3	6.7	3.9	2.9	7.1	3.9
PositionRank	3.7	5.7	4.2	3.6	6.5	4.3	3.4	6.9	4.3	3.3	7.4	4.3	3.2	7.8	4.3
TPR	2.8	4.5	3.2	2.7	5	3.3	2.7	5.5	3.4	2.7	6.1	3.5	2.6	6.5	3.6
Yake	1.3	2.2	1.6	1.3	2.5	1.6	1.3	2.9	1.6	1.2	3.1	1.7	1.3	3.5	1.8
								KDD							
	Pre@1	Re@1	F1@1	Pre@2	Re@2	F1@2	Pre@3	Re@3	F1@3	Pre@4	Re@4	F1@4	Pre@5	Re@5	F1@5
TripleRank	15.2	4	6.2	14.5	7.4	9.5	13.9	10.6	11.5	12.9	12.8	12.3	11.9	14.5	12.5
MultiPartiteRank	16.8	4.5	6.9	14.7	7.6	9.7	12.2	9.3	10.2	11.1	11	10.6	10.1	12.2	10.7
PositionRank	14.6	3.7	5.8	14.7	7.4	9.6	13.7	10.3	11.3	12.5	12.4	11.9	11.7	14.2	12.2
TPR	6.2	1.5	2.4	7.5	3.8	5	7.9	6	6.6	8	7.9	7.7	8.1	9.7	8.5
Yake	4.8	1.4	2.1	3.9	2.1	2.6	3.5	2.7	2.9	3.2	3.3	3.1	3.1	4	3.4
	Pre@6	Re@6	F1@6	Pre@7	Re@7	F1@7	Pre@8	Re@8	F1@8	Pre@9	Re@9	F1@9	Pre@10	Re@10	F1@10
TripleRank	11.5	16.4	12.9	10.5	17.3	12.4	9.9	18.3	12.1	9.5	19.3	12	9.2	19.8	11.8
MultiPartiteRank	9.3	13.1	10.4	8.7	14.2	10.3	8.2	15.3	10.2	7.8	16	10	7.4	16.8	9.7
PositionRank	10.9	15.7	12.3	10.4	17.1	12.3	9.8	18.2	12.1	9.3	19.2	11.8	9	19.9	11.7
TPR	7.8	12.7	9.2	7.6	13.9	9.4	7.6	13.9	9.4	7.6	15.6	9.7	7.4	16.4	9.7
Yake	2.9	4.5	3.5	2.8	5.1	3.5	2.8	5.7	3.7	3	6.8	4	3.5	8.8	4.8
								WWW							
	Pre@1	Re@1	F1@1	Pre@2	Re@2	F1@2	Pre@3	Re@3	F1@3	Pre@4	Re@4	F1@4	Pre@5	Re@5	F1@5
TripleRank	13.9	3.3	5.3	14.5	6.9	9	14.5	9.9	11.3	13.74	12.2	12.4	12.9	14.2	12.9
MultiPartiteRank	19.8	4.5	7.1	17.8	8	10.8	15.3	10	11.7	13.3	11.4	11.8	12.2	12.9	12
PositionRank	13.1	3.2	5	13.5	6.4	8.3	13.7	9.3	10.9	13.3	11.9	12	12.4	13.6	12.3
TPR	5.3	1.4	2.1	7.8	3.7	4.8	9	6.2	7.1	9.1	8.1	8.2	9.4	10.2	9.3
Yake	5.1	1.2	1.9	5.2	2.3	3.1	5.1	3.5	4	4.6	4.1	4.2	4.4	5	4.5
	Pre@6	Re@6	F1@6	Pre@7	Re@7	F1@7	Pre@8	Re@8	F1@8	Pre@9	Re@9	F1@9	Pre@10	Re@10	F1@10
TripleRank	12.2	15.8	13.1	11.4	16.9	12.9	10.8	18.1	12.7	10.3	18.9	12.5	10.1	19.6	12.5
MultiPartiteRank	11.3	14	11.9	10.4	14.9	11.6	9.8	15.7	11.3	9.2	16.4	11.1	8.7	17.1	10.8
PositionRank	11.6	15	12.4	11	16.3	12.4	10.5	17.4	12.3	10.1	18.7	12.3	9.9	19.7	12.3
TPR	9.1	11.7	9.7	8.9	13.1	10	8.7	14.5	10.2	8.6	15.7	10.4	8.5	16.7	10.5
Yake	4.2	5.7	4.6	3.7	6.2	4.6	3.7	6.8	4.6	3.7	7.5	4.7	3.9	8.6	5.1

in Fig. 7(a). The occurrences of author given keyphrases are influential to keyphrase extraction algorithms because the ranking is based on words that appear in documents, and the evaluation is mostly based on author input keyphrases. Once the author given keyphrases did not appear in the documents, the prediction was more likely to be invalid. As Fig. 7(b) shows, once the result of TripleRank is normalised by the average actual occurrence in documents, the curves are closer and distributed rationally,

