Information Sciences 517 (2020) 100–127

Contents lists available at ScienceDirect

Information Sciences

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

Phrase2Vec: Phrase embedding based on parsing

Yongliang Wu[a], Shuliang Zhao[b,c,d,*], Wenbin Li[e]

[a] College of Mathematics and Information Science, Hebei Normal University, Hebei 050024, China
[b] College of Computer and Cyber Security, Hebei Normal University, Hebei 050024, China
[c] Hebei Provincial Key Laboratory of Network and Information Security, Hebei 050024, China
[d] Hebei Provincial Engineering Research Center for Supply Chain Big Data Analytics & Data Security, Hebei 050024, China
[e] College of Information Engineering, Hebei GEO University, Hebei 050024, China

A R T I C L E I N F O

Article history:
Received 12 May 2019
Revised 1 December 2019
Accepted 18 December 2019
Available online 18 December 2019

Keywords:
Text representation
Phrase mining
Phrase embedding
Parsing
Text classification
Text clustering

A B S T R A C T

Text is one of the most common unstructured data, and usually, the most primary task in text mining is to transfer the text into a structured representation. However, the existing text representation models split the complete semantic unit and neglect the order of words, finally lead to understanding bias. In this paper, we propose a novel phrase-based text representation method that takes into account the integrity of semantic units and utilizes vectors to represent the similarity relationship between texts. First, we propose HPMBP (Hierarchical Phrase Mining Based on Parsing) which mines hierarchical phrases by parsing and uses BOP (Bag Of Phrases) to represent text. Then, we put forward three phrase embedding models, called Phrase2Vec, including Skip-Phrase, CBOP (Continuous Bag Of Phrases), and GloVeFP (Global Vectors For Phrase Representation). They learn the phrase vector with semantic similarity, further obtain the vector representation of the text. Based on Phrase2Vec, we propose PETC (Phrase Embedding based Text Classification) and PETCLU (Phrase Embedding based Text Clustering). PETC utilizes the phrase embedding to get the text vector, which is fed to a neural network for text classification. PETCLU gets the vectorization expression of text and cluster center by Phrase2Vec, furthermore extends the K-means model for text clustering. To the best of our knowledge, it is the first work that focuses on the phrase-based English text representation. Experiments show that the introduced Phrase2Vec outperforms state-of-the-art phrase embedding models in the similarity task and the analogical reasoning task on Enwiki, DBLP, and Yelp dataset. PETC is superior to the baseline text classification methods in the F1-value index by about 4%. PETCLU is also ahead of the prevalent text clustering methods in entropy and purity indicators. In summary, Phrase2Vec is a promising approach to text mining.

1. Introduction

Information systems are widespread nowadays, and enormous data is stored as text. English text representation is the most vital work in natural language processing (NLP) tasks [4]. Its goal is to transform a varied-length text (sentence, paragraph, document) into a low-dimension vector, which is used as input to subsequent NLP tasks, such as text classification

* Corresponding author.
E-mail address: shuliangzhao@hebtu.edu.cn (S. Zhao).

https://doi.org/10.1016/j.ins.2019.12.031
0020-0255/© 2020 Elsevier Inc. All rights reserved.

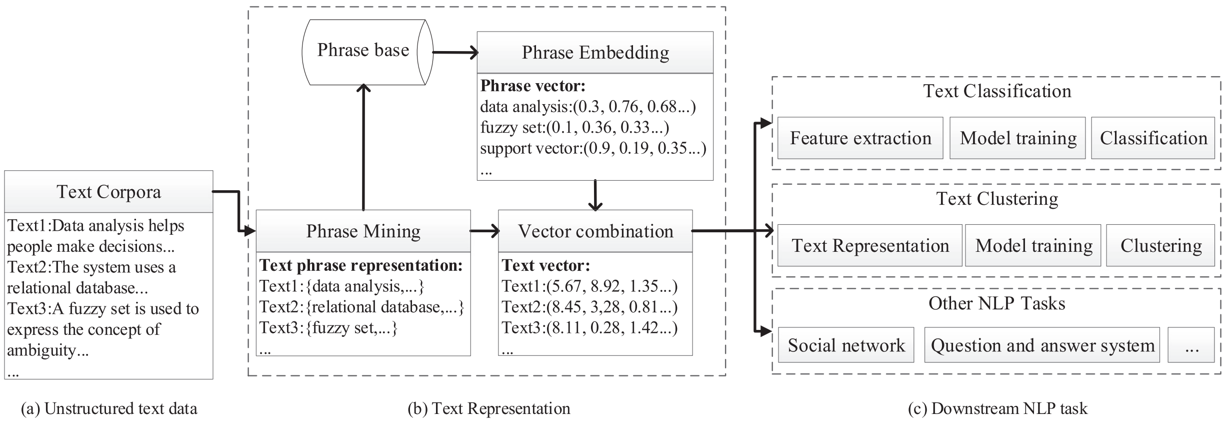

Text Corpora
Text1:Data analysis helps people make decisions...
Text2:The system uses a relational database...
Text3:A fuzzy set is used to express the concept of ambiguity...
...
Phrase base
Phrase Embedding
Phrase vector:
data analysis:(0.3, 0.76, 0.68...)
fuzzy set:(0.1, 0.36, 0.33...)
support vector:(0.9, 0.19, 0.35...)
...
Phrase Mining
Text phrase representation:
Text1:{data analysis,...}
Text2:{relational database,...}
Text3:{fuzzy set,...}
...
Vector combination
Text vector:
Text1:(5.67, 8.92, 1.35...)
Text2:(8.45, 3,28, 0.81...)
Text3:(8.11, 0.28, 1.42...)
...
Text Classification
Feature extraction
Model training
Classification
Text Clustering
Text Representation
Model training
Clustering
Other NLP Tasks
Social network
Question and answer system
...
(a) Unstructured text data
(b) Text Representation
(c) Downstream NLP task

Fig. 1. The overall flowchart of the phrase-based text representation model with some processing examples of core stages.

[35], text clustering [2,28], social network analysis [21,30]. Although many researchers have conducted in-depth research on text representation, it still faces many challenges, such as text semantic loss.

Text representation is refined into three aspects: text composition, vector representation and subsequent application of NLP tasks. Text composition focus on converting original text to an essential element group. The basic unit of English text is the word, so most of the previous studies compress the text into a set of words [24,45], called bag-of-words (BOW). E.g. the *Text1* in Fig. 1(a) is represented as six independent words (‘Data’, ‘analysis’, ‘helps’, ‘people’, ‘make’ and ‘decisions’). These methods split the complete phrase into unordered words thus lost the original semantics of the text. To preserve the order of the words, some researchers propose to use the n-gram method to represent the text [15]. E.g. the *Text1* in Fig. 1(a) is represented by 2-gram as some word sequences (‘Data analysis’, ‘analysis helps’, ‘helps people’, ‘people make’ and ‘make decisions’). These studies preserve the word order, but introduce a large number of meaningless units and increase the computational complexity. Some researchers have proposed other basic elements to represent text [11,48]. However, these methods slight the extraction of the semantic units in the text, so they often cause semantic loss.

Vector representation refers to map essential elements to vector space, and the distance of vectors indicates the similarity of components, i.e., embedding. The most classical vectorization technology is Word Embedding, which finds a mapping relation between word space and continuous vector space [24]. It expresses the relevance of words by the distance between vectors. Afterward, the researchers obtain a distributed representation of the word sequence through basic word combinations [25]. With the introduction of different text elements, the corresponding embedding methods are subsequently proposed [4,47]. These embedding methods all originate from word embedding and represent the relationship of elements. However, they ignore the semantic integrity of the component and reduce the performance of subsequent NLP tasks.

Text representation has laid a solid foundation for various NLP tasks. E.g., researchers utilize embedding technology to improve machine translation performance [7,37,46]. Many researchers have proposed joint solutions for embedding technology and NLP tasks [27,35]. However, there is no universal text representation for various NLP tasks.

Natural language usually employs the phrase as the basic semantic unit. A phrase is a word sequence which represents independent semantics. Summarizing the research of the predecessors, we find that the current work rarely pays attention to the phrase-based text representation. The main reason is no intrinsic separator between phrases. Some related questions are naturally raised: (a) How to identify all the phrases from a given sentence? (b) How to learn a distributed representation for an extracted phrase? (c) How to apply phrasal embedding to the down-stream tasks?

In this paper, we plan to give a better solution for the following three aspects.

1 Phrase mining problem: The traditional text composition model is the BOW model, which divides text into a set of unordered words [35]. Phrase preserves the order of words to the utmost extent and corresponds to independent semantic [33]. So, we propose a bag-of-phrases (BOP) model, which considers words and phrases corresponding to the separate semantic unit to represent text. BOP converts text into a group of phrases that are obtained by the phrase mining task. In this paper, we utilize HPMBP (Hierarchical phrase mining based on parsing) to extract all semantic units in the text. All extracted phrases constitute a phrase base for subsequent tasks.
2 Phrase embedding problem: After phrase mining, the next challenge is the mathematical expression of phrases, called phrase embedding. It is a function that involves a mathematical embedding from the phrase space to low dimensional vector space [16]. In this paper, we extend three models for English phrase embedding, called Phrase2Vec. Then, we get the vector of each phrase in the vocabulary. The vector distance represents the semantic similarity of phrases.
3 Application of Phrase2Vec in subsequent NLP tasks: According to previous research, different text representation methods have a specific impact on downstream NLP tasks [12]. On this study, we apply Phrase2Vec to the classical English text mining tasks (text categorization and text clustering), thus propose PETC (Phrase Embedding based Text Classification)

and PETCLU (Phrase Embedding based Text Clustering). Besides, we also discuss the feasibility of applying Phrase2Vec in other NLP tasks.

The overall flowchart of our model is shown in Fig. 1. Text corpora is a collection of documents from different domains (such as scientific papers, product reviews, or user posts). It consists of unstructured text, e.g., *Text1* is 'Data analysis helps people make and decisions…', as shown in Fig. 1(a). Fig. 1(b) demonstrates the core flow of the phrase-based text representation: phrase mining and phrase embedding. The goal of phrase mining is to mine the core phrases in the text and utilize a phrase group to represent the corresponding text. E.g., *Text1* is represented as {dataanalysis,...}. Next, we collect all the extracted phrases to construct a phrase base. Phrase embedding maps phrases into low-dimensional vector spaces. E.g., the vector of 'data analysis' is (0.3, 0.76, 0.68, …). In the vector combination phase, we obtain the vector representation of the text based on the phrase vector. E.g., *Text1* is (5.67, 8.92, 1.35, …). Fig. 1(c) briefly explains the application of Phrase2Vec in subsequent NLP. The main contributions of our research are as follows:

1 As far as we know, it is the first work to focus on the mining of hierarchical phrases. We put forward HPMBP which integrates the parsing into the phrases mining.
2 We bring about three Phrase2Vec methods for phrase embedding. Different from previous studies, they focus on the vector learning of the hierarchical phrase and use the vector distance to indicate semantic similarity.
3 We present PETC that utilizes Phrase2Vec to get the vector of text and trains a neural network for text classification. It is a fresh attempt to apply phrase-based representation to text categorization.
4 We come up with PETCLU that extends K-means by Phrase2Vec for text clustering. It provides a novel idea to solve the problem of document distance that has long plagued researchers.

Abundant experiments demonstrate that our methods have high effectiveness and efficiency compared with state-of-the-art research.

The rest structure of the article is listed. In Section 2, we describe the development of related work in four aspects: phrase mining, word & phrase embedding, text classification, and text clustering. Section 3 elaborates on the preliminary knowledge of our algorithm and lays a solid foundation for the subsequent algorithm understanding. Section 4 introduces the implementation details of Phrase2Vec, which consists of HPMBP and phrase embedding. We present HPMBP to extract phrases in the corpora. Based on the word embedding, we extend three phrase embedding methods and describe the algorithm implementation in detail. In Sections 5 and 6, we apply phrase-based text representation to text classification and text clustering, respectively. In Section 7, we train the Phrase2Vec on three corpora (Enwiki, DBLP, and Yelp, detailed in Section 7.1.1) and compare them with the state-of-the-art embedding methods in the similarity task and the analogical reasoning task. Then we conduct several experiments to compare the proposed PETC and PETCLU with state-of-the-art baselines and reach an in-depth analysis. We also point out the potential application of Phrase2Vec in different NLP tasks. Finally, Section 8 summarizes our proposed algorithm and briefly describes our future research plans.

2. Related work

2.1. Phrase mining

Accurate extraction of meaningful phrases attracts more and more researchers. We divide existing phrase mining methods into two camps according to the extraction method of candidate phrases: Data-based method and Grammar-based method.

Data-based method typically uses n-gram techniques to obtain candidate phrases and then evaluates them based on statistical metrics. Liu et al. [20] adopted a phrase mining model that mined quality phrases in corpora based multi-word sequences and integrated with phrasal segmentation. Experiments showed that it was a competitive phrase mining method. Shang et al. [33] employed a framework for automated phrase mining, which extracted high-quality phrases by evaluating n-gram candidate phrase. Li et al. [17] mined quality topic phrases through data-driven methods. Data-based methods guarantee the frequency of phrases. However, they introduce several meaningless word sequences and widen candidate phrase space.

Grammar-based method obtains candidate phrases based on the POS or context, so it gets more meaningful candidate phrases. Gebhardt et al. [8] used the hybrid grammar to deal with complicated phrase structures. Jie et al. [11] exploited the structured information conveyed by dependency trees to improve the performance of NER. Grammar-based methods introduce POS into phrase mining, improving the completeness of the phrase. However, they only involve short phrases that satisfy simple grammar but neglect long combination phrases and hierarchical phrases.

Many previous researchers have implemented related phrase extraction functions in their respective research processes. Wang et al. [37] employed the extracted phrase pairs to solve the adaptation in machine translation. Li et al. [17] proposed a joint solution for phrase mining and cohesive topic. Although the phrase related technology has made some progress in different fields and the researchers have also proposed some methods of phrase mining, these methods are all limited by the appropriate field and difficult to extend to other areas.

Syntactic parsing is an important aspect of natural language understanding. Researchers have made in-depth progress on parsing and achieved specific results in different application fields. Zhang et al. [44] proposed a probabilistic graph-based parsing model and achieved exceptional results in both Chinese and English. Gebhardt et al. [8] offered a way to use parsing to find complex structures. Some researchers have gradually tried to combine parsing with the mining of phrases, but still face particular challenges [8]. All the above studies show that syntactic parsing has dramatically promoted the task of natural language understanding, thus laid the foundation for our subsequent parsing-based jobs.

Different from previous research, we incorporate the parsing into the phrase mining process, which combines the advantages of Data-based and Grammar-based methods. We focus on extracting hierarchical phrases in terms of sentences, which are used to train the phrase embedding.

2.2. Word embedding & phrase embedding

Many researchers have done much research in the field of word vectorization. Mikolov et al. [25] first proposed the word embedding model and proved the possibility of phrase embedding. Pennington et al. [29] put forward a global log-bilinear regression model that combined the local content with global information for word embedding. Stein et al. [35] discussed the validity of word embedding on hierarchical text tasks. However, word embedding ignores phrase semantics and loses the order of words in a sentence.

Some researchers have conducted related studies on phrase embedding. Socher et al. [34] introduced a recursive neural network for unseen phrases representation. Wieting et al. [39] proposed to learn word embedding and phrase embedding based on the Paraphrase Database (PPDB), and experiments proved that it had achieved strong competitiveness. Wieting et al. [40] proposed an excellent sentence embedding method based on PPDB. Hashimoto et al. [10] weighted compositional and non-compositional phrase embedding for phrase representation. Li et al. [15] proposed that n-grams could better represent the semantics of text. Zhao et al. [47] verified the advantages of n-gram for textual meaning. Zhuang et al. [48] combined topic recognition with embedded features of words. Li et al. [16] proposed that an implicit hierarchical structure could promote joint learning combination and phrase embedding methods. Zhao et al. [45] raised to quantify the semantic relevance of words based on word embedding. However, the existing phrase embedding methods are based on n-gram to get candidate phrases, ignoring the embedding of hierarchical phrases.

Based on existing phrase research, other natural language understanding tasks are also promoted. For example, many leading machine translation studies have integrated phrase technology. Zhang et al. [43] proposed a recursive encoder to improve phrase embedding and machine translation. Passos et al. [27] applied the phrase embedding to named entity recognition and had achieved excellent experimental results. Durrani et al. [6] focused on the role of phrase and N-gram for statistical machine translation. Yin and Schütze [42] enhanced paraphrase identification performance by the utilization of continuous and discontinuous phrase embeddings. Passban et al. [26] employed four probability scores in phrase tables as bilingual features for machine translation. Sánchez-Cartagena et al. [32] integrated linguistic resources into phrase-based statistical machine translation. Wang et al. [38] conformed a phrase memory into the encoder-decoder architecture for machine translation. Zhao et al. [46] used a phrase table as the recommendation memory so that NMT could make correct predictions. Eriguchi et al. [7] proposed an end-to-end syntactic Neural machine translation, which extended a sequence-to-sequence model with the source-side phrase structure. There are also some other research advances [21,30]. The above researches prove that phrase embedding plays an essential part in the development of natural language understanding. However, few researchers pay attention to hierarchical phrase embedding.

