GKEEP: An Enhanced Graph-Based Keyword Extractor With Error-Feedback Propagation for Geoscience Reports

Qinjun Qiu1,2, Zhong Xie1,2, Hong Xie3, and Bin Wang1,2

1National Engineering Research Center of Geographic Information System, Wuhan, China, 2China School of Geography and Information Engineering, China University of Geosciences, Wuhan, China, 3School of Arts and Communication, China University of Geosciences, Wuhan, China

Abstract As the amount of published geoscience literature grows, reading and summarizing texts of large collections has become a challenging task. Publication keywords can be considered basic components of knowledge structure representations and have been used to reveal knowledge concerning research domains. In contrast to data used in other research domains, the works on textual geoscience data that entail keyword extraction are limited. In this paper, we propose an unsupervised algorithm, the graph-based keyword extractor with error-feedback propagation (GKEEP), that enhances graph-based keyword extraction approaches by using an error-feedback mechanism similar to the concept of backpropagation. The proposed approach comprises the following steps. A preprocessed document is used as the input of the proposed model and is represented as a weighted undirected graph, where the vertices represent words and the edges represent the cooccurrence relationship between the words constrained by a window size. Subsequently, its nodes are ranked by their importance scores calculated by a graph-based ranking algorithm. Consequently, all the words have their own scores, and they are used to compute the scores of keyword candidates. Subsequently, the Word2Vec method is applied to recalculate the scores of keyword candidates and rank the keyword candidates to select the final keyword. It also utilizes error feedback to boost the rankings of the most salient terms that would otherwise be deemed less important. With empirical experiments on two real data sets (including our newly built data set), the proposed GKEEP model outperforms state-of-the-art unsupervised models and the existing graph-based ranking models. The proposed method can effectively reflect intrinsic keyword semantics and interrelationships.

Plain Language Summary The common or frequently used terms receive higher scores in traditional graph-based extraction owing to there are more edges connected to them. This paper proposes a graph-based KE algorithm called KE using error-feedback propagation, which utilizes the semantics of word embedding to assist in extracting keywords from geoscience reports. We hope that our approach will serve as an alternative method that deserves further study.

1. Introduction

The exponentially growing number of geoscience reports contains a wealth of knowledge for geoscience research (Qiu et al., 2018; C. Wang et al., 2016). Such a vast amount of information makes the process of indexing and finding reports a challenging and difficult task. The vast majority of reports are not associated with any keywords (aka keyphrases), which forces users to read an entire report to obtain an overall idea of its content (Ma et al., 2017; Qiu et al., 2019). The process of manually creating descriptive terms for texts can also quickly become a distressful and time-consuming task for those who need to tag their documents. With so many documents available, manual keyword extraction is no longer feasible (Campos et al., 2020; Sung et al., 2020; Zhang et al., 2020). Automatic keyword extraction methods are expected to provide guidelines for extracting a set of phrases that preserve the main theme of a document. Selecting the appropriate keyword is essential in domains that require an effective and fast understanding of large text collections (Qiu et al., 2018, 2019).

The earliest works that focused on automatic keyword extraction employed purely statistical techniques based on term/word frequency to evaluate the importance of the words (Chen & Xiao, 2016; Duari et al., 2019; Zhu & Hua, 2017), such as n-gram statistics, term frequency x inverse document frequency (TF-IDF) (Salton et al., 1974). In recent years, two lines of development of keyword extraction methods
have gained prominence. The first of these is the machine learning (ML)-based approaches and the second is based on the graph representation of text.

ML methods come in supervised and unsupervised flavors (Sarkar et al., 2010; Sung et al., 2020; C. Zhang, 2008; Zhang et al., 2017; Zhang et al., 2020). Supervised learning approaches need labeled training samples to induce the model. Each instance in the training set denotes a term in the document/report with label 1 (that represents keyword) or 0 (that represents not a keyword). Development of a training set requires manual annotation of the text, making the task tedious, subjective, and possibly inconsistent. Due to the intense human intervention required, supervised approaches to the keyword extraction task have not been able to sustain interest and popularity. Based on this reason, unsupervised approaches are favored as an alternative approach for recognizing keywords/keyphrases.

Graph-based methods represent candidate keywords as nodes and the relationship between two nodes as an edge. A set of scoring functions is used to rank the candidates based on a specific graph property (Figueroa et al., 2017; Li et al., 2016; Mihalcea & Tarau, 2004; Rose et al., 2012). Graph-based extraction, however, has the drawback that common or frequently used terms obtain higher scores because they have more edges connected to them. Similarly, rare or infrequent terms obtain lower scores because they are less connected. To overcome this observed weakness on graph-based approaches, we find that the backpropagation concept is a well-suited approach.

Although considerable research has been devoted to this topic over the years, the problem of extracting relevant keywords with high accuracy remains unsolved. Several factors contribute to this. (1) More variations: the text in geoscience reports is complex in terms of text characteristics and patterns and typically written by many different writers/inspectors from various institutions and localities (e.g., “elastic modulus” and “Young’s modulus” represent the same quantity). (2) Domain-specific uniqueness: when referring to domain-specific text, geological reports demonstrate domain-specific uniqueness and include high levels of specialized details, complex concept identifications, and relationship associations about minerals, rocks, and geological times (e.g., identifying complex terms about the Permian–Triassic Boundary Section in Changxing Zhejiang and the Maastrichtian Age). The existing statistics-based and ML-based approaches cannot handle such complexities and variability while still obtaining high precision and recall. (3) Existing state-of-the-art graph-based keyword extraction methods rely only on a co-occurrence relation between the candidate keywords, while completely ignoring the semantic relationship. These observations motivate design of word scoring approaches that account for semantic connectivity among the words.

In this study, we propose a graph-based KE algorithm called KE using error-feedback propagation (GKEEP), which utilizes the semantics of word embedding to assist in extracting keywords from geoscience reports. The key idea of GKEEP is that, when a node with an incorrect score is detected by word embedding, backpropagation learning is used to propagate backward all the nodes of the graph by the amount of the error. GKEEP leverages the benefits of word embedding and error-feedback propagation to undertake the above challenges of KE from geoscience texts, which contain ambiguities due to their lexical variations, free-text narrative style, arbitrary word ordering, and descriptive diversity. The results of our experiments show how backpropagation can be utilized on graph-based keyword extraction methods to produce better-quality keywords. To the best of our knowledge, this work is the first to use an enhanced graph-based approach to mine geoscience keywords from geoscience reports written in the Chinese language.

Our contributions can be summarized as follows.

(1) The proposed method incorporates word embedding to capture the dependency structure as well as the data distribution, and it computes semantic relations to solve the content sparsity problem.

(2) GKEEP utilizes error feedback to boost the most salient terms that graph-based approaches deem less important and designs of a word scoring approach that aims to capture semantic connectivity of the words in the text.

(3) We conduct a set of experiments to verify the effectiveness of the proposed method on two available manually constructed data sets. The experimental results demonstrate that our algorithm outperforms the state-of-the-art methods and obtains better-quality keywords. Although this paper focuses on the geoscience domain, our approach could easily be extended and adapted to a wider variety of fields such as medicine and chemistry.
The remainder of the paper is organized as follows. In Section 2, some related work about KE methods is reviewed. Our proposed GKEEP method is described in Section 3. Experimental results are presented in Section 4. Section 5 presents a discussion, and Section 6 provides conclusions and discusses future work.

2. Literature Review

Highlighting these keywords within the body of a document increases the browsing speed of readers. In this section, we briefly review the literature concerning KE methods. Recent work has categorized them as either supervised or supervised KE approaches.

2.1. Supervised KE

Supervised approaches for keywords extraction make full use of training data sets consist of texts and their corresponding keywords. KEA (Frank et al., 1999) applied a Naïve Bayes classifier developed from two features, which are extracted from phrases in the documents: the TF-IDF and the relative position about the phrase.