Table B.1
Explanation for variables in proposed model.

Keyphrase Coverage		***Position Information***	
V	Vocabulary size	d	Document
N	Hidden layer size	p_{w_i}	Position weight factor
C	Number of words		
$\{x_1, \ldots, x_V\}$	One-hot vectors		
$\{h_1, \ldots, h_n\}$	Hidden layer vectors		
$\{y_{11}, \ldots, y_{CV}\}$	Output layer vectors		
T	Number of words trained		
w_i	A candidate		
Keyphrase Semantic Diversity		***Ranking Scheme***	
W	Corpus	C_i	Normalised keyphrase coverage score
M	Document number	P_i	Normalised position score
N_m	Word number of document M	D_i	Aggregated normalised ranking score
K	Topic numbers	S_i	Final score
z_i	Topic that w_i belonged to		

and the results of scholarly documents KDD and WWW datasets rose to the leading position. Although the supporting evidence exemplified is in DUC datasets, TripleRank still performs better on scholarly documents.

5. Conclusion

In this paper, we developed two innovative features – keyphrase semantic diversity and keyphrase coverage – to solve the problem of synonym phrase occurrence, which exists in state-of-the-art approaches. Using these two features together with the position information, we proposed TripleRank, a method of keyphrase extraction. The experimental results proved that TripleRank outperforms the baseline methods in precision, and the results are mainly improved by keyphrase semantic diversity and keyphrase coverage. Moreover, eliminating the unnecessary iterations that were present in traditional keyphrase extraction models improved the cost and efficiency of calculations. Our excellent results also prove the potential of our approach. Our further work may focus on discovering more efficient evaluation of the mentioned features. In addition, we are inspired by Zeng et al. [32] to use a sequence-to-sequence model based on reinforcement learning to extract the keyphrases not appearing in the text in our future research.

CRediT authorship contribution statement

Tuohang Li: Conceptualization, Methodology, Formal analysis, Investigation, Writing - original draft, Writing - reviewing & editing, Validation. **Liang Hu:** Resources, Project administrating. **Hongtu Li:** Methodology, Validation. **Chengyu Sun:** Software, Investigation. **Shuai Li:** Data curation. **Ling Chi:** Conceptualization, Methodology, Software, Supervision, Resources, Writing - reviewing & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is funded by: National Key R&D Plan of China under Grant No. 2017YFA0604500, and by National Sci-Tech Support Plan of China under Grant No. 2014BAH02F00, and by National Natural Science Foundation of China under Grant No. 61701190, and by Youth Science Foundation of Jilin Province of China under Grant No. 20160520011JH & 20180520021JH, and by Youth Sci-Tech Innovation Leader and Team Project of Jilin Province of China under Grant No. 20170519017JH, and by Key Technology Innovation Cooperation Project of Government and University for the whole Industry Demonstration, China under Grant No. SXGJSF2017-4, and by Key scientific and technological R&D Plan of Jilin Province of China under Grant No. 20180201103GX, and by Project of Jilin Province Development and Reform Commission, China under Grant No. 2019FGWTZC001.

Appendix A. Performances of TripleRank and baseline models

See Table A.1.

Appendix B. Explanation for variables in proposed model

See Table B.1.