2.3. Text classification

Text categorization is a typical text mining technique. The predecessors have done some research on it and applied it to different fields, such as disease diagnosis and spam detection. Burkhardt and Kramer [3] proposed a model combining text classification and topic model. Canuto et al. [5] improved the classification model through a multi-target strategy. Meng et al. [23] employed a weak supervisory approach to address the lack of training data. Salles et al. [31] used an improved random forest classifier to solve high dimensional noise problems. Stein et al. [35] investigated the application of word embeddings on automatic document classification tasks and proved that word embedding could promote text classification model. Wieting et al. [40] suggested that sentence embedding could improve the efficiency of text classification. The above researches combine text classification and semantic understanding. Moreover, some researchers have tried to analyze the role of word embedding in promoting text classification. However, there are few studies on text categorization with phrase embedding.

2.4. Text clustering

Text clustering is a common unsupervised text analysis technique. It learns without data labels and mines in-depth knowledge hidden inside the text, such as emotion analysis and topic discovery. Liang et al. [19] employed text clustering to represent user interests. Mei et al. [22] used high-dimensional fuzzy clustering to solve the problem of document clustering. Xu et al. [41] combined word embedding and convolutional neural networks for short text clustering. Brockmeier et al. [2] employed a predictive network to implement self-tuning document clustering. Pei et al. [28] raised a concept factorization method that integrated an adaptive neighbor's regularization. These researches prove that the key of text clustering is to express the distance between texts reasonably. So, we present a novel text clustering method that calculates text distance based on phrase embedding.

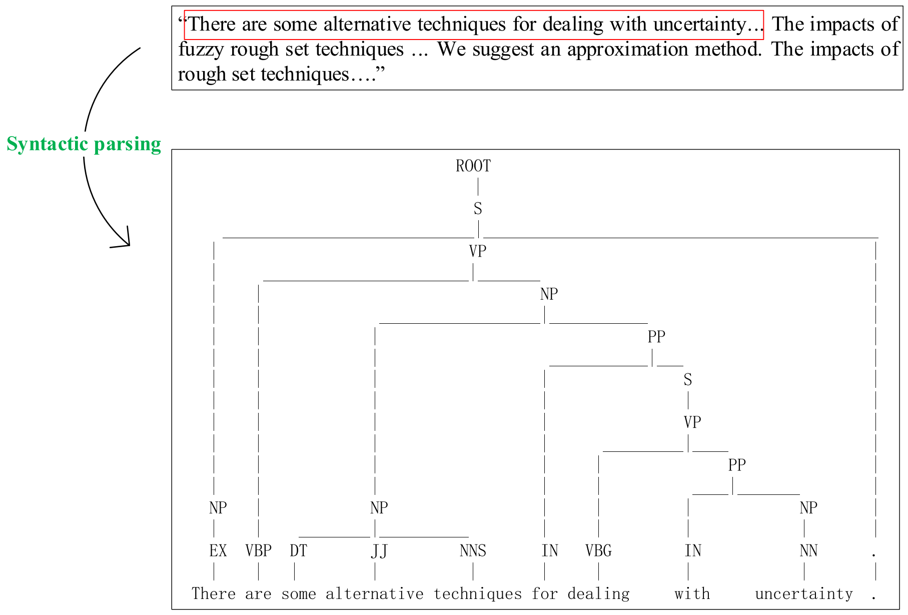

"There are some alternative techniques for dealing with uncertainty... The impacts of fuzzy rough set techniques ... We suggest an approximation method. The impacts of rough set techniques...."
Syntactic parsing
ROOT
S
VP
NP
PP
S
VP
PP
NP
NP
NP
EX VBP DT JJ NNS IN VBG IN NN .
There are some alternative techniques for dealing with uncertainty .

Fig. 2. Convert sentence to parse tree by parsing. (For interpretation of the references to color in the text, the reader is referred to the web version of this article.)

3. Preliminaries

3.1. Basic definition

Word: Word is the smallest unit of text. Each word may have distinct meanings in different sentences. Spaces or punctuations separate words in an English document. In the sample text of Fig. 2, 'fuzzy', 'rough', 'set' are three words. Many of the existing text mining tasks employ word as processing units [35,45]. In our study, we use the symbol w to denote a word in the text.

N-gram: In the NLP field, n-gram represents a set of consecutive words. In the sample text of Fig. 2, 'are some', 'some alternative', 'alternative techniques' are three two-grams (bigram). 'There are some', 'are some alternative' are three-grams (trigram). Many researchers have also conducted in-depth research on n-gram [47].

Phrase: In linguistics, a phrase consists of one or more words (no length limit) in a fixed order that is an independent component in a sentence. Phrases are not naturally separated by delimiters but hidden in the sentence. From the perspective of the role of the phrase in the sentence, there are three common phrase types: noun phrase (NP), prepositional phrase (PP), and verb phrase (VP). According to the composition of the phrase, we divide phrases into Non-Compositional Phrase, Compositional Phrase, and Hierarchical Compositional Phrase. In the sample text of Fig. 2, 'fuzzy rough set' is an NP. 'an approximation method' is a compositional phrase. 'The impacts of rough set techniques' is a hierarchical compositional phrase. Some researchers have begun to study phrase mining [17,33]. The challenges of phrase research are the hierarchical structure and embedding representation of phrases [16]. In our paper, we denote the phrase as p. We expect to employ phrases to represent the text, so we introduce the TFIDF value of the phrase to indicate the importance of the phrase to the text. Given a document set $D = \{d_i\}$ and phrase set $P = \{p_j\}$, the TFIDF value of the phrase p_j relative to the document d_i is expressed as Eq. (1).

$$\mathrm{TFIDF}_{ij} = \frac{n_{ij}}{\sum_{p_k \in d_i} n_{ik}} * \ln \frac{|D|}{|\{i : p_j \in d_i\}|}, \tag{1}$$

where n_{ij} is the number of times p_j appears in d_i. $|D|$ means the number of documents in the corpus.

Text representation is to transform unstructured text into a structured form for convenient calculation. For a corpus $D = \{d_i\}, i = 1, 2...n$, where d_i means a document. Text vectorization maps D into a multidimensional vector space where each text corresponds to a point. The similarity between texts is transformed as the distance of corresponding points. The most commonly used text composition in current research is BOW, which expresses text as a set of words, thereby disregards grammar and word order. It formulates as $D = \{\{w_1, w_2, w_3 \ldots\}_{d_i}\}, i = 1, 2 \ldots n$. In this paper, we propose BOP to express text, which maximizes the preservation of text semantics. It formulates as $D = \{\{p_1, p_2, p_3 ...\}_{d_i}\}, i = 1, 2 \ldots n$. Based on the composition structure, researchers can get the vector representation of the document. Comparing the classic three representation schemes (TFIDF, latent Dirichlet allocation(LDA) [1], document to vector(Doc2Vec) [14], and recursive neural network (RNN) [34]), we employ RNN for combined embedding vector. It combines the vector of each component to repre-

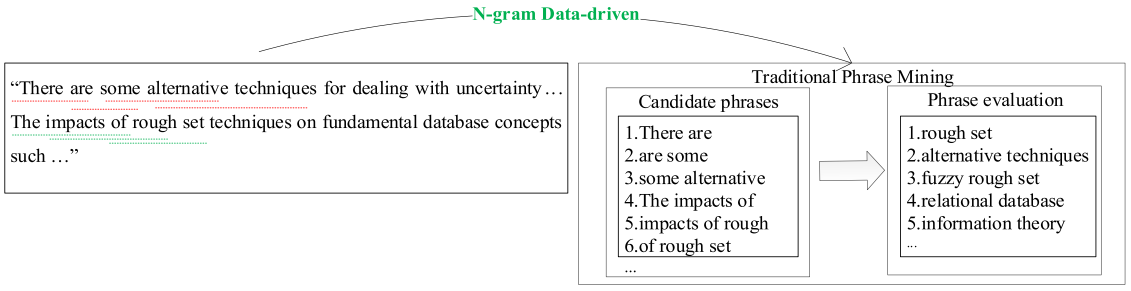


Fig. 3. The overall architecture of the traditional phrase mining method. (For interpretation of the references to color in the text, the reader is referred to the web version of this article.)

sent the document as a vector in the same vector space. In this paper, we split text representation into phrase mining and phrase embedding.

Syntactic parsing aims to analyze the syntactic structure of the sentence based on the part of speech. It is mainly divided into two categories: dependency and constituency. Dependency parsing focuses on the dependencies of the various components in the sentence, e.g., in 'John hit the ball', it expresses a 'hit' relationship between 'John' and 'ball'. Constituency parsing researches the phrase structure of a sentence, e.g., it regards sentence ('John hit the ball') as a combination of a noun phrase ('John') and a verb phrase ('hit the ball'). The goal of our research is to mine the hierarchical phrases, so we choose the constituency parsing. The parsing tree is the common syntactic structure expression that uses a hierarchical tree to express sentence grammatical structure. In Fig. 2, we convert the sentence in the red box into a parse tree, which reflects the syntactic hierarchy with the corresponding string [There are [[some alternative techniques] [for dealing [with uncertainty]]]].

3.2. Phrase mining

The primary purpose of phrase mining is to get a list of high-quality phrases with independent meanings from a large number of corpora, which can more effectively represent the original text. The core steps of the traditional phrase mining framework are as follows [17,20,33]:

1 It enumerates all word sequences based on n-gram technology, usually limiting n to less than 6. In Fig. 3, we mark some bigrams with red dashed lines ('There are', 'are some', ' some alternative'), and some trigrams with green dashed lines ('The impacts of', ' impacts of rough').
2 It filters the list of n-grams by some statistical features (such as frequency, correction frequency) then forms a list of candidate phrases. In Fig. 3, we find that 'There are' and 'The impacts of' appear in the candidate list, which means they satisfy the statistical feature.
3 It sorts the phrases by custom evaluation metrics. The most commonly used evaluation indicators in the field of phrase mining are Popularity, Concordance, Informativeness, Completeness [20,33]. Phrase quality measures the probability that a group of words will form a phrase. Given word sequence $w_1w_2w_3...w_n$, its quality denotes as Eq. (2).

$$Q(w_1w_2w_3...w_n) = p(< w_1w_2w_3...w_n > |w_1w_2w_3...w_n) \in [0, 1], \tag{2}$$

where $< w_1w_2w_3...w_n >$ means the event that word sequence constitutes a phrase. $Q(\cdot)$ expresses phrase quality evaluation function. An excellent phrase indicator evaluates a high-quality phrase close to 1, $Q('roughset') \approx 1$, and an inferior quality phrase close to 0, $Q('aresome') \approx 0$. The output of the phrase mining method is a list of high-quality phrases.

The traditional phrase mining method is limited by the length and statistical characteristics, resulting in neglecting long phrases and introducing meaningless phrases. Phrases in the actual corpus contain long phrases with few occurrences, so we expect to integrate syntactic parsing into the process of phrase mining, which can optimize the number of candidate phrases and improve the accuracy.

3.3. Word embedding

Text representation aims to represent the unstructured text mathematically. The basic unit of the document is word, so previous researchers transform the text representation into word vectorization, which is called word embedding. In mathematics, embedding is an injective and structure-preserving map $f : X \to Y$. The injection means that for any $x \in X$, there must be a distinct $y \in Y$ corresponding to it, $\forall x \in X \to \exists y \in Y : y = f(x)$. "structure-preserving" indicates that the relation of elements is preserved during the mapping process. In NLP, word embedding is a general term for the method of mapping word space to vector space. Previous researchers have proposed a variety of ways to achieve word embedding, such as neural network model training and reduction co-occurrence matrix.

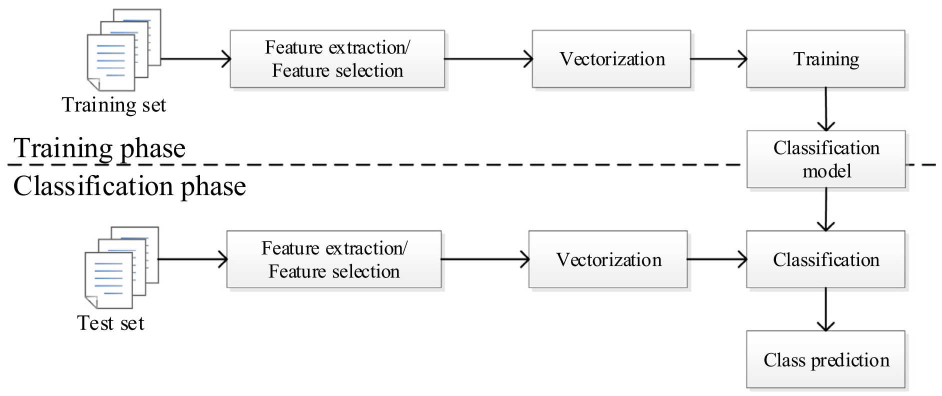


Fig. 4. The overall architecture of the traditional text classification.

Word2Vec is a word embedding model first proposed by Google. It solves the pseudo-task through the three-layer neural network, and the weight matrix of the neural network is used as the result of word embedding [24]. Word2Vec algorithm has many advantages compared to earlier algorithms, such as latent semantic analysis. It has two specific implementation forms: Skip-Gram and continuous bag of words (CBOW) [24,25].

Skip-Gram uses the central word to speculate the surrounding words. Its hidden layer weight matrix is used as the word vector. The distance between word vectors represents the semantic similarity of words. It uses the window size to define 'nearby', usually set to 2. Then the training word pairs are obtained by iterating the sliding window. We train the neural network by the one-hot representation (generally more than 10,000 dimensions) of these word-pairs and finally get word vector (commonly 300 dimensions). The hidden layer has 300 nodes, and the output layer adopts a SoftMax regression classifier to improve training efficiency. We obtain the posterior distribution of words, shown as Eq. (3).

$$p(w_j|w_I) : y_j = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})}, \tag{3}$$

where y_j expresses the jth output. w_I is the central word. w_j is the surrounding word. We employ the standard exponential function $\exp(\cdot)$ to standardize word prediction probability. u_j denotes the jth component of the output V. To improve the training speed of neural networks, the researchers extended multi-frequency sub-sampling and negative sampling based on Word2Vec. Researchers have shown that negative sampling and subsampling techniques not only elevate training efficiency but also improve the quality of the generated word vectors.

The CBOW predicts the central word based on context. Its architecture is similar to Skip-Gram, except how the training word-pairs are obtained. CBOW calculates the likelihood of the particular word in the middle by providing content words appearing nearby (input word). The structure and training method of the neural network is consistent with the Skip-Gram. Through researching, it is superior to Skip-Gram in training speed [24,25].

After Skip-Gram and CBOW, the most outstanding is GloVe which achieves Word2Vec by unique factorization of the co-occurrence matrix [29]. The specific implementation process of GloVe is as follows. First, we extract the word co-occurrence matrix X from the original corpus. X_{ij} denotes the co-occurrence frequency of word i and word j. We traverse each word and the words nearby to form the training word pairs. The GloVe captures the counts of overall statistics about word co-occurrence. Then we factorize the co-occurrence matrix to obtain the lower-dimension word vector.

Word Embedding offers a technique to get the word vector; however, it ignores the relevance between words and phrases. In our work, we express text by BOP which maximizes the preservation of text semantics. After phrase mining, the next challenge to be addressed is the mathematical expression of the phrase, which is called phrase embedding. Similar to the word embedding, the phrase embedding expects to find a mapping that is injective and structure-preserving. Different from word embedding, phrase embedding is the map between the phrases space and the lower-dimensional vector space. Many researchers focus on word embedding, but few researchers consider phrase embedding. Phrases are more reflective of the actual semantics of sentences, so we concentrate on phrase embedding.