Additional innovative supervised methods have been proposed in recent years, ranging from conditional random fields (CRF; C. Zhang, 2008) to neural networks (Sarkar et al., 2010). C. Zhang (2008) proposed a CRF model whose string labeling is used over 20 phrase features. A framework used a maximum-entropy classifier was proposed, it can extract keywords from meeting transcripts (F. Liu et al., 2011). In addition, various ML-based approaches have been constructed to automatically acquire knowledge from training data. In particular, these approaches are used to develop classification models (Onan et al., 2016; Yang et al., 2016). These models are first trained using a training set with labeled tags; then, the model is validated using a test set. In recent years, DL methods have exhibited surprising abilities to learn features and have achieved state-of-the-art performances on NLP tasks in several fields (Chung et al., 2014; Goodfellow et al., 2016; Santos et al., 2018). Convolutional neural networks and recurrent neural networks have been widely applied to NLP tasks and have achieved remarkable results in many fields (Guo et al., 2017; Palafox et al., 2017; H. Zhang et al., 2017).

Supervised keyword extraction approaches, as the current state-of-the-art, still require manually annotated training data sets, which are rather challenging and time consuming.

2.2. Unsupervised KE

Labeled training data are not needed when we use unsupervised approaches for KE. Some of approaches are based on term statistics, which assess the terms’ degrees of importance to the text or collection according to the texts are found (even though lots of these methods require a document corpus for their calculations, because there is no “labeled” corpus available, they are not considered supervised).

Statistics-based approaches, such as n-gram statistics, word frequency, and TF-IDF (Salton et al., 1974) are domain independent and do not require a large amount of labeled data; these methods use statistical techniques to extract keywords (Beliga et al., 2015; Chen & Xiao, 2016; Zhu & Hua, 2017). Niu et al. (2014) applied high-frequency keywords to extract key information from geoscience and environmental documents. Su et al. (2014) selected high-frequency keywords to analyze research hotspots and develop domain trends. Chen et al. (2015) used the active index (AI) to demonstrate whether a country/region has comparative advantages based on its share of total world publication. Hu et al. (2018) presented a domain-keyword analysis approach using a Google Word2Vec model by extending the term frequency-keyword AI. The experimental results showed that the proposed method resulted in more effective analysis of domain knowledge.

Graph-based models generally view a document as a graph, whose vertices denote words and edges denote the relationship between two words constrained by the window size. TextRank is an unsupervised algorithm that uses only the information in the document itself for KE and in lots studies TextRank has been used to improve the graph-based ranking algorithm (Figueroa et al., 2017). Rose et al. (2012) proposed a Rapid Automatic KE algorithm, stop words and delimiters are used to divide a text into some candidate keywords. This algorithm has better performance than TextRank. Wang et al. (2015) used pretrained word em-
beddings as an external knowledge to improve TextRank and extract keywords from scientific publications. The experiments show that training the word embeddings improves the algorithm's performance compared to the original TextRank algorithm.

However, these models have a common drawback. To extract high-quality keywords depends on whether the total document set composition is the same domains. For another one, graph-based models build a graph to extract keyword candidates, and subsequently these candidates are ranked to select the real keywords from a single document.

### 2.3. Motivations and Limitations of the Study

There are considerable research efforts, both inside and outside the geoscience domain, have been undertaken to extract keywords from unstructured text (Gupta et al., 2018; K. Liu & El-Gohary, 2017; C. Wang et al., 2018). Despite their achievements, the existing KE approaches are still limited in their ability to automatically recognize keywords in the geoscience domain with highly heterogeneous and complex text. Hence, the main motivation of this research is to empirically enhance the learning methods. Two primary knowledge gaps have to be settled urgently.

First of all, when extracting keywords from highly heterogeneous and complex text, there is a lack of KE methods that can both reduce manpower input and achieve high performance. Most of the existing KE methods are ruled-based methods (Li et al., 2016; J. Zhang & El-Gohary, 2016; Zhou & El-Gohary, 2017) or supervised ML-based methods (G. Wang et al., 2014; Xia et al., 2011). These approaches can address highly heterogeneous and complex text through a large number of representative samples, but large manual efforts are required to construct rules or annotate training data. In essence, it is extremely challenging and time consuming to build patterns (for rule-based KE) or to tag examples (for the supervised ML-based KE) from geoscience reports.

Second, semantic approaches to support automated KE are lacking. A body of research exists that has explored the use of semantics in addressing various NLP tasks. Formally and explicitly defined semantics can facilitate the recognition and extraction of keywords. Utilizing semantics to assist automated KE is therefore highly important in our study, given the heterogeneous and complex text in geoscience documents. However, semantics have not been well explored in terms of rule-based or ML-based KE.

Starting from this observation and knowledge gap, this paper focuses on a graph-based KE algorithm with error-feedback propagation for use with geoscience reports. The GKEEP algorithm takes advantage of the graphics-based KE by considering the structure and content of a single document. Backpropagation is then used to improve its effectiveness, thereby increasing the score for terms/phrases that are more likely to be relevant and decreasing the score for those that are less likely.

### 3. The Proposed GKEEP Model

To extract keywords from geoscience reports, we begin by analyzing the ways that terms of rocks, structures, and geological time are used. Figure 1 demonstrates an example related to the geological structure. Such as "Triassic," "Rock types," and "Lithology" are keyword terms that illustrate the source document, which refers to rock and geological structures. Other terms such as "Stripes" and "Gray white" in the reports reveal the descriptive aspects of these keywords. By obtaining the keyword terms, it can be inferred that this report focuses on the geological structure.

In this section, we introduce the GKEEP and elaborate on components of it. The overall KE process of GKEEP is graphically presented in Figure 2, and there is a detailed introduction in Algorithm 1. As shown in Figure 2, the KE process of the GKEEP algorithm for an input text is performed in four phases. First, the input text is cleaned, and the graph of the input text's words is constructed. Second, each node is ranked according to the graph structure and ranking algorithm. In the third step, each node is measured and the error-detection method is used to detect whether its rank is higher or lower than it should have. In this step, the basic idea is to identify inaccurate scores and assign expected scores for the nodes in the graph with Word2Vec. In the fourth step, the errors detected in the previous step are applied to adjust the weight of every edge. Finally, a list of candidate keywords is extracted from the input reports for which the ultimate
The stratum is mainly exposed in Qieqiaqu area and its rock types are complex. It is mainly composed of a suite of gneiss, schist, quartzite and marble rock assemblages. The lower lithology is biotite plagioclase gneiss, garnet-bearing biotite plagioclase gneiss, diopside-bearing biotite plagioclase gneiss, streaked dimica longer gneiss with lime white sugar granulated marble and deep lime-green amphibole. A small amount of grey biotite plagioclase gneiss is found in the intrusive rocks of Late Triassic and Late Jurassic in Chayong area.

Figure 1. An example of a geological document segment analyzed by the GKEEP model. Rock types, Lithology, Late Triassic, and Late Jurassic are considered to be manually assigned keywords. GKEEP, graph-based keyword extractor with error-feedback propagation.

scores are calculated. Keywords are ranked by their scores; hence, the top \( N \) candidate keywords are selected as the core and meaningful keywords.

**Algorithm 1.** The pseudo-code of GKEEP.

**Input:** A text document

**Output:** A set of keywords

1. **Step 1:** preprocess and create the graph(text) \( G=(V,E) \) (Section 3.1)
2. **Step 2:** rank each node (Section 3.2)
   
   \[
   \text{pairs} \leftarrow \text{rank	extunderscore nodes}(G) \quad \triangleright \text{TextRank Only}
   \]

3. **Step 3:** detect the error nodes(pairs) (Section 3.3)
   
   for each pair \( w_i, w_j \in V \)
   
   \[
   S_t \leftarrow \text{the score detected by Word2Vec algorithm}
   \]
   
   \[
   S_v \leftarrow \text{the score detected by TextRank algorithm}
   \]
   
   \[
   \text{Calculate expected node scores}
   \]
   
   if \( S_v > S_t \) then
   
   \[
   T_j \leftarrow \text{max}(S_{\text{mid}})
   \quad \triangleright \text{Equation 5}
   \]
   
   else
   
   \[
   T_i \leftarrow \text{min}(S_{\text{mid}})
   \quad \triangleright \text{Equation 4}
   \]
   
   end if

4. **Step 4:** Error feedback and rank nodes (Sections 3.4 and 3.5)
   
   repeat
   
   for each node \( V_j \in G \)
   
   compute \( \delta_j \) \quad \triangleright \text{Equation 13}
   
   end for
   
   for each edge \( w_{ij} \in G \)
   
   compute \( \Delta w_{ij} \)
   
   \[
   \Delta w_{ij} \leftarrow \text{denormalize}(w_{ij})
   \]
   
   \[
   w_{ij} \leftarrow w_{ij} + \Delta w_{ij}
   \quad \triangleright \text{Equation 21}
   \]
   
   end for
   
   rank	extunderscore nodes\((G)\)

   until \( \Delta E < \varepsilon \)
In contrast to the most recent graph-based KE approaches, GKEEP uses Word2Vec to enhance graph-based KE algorithms with backpropagation learning. Thus, while most of the important and relevant keywords are falsely assigned a low score during the initial KE process, the semantic information from Word2Vec is properly balanced through the error-feedback propagation.