References

[1] P.D. Turney, Learning To Extract Keyphrases from Text, National Research Council Canada, Institute for Information Technology, 2002, pp. ERB–1057.
[2] E. Frank, G.W. Paynter, I.H. Witten, C. Gutwin, C.G. Nevill-Manning, Domain-specific keyphrase extraction, in: Proceedings of the 16 th International Joint Conference on Arti Intelligence, 1999, pp. 668–673.
[3] K.S. Hasan, V. Ng, Automatic keyphrase extraction: A survey of the state of the art, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol. 1, 2014, pp. 1262–1273.
[4] A. Hulth, J. Karlgren, A. Jonsson, H. Bostrom, L. Asker, Automatic keyword extraction using domain knowledge, in: Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Text Processing, 2001, pp. 472–482.
[5] W.-T. Yih, J. Goodman, V.R. Carvalho, Finding advertising keywords on web pages, in: Proceedings of the 15th International Conference on World Wide Web, 2006, pp. 213–222.
[6] X. Jiang, Y. Hu, H. Li, A ranking approach to keyphrase extraction, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 756–757.
[7] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order To the Web, Technical Report, Stanford Digital Library Technologies Project, 1998.
[8] R. Mihalcea, P. Tarau, TextRank: bringing order into texts, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2004.
[9] Z. Liu, W. Huang, Y. Zheng, M. Sun, Automatic keyphrase extraction via topic decomposition, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2010, pp. 366–376.
[10] D. Blei, A. Ng, M. Jordan, J. Lafferty, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[11] F. Boudin, Unsupervised keyphrase extraction with multipartite graphs, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics vol. 1, 2018, pp. 1105–1115.
[12] C. Florescu, C. Caragea, PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics 2017, pp. 1105–1115.
[13] H. Zha, Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering, in: Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2002, pp. 113–120.

[14] X. Wan, J. Yang, J. Xia, Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction, in: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, 2007, pp. 855–860.
[15] X. Wan, J. Xiao, Single document keyphrase extraction using neighborhood knowledge, in: Proceedings of the 23rd AAAI Conferenceon Artificial Intelligence, 2008, pp. 855–860.
[16] L. Sterckx, T. Demeester, J. Deleu, C. Develder, Topical word importance for fast keyphrase extraction, in: Proceedings of the 24th International Conference on World Wide Web Companion, 2015, pp. 121–122.
[17] T. Cho, J.H. Lee, Latent keyphrase extraction using LDA model, J. Korean Inst. Intell. Syst. 25 (2) (2015) 180–185.
[18] N. Teneva, W. Cheng, Salience rank: efficient keyphrase extraction with topic modeling, in: Presented at the Meeting of the Association for Computational Linguistics, 2017.
[19] A. Bougouin, F. Boudin, B. Daille, TopicRank: Graph-based topic ranking for keyphrase extraction, in: Presented at the International Joint Conference on Natural Language Processing, 2013.
[20] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, Adv. Neural Inf. Process. Syst. 26 (2013).
[21] D. Mahata, J. Kuriakose, R.R. Shah, R. Zimmermann, Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings, in: Presented at the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2018.
[22] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, M. Jaggi, Simple unsupervised keyphrase extraction using sentence embeddings, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 2, 2018, pp. 634–639.
[23] E. Papagiannopoulou, G. Tsoumakas, A review of keyphrase extraction, in: WIREs Data Mining and Knowledge Discovery, vol. 10, 2020, e1339.
[24] E. Papagiannopoulou, G. Tsoumakas, Local word vectors guiding keyphrase extraction, Inf. Process. Manage. 54 (2018) 888–902.
[25] X. Rong, Word2vec parameter learning explained, Comput. Sci. (2014) [Online]. Available: http://arxiv.org/abs/1411.2738v2.
[26] F. Morin, Y. Bengio, Hierarchical probabilistic neural network language model, in: Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, 2005, pp. 246–252.
[27] G. Heinrich, Parameter Estimation for Text Analysis, Technical Report, 2005.
[28] D.J.C. Mackay, Informationtheory, inference, and learning algorithms, IEEE Trans. Inform. Theory 50 (10) (2003).
[29] S.D. Gollapalli, C. Caragea, Extracting keyphrases from research papers using citation networks, in: Presented at the Twenty-eighth Aaai Conference on Artificial Intelligence, 2014.
[30] A. Hulth, Improved automatic keyword extraction given more linguistic knowledge, in: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 2003, pp. 216–223.
[31] C. Giles, K. Bollacker, S. Lawrence, CiteSeer: An automatic citation indexing system, in: Proceedings of 3rd ACM Conference on Digital Libraries, 2000.
[32] D. Zeng, G. Tong, Y. Dai, F. Li, B. Han, S. Xie, Keyphrase extraction for legal questions based on a sequence to sequence model, J. Tsinghua Univ. 59 (4) (2019) 256–261.