3.4. Text classification

The goal of text categorization is to assign text to predefined categories based on the content. It is the cornerstone of NLP and used in major applications such as spam identification, sentiment analysis, and topic identification. Text classification is implemented in two steps: training and evaluation. The overall framework of the traditional text classification model is shown in Fig. 4. In the training phase, we preprocess the labeled unstructured text data $D = \{(d_1, t_1), (d_2, t_2), \ldots (d_{|D|}, t_{|D|})\}$ where d_i is a training text with its original label t_i, and convert them into structured data. The usual practice is to indicate text based on the BOW model, which obtains the core words (e.g., keywords) of the text by feature extraction and feature

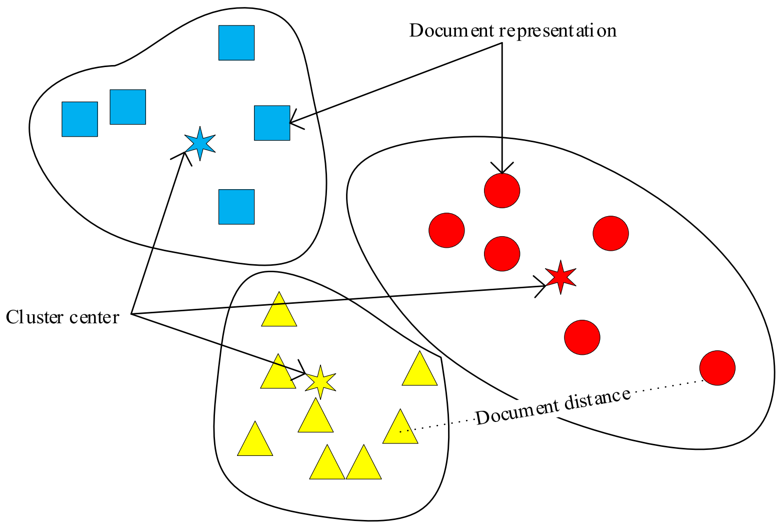


Fig. 5. A schematic diagram of a traditional text clustering.

selection techniques (e.g., TFIDF), $D = \{(\{w_{11}, w_{12}, ...\}, t_1), (\{w_{21}, w_{22}, ...\}, t_2), ...(\{w_{|D|1}, w_{|D|2}, ...\}, t_{|D|})\}$, where w_{ij} denotes the jth word of the ith document. The keywords are converted to the corresponding mathematical vectors by word embedding model, $D = \{(\{\mathbf{w_{11}}, \mathbf{w_{12}}, ...\}, t_1), (\{\mathbf{w_{21}}, \mathbf{w_{22}}, ...\}, t_2), ...(\{\mathbf{w_{|D|1}}, \mathbf{w_{|D|2}}, ...\}, t_{|D|})\}$, where $\mathbf{w}_{ij}$ is the vector of the relevant word. Then, we feed the text vector to the classification model (e.g., Naive Bayesian, k-Nearest Neighbor), as $f : D \rightarrow T, f(d) = t$. In the classification phase, we use the same steps as the first phase for text preprocessing to obtain the vector of the document to be classified. Then, we adopt the text classification model obtained in the early stage to predict the text label. Finally, we get the classification evaluation by comparing the original label with the prediction label.

Through the description of the overall process of text categorization, we find that the accuracy of text categorization is relevant to the rationality of the text vector. However, the existing text representation methods are based on BOW, which causes the loss of phrase semantics and affects classification accuracy. In our research, we propose a novel phrase-based text classification model, which incorporates BOP into the existing text classification method to enhance accuracy.

3.5. Text clustering

Text clustering is the most classic unsupervised text mining method. It obtains text aggregation without text labels. Researchers apply text clustering for topic discovery, related recommendations, etc. The key to text clustering is to define a reasonable text similarity function. In different application areas, text clustering has different levels of granularities, e.g., document, paragraph, sentence, or term. We divide text clustering into three phases: initial definition, cluster training and cluster evaluation. In the initial definition phase, we need to find a reasonable method to express the texts $D = \{d_1, d_2, ...d_{|D|}\}$ mathematically and cluster centers $C = \{c_1, c_2, ...c_k\}$, also, and define the distance from the text to the cluster center $\text{Dist}(d_i, c_j)$, where $d_i \in D$, $c_j \in C$. The usual practice employs the BOW to denote the text $D = \{\{w_{11}, w_{12}, ...\}, \{w_{21}, w_{22}, ...\}, ...\{w_{|D|1}, w_{|D|2}, ...\}\}$, then aggregates the word vector into a text vector $D = \{\{\mathbf{w_{11}}, \mathbf{w_{12}}, ...\}, \{\mathbf{w_{21}}, \mathbf{w_{22}}, ...\}, ...\{\mathbf{w_{|D|1}}, \mathbf{w_{|D|2}}, ...\}\} = \{\mathbf{d}_1, \mathbf{d_2}...\mathbf{d}_{|D|}\}$. According to different clustering methods, the cluster centers are indicated as vectors too. The initial cluster centers are defined as vectors of random texts $C = \{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_k\}$. The similarity function calculates the distance from the text to the cluster center $\text{Dist}(d_i, c_j)$. Fig. 5 depicts the overall architecture of text clustering, where shapes of different colors show text in different clusters, and stars represent cluster centers. In the cluster training phase, we iteratively update the cluster center and the labels of documents until the cluster center no longer changes. Then, we have divided the sample data into different clusters. We can extract standard features of texts in separate groups for classification. In the evaluation phase, we analyze the results of clustering from multiple perspectives and compare various indicators, such as purity and mutual information.

Through the description of the overall process of text clustering, we find that the foundation of the text clustering depends on the reasonable representation of the text and cluster center. However, the existing text clustering methods are based on the BOW model, which neglects the phrase semantics and obtains low-quality results. In our work, we propose a novel phrase-based text representation and incorporate it into the existing text clustering methods to improve clustering quality.

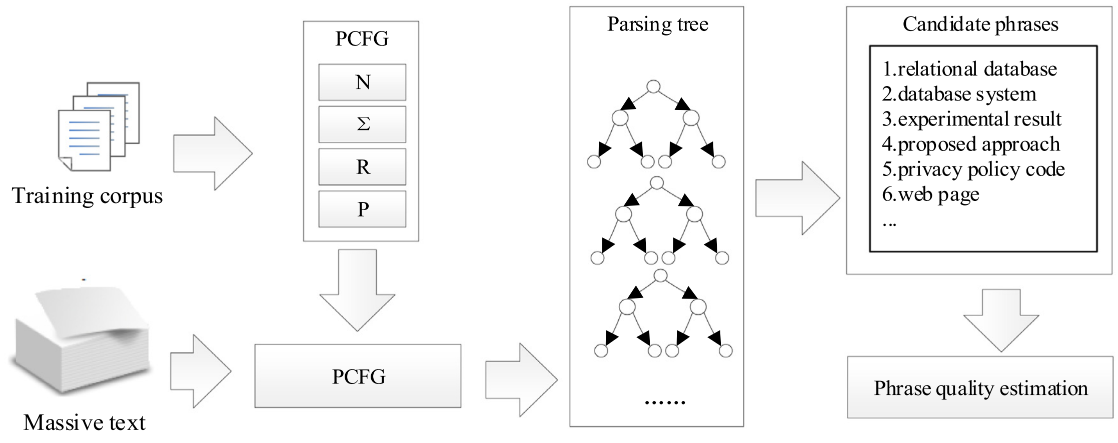


Fig. 6. The overall architecture of the HPMBP.

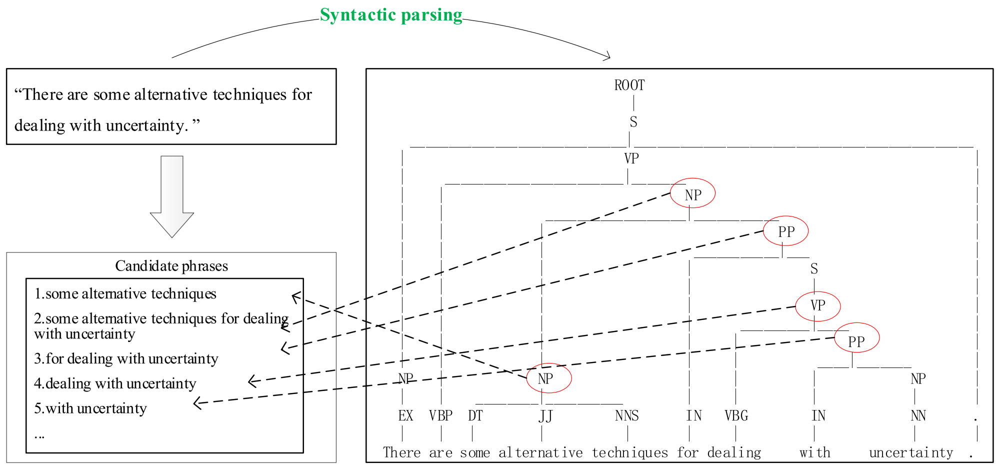

Syntactic parsing
"There are some alternative techniques for dealing with uncertainty."
Candidate phrases
1.some alternative techniques
2.some alternative techniques for dealing with uncertainty
3.for dealing with uncertainty
4.dealing with uncertainty
5.with uncertainty
...
ROOT
S
VP
NP
PP
S
VP
PP
NP
NP
NP
EX VBP DT JJ NNS IN VBG IN NN .
There are some alternative techniques for dealing with uncertainty .

Fig. 7. Hierarchical traversal parse tree to obtain candidate phrases.

4. Phrase embedding based on parsing

In this part, we introduce the Phrase embedding based on Parsing. In Section 4.1, we present the method of hierarchical phrase mining based on parsing, called HPMBP, which extracts hierarchical phrases from the corpus. In Section 4.2, we propose Skip-Phrase and CBOP (Continuous Bag of Phrases) for learning phrase vector. In Section 4.3, we put forward GloVeFP (Global Vectors For Phrase Representation), which employs global vectors for phrase embedding.

4.1. Hierarchical phrase mining based on parsing

The purpose of hierarchical phrase mining is to obtain the hierarchical phrases which have independent semantics. The previous study enumerated all n consecutive words in the corpus as phrase candidates based on the n-gram method. It ignored phrases longer than n (e.g., n usually set to 6) and introduced a large number of meaningless phrases in the candidate set. In this section, we present HPMBP (Hierarchical phrase mining based on parsing), which extracts all semantic units in the parse tree as candidate phrases. Fig. 6 shows the overall architecture of the HPMBP approach. The input of HPMBP includes grammar corpus and text corpus. We get the probabilistic context-free grammar (PCFG) based on the grammar corpus. Then we convert large text into the corresponding parsing trees depends on PCFG. Then we traverse all the parse trees based on the hierarchical traversal method to get the candidate phrase set. Fig. 7 shows the specific flow of obtaining candidate phrases based on the parse tree. The parsing process improves the quality of candidate phrases and reduces the size of the candidate phrase space. Finally, we filter the candidate phrases through the phrase quality estimation module to get the final phrase list.

Algorithm 1 Hierarchical phrase mining based on parsing.

```
Input: Corpus $C = (s_1,t_1), (s_2,t_{2|}) \cdots (s_{|C|},t_{|C|})$.
Output: Quality phrase list.
1: $PL \leftarrow []$ // Initialize the phrase storage list
2: $RF \leftarrow []$ // Initialize candidate phrase frequency dictionary
3: for $s_i$ in $C$ do
4:     $tempstack \leftarrow \emptyset$ //Initialize the auxiliary stack
5:     $tempstack$.push($t_i$) // Push the root of $t_i$ in the stack
6:     while $t \leftarrow tempstack$.pop() is not empty and ($t.lable$ in {NP, VP, PP}) do
7:       if $t.content$ not in $PL$ then
8:         $PL$.add($t.content$)
9:         $RF[t.content] \leftarrow 1$
10:        else do
11:          $RF[t.content] \leftarrow RF[t.content]+1$
12:        End if
13:        $stl \leftarrow t$.subtrees() // Get the subtree list of $t$
14:          if $stl$ is not empty then
15:            for $st_i$ in$stl$ do // Add each subtree to the auxiliary stack in turn
16:              $tempstack$.push($st_i$)
17:            End for
18:          End if
19:        End while
20:    End for
21:      $PL \leftarrow$ Evaluation($PL$, $RF$) //Filter and evaluate candidate phrases
22:    return $PL$, $RF$
```

We obtain the PCFG through the available Treebank dataset (Penn Treebank Syntax).[1] G can be defined by a quintuple $G = (N, \Sigma, R, S, P)$. N denotes the non-terminal symbols set. The elements of N are derived from the phrase level bracket labels of Penn Treebank II Constituent Tags,[2] such as PP, VP, and NP. Because phrase can be composed of other phrases, a multi-level phrase structure constructs the sentence's syntax tree. In the example sentence, the noun phrase 'some alternative techniques for dealing with uncertainty' consists of the noun phrase 'some alternative techniques' and the prepositional phrase 'for dealing with uncertainty'. Σ is the set of terminal symbols. Terminals in grammar are words in the corpus and disjoint from N. We consider the collection of all the words in the Penn Treebank as Σ. R denotes the production ruleset. Each r is expressed $A \rightarrow B$, where A indicates the non-terminal node, B means the combination of elements derived from $(\Sigma \cup N)$. We collect all the extensions of non-terminal nodes in the Penn Treebank to form R. S means start point. P denotes the production probabilities. The rule r is expressed as $A \rightarrow B[p]$, where p shows the likelihood of r, i.e. $\mathrm{P}(B|A)$. Thus, the sum of possible expansion is 1: $\sum_{\gamma \in B} \mathrm{P}(A \rightarrow \gamma) = 1$, γ is arbitrary expansion rules of A. We can calculate the probability of each rule through Penn Treebank, shown in Eq. (4).

$$\begin{aligned} \mathrm{P}(A \rightarrow \beta) &= \frac{\mathrm{Count}(A \rightarrow \beta)}{\sum_{\gamma} \mathrm{Count}(A \rightarrow \gamma)} \\ &= \frac{\mathrm{Count}(A \rightarrow \beta)}{\mathrm{Count}(A)} \end{aligned} \tag{4}$$

We convert sentences into the corresponding parse trees using the Cocke–Younger–Kasami algorithm (CKY), which depends on PCFG. In this paper, the CKY algorithm is used to obtain the parse tree of each sentence, laying the foundation for subsequent candidate phrase extraction. Give a text corpus $C = s_1, s_2 \ldots s_n$, s_i is the $_i$th sentence in the corpus. We obtain the parse tree corresponding to each sentence by the PCFG and CKY algorithm, so the text corpus is further represented as $C = (s_1,t_1), (s_2,t_2) \ldots (s_n,t_n)$. t_i is the parse tree of s_i. Based on this preprocessing, we propose Algorithm 1 to obtain phrases set.

Algorithm 1 shows the implementation of HPMBP. $|C|$ denotes the number of sentences in the corpus. *t.content* means the string content of t. *t.lable* expresses the tag of the phrase tree. In Fig. 7, we map a sentence into a hierarchical tree by syntactic parsing, and each node represents a phrase. Each node contains several subtrees that correspond to sub-phrases. In the parse tree, we employ t.subtrees() to get the direct subtree list of t. In HPMBP, we only limit the types of phrases (NP, VP, and PP) without the length. However, phrases are filtered again in the evaluation process. Following [20,33], we evaluate and filter phrases in four criteria: Popularity, Concordance, Informativeness, and Completeness. Through statistical analysis of the experimental results, we find that the length of the phrase is less than 30. It depends on different data sets and evaluation processes. Because HPMBP requires a certain amount of computing time, it is a non-real-time task. It returns phrases vocabulary which is used in the subsequent algorithm.

[1] https://catalog.ldc.upenn.edu/LDC99T42.
[2] http://www.surdeanu.info/mihai/teaching/ista555-fall13/readings/PennTreebankConstituents.html.

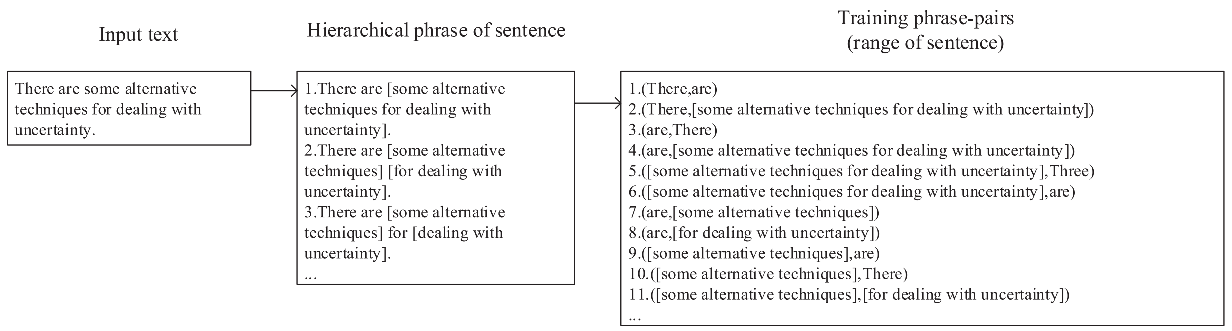

Input text
There are some alternative techniques for dealing with uncertainty.
Hierarchical phrase of sentence
1.There are [some alternative techniques for dealing with uncertainty].
2.There are [some alternative techniques] [for dealing with uncertainty].
3.There are [some alternative techniques] for [dealing with uncertainty].
...
Training phrase-pairs (range of sentence)
1.(There,are)
2.(There,[some alternative techniques for dealing with uncertainty])
3.(are,There)
4.(are,[some alternative techniques for dealing with uncertainty])
5.([some alternative techniques for dealing with uncertainty],Three)
6.([some alternative techniques for dealing with uncertainty],are)
7.(are,[some alternative techniques])
8.(are,[for dealing with uncertainty])
9.([some alternative techniques],are)
10.([some alternative techniques],There)
11.([some alternative techniques],[for dealing with uncertainty])
...