The following subsections describe the proposed GKEEP and its core phases in detail.

### 3.1. Graph Construction

In the graph-based KE method, text units are first identified and added to the graph as nodes. Then, some meaningful relationships between text units are identified and added as edges between nodes. The resulting graph is used to calculate the importance of each node relative to all the others. Similarly, the score of a particular node affects the score of the node to which it is connected.

| Table 1 | Preprocessed Sample Text From the Collection Corpus, and Its Manually Assigned Keywords |
|---------|----------------------------------------------------------------------------------|
| Original text | The formation is mainly exposed in niechaqu area, and its rock types are also complex. It is mainly composed of a set of gneiss, schist, quartzite and marble. It is composed of upper and lower lithology, which cannot be represented on the map. In chayong area of the survey area, only a small amount of biotite plagioclase gneiss is distributed in late Triassic and late Jurassic intrusive bodies. |
| Manually assigned keywords | niechaqu area, rock types, biotite plagioclase gneiss, late Triassic and late Jurassic |

Note. Keywords shown in bold are those actually appear in the text.
3.1.1. Preprocessing

Regional geological reports are known to be very noisy for mining tasks because they include some symbols that do not contain any useful information and make further processing ineffective. Therefore, an effective preprocessing phase is needed to remove meaningless symbols and allow more effective keywords to be extracted. There are some preprocessing differences between Chinese and English texts. Unlike English, Chinese has consecutive characters that are not delimited by word boundaries. The goal of text preprocessing is to convert free raw text into a format, which is more conducive to further analysis. The steps for preprocessing are as follows in this research according to the standard procedure: text cleaning, sentence splitting, tokenization, and stop-word removal. Text cleaning is applied to eliminate tables of results, mathematical formulas, author affiliations, and other noisy lines. Sentence splitting marked text into grammatically meaningful sentences based on sentence boundaries such as periods and question marks. Tokenization segments the continuous raw text into semantically and syntactically meaningful units—words, punctuation, digits, and whitespace. Stop-word removal removes function words and some common words using a standard list of stop words and then eliminates words from the set using a word-matching method.

Table 1 demonstrates a sample text, which is selected from the collection corpus after processing, along with manually assigned (real) keywords.

3.1.2. Graph Construction: TextRank

TextRank technique can be easily configured based according to different applications. In this research, we will look at the configuration used in the experiment to get the best results.

The size of the text added as a node in the graph is n-grams and the size is 1–4. The cooccurrence (i.e., the maximum distance in words) between two text units is used to represent the relation between the nodes. In our research, 2 is applied as a cooccurrence value, which represents that an edge is developed between two nodes if they cooccur within a window of two words, there is an exception when there is a full stop. Since the manner in which GKEEP will later apply backpropagation to the text graph, the weighted and undirected edges are necessary to owe, which are all initialized with a weight of 1. Figure 3 shows the graph-constructed result of TextRank from the selected sample text.

3.2. Node Ranking

TextRank uses a recursive algorithm on the graph, iteratively assigning points to each node until convergence is achieved. In short, the score of a node in the graph affects all scores associated with that node, and vice versa.

For the recursive algorithm, all nodes in the graph are initialized with a score of 1, and all edge weights are set to 1. For each node $V_i$ in graph $G$, $\text{In}(V_i)$ denotes the set of incoming nodes (those that point to it), and
Out($V_i$) denotes the set of outgoing nodes (those that $V_i$ points to). Using TextRank, the rank/relevance of node $V_i$ is calculated by Equation 1 as shown below:

$$S(V_i) = (1 - d) + d \sum_{V_j \in \text{in}(V_i)} \frac{w_{jk}}{\sum_{V_k \in \text{out}(V_j)}} \cdot S(V_j)$$

(1)

Here, $d$ represents a damping factor (usually set to 0.85), which represents the probability of jumping from one node to the next node. Additionally, $(1 - d)$ is the probability of transference to a new node, and $w_{ij}$ is the weight from $V_i$ to $V_j$ represents the edge. This is a recursive ranking algorithm applied to the graph until it converges to within a set threshold (set to 0.005 in these experiments).

3.3. Error Detection

The enhanced graph-based representation step detects whether a node has been assigned the correct score in each individual sentence. However, the common or frequently used terms often have low term frequency, which means that they are assigned a low score, which is sometimes a signal that a word is a key term. Inspired by feedback propagation in neural networks, which often propagate backward, this step is designed by considering the error and the responsible unit between term frequencies. In our work, which aims to recognize and extract keywords at the topic aspect level from each individual geoscience report, leads to the idea that KE would be improved if the reason for an error could be assessed. We noticed that low-frequency words usually contribute much more. Therefore, our goal is to build an error-feedback mechanism to enhance KE.

The next step in the GKEEP model is to detect and identify nodes with inaccurate scores and then assign them the expected score. Many graph-based methods obtain inappropriate scores when using only word frequencies as the measure of a term’s importance. From the constructed graph, GKEEP detects nodes with inappropriate scores (i.e., that were assigned a higher or lower score than they deserve) and then assigns them more accurate scores. To address the lack of domain-specific synonyms resulting from the traditional use of word frequencies, we select the Word2Vec model (Mikolov, Chen, et al., 2013; Mikolov, Sutskever, et al., 2013; Mikolov, Yih, et al., 2013). In this subsection, we used two different measures in our experiment: TF-IDF and Word2Vec.

3.3.1. TF-IDF

TF-IDF represents the importance of the phrase or word to the input document in the collection, and the higher the value, the higher the correlation. Since our test corpus is composed of reports and abstract of research papers belonging to a specific field, TF-IDF measure is a sufficient and simple measure for our backpropagation algorithm. Given each node $V_i$ in graph $G$ (i.e., each phrase or word in an input document), we compute its TF-IDF. The TF-IDF measure for phrase $P$ in a document is calculated as follows:

$$\text{TF-IDF}(P) = \text{TF}(P) \cdot \text{IDF}(P) = \frac{\text{frequency}(P)}{\text{wc}(D)} \cdot \log \frac{|C|}{\text{df}(P)}$$

(2)

where frequency($P$) denotes the number of times $P$ occurs in $D$, wc($D$) denotes the number of words in $D$, $|C|$ denotes the number of documents in document collection $C$, and df($P$) denotes the number of documents in $C$ that include $P$.

In the KE task, since the document being processed is no longer a traditional text document, TF-IDF cannot provide accurate domain-specific representing results. Therefore, in our example, some words are more likely to appear in both the target document and the background document than in a document that consists of a list of keywords.
3.3.2. Word2Vec

The Word2Vec model is based on the corpus but extends the n-gram language model to help determine the semantic distance between two words without any monitoring information. Therefore, we chose the Word2Vec model to dynamically model the semantics behind low-frequency keywords to reduce the impact of the synonym problem.

