Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Principal Component Analysis Algorithm and Spatial Mapping Construction for Feature Extraction of University Language Text Data

Guoxi Lv[1, ✉]

[1] *Neijiang Normal University, Neijiang, Sichuan, 641100, China*

ABSTRACT

With the arrival of the big data era, a huge amount of text data of college language is generated, and how to manage these text data efficiently and mine useful information has become the focus of many scholars. The study first preprocesses and represents the university language text data, proposes a feature screening method based on Shannon entropy and JS-scatter, and then combines the principal component analysis algorithm with the dimensionality reduction of the extracted features on this basis. Subsequently, a pre-trained high-dimensional word vector spatial mapping model is introduced to generate richer semantic representations, and a pre-trained high-dimensional word vector spatial mapping model based on the pre-trained high-dimensional word vector spatial mapping model is designed. Finally, the method proposed in this paper is tested experimentally. Under different feature dimensions, the macro-averages of this paper's method are 72%, 44.2%, 67.1%, and 3.3% higher than those of IG, PMI, ANOVA, and JS methods. At the feature dimension k=350, the macro-mean of this paper's method is 0.853, when the classification effect reaches the optimization. In the spatial mapping relationship of word vectors, the accuracy of the mapping of this paper's method also reaches 11.2% for the words with word frequency sorted from the first 5000 to the first 6000. This proves the effectiveness and feasibility of this paper's method.

Keywords: principal component analysis, spatial mapping model, university language text data, JS-scatter, feature extraction

1. Introduction

As a public compulsory basic course for college students, college language can not only enable college students to master the basic knowledge and basic language skills of Chinese language and literature, improve their literary cultivation, and enhance their literary reading and appreciation ability, but also

✉ Corresponding author.
E-mail address: lvguoxi2008@126.com (G. Lv).
Received 01 March 2024; Revised 25 May 2024; Accepted 10 November 2024; Published Online 15 April 2025.
DOI: 10.61091/jcmcc127a-307

cultivate college students' moral sentiment, aesthetic interest, and cultivate their humanistic spirit [1-4]. Text reading is an indispensable part of the university language learning process, which is an important means to realize the purpose of language teaching and improve the knowledge reserve and comprehensive quality of college students [5-7]. With the acceleration of social informatization process, college students' access to language reading materials has become more diversified, and online learning has become the most popular way of knowledge acquisition [8-9]. Online learning websites cover a wealth of university language reading materials, genres and types, and it is quite difficult for college students to retrieve the university language reading materials they need in the vast language reading resources [10]. The genre of university language reading materials is the embodiment of the specification and form of reading content, plays a certain normative role in the structure of reading content, and has a direct relationship with the author's writing style, social environment, etc. [11-12]. The genre of reading materials reflects the formal information of the article, and reading materials of the same genre may be the same in text format and language style, but their themes may be very different [13-14]. University language reading material genre classification is an effective means to realize the classification of reading resources retrieval. Genre classification of university language reading materials involves the classification of article genres and text genres, the latter belongs to text classification, and text structure is closely related to genre classification [15]. The use of punctuation, lexis, and vocabulary in different genres of university language reading materials has its own characteristics, which is important for realizing the genre classification of university language reading materials by taking them as genre features and exploring the correlation between feature items and genres [16-19].

The article first introduces the overall processing flow of text feature extraction, focusing on data preprocessing and text representation methods. It then designs an algorithm for data anonymization under a given JS-scatter threshold and proposes a method for applying principal component analysis to process vectors. The knowledge graph embedding representation is then pre-trained using the BERT model and the embedding vectors of head and tail entities are extracted by a variant of KGBERT. Then, fine-tuning is performed by pooling technique and cross-entropy loss function to learn the head entity and relation embedding representation vectors and the embedding representation vectors of tail entities. Finally, the university language text dataset is used to test that the model in this paper has some effectiveness in text feature extraction. The invariance between word vector differences and the invariance between clip angles are discussed separately using the high-dimensional word vector space mapping model proposed in this paper. The connection between these two types of invariants and semantics is judged by labeling the set, and explanations are given for these invariants.

2. Principal component analysis algorithm based on data feature extraction

2.1 Data pre-processing

2.1.1. Text cleaning. Text cleaning mainly refers to the removal of noise and redundant characters in the text. Through these text cleaning operations, the text data can be effectively purified to remove the noise and unnecessary information in it, making the text cleaner and more standardized, and providing a good foundation for subsequent text processing and model training.

2.1.2. Segmentation. Segmentation in text preprocessing is the process of segmenting text data into words or subwords, the purpose of which is to convert text data into basic units that can be processed by the model. Currently commonly used segmentation methods for text data are roughly divided into three categories, the first category is rule-based segmentation. The second category is based on statistical segmentation. The third category is deep learning-based segmentation.

2.1.3. De-duplication of words. Deactivation in text preprocessing refers to the removal of deactivated words from text data. The methods of de-duplication usually include the use of a list of deactivated words, word-frequency based de-duplication methods, corpus based de-duplication methods, machine learning based de-duplication methods and so on. Deactivation is one of the important steps in text preprocessing, which can improve the quality of text features and the performance of the model, and make the model more focused on the extraction and analysis of key information.

2.2 Text representation stage

2.2.1. Bag-of-words model. Bag-of-words model is a commonly used text representation method in NLP, which represents the text in the form of word frequency vectors, without considering the order of words in the text, but only the frequency of occurrence of words in the text. The principle is to construct a vocabulary list in the bag-of-words model, which contains all the words that appear in the text data. Then each text sample is represented as a fixed-length vector, each dimension of the vector corresponds to a word in the glossary, and the value of the vector indicates the number of occurrences of the word in the text or the word frequency.

2.2.2. TF-IDF. TF-IDF is a common technique in the field of information retrieval and text mining, often used in keyword extraction and text categorization tasks, which serves to assess the importance of a keyword in a text or corpus. When a keyword occurs more frequently in a text, its importance increases, and conversely, the less frequently a keyword occurs in a text, its importance decreases [20]. The TF-IDF model is divided into two parts, i.e., the frequency of each keyword in the text (TF) and the inverse document frequency (IDF) of that keyword in the whole collection of texts, and the following introduces each of these two concepts.

1) TF. TF represents the relative frequency of occurrence of a keyword in the text, i.e., the number of times a keyword occurs in the text compared to the total number of keywords in the text. The mathematical expression of TF is shown in the following equation (1).

$$TF_{i,j} = \frac{n_{i,j}}{\sum_{k} n_{k,j}} \tag{1}$$

Where, $TF_{i,j}$ is the frequency of the keyword in the text and $n_{i,j}$ is the number of times the keyword appears in the text.

2) IDF. IDF measures the general importance of a word, i.e. a word is more important if it occurs in a document less often in a collection of documents. When calculating the IDF of a keyword, the number of documents in which the keyword is located is first calculated, then the total number of documents is used to divide by the number of documents in which the keyword is located, and finally the result of the calculation is taken as a logarithm. This is shown in equation (2) below.

$$IDF_i = log \frac{|D|}{1+|d|} \tag{2}$$

Where $|D|$ is the total number of documents in the document collection, $|d|$ is the number of documents containing the keyword, and 1 in the denominator is to prevent the denominator from being 0.