Fig. 8. The acquisition process of training phrase-pairs in Skip-Phrase.

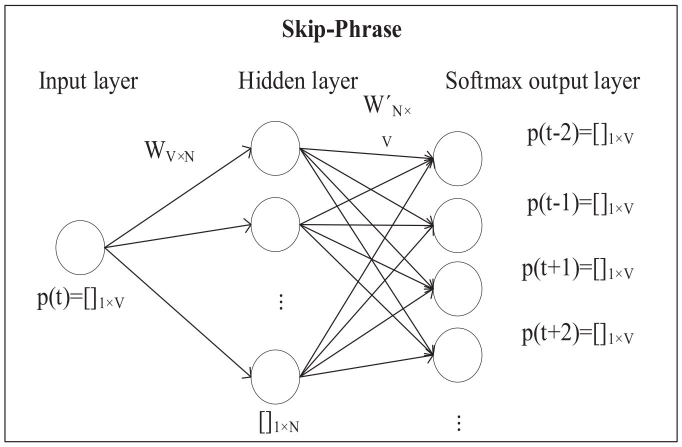


Fig. 9. The training process for Skip-Phrase.

4.2. Phrase2Vec

4.2.1. Skip-Phrase

Skip-Gram is the most classic Word2Vec model, which predicts context words by giving a specific word. The goal of Skip-Gram is to train the weight of the hidden layer that is treated as the embedding of the corresponding word. In this section, we extend the Skip-Gram framework to the Skip-Phrase, which learns the vector of the phrase, i.e., phrase embedding.

The Skip-Phrase model is divided into two parts. The first step is to build a neural network based on the training phrase-pairs. As shown in Fig. 8, we preprocess the input text in units of sentences. Based on HPMBP, we get the BOP representation of the sentence. Different from Skip-Gram, we set "nearby" to the sentence range. Following the above ideas, we acquire a group of phrase-pairs as the training data of the neural network, (input phrase, output phrase). The output represents the possibility that the central phrase and other phrases appear in the same sentence. Fig. 9 shows the training process of the Skip-Phrase. We obtain the one-hot representation of each phrase based on the phrase vocabulary acquired in Section 4.1. The input and output in Skip-Phrase are both V-dimensional vectors where V denotes the number of phrases. The output neurons adopt SoftMax to optimize training speed. During the training of the neural network, the input and output are encoded by one-hot. N is the dimension of the resulting phrase embedding space. We acquire the weight matrix W_{V*N} by Skip-Phrase. In the hidden layer, the V-dimensional vector of the input phrase is converted to an N-dimensional vector, which is fed to the output layer. The output layer converts hidden layer results into phrase vectors. Based on this mechanism, the vectors of phrases with the same context are very similar. The raw input is a corpus $D = \{\{p_1, p_2, p_3 \ldots\}_{s_i}\}, i = 1, 2 \ldots n$, where s_i denotes a sentence. $\{p_1, p_2, p_3 \ldots\}_{s_i}$ indicates the BOP composition of the sentence. P means the phrase vocabulary. Let p and cp denote phrase and context phrases. Skip-Phrase's parameters involve phrase vector dimension and vocabulary size. With N-dimensional phrase embedding, $\mathbf{p} \in R^N$, the total number of parameters is $|V|*N$. Optimization goal of Skip-Phrase is shown in Eq. (5).

$$\max \sum_{t=1}^{||P||} \left[\sum_{cp, p_t \in P} \log \mathrm{p}(cp|p_t) \right] \tag{5}$$

Negative sampling technology can also be applied to speed up training. In this part, we train the phrase embedding model according to the phrase vocabulary. In our algorithm, the phrase is a complete semantic unit with a unique embedded

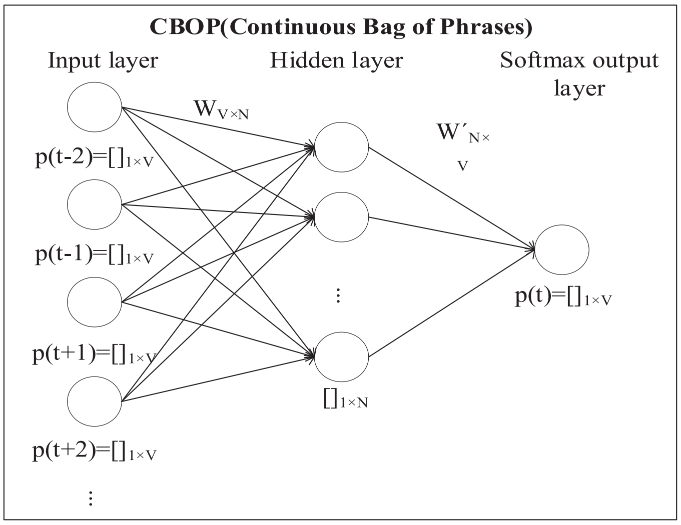
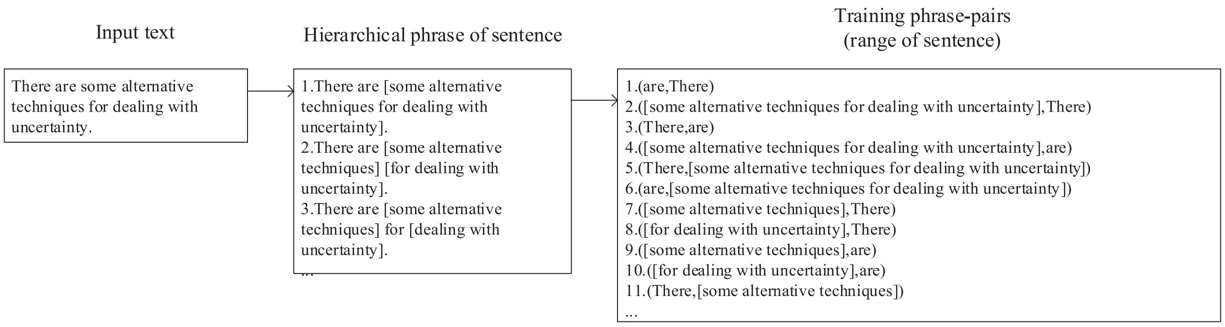


Fig. 10. The training process for CBOP.


Input text

There are some alternative techniques for dealing with uncertainty.

Hierarchical phrase of sentence

1.There are [some alternative techniques for dealing with uncertainty].
2.There are [some alternative techniques] [for dealing with uncertainty].
3.There are [some alternative techniques] for [dealing with uncertainty].
...

Training phrase-pairs (range of sentence)

1.(are,There)
2.([some alternative techniques for dealing with uncertainty],There)
3.(There,are)
4.([some alternative techniques for dealing with uncertainty],are)
5.(There,[some alternative techniques for dealing with uncertainty])
6.(are,[some alternative techniques for dealing with uncertainty])
7.([some alternative techniques],There)
8.([for dealing with uncertainty],There)
9.([some alternative techniques],are)
10.([for dealing with uncertainty],are)
11.(There,[some alternative techniques])
...

Fig. 11. The acquisition process of training phrase pairs in CBOP.

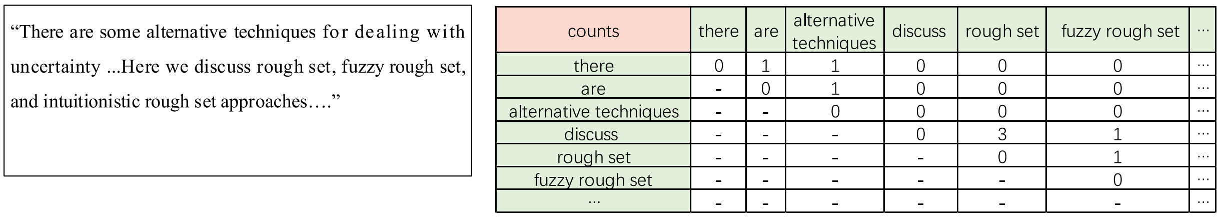
vector. So, we get an N-dimensional embedded space through the Skip-Phrase framework, in which each phrase in the phrase vocabulary is uniquely mapped to a point, as well as the cosine similarity of vectors indicates the semantic similarity between the phrases.

4.2.2. CBOP

In this section, we propose the CBOP model, which predicts a central phrase by contextual phrase. Fig. 10 shows the modification to the neural network architecture. In CBOP, the input is the one-hot representation of contextual phrases. Like Skip-Phrase, the context is defined as the scope of the sentence. In the training process of the neural network, we use the weight of the input layer to the hidden layer as the phrase embedding. Input and output are V-dimensional one-hot vectors. The number of hidden layer nodes determines the dimension of the phrase space. W_{V*N} denotes the weight matrix from the input layer to the hidden layer. W'_{N*V} indicates the weight of the hidden layer to the output layer. We feed the context phrase to the neural network and compare the output with the central phrase. We repeat the above process using multiple context phrases and update the weight. The formation process of the training pair is similar to Skip-Phrase, which defines the context as the inside of the sentence. By switching the center words, all training phrase pairs are obtained. The whole process is shown in Fig. 11. We also get a vector space where each phrase in the phrase vocabulary is uniquely mapped to a point, while the distance between the points denotes the semantic similarity between phrases.

4.3. GloVeFP: global vectors for phrase representation

In this section, we propose GloVeFP which learns the phrase vector by factorizing phrase co-occurrences matrix in a text corpus. It consists of the following steps. First, we extract the phrase co-occurrence matrix X from the corpus. As shown in Fig. 12, X_{ij} denotes the frequency that phrase-pair appears in the same sentence. For example, according to the example shown in Fig. 12, 'discuss' and 'rough set' frequently appear in the same sentence so that they may have some hidden relationship. In the traversal process, we define the sentence as a co-occurrence range, instead of windows_size. Then we



"There are some alternative techniques for dealing with uncertainty ...Here we discuss rough set, fuzzy rough set, and intuitionistic rough set approaches…."

counts	there	are	alternative techniques	discuss	rough set	fuzzy rough set	···
there	0	1	1	0	0	0	···
are	-	0	1	0	0	0	···
alternative techniques	-	-	0	0	0	0	···
discuss	-	-	-	0	3	1	···
rough set	-	-	-	-	0	1	···
fuzzy rough set	-	-	-	-	-	0	···
···	-	-	-	-	-	-	···

Fig. 12. The construction process of the phrase co-occurrence matrix in GloVeFP.

set the constraints of the phrase pair, shown in Eq. (6).

$$\mathbf{p}_i^T\mathbf{p}_j + b_i + b_j = \log(X_{ij}), \tag{6}$$

where $\mathbf{p}_i$ denotes the specific phrase vector. $\mathbf{p}_j$ expresses the context phrase vector. b_i and b_j are a deviation. Finally, the cost function is showed as Eq. (7).

$$J = \sum_{i=1}^{|P|}\sum_{j=1}^{|P|} f(X_{ij})(\mathbf{p}_i^T\mathbf{p}_j + b_i + b_j - \log(X_{ij})), \tag{7}$$

where f represents the weighting function, formulated as Eq. (8).

$$f(X_{ij}) = \begin{cases} \left(\frac{X_{ij}}{x_{max}}\right)^{\alpha}, if X_{ij} < x_{max} \\ 1, otherwise \end{cases} \tag{8}$$

For common phrases, return value cuts off its standard output and return 1, x_{max} is the co-occurrence threshold (Through experimentation, 100 is a better choice.). For other cases, the result is between 0 and 1, which depends on the value of α(usually set to 0.75). The GloVeFP captures the counts of overall statistics about phrase co-occurrence. We factorize this matrix to get a lower-dimension phrase representation, which maintains the similarity between phrases.

4.4. The complexity of the model

The computational complexity of phrase embedding comes mainly from two aspects: phrase mining and phrase embedding. PCFG and CKY parsing are the pre-processing work of our framework, which can be used multiple times after one-time processing. The size of treebank decides the time and space complexities of PCFG. It is $O(|C|)$, where $|C|$ is a constant indicating the number of words in the corpus. The time and space complexities of CKY are $O(|R|*|C|^3)$. $|R|$ denotes the number of rules. So, it is linearly related to $|C|^3$. The complexity of HPMBP is $O(V)$, where V means the size of the syntax tree and is a constant much smaller than $|C|$. The complexity is linear to $O(|C|)$. The complexity of the preprocessing stage is $O(|C|^3)$ but the one-time processing. The complexity of our proposed method is $O(|C|)$. Furthermore, HPMBP is based on sentences so it can be easily parallelized. Phrase vocabulary obtained through phrase mining can be used by other researchers.

CBOP relies on the number of phrases in the entire text (each time the prediction behavior will be a backpropagation, which is often the most time-consuming part). The size of the common phrase vocabulary is thinner than the word vocabulary, so the complexity of Phrase2Vec is less than $O(V)$. Skip-Phrase is more predictive than CBOP because each phrase is used as a central phrase, i.e., it must be predicted once using every surrounding phrase. Its complexity is comparable to CBOP (assuming average k phrases per sentence) $O(kV)$, and the training time is longer than CBOP. Compared to the complex Phrase2Vec, GloVeFP is a log bilinear model that yields excellent results with a simple loss function.

In conclusion, Phrase2Vec has lower computational complexity than Word2Vec, especially in large corpora. Studies have shown that, in complex NLP tasks, researchers tend to use multi-word phrases to express semantics rather than a combination of words [16]. Because the Phrase2Vec tasks are non-real-time, we can use their training results multiple times for other subsequent NLP processing tasks.

5. Phrase embedding based text classification

Text classification has a wide range of applications. The essential part of text classification is the vectorization of the text. Traditional text categorization methods adopt BOW to express text, so the vectorized representation of text relies on word embedding. We propose Phrase Embedding based Text Classification (PETC), which employs BOP to signify the text and incorporates Phrase2Vec into text classification to improve classification manifestation. Fig. 13 shows the architecture of PETC. It is split into two stages: model training and model evaluation. In the off-line model training, we preprocess the labeled unstructured text data $D = \{(d_1, t_1), (d_2, t_2), \ldots(d_{|D|}, t_{|D|})\}$, where d_i is a training text with its original label t_i, then convert them into structured data. We get the composition phrases for each sentence by HPMBP and represent the sentence according to the BOP model. According to the phrase representation of every sentence, we

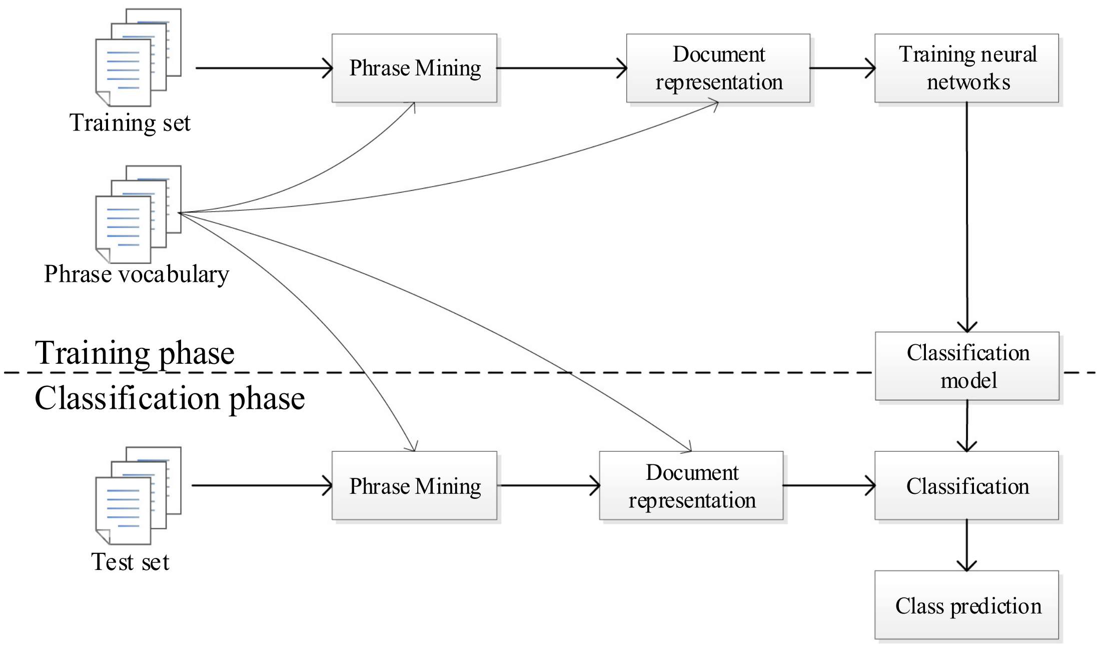


Fig. 13. The overall architecture of the PETC.