Word2Vec (Hu et al., 2018; Mikolov, Chen, et al., 2013) model is a model based on shallow neural network, which is used for word embedding. The model uses a large number of text corpora for training and aims to create a unique vector for each word in a high-dimensional space. More recently, Word2Vec has applied two constructs to create word vector representations: CBOW and skip-gram. The continuous word package USES the surrounding context words to predict the words, while the hops framework predicts the surrounding words in a fixed-size window around the current word. This model works better for uncommon words. According to the characteristic of jump graph structure, the model of jump graph is trained and applied. A corpus of 14 earth science reports (about 1.4 million words) was used to train the model. For each word, calculate the value of its word vector and use it as a word vector.

The built-in Word2Vec calculation treats each text unit as one vertex of the graph and calculates the similarity distribution between text units based on the edges between the vertices. Finally, in the integration of TextRank and Word2Vec, the semantic similarity is used to calculate the edge values between words. Where, in order to replace formula 1 in TextRank, the semantic similarity calculated by Word2Vec distance is used to reset the weight value \( w_{ji} \).

3.3.3. Calculation of Expected Node Scores

After identifying errors in the nodes of graph \( G \), the next step is to calculate the expected scores for each node. Because there are three possibilities for nodes in the graph—accurate scores and two types of inaccurate scores (lower scores than they deserve and higher scores than they deserve)—to obtain the expected scores in the graph \( G \), one of the above-described approaches must be chosen (henceforth referred to as metric \( \mu \)) and computed for each node. Graph \( G \) is split into three mutually disjoint collections: \( G_{\text{low}} \), \( G_{\text{mid}} \), and \( G_{\text{high}} \).

\( \mu \) denotes the mean values in \( G \), and \( \sigma \) denotes the standard deviation in \( G \). The notion \( \mu(V_i) \) is defined based on the metric value of node \( V_i \). The three disjoint subjects, \( G_{\text{low}} \), \( G_{\text{mid}} \), and \( G_{\text{high}} \), are shown in Equation 3:

```
| Node                          | Word2Vec | TF-IDF | TextRank | Expected scores | Direction |
|-------------------------------|----------|--------|----------|-----------------|-----------|
| \( G_{\text{low}} \)          |          |        |          | 0.91            | ↓         |
| Amphibolite facies            | 0.22     | 0.16   | 0.91     | 0.91            | ↓         |
| Undivided                     | 0.15     | 0.98   | 2.11     | 0.91            | ↓         |
| Contact                       | 0.11     | 1.28   | 2.08     | 0.91            | ↓         |
| Near                          | 0.08     | 1.21   | 1.89     | 0.91            | ↓         |
| Boundary of a country         | 0.06     | 0.18   | 1.06     | 0.91            | ↓         |
| Archaean                      | 0.39     | 0.18   | 1.07     | 0.91            | ↓         |
| \( G_{\text{mid}} \)          |          |        |          | 0.91            | ↓         |
| Five Supergroups              | 0.81     | 0.03   | 1.05     | ∅               | –         |
| Jingangku Formation           | 0.80     | 0.02   | 1.00     | ∅               | –         |
| \( G_{\text{high}} \)         |          |        |          | 3.85            | ↑         |
| Biotite amphibole plagioclase gneiss | 1.50 | 0.05   | 3.85     | 3.85            | ↑         |
| Gneiss                        | 1.20     | 0.12   | 0.98     | 3.85            | ↑         |
| Regional metamorphism         | 1.13     | 0.19   | 1.91     | 3.85            | ↑         |
| Quartzite                     | 1.31     | 0.04   | 2.78     | 3.85            | ↑         |
```
If the standard deviation is less than $\mu$, the $\lambda$ will be defined with a lower bound of the standard deviation, and if the standard deviation is greater than $\mu$, the $u$ will be defined with an upper bound of the standard deviation.

The nodes in $G_{low}$ that have relatively low metric value would be expected to have lower assigned scores than other nodes. $S_{mid}$ denotes the set of node scores in $G_{mid}$. $T_i$ denotes the expected score in $G_{low}$ and is defined as shown in Equation 4:

$$T_i = \begin{cases} \min(S_{mid}) & \text{if } S(V_i) \geq \min(S_{mid}) \\ \emptyset & \text{if } S(V_i) < \min(S_{mid}) \end{cases}$$

(4)

Analogously, nodes from $G_{high}$ are expected to have higher scores than other nodes. The expected score $T_i$ for every node $V_i$ in $G_{high}$ is defined as shown in Equation 5:

$$T_j = \begin{cases} \max(S_{mid}) & \text{if } S(V_i) < \max(S_{mid}) \\ \emptyset & \text{if } S(V_i) \geq \max(S_{mid}) \end{cases}$$

(5)

Because terms/phrases in $G_{mid}$ are in the middle range of metric values, their importance is unlikely to increase or decrease; their expected scores are then left empty (set to $\emptyset$).

Table 2 shows the three subsets obtained from graph $G$. For brevity, only the subsets obtained using KE’s TextRank algorithm and Word2Vec as error-detection methods are shown. It can be observed that some of the general terms from Table 2, the terms and conditions in the absence of Wefen, Jiechu and Linjin, obtain high TextRank scores, while its Word2Vec values indicate that they have lower ranking. By contrast, domain-specific terms such as pianmayan and shiyingyan—which are actual keywords—obtained lower TF-IDF and TextRank scores than they deserved. It is worth noting that this metric (in this case, Word2Vec) can be replaced by another metric, such as clustered values. More importantly, a supervised approach can be used to obtain the ranking target, assuming that the output is a probability score and a training data set is available.

### 3.4. Error Feedback

Once achieving the expected scores in the graph, the next step is to use the error-feedback strategy to update the edge weights iteratively. For convenience, $A_i$ denotes $S(V_i)$, and $T_i$ denotes the expected score of $V_i$.

For further clarity, we change index $i$ and index $j$ in the TextRank score formula (in Equation 1); we can use $w_{ij}$ to represent the edge weights in the sum molecule. Then, we can obtain

$$A_j = S(V_j) = (1-d) + d \cdot \sum_{V_j \in \text{Out}(V_i)} w'_{ij} A_i$$

(6)

For convenience, we use $w'_{ij}$ to denote the normalized weight of $w_{ij}$ in Equation 1. In addition, the notation $w'_{ij}$ is set as follows:

$$w'_{ij} = \frac{w_{ij}}{\sum_{V_k \in \text{Out}(V_i)} V_k}$$

(7)

Many loss functions can be used in the error-feedback phase, including $L_1$-loss, $L_2$-loss, and log loss. Motivated by Figueroa et al. (2017), we select $L_2$-loss and cross-entropy loss as the loss functions.
The $L_2$-loss is defined as follows:

$$E_1 = \frac{1}{2} \sum_j (T_j - A_j)^2$$

(8)

where the factor of $\frac{1}{2}$ is used to cancel the exponent after differentiating.

The cross-entropy loss is defined as

$$E_2 = -\frac{1}{n} \sum_j \left( T_j \ln A_j + (1 - T_j) \ln (1 - A_j) \right)$$

(9)

In this stage, the goal is to minimize the error function. To reach this goal, gradient descent (Rumelhart et al., 1985) is used. During gradient descent, the modification value $-\Delta w'_ij$ is iteratively computed and applied to update each normalized edge weight.

$$\Delta w'_ij = -\eta \frac{\partial E}{\partial w'_ij}$$

(10)

where $\eta$ denotes the learning rate, which is used to control the rate of gradient descent. Usually, the learning rate ranges from 0 to 1.