Finally, TF is multiplied with IDF to obtain the TF-IDF weight of the word. This weight indicates the importance of the word in the current document, taking into account its general importance in the whole collection of texts, as shown in equation (3) below.

$$TF - IDF = TF * IDF \tag{3}$$

2.2.3. BERT. BERT is a pre-trained model based on the Transformer architecture, and each module of BERT is described in detail below [21].

1) Input representation. The input of BERT is a text sequence consisting of multiple word fragments. Each word fragment is first converted into a word embedding vector, which represents the position of the word fragment in a high-dimensional space. The input to BERT also includes some special tokens, such as CLS (used to denote the beginning of the text), SEP (used to denote the separation of different sentences), etc. The input to BERT is a sequence of text consisting of multiple word fragments, each of which is represented as a word embedding vector.

2) Encoder. The core architecture of the BERT model is multiple bi-directional Transformer encoders, and each Transformer encoder layer incorporates a multi-head self-attention mechanism and a feed-forward neural network. The multi-head self-attention mechanism is one of the core components in the Transformer model, which enables the model to establish global dependencies at different locations and generate corresponding contextual representations. Multi-head self-attention mechanism means applying the self-attention mechanism to several different representation spaces and splicing or weighted summing the outputs of each head to get the final representation. In Transformer, it is common to use multiple independent self-attention mechanisms where each head learns a different feature representation, thus improving the expressive and generalization capabilities of the model.

Self-attention mechanism:

The self-attention mechanism is used to compute the correlation between each position in the sequence and all other positions. For each position in the input sequence, the self-attention mechanism computes its correlations with other positions in the sequence and uses these correlations as weights, weighted and summed to obtain a contextual representation of that position.

The multi-head self-attention mechanism, i.e., the simultaneous use of multiple attention heads and the independent computation of the attention weights of each head, thereby improving the expressive and generalization capabilities of the model, is computed as follows:

Step 1: Input Sequence Representation: Assume the input sequence is $x = \{x_1, x_2, \cdots\cdots, x_n\}$, where each x_i is a word vector representing a position in the sequence.

Step 2: Linear transformation: the input sequence is linearly transformed to obtain a query vector (Query, Q), a key vector (Key, K) and a value vector (Value, V). Use these three sets of vectors to calculate the attention weights of the input data. The specific formulas are shown in Eqs. (4) to (6) below.

$$Q = x_i W^Q \tag{4}$$

$$K = x_i W^K \tag{5}$$

$$V = x_i W^V \tag{6}$$

where W^Q, W^K, W^V is the linear transformation matrix of the query vector, key vector, and value vector, respectively.

Step 3: Calculate Attention Score: the attention score between each location and all other locations is calculated, which can be obtained by the dot product of the query vector Q and the key vector K with scaling. As shown in equation (7), the attention score A_{ij} indicates the degree of association between location i and location j.

$$A_{ij} = \frac{q_i \cdot k_j^T}{\sqrt{d_k}} \tag{7}$$

where d_k is the key vector dimension, usually the number of columns of W^Q or W^K.

Step 4: Calculate the attention weights, the calculated attention scores are normalized using softmax to obtain the attention weights of each position relative to all other positions, which will be used to weight and sum the value vectors v_i to obtain the contextual representation of each position. As shown in equation (8).

$$A_{ij}^{'} = softmax(A_{ij}) \tag{8}$$

Step 5: Weighted summation, as shown in Equation (9), weights and sums the value vectors v_j using the attention weights $A_{ij}^{'}$ to obtain the contextual representation z_i for each position:

$$z_i = \sum_{j=1}^{n} A_{ij}^{'} v_j \tag{9}$$

3) Pre-training tasks. BERT has two main tasks in the pre-training phase, which are MLM pre-training task and NSP pre-training task.

The MLM task is for the BERT model to randomly replace some word fragments in the input text with special masked tokens, and then try to predict the masked word fragments, so as to learn the contextual information of the words.

The task of NSP is to make the model learn the continuity and contextual relationship of the text sequence so that the BERT model can determine whether the two input sentences are consecutive sentences in the original text sequence.

4) Output layer. The output layer of a BERT model is usually a fully-connected layer that is used to convert the output of the multilayer Transformer encoder into final textual representations that can be used for various natural language processing tasks.

2.3 Shannon entropy (information entropy) and JS-dispersion

Information theory recognizes that the degree of uncertainty of information determines the amount of information it can contain. As an intuitive example, when we are learning a new theory, at first, because we know very little about the theory, we must learn a great deal of information about it in order to figure it out. Conversely, if we know more about the theory, we need much less information to figure it out. Uncertainty can indeed account for the amount of information.

For any random variable X, the magnitude of entropy is positively correlated with its uncertainty, and Shannon gives specific formulas for quantifying the entropy of a random variable X:

$$H(X) = -\sum_{x} P(x) \log_2 [P(x)] \tag{10}$$

In probability theory and statistics, the Jensen-Shannon scatter is a measure of the distance (degree of similarity) of probability distributions. For distributions of more than two, the JS-scatter has the following formula:

$$JS(P_1, P_2, ..., P_n) = H\left(\sum_{i=1}^{n} \pi_i P_i\right) - \sum_{i=1}^{n} \pi_i H(P_i) \tag{11}$$

where π_i is the weight of distribution P_i and has $\sum \pi_i = 1$.

In this paper, we use JS-scatter to measure the distance of distributions mainly because JS-scatter has three better properties compared to some other distance measures:

1) JS-scatter can be applied to the set of more than two probability distributions [22].

2) The JS-scatter value is always a definite number.

3) The JS-scatter is symmetric with respect to the order of the parameters, i.e., the result is independent of the order in which the parameters are arranged.

2.4 Principal component analysis

Principal component analysis is a widely used method for pattern recognition and signal processing, and also appears to be particularly effective in data compression and feature extraction. In this paper, we apply the principal component analysis method to text categorization in the expectation of obtaining text vectors with low-dimensional representations. Considering the particularly high complexity of employing principal component analysis in the high-dimensional space, we apply principal component analysis in the conceptual indexing subspace, i.e., we first apply conceptual indexing in the pure document vectors, and then apply principal component analysis in the conceptual indexing in the low-dimensional subspace. We define the subspace after the processing of concept indexing and principal component analysis as the CP subspace. We want to obtain text vectors with low-dimensional representations at a relatively low cost. The specific steps of principal component analysis are as follows:

Step1: Let there be M classification in the training set sample, calculate the covariance matrix of each internal document classification separately as follows:

$$X_i = E[(x-\mu)(x-\mu)^T] \tag{12}$$

Where X_i is the covariance matrix of internal document classification for the ind classification, x is the text vector in the ith classification, and μ_i represents the prototype vector in the ith classification.

Step2: Calculate the covariance matrix for weighted average internal text classification S_w, which is calculated as follows:

$$S_w = \sum_{i=1}^{M} P_i X_i \tag{13}$$

where P_i is the a priori probability of the ind classification, which can be inferred from the training set samples to take its value.

Step3: Compute eigenvalue d_j and eigenvector V_j for eigenvalue S_w:

$$S_w V_j = d_j V_j \tag{14}$$

Since S_w is a real symmetric matrix, then there exist independent linear eigenvectors and make their dimension M.