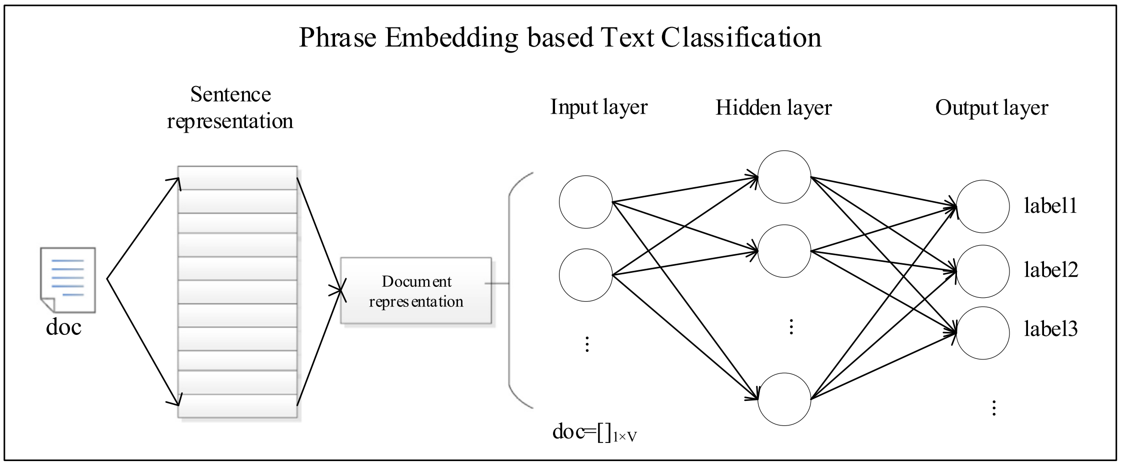


Fig. 14. The training process for PETC.

get the constituent phrases of the text and sort them in descending order according to the TFIDF value of phrases, $D=\{(\{p_{11},p_{12},...\},t_1),(\{p_{21},p_{22},...\},t_2),...(\{p_{|D|1},p_{|D|2},...\},t_{|D|})\}$, where p_{ij} denotes the jth phrase in the ith document. We use vectors to represent phrases $\mathbf{D}=\{(\{\mathbf{p}_{11},\mathbf{p}_{12},...\},t_1),(\{\mathbf{p}_{21},\mathbf{p}_{22},...\},t_2),...(\{\mathbf{p}_{|D|1},\mathbf{p}_{|D|2},...\},t_{|D|})\}$, where $\mathbf{p}_{ij}$ indicates the phrase vector. We use the RNN to combine phrase embeddings into document vectors $\mathbf{D}=\{(\mathbf{d}_1,t_1),(\mathbf{d}_2,t_2),...(\mathbf{d}_{|D|},t_{|D|})\}$. Then, we train the neural network with the processed text vector and the corresponding label as training data. Fig. 14 shows the overall framework of the neural network which includes the input layer, the hidden layer, and the output layer. The input layer has V nodes, corresponding to the V-dimensional vector of the text. The hidden layer also has N nodes fully connected to the input layer with a weighted summation function. The number of output layer nodes is the number of categories. By training the neural network, we get the classification model. In the real-time classification phase, we use HPMBP to obtain the vector representation of the text to be classified. If a phrase does not appear in the training phase during the parsing process, we will replace it with the phrase that is closest to it in phrase embedding. This mechanism makes PETC have a fuzzy classification effect and improve classification accuracy. Then, we adopt the text classification model obtained in the first stage to predict the text label. Finally, we evaluate the classifier by contrasting the original label and the prediction label of the text.

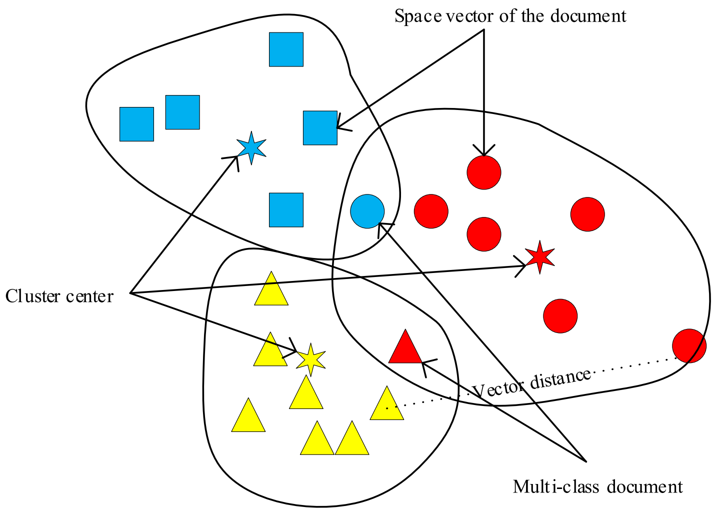


Fig. 15. A schematic diagram of PETCLU. (For interpretation of the references to color in the text, the reader is referred to the web version of this article.)

6. Phrase embedding based text clustering

Text clustering is the most classic unsupervised text mining method. Its training process does not require labeled data, so it is widely used in various research, such as topic clustering, news recommendation, etc. The key of the text clustering is the vectorization of the texts and cluster centers. Traditional text clustering methods utilize BOW technology to express text, so the vectorization of documents and cluster centers is implemented by word embedding. In this section, we put forward Phrase Embedding based Text Clustering (PETCLU), which introduces BOP to signify the text and incorporates Phrase2Vec into the text clustering to improve the quality of text clustering. Fig. 15 depicts the schematic diagram of text clustering, where shapes of different colors show the text in different clusters, and stars denote cluster centers. In the cluster training phase based on the K-means method, we iteratively update the cluster center and the labels of texts until the cluster center no longer changes. Algorithm 2 shows the algorithm pseudo-code of PETCLU. It is divided into three phases: initial definition, cluster training, cluster evaluation. In the initial definition phase, we adopt the BOP to denote the texts mathematically. According to the phrase composition of the sentence, we get the constituent phrases of the text, $D = \{d_1, d_2, ...d_{|D|}\} = \{\{p_{11}, p_{12}, ...\},\{p_{21}, p_{22}, ...\}...\{p_{|D|1}, p_{|D|2}, ...\}\}$, where p_{ij} is the jth phrase in ith document. We employ the RNN to combine phrase embeddings into document vectors $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \ldots \mathbf{d}_{|D|}\}$. The cluster centers are initialized to k random documents, as $C = \{c_1, c_2, \ldots c_k\} = \{d_1, d_2, \ldots d_k\}$. Then the vectors of the cluster centers are denoted as $C = \{\mathbf{d}_1, \mathbf{d}_2, ...\mathbf{d}_k\}$. We utilize cosine similarity to calculate the correlation between the text and the cluster center, shown as Eq. (9).

Algorithm 2 Phrase embedding based text clustering.

```
Input: Corpus $D = \{d_1, d_2, ...d_{|D|}\}$, Number of clusters $k$.
Output: Cluster center vectors $C = \{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_k\}$.
1: $D \leftarrow \{\mathbf{d}_1, \mathbf{d}_2, ...\mathbf{d}_{|D|}\}$ //combine phrase embeddings from Alg.1 by RNN
2: $C' \leftarrow$ Randomkdocuments$\{\mathbf{d}_1, \mathbf{d}_2, ...\mathbf{d}_k\}$// Initialize cluster center vectors
3: do
4:   $C \leftarrow C'$//updating the cluster center
5: for $d'$ in $D$//updating document cluster
6:      $dist_{\min} \leftarrow \infty$
7:       for $c'$ in $C'$//finding the most nearby cluster of $d'$
8:         $dist \leftarrow$ Dist$(d', c')$
9:           if $dist_{\min} > dist$
10:             $dist_{\min} \leftarrow dist$
11:             set $d'$ in the cluster $c'$
12:          End if
13:       End for
14:       End for
15:       for $c'$ in $C'$//calculating cluster centers
16:          updating $\mathbf{c}'$ by averaging all document vectors in the cluster
17:       End if
18:    End while
19: Until $C$==$C'$//Until the cluster center no longer changes
20: return $C$
```

Table 1
Statistics of the three corpora.

Corpus	Enwiki	DBLP	Yelp
Size	5.8G	2.47G	3.36G
Words	1.491B	635.33M	864.25M
Vocabularies	249,993	106,462	144,838

$$\mathrm{Dist}\left(d_i, c_j\right) = \frac{1}{\cos(\theta)} = \frac{\|d_i\| \left\|c_j\right\|}{\mathbf{d}_i \,.\, \mathbf{c}_j}, \tag{9}$$

where $d_i \in D$, $c_j \in C$. In the cluster training phase, we iteratively change the label of each document until the cluster center remains stable. In the cluster evaluation phase, we evaluate the clustering results through different evaluation indicators. The cluster model that meets assessment requirements is often used for real-time document clustering tasks. In Fig. 15, we also find a common problem in text clustering, which is called the multi-label cluster. It means that the distance from the text to multiple cluster centers is similar. Traditional text clustering methods set the sample into a specific category, which is called hard clustering. In this paper, when the distance from a sample to multiple cluster centers is less than a certain threshold, we use the degree in which the sample belongs to a cluster as the weight calculated by the cluster center, which is called soft clustering. In the cluster evaluation phase, we analyze the results of clustering by purity and entropy. The subsequent experiments show that PETCLU has individual competitiveness.

7. Experimental results

In this section, we certify the validity of phrase-based text representation by experiments. In Section 7.1, we introduce the data sets for phrase mining and evaluate the quality of phrases. In Section 7.2, we assess Phrase2Vec methods (Skip-Phrase, CBOP, and GloVeFP) in a similarity task and an analogical reasoning task on Enwiki, DBLP, and Yelp dataset, detailed in Section 7.1.1. In Section 7.3, we demonstrate the validity of PETC by comparing it with the most advanced text categorization on five data sets. In Section 7.4, we compare PETCLU with the trendiest text clustering methods on three baseline datasets. Section 7.5 summarizes the experimental results and brings out some potential application of Phrase2Vec in other NLP tasks.

7.1. Evaluation of phrase quality

7.1.1. Training corpora descriptions

We select three data sets as training corpora for phrase embedding. Table 1 lists the statistics of the data sets. (1) Enwiki is a dump of the content of the English Wikipedia page article.[3] We preprocess the raw data to extract article content for phrase embedding. (2) DBLP is a computer science bibliographic website that contains information on popular computer science papers. We preprocess the raw data to extract 3,079,007 titles and 2,548,532 abstracts for phrase embedding.[4] (3) The Yelp dataset is a corpus of business reviews. We preprocess the raw data to obtain 5,996,996 valid reviews for embedded learning.[5] In the preprocessing stage, we remove real numbers and special symbols. For convenience, we convert the words to lowercase. We define the range of sentences as the 'nearby' phrase.

7.1.2. Compared methods

To verify the validity of HPMBP, we choose three excellent baseline methods for comparison.

SegPhrases+ [20]: It proposes a phrase mining framework that extracts quality phrases from massive text corpora, integrated with phrasal segmentation.

AutoSegPhrase [33]: It proposes a phrase mining framework based on the POS tagger. Experiments show that it has individual competitiveness.

CQMine [17]: It proposes an efficient method for high-quality phrase mining, which combines topical cohesion.

7.1.3. Experimental setting

The first stage of Phrase2Vec is to mine phrases in the corpus, so we compare HPMBP with excellent phrase mining methods. The necessary hyperparameters in the algorithm implementation are as follows. The general parameter for all comparison methods is the frequency threshold of the phrase. When phrase frequency reaches the threshold, it satisfies the Popularity of the phrase [20]. It is proportional to the Precision and inversely proportional to the Recall. By comparing multiple experiments, we find that when the frequency threshold is 120, we get a compromise between precision and recall, shown in Fig. 16. For n-gram based methods (SegPhrases+ and AutoSegPhrase), n is set to 6. It is consistent with the previous study of the phrase mining [33]. HPMBP does not manually limit the length of the phrase during algorithm

[3] https://dumps.wikimedia.org/enwiki/.
[4] https://aminer.org/citation.
[5] https://www.yelp.com/dataset.

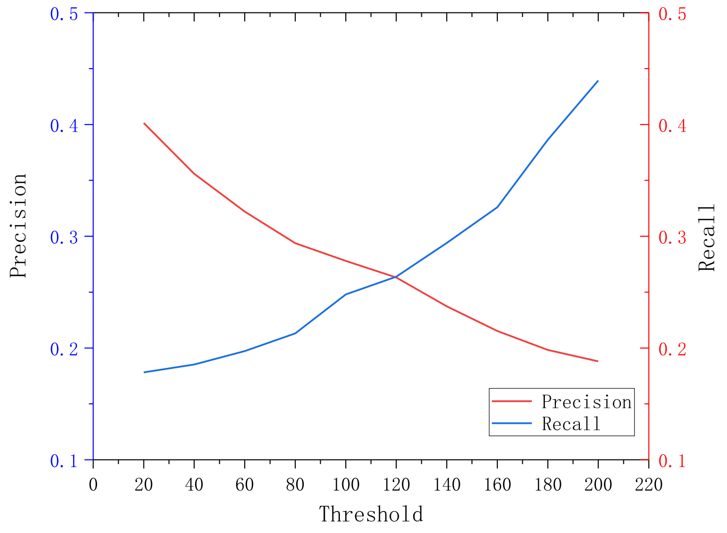


Fig. 16. The Precision and Recall of HPMBP at different frequency thresholds on EnWiki.

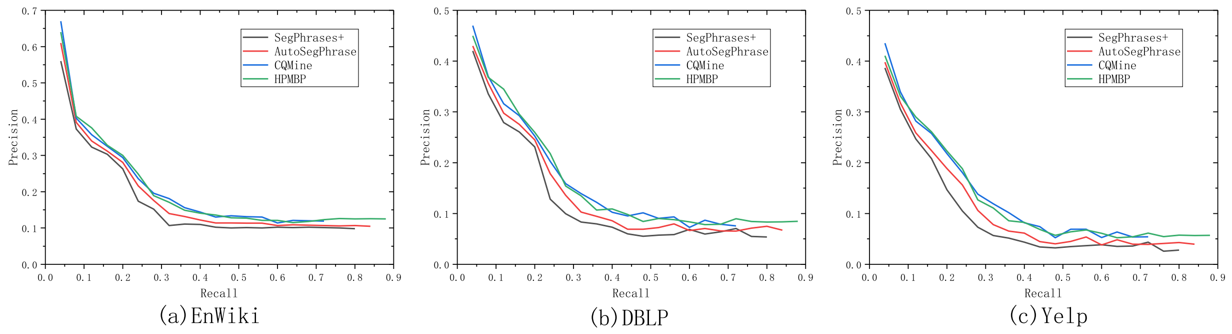


Fig. 17. Precision-recall curves of all baselines by Wiki-phrases on three datasets.

implementation. Parameters not explicitly stated in the comparison algorithms are from the original paper or its open-source project.

7.1.4. Phrase quality evaluation

To verify the quality of the phrases obtained by HPMBP, we use two metrics [17]: Wiki-phrases benchmark and Expert Evaluation.

Wiki-Phrases [20]: Wiki-phrases is a collection of popular mentions of entities by crawling intra-Wiki citations within Wiki content. It contains a large number of commonly used short phrases. In evaluation, we treat Wiki-phrases as ground truth phrases. If the phrase appears in Wiki-phrases, we mark it positive, otherwise negative. For precision and recall, we firstly merge all the phrases obtained by baseline methods. Then we get the intersection between the Wiki phrases and the merged phrases as the evaluation set. Fig. 17 shows the Precision-recall curves (PR-curves) of different methods evaluated by Wiki-phrases. The PR curves are created by plotting precision-recall pairs at various frequency thresholds. Comparing the different methods, we find that SegPhrase+ only based on frequency statistics is the worst. AutoSegPhrase considers the POS feature on SegPhrase+, so its performance has improved. CQMine achieves the best performance because it focuses on cohesive topic phrases. That is, CQMine and HPMBP consider the relationship of hierarchical phrases to some extent. HPMBP performs better than statistical or simple POS-based methods but delivers comparably to CQMine. The main reason is that HPMBP mines a large number of long hierarchical phrases that are not included in Wiki-Phrase. The excellent Recall of HPMBP also proves that it has specific mining performance for short common phrases. Comparing the performance of the baseline methods in different data sets, we get some interesting conclusions. Phrase mining algorithm performs better on EnWiki than other data sets. We infer that large data sets are more conducive to the performance improvement of phrase mining results. The overall performance of the baseline methods in DBLP is better than Yelp. The main reason is that the description of the title and abstract are more standardized than the comments. In summary, HPMBP performs comparably to the existing optimal phrase mining algorithms in commonly used short phrase extraction.

Expert Evaluation: Wiki-phrases mainly cover standard terms and professional noun phrases but do not include complex phrases or long hierarchical phrases with specific meanings, so we also conduct an expert evaluation to examine whether

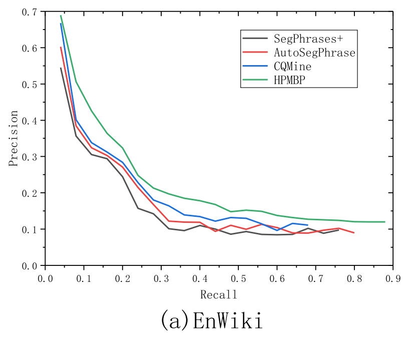


Fig. 18. Precision-recall curves of all benchmarks by Expert Evaluation on three datasets.