The chain rule is used to compute the derivative as follows:

$$\frac{\partial E}{\partial w'_ij} = \frac{\partial E}{\partial A_j} \frac{\partial A_j}{\partial w'_ij}$$

(11)

Using the Figueroa et al. (2017) approach, when $E_1$ is selected as the loss function, we obtain

$$\Delta w'_ij = -\eta \frac{\partial E_1}{\partial w'_ij} = -\eta \left( -\delta_j \cdot dA_j \right) = \eta \cdot \delta_j \cdot dA_j$$

(12)

in which

$$\delta_j = \begin{cases} T_j - A_j & \text{if } T_j \neq \emptyset \\ d \sum_{k \in \partial \ Marines(V_j)} w'_{ik} & \text{if } T_j = \emptyset \end{cases}$$

(13)

Then, the next step is to calculate $E_2$. Let

$$\frac{\partial E_2}{\partial A_j} = -\delta_k$$

(14)

Then, by combining Equations 1 and 11, we obtain

$$\frac{\partial E_2}{\partial A_j} = \frac{\partial}{\partial w'_ij} \left( 1 - d \right) + d \cdot \sum_{k \in \partial \ Marines(V_j)} w'_{ik} \frac{\partial A_k}{\partial w'_ij} = dA_k$$

(15)

When minimizing the cross-entropy loss $\frac{\partial E_2}{\partial A_j}$, there are two possible cases. When $V_j$ has an expected score, we compute it as follows:

$$\frac{\partial E_2}{\partial A_j} = \frac{\partial}{\partial A_j} \left( -\frac{1}{n} \sum_j \left( T_j \ln (A_j) + (1 - T_j) \ln (1 - A_j) \right) \right)$$

(16)
\[
\delta_j = \frac{1 - T_j}{1 - A_j} - \frac{T_j}{A_j}
\] (17)

and let
\[
\delta_j = \frac{1 - T_j}{1 - A_j} - \frac{T_j}{A_j}
\] (18)

When \(V_j\) does not have an expected score, we calculate
\[
\frac{\partial E_2}{\partial W_{ij}} = \frac{\partial E_2}{\partial A_j} \frac{\partial A_j}{\partial W_{ij}} = \left(-d\right) \sum_{V_k \in \text{Out}(V_j)} \delta_k w'_{ij} dA_k = -\delta_j \cdot dA_j
\] (19)
in which
\[
\delta_j = d \sum_{V_k \in \text{Out}(V_j)} \delta_k w'_{ij}
\] (20)

We obtain the final modification value:
\[
\Delta w'_{ij} = -\eta \frac{\partial E_2}{\partial W_{ij}} = -\eta \left(-\delta_j \cdot dA_j \right) = \eta \cdot \delta_j \cdot dA_j
\] (21)

where
\[
\delta_j = \begin{cases} 
\frac{1 - T_j}{1 - A_j} - \frac{T_j}{A_j} & \text{if } T_j \neq \emptyset \\
\frac{d \sum_{V_k \in \text{Out}(V_j)} \delta_k w'_{ij}}{V_k \in \text{Out}(V_j)} & \text{if } T_j = \emptyset
\end{cases}
\] (22)

In Equations 13 and 22, \(\delta_j\) denotes the difference between the current node score \(A_j\) and the expected node score \(T_j\).
Finally, the proposed algorithm converges when the difference between the current node scores $A_j$ and the expected node scores $T_j$ falls below a threshold $\epsilon$ (set to 0.01 in our experiments).

### 3.5. Candidate KE, Scoring, and Ranking

When the error-feedback method converges, the nodes will be given new and more accurate scores. The keywords were then ranked in descending order of their scores. Arrange keywords in descending order and select them. The number of keywords can be selected at will (e.g., 3, 5, 7), or as a percentage of the total number of words in the text. Finally, there are many strategies that combine a keyword’s score and obtain the final score for each candidate keyword. In GKEEP, the final score for a candidate keyword is a new, updated, and more accurate score. Then, the extracted candidate keywords will be sorted according to the final scores, and $N$ best keywords will be returned as the ultimate keywords. Table 3 illustrates the keyword list after executing the entire GKEEP algorithm. For each keyword, the table shows the original TextRank score, the final GKEEP score, and the direction in which the original score was taken. As you can see from this example, the original ranking (TextRank Score) and error-detection method (Word2Vec) play an irreplaceable role in the final ranking obtained by backpropagation of the graph application.

### 3.6. Computational Complexity

The complexity of the first step of GKEEP is $O(\log_2 V)$, where $V$ is the number of words in the corpus. In its second step, the complexity is $O(\max(V^2, WL))$, where $V$ denotes the number of vertices of the graph, $W$ denotes the window size, and $L$ denotes the number of words in the input reports. In the last step of GKEEP, in which the initial set of candidates is acquired from the input text, the complexity is $O(L)$.

Considering the all above analysis, the overall computational complexity of GKEEP is $O(\max(V^2, WL, L))$, which in practice, translates to $O(V^2)$ in real-world applications because the other terms are smaller than $V^2$.

### 4. Experimental Results and Evaluations

#### 4.1. Experiment Setup

##### 4.1.1. Data Sets and Evaluation Metrics

In the experiments, we collected two data sets for use. The first data sets collect abstracts and keywords of journal articles from China National Knowledge Infrastructure (http://cnki.net/). These journals are geoscience journal papers published from 2012 to 2017 described in this section, two data sets are constructed and used. This data set contains 5,500 abstracts and is denoted by GEOA.

The second data set used in this study is a collection of 17 regional geological reports from the National Geological Archives of China (http://www.ngac.org.cn/). These reports include valuable and representative thematic information on stratigraphic distribution, lithology, tectonics, tectonics, minerals, rocks, and geological attitudes of various types. These reports were selected because they are representative: (1) regional geological information recorded by different authors over different years; (2) from different regions; and (3) to present the complexity of a specific field, the text varies from simple to complex. We represent this data set as RGDS.

The data sets were divided into tenfolds, and we use tenfold cross-validation for the proposed model. Six workers with rich knowledge of both geosciences and KE read each sentence and then annotated all the keywords. A voting mechanism was applied to solve any disagreements. Each geological report includes more than 10 chapters and more than 100,000 Chinese characters. Finally, after applying these criteria, we obtained a corpus of 5,000 documents.
We conducted statistical analyses in our experiments using these data collections. Table 4 shows these statistics for each collection.

The precision, recall, and $F_1$-score are applied as performance indicators, which are calculated after lemmatizing both the extracted and the real keywords. Let $E_D$ be the set of keywords extracted from document $D$, and $KEY_D$ be the set of manually assigned keywords; the precision, recall, and $F_1$-score are calculated as follows:

\[
\text{precision} = \frac{|E_D \cap KEY_D|}{|E_D|},
\]

\[
\text{recall} = \frac{|E_D \cap KEY_D|}{|KEY_D|},
\]

\[
F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.
\]

**4.1.2. Baselines**

To better illustrate the performance of the model, we compared it with the following generic baseline $KE$ algorithms.

**TF-IDF** (Salton et al., 1974) is calculated based on the term and inverse document frequencies in the given documents.

**NB** (Naïve Bayes algorithm) (Han & Kamber, 2006) is based on the class conditional independence assumption, which reduces the required computational cost.

**SVM** (Support Vector Machine) (Joachims, 1998) uses the nonlinear matching method to transform the original data set into the high-dimensional data set. In this dimension, a hyperplane that can properly partition the data is checked. Hyperplanes are decision boundaries that divide data into classes.

**RF** (random forest algorithm) (Breiman, 2001) is an ensemble of classification and regression trees induced from bootstrap samples of the training data.

**SingleRank** (Wan & Xiao, 2008) is a novel method to collaboratively perform single-document keyphrase extraction by making use of mutual influences of multiple documents within a cluster context.

**TopicRank** (Bougouin et al., 2013) is a graph-based keyphrase extraction method that relies on a topical representation of the document. Candidate keyphrases are clustered into topics and used as vertices in a complete graph. A graph-based ranking model is applied to assign a significance score to each topic. Keyphrases are then generated by selecting a candidate from each of the top-ranked topics.

**TopicalPageRank** (Sterckx et al., 2015) incorporates topical information from topic models and then computes a single PageRank for each text regardless of the number of topics in the model.

**PositionRank** (Florescu & Caragea, 2017) is an unsupervised graph-based model that incorporates information from all positions of a word’s occurrences into a biased PageRank to score keywords that are later used to score and rank keyphrases in research papers.

**MultipartiteRank** (Boudin, 2018) is an unsupervised keyphrase extraction model that encodes topical information within a multipartite graph structure.