Step4: Change the set of vectors by the eigenvectors of S_w. Sort the eigenvalues of S_w into $d_1 \geq d_2 \geq \ldots \geq d_M$ in descending order and include the corresponding eigenvectors in the transformation matrix, i.e:

$$W = \{V_1, V_2, \ldots V_M\} \tag{15}$$

Then for each vector x, it can be transformed into a new subspace:

$$y = W^T x \tag{16}$$

where W is the matrix formed by the M independent linear eigenvectors in S_w.

Step5: Compress the vectors into the transformation space, i.e., the principal component analysis subspace. According to the dimension D defined earlier, we only need to save the vectors in the first D dimensions and ignore the vectors in the other dimensions.

Through Step1 to Step5, we can get the low-dimensional representation vectors in the principal component analysis subspace [23]. The key of Principal Component Analysis is that it always obtains the best linear mapping from the M-dimensional space to the D-dimensional space, due to the fact that the algorithm obtains the data in such a way that the sum of squares of the errors is always minimized, and at the same time, the interaction information between the original vectors x and their mapping vectors y is maximized:

$$I(x,y)=\frac{1}{2}\ln((2\pi e)^{D}\lambda_1\lambda_2\cdots\lambda_D \tag{17}$$

where $\lambda_I,\lambda_2,...\lambda_D$ is the first D eigenvalues of the covariance matrix.

3. Construction of high-dimensional word vector space mapping based on pre-trained models

3.1 Embedding extraction based on pre-trained models

3.1.1. Marker embedding. The task of knowledge mapping is introduced in the pre-training phase. Unsupervised learning approach is first used in KG-BERT to pre-train BERT using a large-scale text corpus. It uses the prediction task of language modeling, and also introduces the task of knowledge mapping based on knowledge mapping to enhance the learning of the representation of Tongue information. For the input triples, the model treats them as a sequence, distinguishing head entities, relations and tail entities by adding special tags [CLS] and [SEP]. It is worth noting that a [SEP] tag is also added between the head entity and the relation to allow the model to distinguish between them.

Different blocks are separated into different parts by "SEP", and the token embedding vectors of tokens for each part are obtained from the token, the head entity $\vec{h}$ bronze dalit statue is mapped as $E_{h1}\cdots E_{hL}$, the relation $\vec{r}$ excavated in is mapped as $E_{r1}\cdots E_{ro}$, and the tail entity $\vec{t}$ Guanghan is mapped as $E_{t1}\cdots E_{tp}$. The token embedding vectors are passed through multiple encoders of BERT to form the feature vectors R, and the pre-trained BERT model of the knowledge graph is shown in Fig. 1, and note that what is labeled in the figure is the tokens and embedding vectors, not the id in the next sentence prediction.

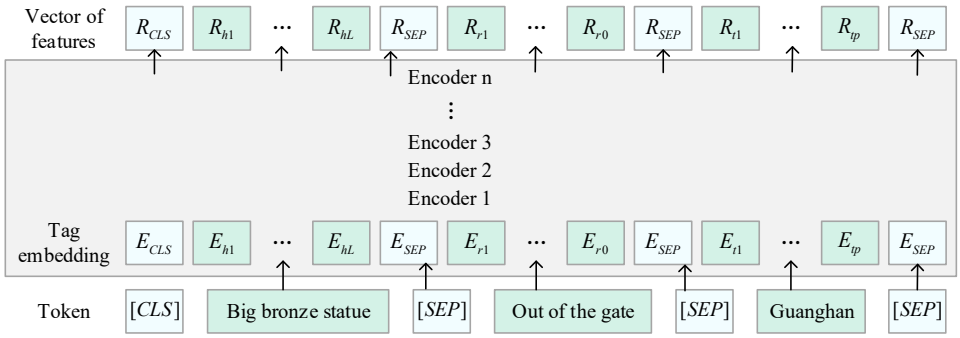


Fig 1. Pre-trained knowledge graph BERT model

The model is different from the knowledge graph representation learning model of KG-BERT, although the inputs are all ternaries in the knowledge graph, and the ternaries are used as sequences, both of which utilize the pre-trained language model and the graph information to learn the entities and relations.

According to such design idea, I divide the embedded token into two $tokens_m$ and $tokens_n$, the first one is head entity and relation, and the second one is tail entity, the design idea merges the head entity and relation into one tokens encoding them using the pre-trained BERT model to obtain the representation of the head entity and relation in the same dimension. Meanwhile, the tail entity constitutes a separate token and is also fed into the BERT encoder for encoding. This better captures the semantic links between entities and relations. We can represent this using the following equation:

$$tokens_m=\left[\left[CLS\right],\text{Jobs},\left[SEP\right],\text{Create},\left[SEP\right]\right] \tag{19}$$

Similarly, for $tokens_n$ of the tail entity, we can use the following expression (20):

$$tokens_n = [[CLS], \text{Apple}, [SEP]] \tag{20}$$

The pre-training model for the head entity and relationships is shown in Figure 2. The pre-training model for the tail entity is shown in Figure 3.

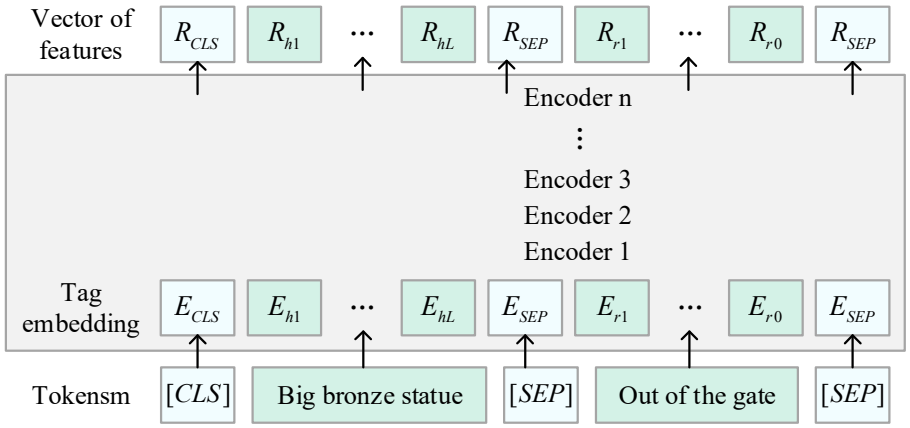


Fig 2. Pre-trained model of header entities and relationships

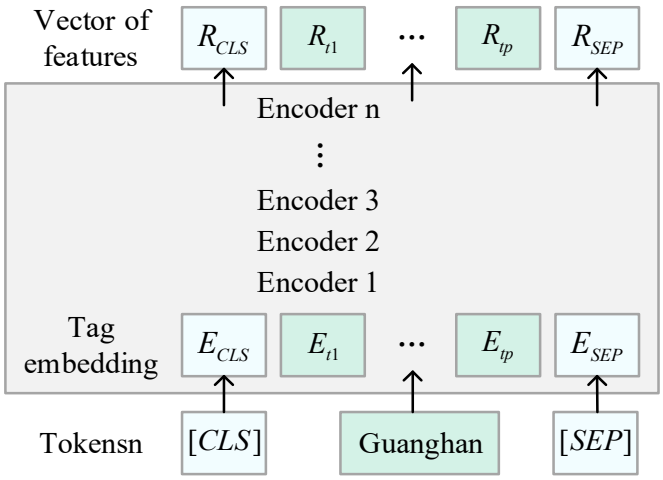


Fig. 3. Pre-trained model of the tail entity

3.1.2. Mask Language Model. Masked Language Modeling (MLM) is a model pre-trained on a large-scale raw corpus by self-supervised learning to obtain a powerful text encoder, which is widely used in natural language processing tasks. The main purpose of the masking model is to avoid the model from accessing any information after any given position when generating vectors for that position. This is achieved by masking out all vectors after that position when generating vectors at each position. The mask is represented by a binary matrix, where a mask value of 1 means that the position needs to be masked, and a mask value of 0 means that the position does not need to be masked.