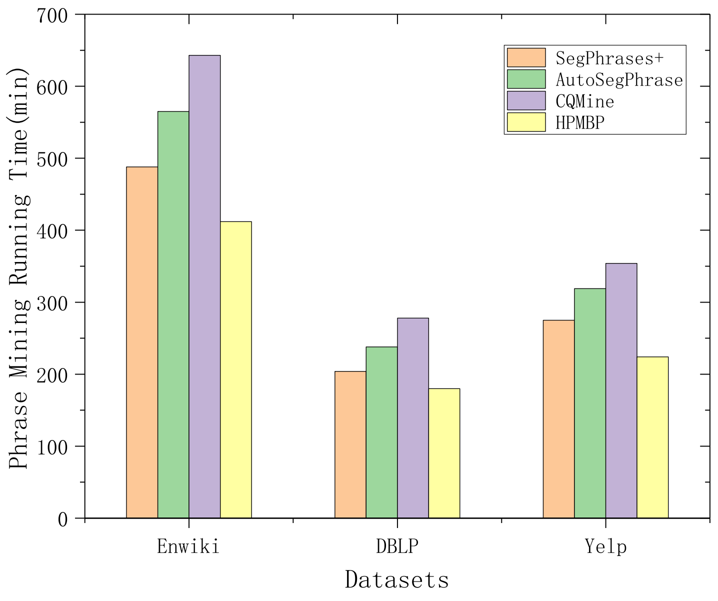


Fig. 19. The running time comparison of Phrase mining models.

the phrases are natural, meaningful, complete. Phrases are diverse and combinable so that expert evaluation can provide an excellent complement to Wiki-Phrases. For each dataset, we randomly sample 1000 phrases that Wiki-phrases uncovered. Five reviewers (computer science Ph.D. candidates in year two or above) independently mark the phrase positive or negative, based on their background knowledge and search engine. We summarize their markup results to calculate precision and recall. The Expert Evaluation results of the baseline methods in different data sets are shown in Fig. 18. Comparing the performance of diverse baseline methods, we find some exciting phenomena different from Wiki-Phrases. HPMBP outperforms other baseline methods about 5% in Precision at the same Recall because long hierarchical phrases ignored in Wiki-Phrases are positive in Expert Evaluation. SegPhrase+, AutoSegPhrase, and CQMine have limited ability to excavate long phrases, so their performance degrades. Comparing the performance of HPMBP in different data sets, we also find that it gets the optimum performance boost in DBLP, which indicates that there are a large number of standardized long hierarchical phrases in DBLP. It is more fluctuating in Yelp, which means that Yelp contains long hierarchical phrases with widely frequency distribution. In summary, HPMBP shows a particular advantage under the Expert Evaluation. The most important reason is that it focuses on long hierarchical phrases that humans can recognize.

Summarizing the above experiments, HPMBP has considerable short-phrase mining ability and has apparent advantages in hierarchical phrase mining. It is proved that HPMBP can effectively identify phrases in texts (sentences, paragraphs).

7.1.5. Performance analysis

To verify the performance of the HPMBP algorithm, we compare its runtime with the benchmark methods, shown in Fig. 19. By comparison, we find that HPMBP has a specific runtime advantage. In different data sets, HPMBP saves about 10% of the run time compared to the closest SegPhrases+. The main reason is that HPMBP relies on parsing to obtain candidate phrases, which optimizes the number of candidate phrases and improves operational efficiency. Both SegPhrases+ and AutoSegPhrase acquire candidate phrases based on n-gram methods and obtain a large number of meaningless word sequences. This process increases the running time. CQMine costs more time because it took extra time to guarantee the consistency of topics in the process of mining phrases. From the perspective of different data sets, we find that phrase mining tasks on Enwiki consume the most time because it has the most substantial volume of documentation. In summary, HPMBP has some performance advantages over other baseline algorithms.

7.2. Evaluation of phrase embedding

7.2.1. Compared methods

To verify the validity of Phrase2Vec (Skip-Phrase, CBOP, GloVeFP), we choose five excellent baseline methods for comparison.

SGNS [25]: Skip-Gram is a classical algorithm of the word-level representation method. SGNS adds subsampling and negative sampling to the original Skip-Gram to speed up the training process. It presents the idea of the earliest phrase vectorization and proves that learning phrase embedding is possible, so we select SGNS as a baseline method for word similarity and phrase similarity experiments.

PARAGRAM [39]: It learns word embedding from PPDB, then gets phrase embedding through RNN. It improves the word embedding obtained by Skip-Gram and proves that phrase embedding can be obtained by merging word embedding through RNN. Experiments show that RNN is an excellent combination.

SEING [36]: Based on the SGNS model, it defines continuous words that satisfy the constraint as a phrase, and does not pay attention to the combination of phrases.

SVO [10]: It learns the compositional and non-compositional phrases embedding, and then adopts the constitutive scoring function to get the final embedding. It can get the vector representation of the short phrase, but there are certain defects for the long-phrase.

HCPE [16]: It utilizes an implicit hierarchical structure to acquire the phrases compositionality and phrase embedding simultaneously. It shows that the phrase has a hierarchical structure but does not pay attention to the embedding of the hierarchical phrase.

7.2.2. Experimental setting

Most of the baseline methods come from the Skip-Gram architecture, so they all use a three-layer neural network for embedded learning. The hyperparameters of all the ways are derived from the original paper [24,25]. For optimization, we use the Negative Sampling technique to speed up training; then, all methods use the same vector dimension and default setting. Negative examples are limited to 25, as well as iterations (number of epochs) to be 5 [24]. The initial learning rate is set to 0.05. It determines the convergence speed of the algorithm. High vector dimensions can represent more remarkable semantic similarities but bring training consumption. Following the previous researchers [25], we set 300 as the dimension of the embedded space. The length of the context window is set to 5. The context of Phrase2Vec is defined as the range of sentences because it is the unit of phrase extraction. Parameters without specified instructions take the default values from the original article.

7.2.3. Evaluation of similarity task

In the similarity tasks, each model needs to compute semantic similarity (Cosine similarity) for any given the word/phrase pair. We compare the effects of the model through the correlation between the results and the gold standard. We adopt two correlation measurements: Pearson correlation γ and Spearman correlation ρ.

Pearson coefficient γ indicates the correlation between two sets of variables. Its value ranges between +1 and −1.0 means no association. An amount higher than 0 means the data is positively correlated, and vice versa means a negative correlation. It is employed to evaluate the accuracy of the phrase embedding. It is calculated by dividing the covariance of two variables by the product of their standard deviations. A pair of random variables (X, Y) is given, and the formula for γ is shown as Eq.(10).

$$\gamma = \frac{\mathrm{cov}(X,Y)}{\sigma X \sigma Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma X \sigma Y}, \tag{10}$$

where cov(**.**) denotes the covariance. σ indicates the standard deviation. The average is denoted by μ. $E[\ \cdot\ ]$ marks the expectation.

Spearman coefficient expresses a nonparametric measure of rank correlation. It adopts a monotonic function to represent the correlation of two variables, requiring data values to be unique. In general, the Spearman correlation coefficient is consistent with the Pearson correlation coefficient of the sorting variable. It is computed from Eq. (11).

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}, \tag{11}$$

where n denotes sample size. d_i is $g(X_i) - \mathrm{rg}(Y_i)$. X_i, Y_i is raw scores. $\mathrm{rg}(X_i)$, $\mathrm{rg}(Y_i)$is correlation rank.

We test our model on both word and phrase similarity tasks with the same trained vectors to demonstrate our model could effectively reduce the sparsity problem and improve word similarity and phrase similarity.

Table 2
The experiment compares our proposed Phrase2Vec methods (Skip-Phrase, CBOP, and GloVeFP) with the baseline methods. Results about the word similarity task on three corpora, the optimal results are signed in bold font.

Corpus	Model	SimLex-999		Sim-353		WP-130	
		ρ	γ	ρ	γ	ρ	γ
Enwiki	SGNS	0.356	0.350	0.622	0.613	0.554	0.487
	PARAGRAM	0.368	0.359	0.631	0.626	0.559	0.537
	SVO	0.348	0.338	0.645	0.626	0.540	0.507
	SEING	0.363	0.373	0.646	0.636	0.555	0.502
	HCPE	0.380	0.366	0.653	0.647	0.558	0.512
	Skip-Phrase	0.393	**0.378**	**0.659**	0.642	**0.578**	0.516
	CBOP	0.382	0.373	0.655	0.647	0.568	**0.519**
	GloVeFP	**0.401**	0.367	0.656	**0.662**	0.562	0.512
DBLP	SGNS	0.314	0.359	0.606	0.608	0.498	0.492
	PARAGRAM	0.333	0.393	0.621	0.624	0.507	0.513
	SVO	0.327	0.360	0.605	0.648	0.557	0.511
	SEING	0.362	0.364	0.646	0.631	0.548	0.503
	HCPE	0.372	0.361	0.649	0.640	0.548	0.505
	Skip-Phrase	**0.374**	0.364	0.649	0.638	0.550	**0.537**
	CBOP	0.370	**0.368**	**0.665**	0.665	**0.566**	0.529
	GloVeFP	0.369	0.364	0.650	**0.669**	0.563	0.509
Yelp	SGNS	0.355	0.289	0.575	0.552	0.542	0.499
	PARAGRAM	0.374	0.315	0.583	0.574	0.551	0.522
	SVO	0.364	0.319	0.585	0.574	0.556	0.524
	SEING	0.369	0.316	0.586	0.570	0.563	0.527
	HCPE	0.381	0.315	0.587	0.579	0.565	0.527
	Skip-Phrase	0.384	0.320	0.589	0.581	**0.571**	0.530
	CBOP	0.382	0.318	**0.608**	**0.586**	0.566	**0.537**
	GloVeFP	**0.391**	**0.325**	0.589	0.580	0.567	0.535

Word Similarity: We learn phrase embedding on the Enwiki, DBLP, and Yelp corpora. Then we validate the phrase embedding with γ and ρ on the three datasets: (1) SimLex-999[6] is used to evaluate the similarity of word meaning. (2) Sim-353[7] involves 353 word-pairs. Each word pair is labeled as a similarity score by a human expert. (3) WP-130 [18] contains 130 words/phrases pairs. Field experts manually mark similarity scores.

Table 2 displays the upshots of word similarity tasks at three corpora. Phrase2Vec models achieve better performance than other baselines. When the training corpus is Enwiki, Skip-Phrase behaves with the best results comparing with other embedding methods. In DBLP or Yelp, CBOP achieves the optimal result. Compared with baseline word embedding methods, Skip-Phrase, CBOP, and GloVeFP are superior in word similarity tasks. Through the analysis of the results, we find that SGNS can obtain word embedding, but the following methods have improved it and achieved better results. PARAGRAM gets word embedding through PPDB, so it is better than SGNS. SEING, SVO, and HCPE are all phrase embedding methods that rectify word embedding while implementing phrase embedding and incorporate words that should not appear alone into phrase unit statistics, so they perform better than simple word embedding methods. Our proposed Phrase2Vec methods focus on the hierarchical phrase embedding in the corpus, which makes a second correction to the word embedding. They optimize the phrase set and correct the combined phrase of words and phrases, so they are ahead of the approaches that only focus on basic phrases. From the perspective of validating the data set, the result of the Sim-353 data set is better than the other two data sets. It indicates that the word similarity is more reasonable in the Sim-353. From the perspective of training corpus selection, the phrase embedding obtained by Enwiki has a higher correlation index. Analyzing specific statistics for different corpora, we find that Enwiki is the most suitable training corpus. We infer that the effect of phrase embedding is proportional to the number of the learning corpus. Summary of experimental results, the three Phrase2Vec methods are superior to the existing word embedding methods in word similarity tasks.

Phrase similarity: For phrase similarity tasks, we adopt three test datasets: (1) TR9856[8] contains human-labeled phrase-relatedness values for 9856 phrase pairs. (2) PhraseSim[9] provides human-rated similarity at the scale of 0 to 9 for 324 phrases pairs. (3) WP-300 [18] includes 200 phrase pairs and 100 word-pairs with human-assigned similarity judgment.

Table 3 lists the consequents of phrase similarity tasks on three corpora. Phrase2Vec achieves the best performance. When the training corpus is Enwiki or DBLP, CBOP demonstrates the optimal results compared to other embedding models. When the training corpus is Yelp, GloVeFP performs best. Compared with other phrase embedding methods, Skip-Phrase, CBOP, and GloVeFP have achieved better results. SGNS has the worst outcome because it only aggregates word embedding to get phrase embedding. The result of PARAGRAM is better than SGNS because it uses the RNN method to get the phrase

[6] https://fh295.github.io/simlex.html.
[7] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/.
[8] https://www.research.ibm.com/haifa/dept/vst/debating_data.shtml.
[9] http://homepages.inf.ed.ac.uk/mlap/index.php?page=resources.

Table 3
Results on the phrase similarity tasks, the best results are marked in bold font.

Corpus	Model	TR9856		PhraseSim		WP-300	
		ρ	γ	ρ	γ	ρ	γ
Enwiki	SGNS	0.474	0.498	0.709	0.717	0.644	0.620
	PARAGRAM	0.485	0.523	0.724	0.732	0.664	0.645
	SVO	0.496	0.552	0.783	0.727	0.672	0.673
	SEING	0.505	0.527	0.729	0.775	0.683	0.651
	HCPE	0.520	0.580	0.812	0.830	0.691	0.710
	Skip-Phrase	0.607	**0.661**	0.852	0.849	**0.774**	0.786
	CBOP	0.595	0.622	0.836	**0.909**	0.761	**0.789**
	GloVeFP	**0.610**	0.591	**0.871**	0.889	0.749	0.754
DBLP	SGNS	0.444	0.487	0.739	0.822	0.624	0.618
	PARAGRAM	0.472	0.493	0.752	0.847	0.652	0.674
	SVO	0.485	0.559	0.780	0.861	0.673	0.684
	SEING	0.512	0.536	0.832	0.864	0.662	0.689
	HCPE	0.513	0.573	0.829	0.871	0.706	0.744
	Skip-Phrase	0.588	0.634	0.846	0.897	0.784	**0.816**
	CBOP	0.586	**0.648**	0.845	**0.901**	**0.794**	0.777
	GloVeFP	**0.633**	0.609	**0.868**	0.887	0.741	0.769
Yelp	SGNS	0.497	0.518	0.696	0.783	0.595	0.608
	PARAGRAM	0.521	0.532	0.721	0.803	0.605	0.654
	SVO	0.541	0.557	0.785	0.831	0.634	0.664
	SEING	0.532	0.542	0.736	0.848	0.656	0.684
	HCPE	0.531	0.559	0.789	0.832	0.670	0.693
	Skip-Phrase	0.574	**0.629**	0.806	0.851	0.742	0.792
	CBOP	0.556	0.589	0.797	**0.881**	0.763	**0.803**
	GloVeFP	**0.638**	0.601	**0.819**	0.864	**0.767**	0.768

Table 4
Precision@top-1 result in the analogical reasoning task; the optimal results are marked in bold font.

Corpus	Dimension	SGNS	SVO	SEING	HCPE	Skip-Phrase	CBOP	GloVeFP
Enwiki	100	32.88	33.72	34.9	35.11	**38.12**	35.12	37
	200	36.12	36.37	34.81	36.28	36.79	38.54	**38.78**
	300	34.94	37.44	37.07	37.72	37.93	**38.9**	38.81
DBLP	100	28.43	29.82	29.46	30.68	**32.61**	31.49	30.74
	200	30.94	31.04	30.24	31.53	**33.02**	31.46	31.8
	300	28.96	30.36	31.12	32.92	33.19	32.75	**33.92**
Yelp	100	29.91	31.03	31.86	33.72	34.02	**34.28**	33.91
	200	31.96	32.54	32.53	34.05	35.3	34.07	**36.95**
	300	29.62	30.63	31.64	34.98	35.48	**35.99**	35.07

embedding. SVO, SEING, and HCPE all consider the compositionality characteristics of phrases to varying degrees, so the result is better than the word embedding combination method. Our proposed Phrase2Vec methods consider the embedding of hierarchical phrases, which is equivalent to multiple combinations of phrases, so they achieve the best performance. In different verification data sets, it displays that the general performance of the PhraseSim is better than the others. It indicates that the phrase similarity between phrases is more reasonable in the PhraseSim. In different embedded training sets, we find that the performance of the Phrase2Vec methods is similar. We infer that under smaller verification data sets, it is difficult to obtain substantial similarity differences. Through the experimental results, it is proved that hierarchical phrases widely exist in the real corpus, and the mining of hierarchical phrases can also effectively improve the accuracy of phrase embedding.

7.2.4. Evaluation of analogical reasoning task

In the analogical reasoning task, each model needs to infer unknown words/phrases based on the relationship of known words [25]. We adopt Google semantic analogy dataset as the validation data set,[10] which contains 3223 phrase analogy questions among five relations. We use the precision@top-1 as measurement, which means the model is only allowed to return its top-1 similar phrase as an answer. Table 4 reports the results of the phrase embedding models.