**RankUp** (Figueroa et al., 2017) is an unsupervised method that applies TF-IDF to identify inaccurate scores of enhanced graph-based keyphrase extraction approaches. It includes an error-feedback mechanism that iteratively applies a chain effect to all nodes.

| Table 5 | The Impact of $\lambda$ and $u$ on GKEEP's Performance on the GEOA Data Set |
|---------|----------------------------------|
| $F_1$-measure | $\lambda = 0.2$ | $\lambda = 0.4$ | $\lambda = 0.6$ | $\lambda = 0.8$ | $\lambda = 1.0$ |
| $u = 0.2$ | 23.99 | 29.30 | 29.15 | 28.02 | 27.00 |
| $u = 0.4$ | 24.22 | 28.98 | 28.15 | 28.44 | 27.48 |
| $u = 0.6$ | 26.78 | **30.75** | 30.00 | 29.78 | 28.06 |
| $u = 0.8$ | 26.87 | 29.45 | 29.05 | 28.45 | 28.00 |
| $u = 1.0$ | 25.48 | 28.87 | 27.44 | 26.98 | 26.77 |

**Note.** The bold represents the highest performance.

| Table 6 | The Impact of $\lambda$ and $u$ on GKEEP's Performance on the RGDS Data Set |
|---------|----------------------------------|
| $F_1$-measure | $\lambda = 0.2$ | $\lambda = 0.4$ | $\lambda = 0.6$ | $\lambda = 0.8$ | $\lambda = 1.0$ |
| $u = 0.2$ | 30.15 | 31.44 | 31.58 | 30.25 | 29.59 |
| $u = 0.4$ | 31.42 | 31.46 | 31.49 | 31.52 | 31.56 |
| $u = 0.6$ | 33.00 | 34.78 | 31.65 | 31.69 | 31.72 |
| $u = 0.8$ | 33.04 | **34.82** | 31.69 | 31.73 | 31.76 |
| $u = 1.0$ | 33.20 | 34.28 | 31.86 | 31.89 | 31.92 |

**Note.** The bold represents the highest performance.
EnhancedWord2Vec (Qiu et al., 2019) is an ontology and enhanced word embedding-based methodology for automatic KE.

4.2. Parameter Setting

To obtain desired keywords, GKEEP offers four effective parameters in the process of keyword extraction. Appropriate setting of these important parameters could lead to meaningful and high-quality keywords. Four important parameters of GKEEP are as follows: (1) the weighting factors $\lambda$ and $u$, (2) vector dimensionality $d$, (3) context window $k$, and (4) the number of keywords extracted $n$. To find the best parameter settings, we first created a set of promising configurations based on our knowledge about the model. The best configuration was then selected and improved by varying parameter values one at-a-time.

In the rest of this section, we investigate the impact of GKEEP’s parameters on the performance of this method. To evaluate a certain parameter, the other parameters are set to their best values in the conducted experiments.

4.2.1. The Weighting Factors

As mentioned in Section 3.2.1, the Stdv lower bound $\lambda$ and Stdv upper bound $u$—a word importance indicator—is used to establish a balance between the importance of different words. We conducted a set of experiments to study how different weights could affect the KE performance and find an optimal weight that improves the extraction performance.

A total of 50 controlled experiments were conducted, in which the weight $\lambda$ was tested with linearly separated values between 0.2 and 1.0 with a step of 0.2, and the weight $u$ ranged from 0.2 to 1.0 with a step size of 0.2. Tables 5 and 6 represent the average accuracy of the experimental results of the GEO and RGDS data sets.
Figure 5. Impacts of context window on both the GEOA and the RGDS data sets.

Figure 6. Impacts of the number of keywords extracted in both the GEOA and the RGDS data sets.
As indicated in Table 5, increasing the weight increases the average performance. For the GEOA data set, a rational range ($\lambda = 0.4$ and $u = 0.6$) provides the best results (the algorithm achieved an average precision of 30.75%). The proposed algorithm obtains its best performance when the optimal weight is set between $\lambda = 0.4$ and $u = 0.8$ on the RGDS data set (the algorithm achieved the average precision of 34.82%). Our results show that raising and lowering keyword ratings after application of error feedback can elevate the most prominent keywords to the highest position, while demoting those words and phrases that are considered less important.

### 4.2.2. The Vector Dimensionality

We also conducted some experiments to verify the impacts of several key hyperparameters in GKEEP, that is, vector dimensionality $d$, context window $k$, and the number of keywords extracted $n$. We selected $P$, $R$, and $F$-measure to demonstrate the impacts of various hyperparameter values on the performance of GKEEP. GKEEP_1 denotes the proposed approach with cross-entropy loss, and GKEEP_2 denotes the proposed method with $L_2$-loss.

The impact of vector dimensionality on the performance of GKEEP (including GKEEP_1 and GKEEP_2) is demonstrated in Figure 4. As seen in Figure 6, increasing vector dimensionality provides performance improvements for keyword extractions from geosciences reports. For example, the proposed approaches achieved the least satisfactory performance with a 50-dimensional vector on the GEOA data set and increasing this value to 300 improved the $F_1$-measure by 5.2% and 4.2%. GKEEP achieves its best performance when vector dimensionality is 300 on the GEOA data set and when vector dimensionality is 200 on the RGDS data set. Thus, we set $d = 300$ and 400 for the GEOA and RGDS data sets, respectively.

### 4.2.3. The Context Window

Figure 5 presents the impacts of context window $k$ on both the GEOA and the RGDS data sets. We can observe that performance improves with $k$, as the high-context window can capture further relationships and achieve better performance. For example, when the context window is set to 2 on the GEOA data set, GKEEP achieves the least satisfactory performance, and increasing $k$ to 7 improved the $F_1$-measure by 31.78% and 30.19%. However, a larger $k$ requires longer training times as well. Thus, we set $k = 7$ to achieve a trade-off between KE performance and training times.

### 4.2.4. The Number of Extracted Keywords

The experimental results from Figure 6 suggest that the number of keywords extracted is important for determining the overall performance on both the GEOA and the RGDS data sets. As seen in Figure 6, when the number of extracted keywords varies from 3 to 5, the $F_1$-measure values increase because increasing the number of keywords leads to the extraction of more correct keywords that would help increase the $F_1$-measure value. However, the growth level of the number of correct keywords is less than the growth level of the number of extracted keywords. When the number of extracted keywords is 6 and 5 on the GEOA and RGDS data sets, respectively, the curve reaches the peak. Thus, we set $n = 6$ and 5 for GEOA and RGDS data sets, respectively.

### 4.3. Comparison With Other Methods

We conducted a total of 14 experiments using the GEOA and RGDS data sets described in Section 4.1.1 to evaluate GKEEP’s performance compared to other KE approaches. Taking the average number of manually annotated keywords in the two data sets into account, the top keyword count is set to $N = 5$ for each data set.

Comparing the results of GKEEP with 12 baseline technical methods, the following results are obtained: TF-IDF, three supervised methods (NB, SVM, RF and Word2Vec), TopicRank, SingleRank, TopicalPageRank, PositionRank, MultipartiteRank, RankUp, TextRank, and EnhancedWord2Vec. Tables 7 and 8 report the performance of GKEEP and the other 12 KE approaches in terms of the average precision, recall, and $F$-measure.

---

**Table 7**

Comparison of Different Supervised Methods With the Precision ($P$), Recall ($R$), and $F$-Measure ($F$) of the Extracted Keywords on the GEOA and RGDS Data Sets

| Methods  | GEOA ($P$) ($R$) ($F$) | RGDS ($P$) ($R$) ($F$) |
|----------|------------------------|------------------------|
| NB       | 31.8 32.6 32.20        | 36.9 36.45 36.67       |
| SVM      | 35.16 36.07 35.61      | 39.59 38.09 38.83      |
| RF       | 34.11 34.12 34.11      | 36.89 36.15 36.52      |
| Word2Vec | 21.60 19.70 20.61      | 24.1 23.5 23.8         |
| GKEEP_1  | 31.80 29.70 30.71      | 35.80 33.90 34.82      |
| GKEEP_2  | 30.20 28.90 29.54      | 33.60 32.00 32.78      |

*Note.* The best results are shown in bold.
Compared to the three supervised algorithms (NB, SVM, and RF) (Table 7), these KE methods obtained the highest predictive performance in all compared cases. For all of the compared cases, the SVM method outperforms the naïve Bayes algorithm and the random forest algorithm. Also, comparing the supervised-based and unsupervised-based approaches in Table 8 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect.