According to the above principle, I will feed the sequence of header entities and relationships into the Transformer encoder, and continue to use "Bronze Dali figure excavated in Guanghan" as an example, which is expressed by the following formula:

$$Transformer - Enc\left([E_{CLS}, E_{\text{Big bronze statue}}, E_{SEP}, E_{\text{Out of the gate}}, E_{SEP}]\right) \tag{21}$$

Similarly the sequence of tail entities is fed into the Transformer encoder, represented by the following equation:

$$Transformer - Enc\left([E_{CLS}, E_{\text{Guanghan}}, E_{SEP}]\right) \tag{22}$$

3.2 Fine-tuning of pre-trained models

On the basis of the pre-trained model, the pooling layer in the fine-tuning process is used to construct a bi-directional classifier for determining whether the head entity and relation sequences and the tail entity sequences are reasonable. Specifically, we will extract the embedding vectors of head entities and relations and the embedding vectors of tail entities from the embedding extraction output of the pre-trained model and splice them separately. I will then pass the merged vectors to the classifier for classification to determine if they are factually correct. The pre-trained fine-tuned model for the head entity and the relation is shown in Figure 4. The pre-trained fine-tuned model for the tail entity is shown in Figure 5.

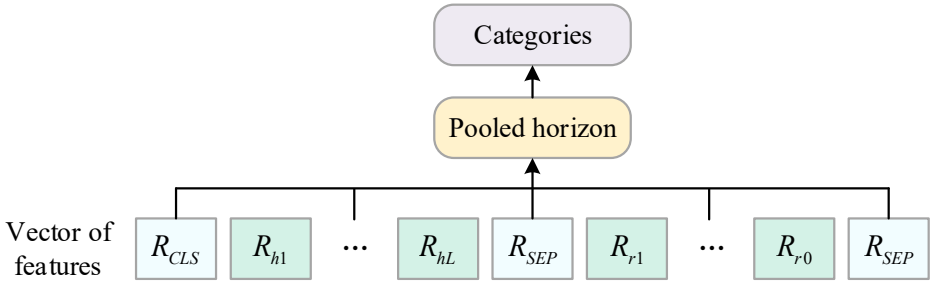


Fig 4. Pre-trained fine-tuning model of head entities and relationships

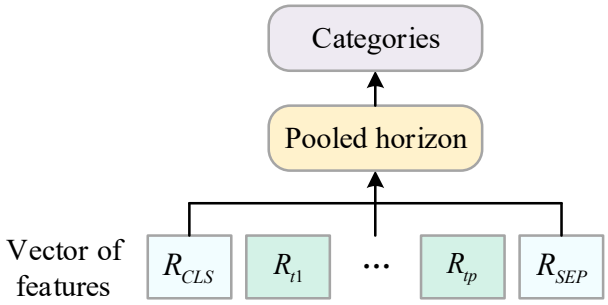


Fig. 5. Pre-trained fine-tuning model of the tail entity

In BERT, MeanPooling is a Pooling method for averaging all token vectors in a sequence of embedding vectors to produce a single vector representing the entire sentence. This vector can be used for downstream tasks. In a concrete implementation, we can represent the output of the embedding extraction of a pre-trained model as a sequence, see Eq:

$$R = [R_1, R_2, ..., R_L] \tag{23}$$

where R_i denotes the feature vector representation of the i nd token. We use MeanPooling to aggregate R and this vector can be used to train downstream tasks.

$$pool = \frac{1}{L}\sum_{i=1}^{L} R_i \tag{24}$$

For my model, I need to compute to get the vector representation of the head entity with the relation and the tail entity, please note that the feature vectors of the head entity with the relation and the tail entity are computed separately in the model. The output of the embedding extraction of the head entity and relation is equation (25).

$$Transformer - Enc\left([R_{CLS}, R_1, R_2, ..., R_L, R_{SEP}, R_{L+1}..., R_{L+o}, R_{SEP}]\right) \tag{25}$$

Where the head entity has L feature vector and the relationship has o. Transformer-Enc is the Transformer encoder described in the previous subsection.

The embedding of the tail entity extracts the output as equation (26).

$$Transformer-Enc\left([R_{CLS},R_1,R_2,...,R_p,R_{SEP}]\right) \tag{26}$$

The embedding of header entities and relations undergoes pooling into Eq. (27).

$$m=pool\left(Transformer-Enc\left([R_{CLS},R_1,R_2,...,R_L,R_{SEP},R_{L+1}...,R_{L+o},R_{SEP}]\right)\right) \tag{27}$$

The tail entity embedding is pooled into equation (28).

$$n=pool\left(Transformer-Enc\left([R_{CLS},R_1,R_2,...,R_p,R_{SEP}]\right)\right) \tag{28}$$

4. Experimentation and analysis

4.1 Experiments on Principal Component Analysis

In this section, the traditional feature dimensionality reduction methods Information Gain (IG), Point Mutual Information (PMI), Analysis of Variance (ANOVA), and the JS method were chosen to compare with the method of this paper (JS-PCA).

The k=300 time λ with the corresponding trend of model error (JS) is shown in Fig. 6. Here in this paper, the penalty parameter that makes the model error to be the smallest is chosen with the value of 0.0021.

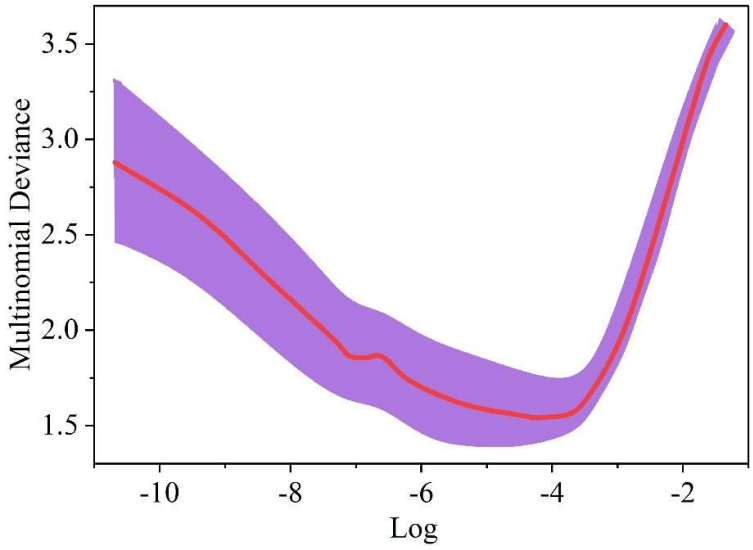


Fig. 6. Model error corresponding trend

The comparison of the macro-averages of the five feature dimensionality reduction methods is shown in Fig. 7. From the figure, it can be found that the macro-averages of this paper's method are higher than those of IG, PMI, ANOVA, and JS methods in different feature dimensions, which are 72%, 44.2%, 67.1%, and 3.3% higher on average, respectively. In the feature dimension k=350, the macro-mean value of this paper's method is the highest, 0.853, at this time the classification effect reaches the optimum, when the feature dimension k=420, the macro-mean value of the JS and this paper's method decreases, which indicates that at this time, unimportant variables may have been selected in the features, which makes the classification effect decrease. At this time, although the macro-mean of other methods increases, the increase is not large.