Table 4 shows that the three Phrase2Vec methods are generally superior to other methods. Skip-Phrase, CBOP, and GloVeFP show slightly different performance under different training corpora in different dimensions. Under the condition of Enwiki as the training corpus, the performance of the phrase embedding methods is generally better than the other two corpora. We speculate that the return of the analogical reasoning task is more affected by the corpus size. The effect of the

[10] code.google.com/p/word2vec/source/browse/trunk/questions-words.txt.

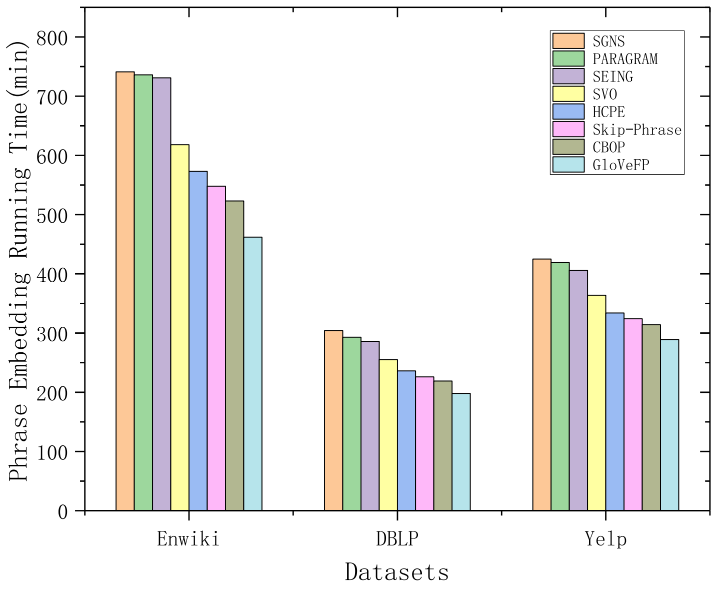


Fig. 20. The running time comparison of Phrase embedding models. The dimension of the vector space is 300.

Table 5
Related information of five datasets.

Statistics	BBC	20NG	Reuters-21578	RCV1-RCV2	Amazon
File size	2.7M	16.53M	26.7 M	2.5G	18G
Vocabulary Size	13,597	32,716	51,552	107,756	3,450,041
Doc Number	2225	19,997	21,578	111,740	34,686,770
Class Number	5	20	5	6	6
Word Counts	0.69M	4.25M	6.87M	643.05M	4631.03M
Training/Testing Splits	1780/445	15,998/3999	17,262/4316	89,392/22,348	27,749,416/6,937,354

analogical reasoning task is positively related to the size of the corpus. Skip-Phrase performs better in low-dimensional embedded representations. CBOP and GloVeFP perform better in high-dimensional vectorization. We find that the Phrase2Vec methods are superior to the existing embedding methods in an analogical reasoning task.

Combining the experimental results of the similarity task and analogical reasoning task, it is proved that Phrase2Vec models are superior to the existing embedding methods. They can effectively solve the problem of phrase embedding.

7.2.5. Performance analysis

We also have some interest in the performance of Phrase2Vec. Fig. 20 shows the runtime comparison of different phrase embedding methods. Our proposed Phrase2Vec models (Skip-phrase, CBOP, and GloVeFP) are ahead of other baseline methods at least 3% in running time. The critical factor is that Phrase2Vec models set the scope of the context to the inside of the sentence, which effectively improves the training efficiency. Traditional word embedding methods (SGNS, PARAGRAM, and SEING) have the same network architecture, so they have a similar running time. SVO and HCPE consider the combination of phrases in the phrase embedding process, so their runtime is improved. Phrase2Vec models enhance the definition of context scope and consider hierarchical phrases, so they achieve the best performance. Comparing different data sets, we find that the production of phrase embedding is proportional to the size of the data set. To sum up, the performance of the Phrase2Vec models is superior to the existing embedding methods.

7.3. Evaluation of phrase embedding based text classification

We propose PETC that incorporates Phrase2Vec models into a neural network classification model. In this part, we contrast PETC with the representative text categorization models on five data sets.

7.3.1. Data set descriptions

We choose Enwiki with a better overall effect as the phrase training corpus, then get three PETC models with different phrase embedding methods: $\text{PETC}_{\text{Skip-Phrase}}$, $\text{PETC}_{\text{CBOP}}$, and $\text{PETC}_{\text{GloVeFP}}$. To get a fair classification result, we select five real classification data sets. The following detail information is listed in Table 5.

BBC[11] contains 2225 BBC news documents from 2004–2005. They are divided into five areas: tech, sport, politics, business, and entertainment [9].

20Newsgroups[12] is an integration of 20,000 news documents from 20 different categories. We use a subset of 20 newsgroups that remove duplicates and sort the titles. The corpus includes 18,846 texts. The vocabulary size achieves 32,716, keeping the words that appear more than five times.

Reuters-21,578[13] includes Reuters newswire documents in 1987 with five categories: EXCHANGES, ORGS, PEOPLE, PLACES, and TOPICS. It is often used as a baseline data set for text classification experiments.

RCV1-RCV2[14] contains Reuters news stories, often used in comparative experiments on natural language processing tasks. We employ 810,000 English news stories. We choose the most commonly six categories (E21, CCAT, GCAT, ECAT, C15, and M11) as the classification labels.

Amazon[15] involves 142.8 million commodity comments between 1996 and 2014. We select the top six product categories as the category label: Books, Electronics, Movies, CDs, Clothing, and Kitchen.

During the experiment, we partition the data set according to the 2/8 ratio. To obtain accurate results, we split the data set into training/testing sub-datasets 10 times, then calculate the metrics separately, finally receive the final classification results by averaging.

7.3.2. Compared methods

To show the property and efficiency of PETC methods (PETC$_{\text{Skip-Phrase}}$, PETC$_{\text{CBOP,}}$ and PETC$_{\text{GloVeFP}}$), we contrast them with the following seven models.

PROJ [40]: It compares multiple methods to achieve sentence embedding, and experiments prove that PROJ has a strong performance. We employ PROJ instead of RNN to implement sentence embedding based the Skip-Gram, then compare the text classification results.

LazyNN_RF [31]: It improves the traditional random forest classifier for the high-dimensional noise classification task.

WESTClass [23]: It uses a weakly supervised approach to solve the lack of training data in text classification.

SPEA2SVM [5]: It cuts back the number of meta-features and improves the effectiveness of classification through multi-objective strategies.

Fast-Dep.-LLDA [3]: It integrates probabilistic generative models and deep learning to prevent overfitting and achieves better classification accuracy.

FBoWC [45]: It proposes a fuzzy semantic similarity to improve the accuracy of the classification algorithm.

XGBoost+Word2Vec [35]: It uses Word2Vec and XGBoost for text categorization.

7.3.3. Experimental setting

Based on different Phrase2Vec models, we implement three PETC algorithms. Following the experimental results of phrase embedding, we choose Enwiki as the embedding corpus and 300 as the dimension. PETC uses a traditional three-layer neural network architecture. The number of input layer nodes is 300, which is the same as the dimension of the embedded result [25]. The number of hidden layer nodes reflects the complexity of the network and determines the training time. There is no uniform standard for the number of hidden layer nodes. We set it 20, which is approximately equal to the square root of the number of nodes in the input layer [23]. The size of the output layer node is the number of classification labels. The learning rate affects neural network training time. Our initial learning rate is 0.1, 0.05 for the learning rate step [12]. When it is 0.5, the PETC achieves better results. The hyperparameters of the baseline methods not mentioned are derived from the original paper.

7.3.4. Classification evaluation

To verify the effect of PETC, we assess different classification models by comparing the prediction labels of the text with the original tags. F1 score is the classic classification indicator. During the experiment, we divide the training set 10 times and calculate the F1 value, respectively. Finally, the average is used as the final evaluation index.

Table 6 shows that the three PETC methods outperform other baselines at least 5% in F1-value. PETC$_{\text{GloVeFP}}$ achieves optimal performance. In most cases, word-based text categorization methods are weaker than phrase-based text representation methods (PROJ and PETC). It indicates that the phrase retains more document semantics in the classification task. There are two reasons why PETC outperforms PROJ. First, the phrase vocabulary of PETC is derived from HPMBP, while PROJ is based on a combination of words, regardless of the independent semantics of the phrase. Second, some literature suggests that embedding phrases through RNN is more conducive to text classification tasks [12,40]. Because PETC is based on hierarchical phrase embedding, we prove that hierarchical phrases can better preserve semantics in a real document. Comparing different corpora, we conclude that the performance of the text classification task is proportional to the size of the classification data set. In the case of the same data set size, as the number of categories increases, the accuracy decreases. Experiments

[11] http://mlg.ucd.ie/datasets/bbc.html.

[12] http://qwone.com/~jason/20Newsgroups/.

[13] https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection.

[14] https://archive.ics.uci.edu/ml/datasets/Reuters+RCV1+RCV2+Multilingual+Multiview+Text+Categorization+Test+collection.

[15] http://jmcauley.ucsd.edu/data/amazon/.

Table 6
Comparison of average F1 values of different methods on five datasets, the best results are marked in bold font.

Model	BBC	20NG	Reuters-21578	RCV1-RCV2	Amazon
PROJ	83.02	80.27	85.29	83.84	87.93
LazyNN_RF	78.18	73.77	78.06	82.05	83.05
WESTClass	81.23	75.2	76.41	81.72	80.49
SPEA2SVM	76.3	76.09	81.48	78.42	82.85
Fast-Dep.-LLDA	78.01	76.57	81.96	79.24	83.55
FBoWC	82.36	79.97	82.16	84.93	86.21
XGBoost+Word2Vec	80.58	78.35	78.85	82.21	83.83
$\text{PETC}_{\text{Skip-Phrase}}$	85.97	83.32	**91.23**	87.46	94.58
$\text{PETC}_{\text{CBOP}}$	**87.25**	84.21	88.77	86.73	89.34
$\text{PETC}_{\text{GloVeFP}}$	87.15	**84.95**	89.22	**91.93**	**95.51**

Table 7
The running time of different classification methods. The best results are marked in bold font. The following data is in minutes.

Model	BBC	20NG	Reuters-21578	RCV1-RCV2	Amazon
PROJ	1.95	6.03	2.53	13.84	887.02
LazyNN_RF	1.75	5.91	2.32	14.26	870.61
WESTClass	1.84	5.83	2.48	14.83	864.93
SPEA2SVM	1.79	5.72	2.38	13.45	860.38
Fast-Dep.-LLDA	1.86	5.54	2.29	13.74	863.58
FBoWC	1.72	5.68	2.64	15.83	873.37
XGBoost+Word2Vec	1.83	5.92	2.53	14.24	857.48
$\text{PETC}_{\text{Skip-Phrase}}$	**1.55**	5.28	2.13	12.27	853.93
$\text{PETC}_{\text{CBOP}}$	1.58	5.37	**2.09**	**12.18**	**849.23**
$\text{PETC}_{\text{GloVeFP}}$	1.57	**5.16**	2.16	12.26	850.78

show that the phrase-based text representation has a positive influence on the text classification task. Also, PETC methods are better than the previous word-based classification algorithms.

7.3.5. Performance analysis

We obtain three PETC models by applying Phrase2Vec to text classification. Table 7 demonstrates the running time details of different text categorization methods. Because the three PETC models all adopt phrase-based text representation, they have similar running time. Compared to other baseline methods, PETCs save at least 1% of time consumption. The main reason is that they acquire document vectors based on Phrase2Vec, which is more efficient than word embedding. PROJ considers the embedding of sentences, so various sentence representations cause its inefficiency. Text classification tasks based on traditional machine learning algorithms (LazyNN_RF, WESTClass, SPEA2SVM, and Fast-Dep.-LLDA) focus on improving classification accuracy through data preprocessing while ignoring operational efficiency. FBoWC and XGBoost+Word2Vec consider the performance of text categorization, but they all utilize word-based text representations, which lead them to have low operational efficiency. Through the above analysis, it is stated that PETC models have certain classification performance advantages.

7.4. Evaluation of phrase embedding based text clustering

To verify the importance of Phrase2Vec for downstream unsupervised NLP tasks, we propose the PETCLU approach that integrates Phrase2Vec into the K-means model. In this part, we demonstrate the validity of PETCLU on three data sets.

7.4.1. Data set descriptions

We choose Enwiki with better full effect as the phrase embedding corpus and get three PETCLU models with different phrase embedding methods: $\text{PETCLU}_{\text{Skip-Phrase}}$, $\text{PETCLU}_{\text{CBOP}}$, and $\text{PETCLU}_{\text{GloVeFP}}$. We select three widely used cluster datasets as baselines. Details are as follows.

TDT2 data set[16] comes from nine bilingual news sites over six months. It contains more than 9394 samples, each with an average of 36,771 words. The preprocessed TDT2 document set is used in our experiment.

Reuters-21578[17] is collected in the 1987 Reuters news line. It is often used for text categorization and text clustering tasks.

20Newsgroups[18] contains 18,846 documents with a total of 32,716 words. We adopt it as the baseline data set of the cluster task.

[16] https://catalog.ldc.upenn.edu/LDC2001T57.
[17] https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection.
[18] http://qwone.com/~jason/20Newsgroups/.

7.4.2. Compared methods

Three popular clustering methods were used in our experiments to compare with PETCLU.

STCNN [41]: It solves the short text clustering through an improved convolutional neural network framework.

STDDC [2]: It proposes an automatic encoder network to solve text clustering.

CFAN [28]: It accomplishes text clustering by constructing a matrix of neighbor graphs of similar text.

7.4.3. Experimental setting

Following different Phrase2Vec models, we achieve three PETCLU algorithms with the optimal embedding result. They extend the K-means method and calculate the distance through the text embedding. We choose Enwiki as the phrase embedding corpus. According to previous studies [25], the embedding dimension is 300. The choice of hyperparameter K refers to the number of real tags in the training set [2,41]. In the comparative experiment, the samples in the data set have clear category labels, so PETCLU does not consider fuzzy clustering parameters. The fuzzy clustering performance of PETCLU will be explored in future research. The parameters of other baseline methods are derived from the original paper.

7.4.4. Clustering evaluation

In our experiments, Entropy and Purity are used for clustering evaluation indicators. Entropy measures the distribution of samples in each cluster. High-quality text clustering results have smaller entropy values. It is defined as Eq. (12).

$$\text{entropy}(\Omega) = \sum H(\omega)\frac{N_\omega}{N}, \tag{12}$$

where clusters are denoted as $\Omega = \{\omega_1, \omega_2, ..\omega_k\}$. $H(\omega)$ is the entropy of ω. N_ω means the number of texts in the group ω. N indicates the total number of points. The cluster entropy is shown as Eq. (13).

$$H(\omega) = -\sum_{c\in C} P(\omega_c) * \log_2(P(\omega_c)) \tag{13}$$

where C is real category set. $P(\omega_c)$ is the probability of a text being classified as c in the cluster ω.

The second measure is Purity, which shows the extent of clusters contains a single class. The clustering result is proportional to the purity value. We find the label with the most samples in each group. Then we calculate the purity by dividing the number of label samples by the total number of samples in the group. It is formulated as Eq. (14)

$$\text{purity}(\Omega) = \frac{1}{N}\sum_{\omega\in\Omega}\max_{c\in C}(\omega_c) \tag{14}$$

where ω_c is the number of samples labeled c in ω.

Table 8 shows that the three PETCLU methods perform better than other clustering methods in Entropy and Purity. Although the PETCLU methods obtained by different phrase embedding methods are distinct, they are better than other baselines. Neural network-based STCNN and STDDC use word embedding to represent text, which leads to aggregate text into the incorrect category. CFAN constructs a text similarity graph based on word embedding, which loses the phrase meaning in the original semantic text, resulting in poor clustering results. Focusing on each clustering method separately, we find that clustering performance is inversely proportional to the number of clusters k. In different cluster corpora, the clustering methods perform significantly better in TDT2 than other corpora. We infer that the text distribution in the TDT2 data set is regular. Experiments show that text representation and text distance calculation are the key of text clustering tasks. In summary, Phrase2Vec effectively improves text clustering performance.

7.4.5. Performance analysis

We obtain three PETCLU models by applying Phrase2Vec to text clustering. Experiments analyze the clustering performance of PETCLU and other baseline methods, shown in Fig. 21. By comparison, we find that the performance of the three PETCLU models is similar and ahead of other text clustering methods at least 4.5% in running time. The principal cause is that PETCLU uses phrase-based text representations, while other methods are based on words. STCNN and STDDC adopt neural network architecture to complete text clustering. However, word-based input increases training time. CFAN uses the graph to perform text clustering, which affects the overall clustering performance. Comparing different datasets, we find that the performance of clustering is also affected by the amount of data and the number of target categories. The above experiments demonstrate that the PETCLU models are superior to other clustering methods in terms of running time.