As can be seen from Table 8, different GKEEP implementations are always superior to their baseline methods in terms of accuracy, recall rate, and F1-measure. Again, our approach yields better performance than other seven graph-based approaches and is compared to the results in Table 7 reveals that the keywords are indeed more difficult to detect. These results indicate that when GEOA and RGDS data sets are used in geological domains, word embedding can better obtain semantic meaning.

---

**Table 8**

| Methods       | GEOA P (%) | GEOA R (%) | GEOA F (%) | RGDS P (%) | RGDS R (%) | RGDS F (%) |
|---------------|------------|------------|------------|------------|------------|------------|
| TF-IDF        | 20.00      | 14.10      | 16.54      | 21.80      | 20.30      | 21.02      |
| TopicRank     | 25.60      | 20.80      | 22.95      | 28.90      | 27.60      | 28.24      |
| SingleRank    | 12.20      | 12.50      | 12.35      | 20.10      | 19.80      | 19.95      |
| TopicalPageRank | 28.90    | 27.70      | 28.29      | 33.20      | 30.10      | 31.57      |
| PositionRank  | 22.50      | 20.80      | 21.62      | 25.10      | 24.10      | 24.59      |
| MultipartiteRank | 21.80   | 17.30      | 19.29      | 22.80      | 21.00      | 21.86      |
| RankUp        | 25.40      | 23.90      | 24.63      | 26.10      | 24.90      | 25.49      |
| TextRank      | 20.15      | 21.26      | 20.69      | 21.11      | 21.56      | 21.33      |
| EnhancedWord2Vec | 27.9    | 30.5       | 29.14      | 26.8       | 29.7       | 28.2       |
| GKEEP_1       | 31.80      | 29.70      | 30.71      | 35.80      | 33.90      | 34.82      |
| GKEEP_2       | 30.20      | 28.90      | 29.54*     | 33.60      | 32.00      | 32.78      |

Note: The best results are shown in bold.
Based on these experimental results, utilizing error feedback and Word2Vec can augment the process of promoting and demoting keyword scores and enhances the most salient terms, while relegating those words and phrases that are deemed less important to lower positions. The simple reason is that error feedback results in a chain effect that affects the terms connected to the promoted and demoted keywords, thus modifying the graph structure. Additionally, GKEEP could easily be extended to other graph-based KE algorithms as well as node scoring approaches if an expected score can be obtained.

5. Discussion

As shown in Tables 9 and 10, we detailed the extracted results by our proposed GKEEP model and the baselines. And, we compare our model with TextRank because our model is based on this algorithm. In Tables 9 and 10, the correctly extracted keywords are marked with purple and the incorrectly extracted keywords are marked with blue highlights. From the table below, we can draw the following conclusions. (1) All approaches will extract some incorrect keywords (marked with blue highlights), which further shows that...
keyword extraction is indeed a challenge. (2) The baseline approach TextRank extracted few and incorrect keywords, while the proposed GKEEP has the ability to extract richer, more comprehensive keywords. We attribute this to the integrated backpropagation mechanism. As shown in Table 9, some keywords (such as “Magmatism” and “Mesozoic”) are interrelated with each other. Thus, the GKEEP model can learn the relevance of these keywords and feedback the proposed method to extract useful keywords that can provide more information of the given document. This further demonstrates the validity of the our GKEEP model.

Traditional graph-based approaches use distributional semantics learned from word co-occurrences in the corpus context, and they rely on a large and high-quality corpus. Our proposed semantic approach, GKEEP, on the other hand, uses background knowledge maintained in the geoscience domain with word embeddings. For instance, Breccia, Basal Conglomerate, and Syngenetic Conglomerate are all part of Conglomerate; therefore, these words can be regarded as the same semantic unit. We found that our proposed approach had an advantage over previous graph-based approaches. This finding is likely because in traditional graph-based extraction, the common words score higher because they have more edges attached to them, which is a disadvantage of this approach. Similarly, rare or uncommon words scored lower because they had fewer connections. Our approach combines this concept with graphics-based keyword extraction because backpropagation can be applied to other types of networks. If any node score is found to be incorrect, all edge weights of the graph will be modified with the score and error to make the final result more accurate.

The traditional graph-based methods assign higher scores to these terms, which are more common or more frequently used since more edges point to them. Furthermore, our results show that after the application of error feedback, the increase and decrease of keyword scores can elevate the most prominent keywords to the highest position, while demoting those words and phrases that are considered less important. Error feedback can also have a knock-on effect, affecting terms associated with promotion and degradation keywords, thus modifying the structure of the entire diagram. Finally, GKEEP can easily adapt graph-based keyword extraction algorithms and other graphics-based scoring methods if the desired score of the node is known. This research has several limitations. The enhanced graph-based method has limitations because of the vagueness of the semantics and the corpus size. Existing semantic units might not have the same meanings, which directly affects the feedback and modification process; therefore, the granularity of the semantic units is difficult to determine. As shown in Figure 7, different numbers of keywords result in different performances on the GEOA and RGDS data sets. Hence, words with rankings lower than the threshold will be ignored in the analysis. And, our further work can attempt to design a self-adapting method for selecting the number of keywords.

6. Conclusions and Future Works

The keywords of a text consisting of a concise set of high-quality and meaningful words from the main content of that text. KE from geoscience texts is a critical and challenging task. Previous studies, especially graph-based approaches, suffer a potential drawback from their frequency-based analysis. They calculate ranking scores by frequency, which may lead to higher ranking scores for more common but less relevant terms. However, some less common but probably more relevant terms may obtain lower ranking scores. In this paper, we propose a KE model called GKEEP, which includes the following three stages: preprocessing, enhanced graph-based representation, and keyword ranking and output. The preprocessing phase involves unwanted noise removal, stop-word removal, and tokenization. The enhanced graph-based representation involves error detection and error feedback. At the error-detection step, the Word2Vec model is used to detect errors in the node weights in the graph. Then, this error feedback is used to update the edge weights iteratively. Finally, keywords are extracted using ranking scores. The proposed GKEEP model is compared with seven existing graph-based models. The experimental results show that the performance of the GKEEP model significantly improves the performance of the existing models. As far as we know, no other work has attempted to employ word embedding to aid the TextRank algorithm.

In the future, new weighting methods may be proposed that can be used in graph-based methods to find high-quality, meaningful keywords, and more semantically based approaches to KE can be explored using the proposed model to obtain better-quality results.
Conflicts of Interest

The authors declare no conflict of interest.

Data Availability Statement

All original data can be found in the Zenodo (https://zenodo.org/record/4312204#.X9Ay3zNn1so).

References

Beliga, S., Mestrovic, A., & Martincic-Ispic, S. (2015). An overview of graph-based keyword extraction methods and approaches. *Journal of Information and Organizational Sciences, 39*(1), 1–20.

Boudin, F. (2018). Unsupervised keyphrase extraction with multipartite graphs. *Advances in Neural Information Processing Systems, 31*(5), 4509–4519.

Bogunovic, A., Boudin, F., & Daille, B. (2013). TopicRank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing* (pp. 543–551).

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Campos, R., Mangaravite, V., Pasqualli, A., Jorge, A. M., Nunes, C., & Jatowt, A. (2020). YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences, 509*, 257–289.

Chen, G., & Xiao, L. (2016). Selecting publication keywords for domain analysis in bibliometrics: A comparison of three methods. *Journal of Information & Knowledge Management, 15*(1), 212–223. https://doi.org/10.1145/2932185

Chen, G., Xiao, L., Hu, C.-P., & Zhao, X.-Q. (2015). Identifying the research focus of Library and Information Science institutions in China with institution-specific keywords. *Scientometrics, 103*(2), 707–724. https://doi.org/10.1007/s11192-015-1545-2

Chung, J., Gulcehre, C., Cho, K. H., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *Eprint Arxiv.*

Duari, S., & Bhatnagar, V. (2019). sCAKE: Semantic Connectivity Aware Keyword Extraction. *Information Sciences, 100–117.*

Figueroa, G., Chen, P. C., & Chen, Y. S. (2017). Rankup: Enhancing graph-based keyphrase extraction methods with error-feedback propagation. *Computer Speech & Language, 47*, 112–131.