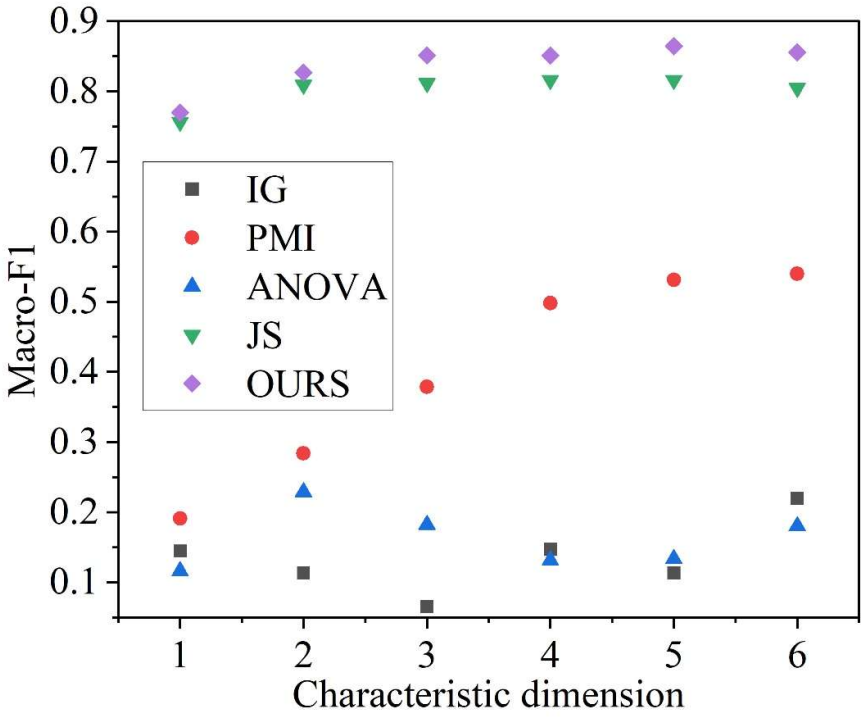


Fig. 7. The average comparison of the five characteristics

In order to further compare the classification effect of the methods in this paper on each category, the following chooses to show the F1 value, precision and recall of the five methods on each category at the optimal feature dimension k=350, respectively. The comparison of F1 values of the five feature dimensionality reduction methods is shown in Fig. 8. It can be found that both this paper's method and the JS method reach the highest F1 value on the Health category, due to the fact that the Health category of the data contains more distinguishable feature words.

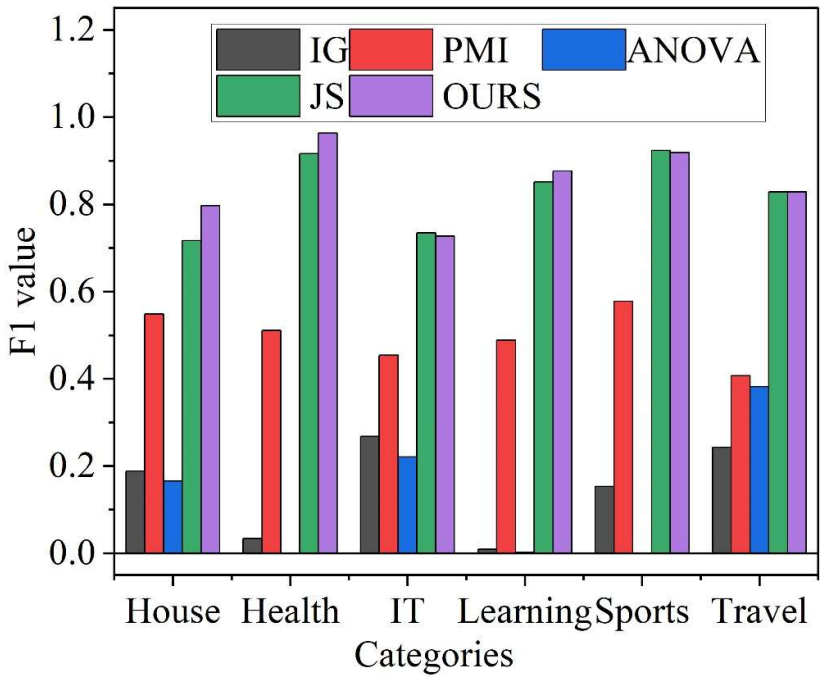


Fig. 8. Comparison of F1 values of five characteristics of descending dimension method

The accuracy rate (k=300) of the five feature dimensionality reduction methods on each category is shown in Table 1. As can be seen from the table, both this paper's method and the JS method have higher accuracy rates on the Health and Sports categories and lower accuracy rates on the IT category. The accuracy rate of this paper's method on each category is mostly higher than other methods, which indicates that this paper's method has better classification performance on each category.

Table 1. The accuracy of five characteristics of the descending dimension method

	IG	PMI	ANOVA	JS	Ours
House	0.205	0.724	0.166	0.91	0.868
Health	0.042	0.489	0.000	0.89	0.929
IT	0.345	0.445	0.321	0.68	0.673
Learning	0.000	0.45	0.000	0.853	0.923
Sports	0.108	0.464	0.000	0.878	0.911
Travel	0.272	0.359	0.492	0.808	0.748

The metric of most interest in the college language classification problem in this paper is the recall rate, which reflects the proportion of samples correctly judged in a given category. The recall (k=300) of the five feature dimensionality reduction methods on each category is shown in Table 2. It can be seen that the recall of JS and this paper's method is significantly higher than that of the other methods, and on the Health category, the recall of both methods reaches 100%, which indicates that the Health category has a high level of feature word category differentiation. On the other hand, on the House category, the recall rates of both methods are lower, indicating that the feature word category distinction of the House category is lower. Overall, the classification effect of this paper's method is better than other methods, and it is effective and feasible to use this paper's method for text categorization.