7.5. Summary and probable applications

Summarizing the above experimental results, we verify that Phrase2Vec is a valid text representation method that employs phrases to represents text, thus preserving the core text semantics. It has favorable phrase mining and embedding performance, contrasting with state-of-the-art phrase techniques. Phrase2Vec can also seamlessly integrates with subsequent NLP tasks. Comparative experiments show that PETC and PETCLU based on Phrase2Vec are superior to baseline methods.

Besides, Phrase2Vec can be potentially used for other NLP tasks as well. As shown in Fig. 1, the text representation is the primary phase of other NLP tasks. Next, we propose some ideas for applying Phrase2Vec to other tasks. Following

Table 8
Entropy and purity for compared methods on three datasets. Boldface indicates the best performance. CL1, CL2, and CL3 are short for PETCLU$_{\text{Skip-Phrase}}$, PETCLU$_{\text{CBOP}}$, and PETCLU$_{\text{GloVeFP}}$, respectively.

TDT2

k	Entropy						Purity					
	STCNN	STDDC	CFAN	CL1	CL2	CL3	STCNN	STDDC	CFAN	CL1	CL2	CL3
2	0.000	0.000	0.000	0.000	0.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000
3	0.228	0.127	0.017	0.000	0.000	0.000	0.887	0.879	0.841	0.923	0.943	0.931
4	0.313	0.255	0.117	0.158	0.083	0.142	0.902	0.875	0.855	0.893	0.915	0.877
5	0.319	0.271	0.152	0.103	0.180	0.127	0.798	0.818	0.865	0.924	0.905	0.906
6	0.431	0.245	0.141	0.088	0.153	0.147	0.840	0.752	0.837	0.947	0.828	0.846
7	0.364	0.222	0.213	0.104	0.222	0.085	0.788	0.755	0.774	0.837	0.846	0.829
Avg.	0.276	0.187	0.107	**0.076**	0.106	0.084	0.869	0.847	0.862	**0.921**	0.906	0.898

Reuters-21578

k	Entropy						Purity					
	STCNN	STDDC	CFAN	CL1	CL2	CL3	STCNN	STDDC	CFAN	CL1	CL2	CL3
2	0.002	0.002	0.002	0.000	0.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000
3	0.255	0.141	0.019	0.027	0.002	0.008	0.849	0.857	0.806	0.864	0.950	0.851
4	0.307	0.318	0.154	0.189	0.112	0.142	0.832	0.826	0.837	0.875	0.822	0.919
5	0.318	0.270	0.165	0.100	0.102	0.151	0.801	0.854	0.806	0.816	0.840	0.835
6	0.453	0.286	0.213	0.193	0.160	0.159	0.749	0.680	0.887	0.817	0.744	0.838
7	0.381	0.278	0.244	0.114	0.222	0.144	0.722	0.743	0.815	0.815	0.761	0.850
Avg.	0.286	0.216	0.133	0.104	**0.100**	0.101	0.825	0.827	0.859	0.865	0.853	**0.882**

20 Newsgroups

k	Entropy						Purity					
	STCNN	STDDC	CFAN	CL1	CL2	CL3	STCNN	STDDC	CFAN	CL1	CL2	CL3
2	0.003	0.003	0.011	0.003	0.001	0.007	0.991	0.999	0.991	0.990	0.999	1.000
3	0.266	0.153	0.024	0.046	0.019	0.018	0.836	0.851	0.791	0.857	0.945	0.844
4	0.307	0.335	0.158	0.200	0.117	0.146	0.825	0.818	0.827	0.865	0.819	0.907
5	0.331	0.272	0.173	0.101	0.105	0.168	0.796	0.848	0.791	0.805	0.839	0.827
6	0.462	0.302	0.230	0.198	0.170	0.178	0.744	0.670	0.887	0.811	0.839	0.835
7	0.398	0.284	0.254	0.130	0.238	0.150	0.708	0.728	0.801	0.813	0.754	0.749
Avg.	0.295	0.225	0.142	0.113	**0.108**	0.111	0.817	0.819	0.848	0.857	**0.866**	0.860

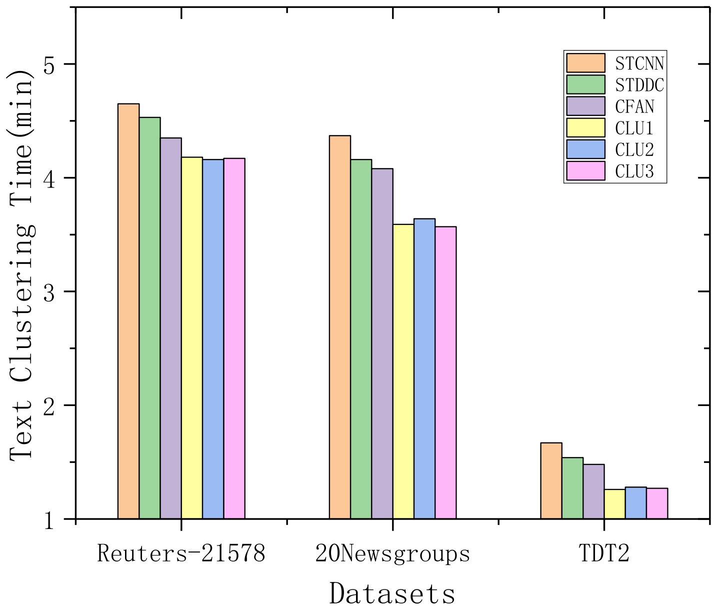


Fig. 21. Running time comparison of different clustering methods. CL1, CL2, and CL3 are short for PETCLU$_{\text{Skip-Phrase}}$, PETCLU$_{\text{CBOP}}$, and PETCLU$_{\text{GloVeFP}}$, respectively.

the previous researcher [11,27], we believe Phrase2Vec can improve named entity recognition. E.g. It can represent the correlation between named entities or find similar entities. Referring to related literature [21,30], we consider Phrase2Vec can also ameliorate social network analysis tasks. E.g. It is able to analyze the semantic relationship of social information posted by users to better predict user behavior. Of course, Phrase2Vec can also promote text representations in different granularities, such as sentences [44], paragraphs [13], etc.

8. Conclusions

To sum up, we propose a novel phrase-based text representation. First, we introduce HPMBP which utilizes parsing technology to mine hierarchical phrases from the corpus, then construct a phrase base. Based on the phrase base, we propose three Phrase2Vec methods: Skip-Phrase, CBOP, GloVeFP. They learn the phrase vector with semantic similarity, thus achieve the phrase embedding. Then we apply the Phrase2Vec to text classification and text clustering, thus propose PETC and PETCLU. PETC utilizes the Phrase2Vec to obtain the vector of the text, which is fed to the classification network. The fuzzy matching problem for unknown phrases is also considered. PETCLU exploits the Phrase2Vec to express text and cluster center, then extends the K-means model for text clustering. Experiments with a similarity task and analogical reasoning task show that the Phrase2Vec methods outperform Word2Vec methods on Enwiki, DBLP, and Yelp dataset. Comparative experiments between PETC and the baselines demonstrate that Phrase2Vec improves the text classification performance. Another set of experiments among PETCLU and the optimal text clustering methods illustrate that PETCLU obtains excellent clustering results. In summary, Phrase2Vec is an excellent text representation model and improves the performance of text categorization and text clustering. Finally, we also discuss the potential application of Phrase2Vec in other NLP tasks.

Looking forward to future work, it is attractive to (1) verify the feasibility of Phrase2Vec models in other languages, e.g., Chinese; (2) find phrase mining methods for some specific fields, e.g., the phrase mining method for question semantic; (3) apply Phrase2Vec to more composite NLP tasks, such as social media analysis, dialog system.

Declaration of Competing Interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled, "Phrase2Vec: Phrase Embedding based on Parsing".

CRediT authorship contribution statement

Yongliang Wu: Conceptualization, Methodology, Software, Validation, Investigation, Writing - original draft, Visualization. **Shuliang Zhao:** Conceptualization, Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition. **Wenbin Li:** Resources, Writing - review & editing.

Acknowledgments

This research is partially supported by The National Social Science Fund of China No. 18ZDA200, No. 13&ZD091.

References

[1] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[2] A.J. Brockmeier, T. Mu, S. Ananiadou, J.Y. Goulermas, Self-tuned descriptive document clustering using a predictive network, IEEE Trans. Knowl. Data Eng. 30 (2018) 1929–1942.
[3] S. Burkhardt, S. Kramer, Online multi-label dependency topic models for text classification, Mach. Learn. 107 (2018) 859–886.
[4] J. Camacho-Collados, M.T. Pilehvar, From Word To Sense Embeddings, A survey on vector representations of meaning, J. Artif. Intell. Res. 63 (2018) 743–788.
[5] S.D. Canuto, D.X. de Sousa, M.A. Gonçalves, T.C. Rosa, A thorough evaluation of distance-based meta-features for automated text classification, IEEE Trans. Knowl. Data Eng. 30 (2018) 2242–2256.
[6] N. Durrani, H. Schmid, A.M. Fraser, P. Koehn, H. Schütze, The operation sequence model - combining n-gram-based and phrase-based statistical machine translation, Comput. Linguist. 41 (2015) 185–214.
[7] A. Eriguchi, K. Hashimoto, Y. Tsuruoka, Incorporating source-side phrase structures into neural machine translation, Comput. Linguist. 45 (2019) 267–292.
[8] K. Gebhardt, M.-J. Nederhof, H. Vogler, Hybrid grammars for parsing of discontinuous phrase structures and non-projective dependency structures, Comput. Linguist. 43 (2017) 465–520.
[9] D. Greene, P. Cunningham, Practical solutions to the problem of diagonal dominance in kernel document clustering, in: Proceedings of the Twenty-Third International Conference, ACM, 2006, pp. 377–384.
[10] K. Hashimoto, Y. Tsuruoka, Adaptive joint learning of compositional and non-compositional phrase embeddings, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL, 2016, pp. 205–215.
[11] Z. Jie, A.O. Muis, W. Lu, Efficient dependency-guided named entity recognition, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI, 2017, pp. 3457–3465.
[12] D. Kim, D. Seo, S. Cho, P. Kang, Multi-co-training for document classification using various document representations: TF-IDF, LDA, and Doc2Vec, Inf. Sci. 477 (2019) 15–29.
[13] H. Kim, M. Bansal, Improving visual question answering by referring to generated paragraph captions, in: Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL, 2019, pp. 3606–3612.

[14] Q.V. Le, T. Mikolov, Distributed representations of sentences and documents, in: Proceedings of the 31th International Conference on Machine Learning, ACM, 2014, pp. 1188–1196.
[15] B. Li, T. Liu, Z. Zhao, P. Wang, X. Du, Neural Bag-of-Ngrams, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI, 2017, pp. 3067–3074.
[16] B. Li, X. Yang, B. Wang, W. Wang, W. Cui, X. Zhang, An adaptive hierarchical compositional model for phrase embedding, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 2018, pp. 4144–4151.
[17] B. Li, X. Yang, R. Zhou, B. Wang, C. Liu, Y. Zhang, An efficient method for high quality and cohesive topical phrase mining, IEEE Trans. Knowl. Data Eng. 31 (2019) 120–137.
[18] P.-.P. Li, H. Wang, K.Q. Zhu, Z. Wang, X. Wu, Computing term similarity by large probabilistic isA knowledge, in: 22nd ACM International Conference on Information and Knowledge Management, ACM, 2013, pp. 1401–1410.
[19] S. Liang, Z. Ren, Y. Zhao, J. Ma, E. Yilmaz, M.d. Rijke, Inferring dynamic user interests in streams of short texts for user clustering, ACM Trans. Inf. Syst. 36 (2017) 10.
[20] J. Liu, J. Shang, C. Wang, X. Ren, J. Han, Mining quality phrases from massive text corpora, in: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, ACM, 2015, pp. 1729–1744.
[21] Y. Liu, L. Zhang, L. Nie, Y. Yan, D.S. Rosenblum, Fortune Teller, Predicting your career path, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI, 2016, pp. 201–207.
[22] J.-P. Mei, Y. Wang, L. Chen, C. Miao, Large scale document categorization with fuzzy clustering, IEEE Trans. Fuzzy Syst. 25 (2017) 1239–1251.
[23] Y. Meng, J. Shen, C. Zhang, J. Han, Weakly-Supervised neural text classification, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, ACM, 2018, pp. 983–992.
[24] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, CoRR (2013) abs/1301.3781.
[25] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: 27th Annual Conference on Neural Information Processing Systems, MIT Press, 2013, pp. 3111–3119.
[26] P. Passban, Q. Liu, A. Way, Enriching phrase tables for statistical machine translation using mixed embeddings, in: 26th International Conference on Computational Linguistics, ACM, 2016, pp. 2582–2591.
[27] A. Passos, V. Kumar, A. McCallum, Lexicon infused phrase embeddings for named entity resolution, in: Proceedings of the Eighteenth Conference on Computational Natural Language Learning, ACL, 2014, pp. 78–86.
[28] X. Pei, C. Chen, W. Gong, Concept factorization with adaptive neighbors for document clustering, IEEE Trans. Neural Netw. Learn. Syst. 29 (2018) 343–352.
[29] J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, ACL, 2014, pp. 1532–1543.
[30] D. Preotiuc-Pietro, Y. Liu, D. Hopkins, L.H. Ungar, Beyond Binary Labels, Political ideology prediction of twitter users, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL, 2017, pp. 729–740.
[31] T. Salles, M.A. Gonçalves, V. Rodrigues, L.C. da Rocha, Improving random forests by neighborhood projection for effective text classification, Inf. Syst. 77 (2018) 1–21.
[32] V.c.M. Sánchez-Cartagena, J.A. Pérez-Ortiz, F. Sánchez-Martıńez, Integrating rules and dictionaries from shallow-transfer machine translation into phrase-based statistical machine translation, J. Artif. Intell. Res. 55 (2016) 17–61.
[33] J. Shang, J. Liu, M. Jiang, X. Ren, C.R. Voss, J. Han, Automated phrase mining from massive text corpora, IEEE Trans. Knowl. Data Eng. 30 (2018) 1825–1837.
[34] R. Socher, C.D. Manning, A.Y. Ng, Learning continuous phrase representations and syntactic parsing with recursive neural networks, in: Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop, MIT Press, 2010, pp. 1–9.
[35] R.A. Stein, P.A. Jaques, J.F. Valiati, An analysis of hierarchical text classification using word embeddings, Inf. Sci. 471 (2019) 216–232.
[36] F. Sun, J. Guo, Y. Lan, J. Xu, X. Cheng, Inside Out, Two jointly predictive models for word representations and phrase representations, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI, 2016, pp. 2821–2827.
[37] R. Wang, H. Zhao, B.-.L. Lu, M. Utiyama, E. Sumita, Connecting phrase based statistical machine translation adaptation, in: 26th International Conference on Computational Linguistics, ACM, 2016, pp. 3135–3145.
[38] X. Wang, Z. Tu, D. Xiong, M. Zhang, Translating phrases in neural machine translation, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, ACL, 2017, pp. 1421–1431.
[39] J. Wieting, M. Bansal, K. Gimpel, K. Livescu, From paraphrase database to compositional paraphrase model and back, TACL 3 (2015) 345–358.
[40] J. Wieting, M. Bansal, K. Gimpel, K. Livescu, Towards universal paraphrastic sentence embeddings, in: 4th International Conference on Learning Representations, 2016, pp. 1–19.
[41] J. Xu, B. Xu, P. Wang, S. Zheng, G. Tian, J. Zhao, Self-Taught convolutional neural networks for short text clustering, Neural Netw. 88 (2017) 22–31.
[42] W. Yin, H. Schütze, Discriminative phrase embedding for paraphrase identification, CoRR (2016) abs/1604.00503.
[43] J. Zhang, S. Liu, M. Li, M. Zhou, C. Zong, Bilingually-constrained phrase embeddings for machine translation, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL, 2014, pp. 111–121.
[44] Z. Zhang, H. Zhao, L. Qin, Probabilistic graph-based dependency parsing with convolutional neural network, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL, 2016, pp. 1382–1392.
[45] R. Zhao, K. Mao, Fuzzy bag-of-words model for document representation, IEEE Trans. Fuzzy Syst. 26 (2018) 794–804.
[46] Y. Zhao, Y. Wang, J. Zhang, C. Zong, Phrase table as recommendation memory for neural machine translation, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Morgan Kaufmann, 2018, pp. 4609–4615.
[47] Z. Zhao, T. Liu, S. Li, B. Li, X. Du, Ngram2vec: learning improved word representations from ngram co-occurrence statistics, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, ACL, 2017, pp. 244–253.
[48] Y. Zhuang, H. Wang, J. Xiao, F. Wu, Y. Yang, W. Lu, Z. Zhang, Bag-of-Discriminative-Words (BoDW) representation via topic modeling, IEEE Trans. Knowl. Data Eng. 29 (2017) 977–990.