Florescu, C., & Caragea, C. (2017). PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Meeting of the association for computational linguistics* (pp. 1105–1115).

Frank, E., Paynter, G. W., & Witten, I. H. (1999). Domain-specific keyphrase extraction. *Paper presented at Proceedings of the International Joint Conference on Artificial Intelligence.*

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* Cambridge, MA: MIT Press.

Guo, L., Li, N., Jia, F., Lei, Y., & Lin, J. (2017). A recurrent neural network based health indicator for remaining useful life prediction of bearings. *Neurocomputing, 240*, 98–109. https://doi.org/10.1016/j.neucom.2017.02.045

Gupta, A., Banerjee, I., & Rubin, D. L. (2018). Automatic information extraction from unstructured mammography reports using distributed semantics. *Journal of Biomedical Informatics, 78*, 78. https://doi.org/10.1016/j.jbi.2017.12.016

Han, J., & Kamber, M. (2006). *Data mining: Concepts and techniques.* San Francisco, CA: Morgan Kaufmann Publishers.

Hsu, K., Wu, H., Qi, K., Yu, J., Yang, S., Yu, T., et al. (2018). A domain keyword analysis approach extending term frequency-keyword active index with Google Word2Vec model. *Scientometrics, 114*(1), 1–38. https://doi.org/10.1111/s11192-017-2574-9

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the tenth European conference on machine learning* (pp. 137–142).

Li, S., Cai, H., & Kamat, V. R. (2016). Integrating natural language processing and spatial reasoning for utility compliance checking. *Journal of Construction Engineering and Management, 142*(12), 04016074. https://doi.org/10.1061/(asce)co.1943-7862.0001199

Liu, F., Liu, F., & Liu, Y. (2011). A supervised framework for keyphrase extraction from meeting transcripts. *IEEE Transactions on Audio, Speech and Language Processing, 19*(3), 538–548. https://doi.org/10.1109/taslp.2010.2052119

Liu, K., & El-Gohary, N. (2017). Ontology-based semi-supervised conditional random fields for automated information extraction from bridge inspection reports. *Automation in Construction, 78*. https://doi.org/10.1016/j.autcon.2017.09.009

Ma, X., Hummer, D., Golden, J. J., Fox, P. A., Hazen, R. M., Morrison, S. M., et al. (2017). Using visual exploratory data analysis to facilitate collaboration and hypothesis generation in cross-disciplinary research. *International Journal of Geo-Information, 6*(11). https://doi.org/10.3390/ijgi6110368

Mihalea, R., & Tarau, P. (2004). TextRank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404–411).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv Preprint. arXiv:1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Mikolov, T., Yih, W. T., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. In: *HLT-NAACL*, June (Vol. 13, pp. 746–751).

Niu, B., Hong, S., Yuan, J., Peng, S., Wang, Z., & Zhang, X. (2014). Global trends in sediment-related research in earth science during 1992–2011: A bibliometric analysis. *Scientometrics, 98*(1), 511–529. https://doi.org/10.1007/s11192-013-1065-x

Onan, A., Korukoglu, S., & Bulut, H. (2016). Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications, 57*, 232–247. https://doi.org/10.1016/j.eswa.2016.03.045

Palandik, L. F., Hamilton, C. W., Scheidt, S. P., & Alvarez, A. M. (2017). Automated detection of geological landforms on Mars using convolutional neural networks. *Computers & Geosciences, 101*, 48–56. https://doi.org/10.1016/j.cageo.2016.12.015

Qiu, Q., Xie, Z., Wu, L., & Li, W. (2018). DGeoSegmenter: A dictionary-based Chinese word segmenter for the geoscience domain. *Computers & Geosciences, 121*, 1–11. https://doi.org/10.1016/j.cageo.2018.08.006

Qiu, Q., Xie, Z., Wu, L., & Li, W. (2019). Geoscience keyphrase extraction algorithm using enhanced word embedding. *Expert Systems with Applications, 125*, 157–169. https://doi.org/10.1016/j.eswa.2019.02.001
Rose, S. J., Cowley, W. E., Crow, V. L., & Cramer, N. O. (2012). Rapid automatic keyword extraction for information retrieval and analysis. US, US 8131735 B2.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation (Technical report). San Diego, CA: University of California, San Diego, La Jolla Institute for Cognitive Science.

Salton, G., Wong, A., & Yang, C. S. (1974). A vector space model for automatic indexing. ACM.

Santos, R., Muñoz-Castells, F., Calado, P., & Martins, B. (2018). Toponym matching through deep neural networks. International Journal of Geographical Information Science, 32(2), 324–348.

Sarkar, K., Nasipuri, M., & Ghose, S. (2010). A new approach to keyphrase extraction using neural networks. arXiv. Information Retrieval.

Sterckx, L., Demeester, T., Deleu, J., & Develder, C. (2015). Topical word importance for fast keyphrase extraction (pp. 121–122).

Su, X., Deng, S., & Shen, S. (2014). The design and application value of the Chinese social science citation index. Scientometrics, 98(3), 1567–1582. https://doi.org/10.1007/s11192-012-0921-4

Sung, Y. Y., & Kim, S. R. (2020). Topical keyphrase extraction with hierarchical semantic networks. Decision support systems, 128, 113163.

Wan, X., & Xiao, J. (2008). CollabRank: Toward a collaborative approach to single document keyphrase extraction. In COLING 2008, International conference on computational linguistics, proceedings of the conference (pp. 969–976). Manchester, UK: DBLP.

Wang, C., Chen, J., Xiao, F., Fode, T., & Li, L. (2016). Radioelement distributions and analysis of microtopographical influences in a shallow covered area, inner Mongolia, China: Implications for mineral exploration. Journal of Applied Geophysics, 133, 62–69. https://doi.org/10.1016/j.jappgeo.2016.06.013

Wang, C., Ma, X., Chen, J., & Chen, J. (2018). Information extraction and knowledge graph construction from geoscience literature. Computers & Geosciences, 112, 112–120. https://doi.org/10.1016/j.cageo.2017.12.007

Wang, G., Sun, J., Ma, J., Xu, K., & Gu, J. (2014). Sentiment classification: The contribution of ensemble learning. Decision Support Systems, 57, 77–93. https://doi.org/10.1016/j.dss.2013.08.002

Wang, R., Liu, W., & McDonald, C. (2015). Corpus-independent generic keyphrase extraction using word embedding vectors. In Proceedings of the Software Engineering Research Conference, (Vol. 39, pp. 1–8).

Xia, R., Zong, C., & Li, S. (2011). Ensemble of feature sets and classification algorithms for sentiment classification. Information Sciences, 181(6), 1138–1152. https://doi.org/10.1016/j.ins.2010.11.023

Yang, S., Lu, W., Yang, D., Li, X., Wu, C., & Wei, B. (2016). KeyphraseDS: Automatic generation of survey by exploiting keyphrase information. Neurocomputing, 224.

Zhang, C. (2008). Automatic keyword extraction from documents using conditional fields. International Journal of Computer Science Issues, 7(2), No. 3.

Zhang, H., Wang, Z., & Liu, D. (2017). A comprehensive review of stability analysis of continuous-time recurrent neural networks. IEEE Transactions on Neural Networks & Learning Systems, 28(7), 1229–1262.

Zhang, J., & El-Gohary, N. M. (2016). Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking. Journal of Computing in Civil Engineering, 30(2), 14101306441000.

Zhang, Y., Tuo, M., Yin, Q., Qi, L., Wang, X., & Liu, T. (2020). Keyphrase extraction with deep neural network model. Neurocomputing, 113–121.

Zhou, P., & El-Gohary, N. (2017). Ontology-based automated information extraction from building energy conservation codes. Automation in Construction, 74, 103–117.

Zhu, J., & Hua, W. (2017). Visualizing the knowledge domain of sustainable development research between 1987 and 2015: A bibliometric analysis. New York, NY: Springer-Verlag.