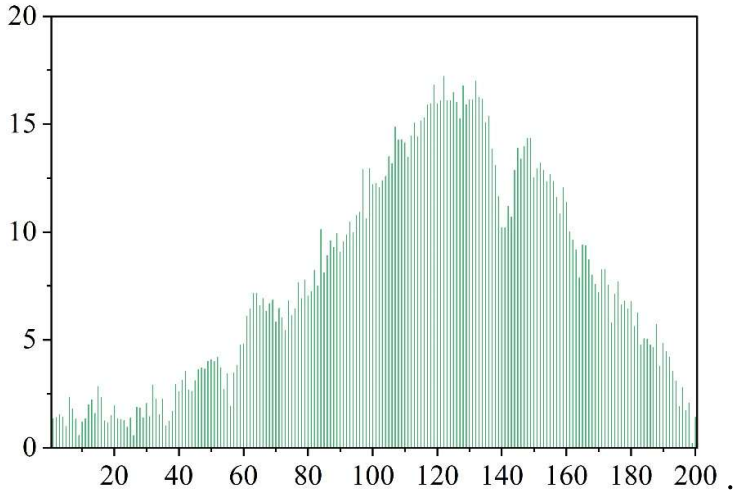
Table 2. Five characteristics degradation methods are recalled in various categories

	IG	PMI	ANOVA	JS	Ours
House	0.174	0.477	0.155	0.605	0.731
Health	0.024	0.517	0.000	1.000	1.000
IT	0.228	0.406	0.183	0.84	0.77
Learning	0.000	0.52	0.000	0.91	0.812
Sports	0.182	0.668	0.000	0.941	0.906
Travel	0.201	0.409	0.318	0.881	0.889

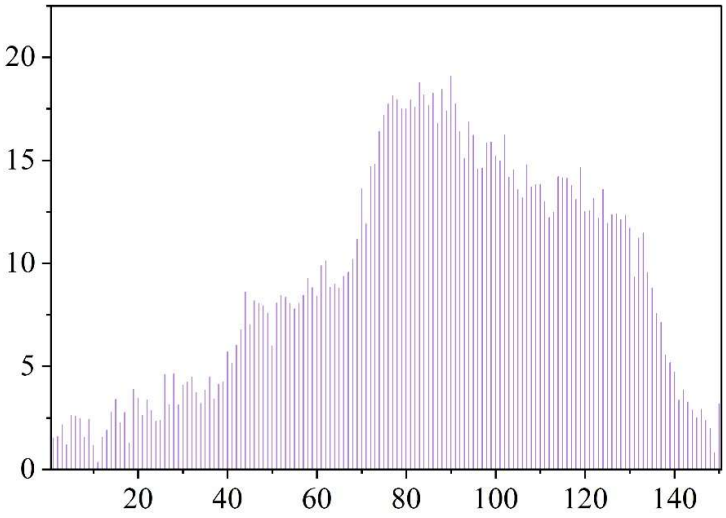
4.2 Experiments on representation invariance of word-vector space mappings

4.2.1. Basic statistics for representation invariance of word vectors. In this section, we first start from the statistical properties of word vectors, find out the basis for word vectors to have representation invariance, and start exploring the multiple representation invariance of word vectors in the following. Firstly, the corpus set for comparison needs to be determined, and in this paper, word vectors trained from the university language corpus are used for statistics. In order to make the training of word vectors adequate, the sizes of the two corpus sets are 42GB and 6GB, respectively, and the sizes of the word lists of the two are 400,000 words and 190,000 words, respectively. For the comparison of word vectors, only 190,000 words of the common word lists of the two are used for statistics. From this we obtain the word vectors trained on the two different corpus sets. In order to make a comparison of the differences of word vectors in different corpus sets, the following method is used in this paper. In each set of word vectors, the remaining string similarity is calculated for each word in the word list two by two.

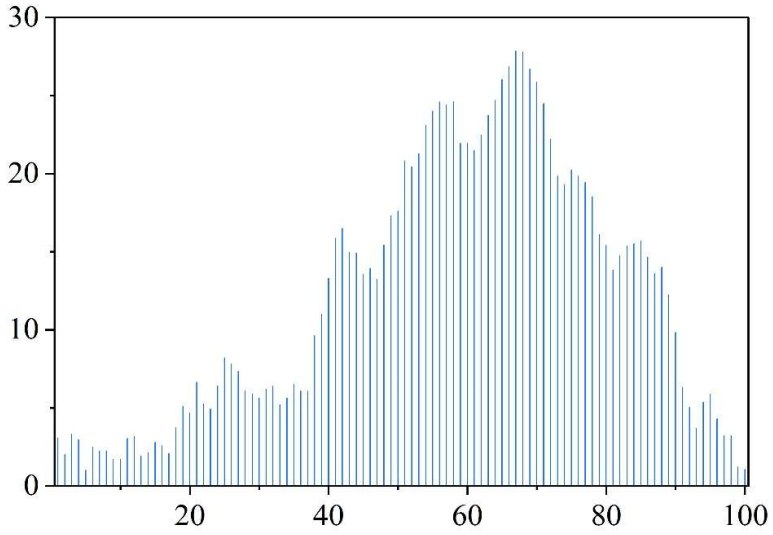
And then according to the calculated similarity, the similar words of each word are sorted in descending order of similarity, and the N words with the highest similarity are filtered out. The part of the two sets of word vectors in which the similar words of the same word w overlap is counted. In this paper, in order to reduce the amount of computation in the comparison process and make the statistical results more meaningful, the 1,000 words with the highest frequency of occurrence in the corpus are used for comparison. That is, the degree of consistency of a word in the two corpus sets is defined as: common words share similar word distributions as shown in Fig. 9 (Figs. a~d are top200 similar word distributions, top150 similar word distributions, top100 similar word distributions, top50 similar word distributions, respectively).



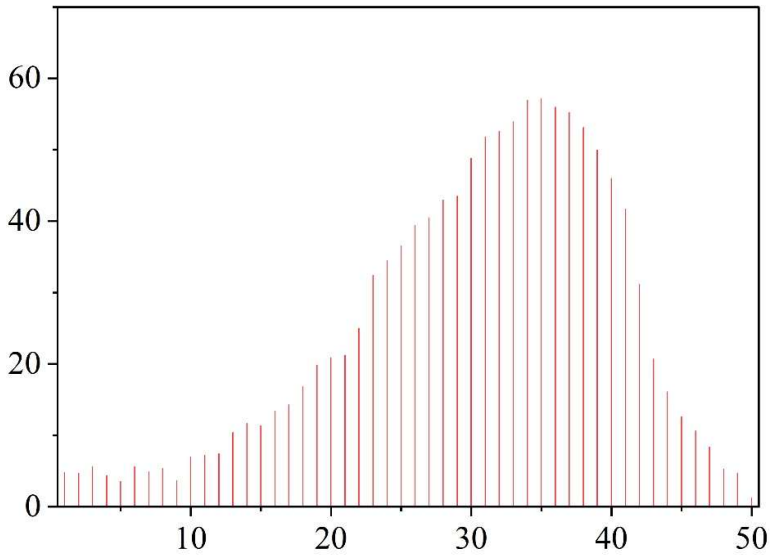
(a) Top200 similar word distribution



(b) Top150 similar word distribution



(c) Top100 similar word distribution



(d) Top50 similar word distribution

Fig. 9. Common words are common in similar words

4.2.2. Spatial mapping relations for word vectors:

1) Gradient descent to solve the mapping matrix. In this section, the mapping accuracy of this university language corpus is compared with the accuracy of Mikolov's translation between English and Spanish. The accuracy of the matrix mapping method is shown in Table 3. It can be seen that the overall accuracy rate is lower than the translation results, this phenomenon is due to the fact that Mikolov's experiment is the entire corpus set of sentence-by-sentence translation, the differences between the corpus is relatively small, and has a strong one-to-one correspondence, which is obviously very helpful for the accuracy of the mapping matrix to improve. However, we can also see that even in our corpus, the accuracy of the mapping reaches 11.2% for words with word frequencies ranked from the top 5000 to the top 6000, which shows that the transformation between the two vector spaces can be realized better to some extent in this way. As the word frequency decreases, the accuracy of the mapping also begins to decrease, this is because as the word frequency decreases, the episodicity in the training process of word vectors increases significantly, and the training results of the word vectors

are more unstable, so the difference of the word vectors themselves between the two corpus sets is widened, and the accuracy of the mapping decreases as a result.

Table 3. Matrix mapping method accuracy

Result	P@1	P@5
Translation:en->sp	35%	50%
News:sgd	11.2%	23.6%
News:sgd+momentum	12.6%	27.5%

2) Matrix decomposition to solve the mapping matrix. When applied to the current problem, the word vectors in the original vector space form matrix A, and the word vectors in the target vector space form matrix B. The mapping matrix R can be obtained according to the above method, and the mapping result of "they" in the two corpora is shown in Fig. 10. It can be seen that this matrix can well complete the transformation between semantic spaces, and when using the 5000 words with the highest word frequency before the transformation, the P@1 accuracy reaches 19.2%, and the P@5 reaches 31.5%. Gradient descent methods that use only some words as training data, the matrix decomposition approach on the one hand applies the global information of word vectors, which is less affected by the error of the training data, and therefore improves the accuracy of the mapping. On the other hand the obtained mapping matrix in the matrix decomposition method also has a higher dimension. The dimension of the matrix in the gradient descent method is nXn (n is the vector dimension), while the dimension of the matrix in the matrix decomposition method is NXN (N is the size of the common word list in the corpus). At this point, the operation accomplished by the mapping matrix is actually a linear combination of the vectors in the current space to generate word vectors in another space. Because of its higher dimension, the mapping matrix is also able to express more complex transformation relations. It is for this reason that the mapping matrix obtained through matrix factorization achieves a higher mapping accuracy.

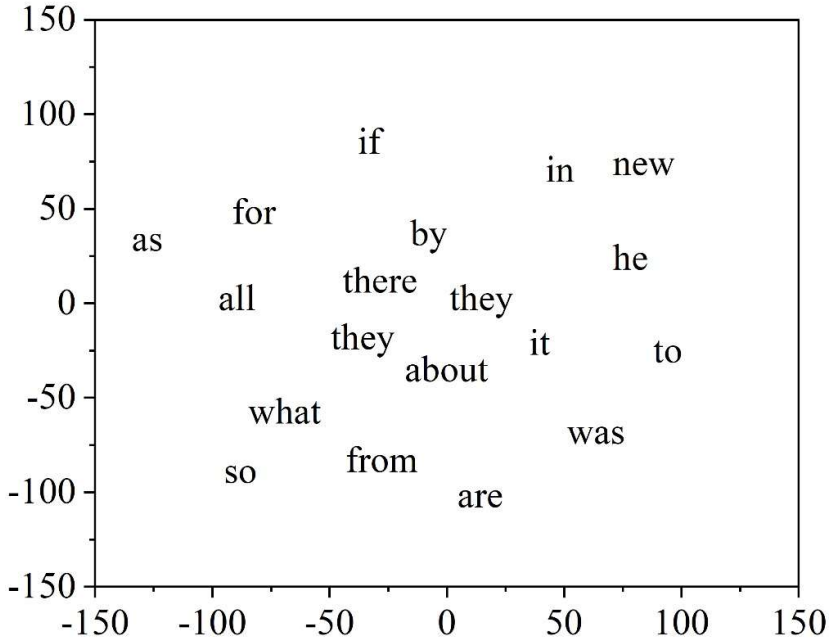


Fig. 10. The mapping results of the two materials

4.2.3. Invariance between word vector differences. The changes in the difference vectors of semantically related word pairs are shown in Table 4, and the changes in the difference vectors of semantically unrelated word pairs are shown in Table 5. We can notice that the degree of change in the

difference vectors of the semantically related words is very small whether the cosine distance is used for the metric, which can be considered as basically relatively unchanged, while the difference of the vectors of the two words that are not semantically related varies relatively more with the corpus. For example, the cosine value of Tokyo-Japan is 0.004 and the Euclidean distance is 9.309 in the change of the difference vectors of semantically related word pairs, and the two types of word pairs are very poorly differentiated when using the Euclidean distance for the metric, so it can be seen that the cosine distance is more effective in the comparison of word vectors. It can also be seen that the difference of semantically related words has invariance. This is consistent with the assumption that word vectors can be transformed by mapping matrices.

Table 4. Semantic correlation words change vector difference

Pair	Cosine	Euclid
Tokyo-Japan	0.004	9.309
Go-Going	0.004	8.579
Fly-Flying	0.009	7.343
Look-Looking	0.003	6.675
Slow-Slowing	0.012	7.85
Describe-Describing	0.005	6.969
Bern-Switzerland	0.01	10.662
Santiago-Chile	0.013	8.506
Increase-Increasing	0.011	7.539
Madrid-Spain	0.013	8.625
Cairo-Egypt	0.019	8.428
Say-Saying	0.019	8.331
Generate-Generating	0.021	8.086
Lisbon-Portugal	0.026	8.922

Table 5. The semantic independent vector is changing the difference

Pair	Cosine	Euclid
Monkey-Computer	0.062	8.3
Air-Keyboard	0.046	10.395
Tea-Calendar	0.057	11.23
Travel-Medicine	0.08	10.073

4.2.4. Invariance between word-vector pinch points. For the purpose of conducting experiments to compare the angular relationships between word vectors, a classic corpus of university language texts was used in this section. The corpus contains 350 sets of word pairs with ratings ranging from 0 to 10, and the ratings are averaged. We calculate the vector pinch angle of these word pairs in each of the two corpora, and calculate the magnitude of the pinch angle change. The relationship between similar word pinch angle change and manual evaluation (positive example) is shown in Table 6. The similar words pinch angle change in relation to manual evaluation (negative example) is shown in Table 7. From the table, it can be seen that in the similar word pinch angle change versus manual evaluation relationship (positive example), the rate of pinch angle between training and automobile is 6.28.

Table 6. Similar words Angle change and artificial evaluation relationship

Word 1	Word 2	Arcoserr	Rate
Train	Car	0.001752	6.28

Vodka	Brandy	0.00385	8.07
Drink	Eat	0.00307	7.03
Journey	Voyage	0.00225	9.27
Coast	Shore	0.00846	8.94
Tool	Implement	0.01076	6.57
Money	Wealth	0.01176	8.15
Money	Withdrawal	0.000823	6.85
Tiger	Jaguar	0.00956	7.95
Tiger	Animal	0.01469	6.94

Table 7. Similar words Angle change and artificial evaluation relationship

Word 1	Word 2	arcoserr	rate
Psychology	Cognition	0.21326	7.33
Drug	Abuse	0.21051	6.92
Rock	Jazz	0.19252	7.51
Soap	Opera	0.19122	7.85
Stroke	Hospital	0.19112	6.97
Network	Hardware	0.18346	8.43
Shower	Thunderstorm	0.17947	6.24
Television	Film	0.16855	7.86
Aluminum	Metal	0.16797	8.05
Credit	Card	0.16897	7.94

A plot of the relationship between manual ratings of similarity and clip change in Wordsim 350 is shown in Figure 11. Scatter plots of the relationship between the magnitude of change in cosine distance and manual similarity ratings were plotted for all words in Wordsim 350. As can be seen from the plot, the similarity of manually evaluated word pairs roughly decreases with increasing cosine distance variation when sorted by the magnitude of change in clip angle between the two corpora. It is found that the more semantically similar the words are their positional relationship in the semantic space is more stable, i.e., the angle invariance between them is stronger. Therefore, it can be inferred that the representation invariance of word vectors is positively related to the semantics of the words.

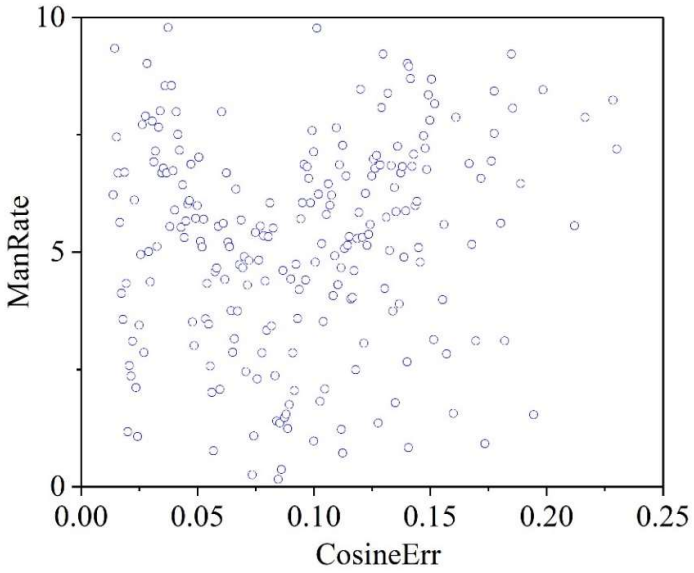


Fig 11. Artificial evaluation similarity and clamping

5. Conclusion

The study proposes a JS-Principal Component Analysis algorithm based on textual data features and constructs a high dimensional word vector space mapping construction based on a pre-trained model, aiming to improve the performance of the method for processing textual data of university languages. The contents of the conclusion are as follows:

The experimental results show that under different feature dimensions, the effect of this paper's method is better than that of JS, IG, PMI and ANOVA methods, which reflects the superiority of this paper's method in feature dimensionality reduction. The method of this paper can not only play a role in reducing the feature correlation, but also further reduce the number of feature dimensions.

In the analysis of the spatial mapping relationship of word vectors, the accuracy of the mapping reaches 11.2% in the words with word frequency sorted from the top 5000 to the top 6000, which can be obtained that the use of this paper's method can to a certain extent better realize the transformation between two vector spaces.

References

[1] Sun, S. (2020). Improving Students' Humanistic Quality in the Teaching of Chinese Language and Literature in Vocational Higher Education Institution. Journal of Contemporary Educational Research, 4(7).

[2] Gong, Y., Lyu, B., & Gao, X. (2018). Research on teaching Chinese as a second or foreign language in and outside mainland China: A bibliometric analysis. The Asia-Pacific Education Researcher, 27(4), 277-289.

[3] Liu, C., & Yang, P. (2024). A performance evaluation index for student satisfaction in online live classes of Chinese language and literature. European Journal of Education, 59(4), e12742.

[4] Gong, Y. F., Gao, X. A., & Lyu, B. (2020). Teaching Chinese as a second or foreign language to non-Chinese learners in mainland China (2014–2018). Language Teaching, 53(1), 44-62.

[5] Poole, A. (2019). Reading strategies and academic success: The case of first-semester college males. Journal of College Student Retention: Research, Theory & Practice, 21(1), 2-20.

[6] Angraini, W., & Rozimela, Y. (2020, August). The implementation genre-based approach in teaching reading at senior high school. In Eighth International Conference on Languages and Arts (ICLA-2019) (pp. 108-112). Atlantis Press.

[7] Dai, Y. (2021). The Impact of Genre-Based Pedagogy on Students' Reading Ability and Critical Thinking Quality Development in Content and Language Integrated Learning. Adv. Educ. Technol. Psychol, 23, 200-204.

[8] Bauer-Kealey, M., & Mather, N. (2019). Use of an online reading intervention to enhance the basic reading skills of community college students. Community College Journal of Research and Practice, 43(9), 631-647.

[9] Breakstone, J., Smith, M., Connors, P., Ortega, T., Kerr, D., & Wineburg, S. (2021). Lateral reading: College students learn to critically evaluate internet sources in an online course. The Harvard Kennedy School Misinformation Review.

[10] Chang, L., Wang, Y., Liu, J., Feng, Y., & Zhang, X. (2023). Study on factors influencing college students' digital academic reading behavior. Frontiers in psychology, 13, 1007247.

[11] Jindal, R., Malhotra, R., & Jain, A. (2015). Techniques for text classification: Literature review and current trends. webology, 12(2).

[12] Saarti, J. (2019). Fictional literature, classification and indexing. Ko Knowledge Organization, 46(4), 320-

332.

[13] Pintas, J. T., Fernandes, L. A., & Garcia, A. C. B. (2021). Feature selection methods for text classification: a systematic literature review. Artificial Intelligence Review, 54(8), 6149-6200.

[14] Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). Deep learning--based text classification: a comprehensive review. ACM computing surveys (CSUR), 54(3), 1-40.

[15] Wang, H. (2019). From construction of meanings to meaning design: A literacy-and genre-focused approach to academic Chinese. Chinese for Specific and Professional Purposes: Theory, Pedagogical Applications, and Practices, 3-24.

[16] Shavrina, T. (2018). Genre Classification on Text-Internal Features: a Corpus Study. ARANEA 2018, 134.

[17] Shen, H. H. (2018). Chinese as a second language reading: lexical access and text comprehension. The Routledge handbook of Chinese second language acquisition, 134-150.

[18] Sun, Y., Liu, J., Liu, W., Han, J., Ding, E., & Liu, J. (2019). Chinese street view text: Large-scale chinese text reading with partially supervised learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 9086-9095).

[19] Biduri, F., Rasyid, Y., & Emzir, E. (2018). The analysis of needs on learning materials in context-based reading Mandarin languages and culture. Journal of education, teaching and learning, 3(1), 9-16.

[20] Xue Jing & Shankar Achyut. (2024). Research on Korean literature corpus processing based on computer system improved TF-IDF algorithm. Intelligent Decision Technologies(4),3011-3024.

[21] Xiao Zunlan & Ning Xiaohong. (2025). Dynamic BERT-SVM Hybrid Model for Enhanced Semantic Similarity Evaluation in English Teaching Texts. Journal of English Language Teaching and Applied Linguistics(1),01-11.

[22] Cecilia N. Isonguyo,Eno E. Ituen,Ituen B. Okon,Monday E. Udoh,Akaninyene D. Antia & Akpan N. Ikot. (2024). Effects of magnetic and Aharonov–Bohm flux fields on information entropy with modified Kratzer plus screened Coulomb potential for selected diatomic molecules. Molecular Physics(24).

[23] Jikui Wang,Cuihong Zhang,Wei Zhao,Xueyan Huang & Feiping Nie. (2025). Fast anchor graph optimized projections with principal component analysis and entropy regularization. Information Sciences121797-121797.