Chinese Journal of Electronics
Vol.29, No.3, May 2020

Words in Pairs Neural Networks for Text Classification

WU Yujia, LI Jing, SONG Chengfang and CHANG Jun

(*School of Computer Science, Wuhan University, Wuhan 430072, China*)

Abstract — Existing methods utilized single words as text features. Some words contain multiple meanings, and it is difficult to distinguish its specific classification according to a single word, which probably affects the accuracy of the text classification. Propose a framework based on Words in pairs neural networks (WPNN) for text classification. Words in pairs include all single word combinations which have a high mutual association. Mine the crucial explicit and implicit Words in pairs as text features. These words in pairs as a text feature are easily classified. The words in pairs are utilized as the input of the neural network, which provides a better classification ability to the model, because they are more recognizable than the single word. Experimental results show that our model outperforms five benchmark algorithms.

Key words — Data mining, Neural networks, High utility itemset, Text classification, Words in pairs.

I. Introduction

Nowadays, most part of the information is presented in the form of text on the internet. Hence, text mining methods such as text classification have become attractive as research tasks of internet information processing[1]. Specifically, text classification is a method that automatically classifies the unknown text into predefined classifications[2]. Text classification is divided into topics (such as technology, entertainment) and sentiment classifications (sentimental ratings in product or movie reviews). According to the text granularity, it is divided into documents and sentence classifications.

The deep neural network model represented by Convolutional neural networks (CNNs)[3] has acquired achievement in various Natural language processing (NLP), and achieved an excellent experimental result. The main idea was to extract features of text using convolution and then constructed the neural network for sentence classification. The CNNs model was designed to solve the classification issues that belong to the field of computer vision and obtained immense success. Recently, CNNs model was applied in NLP, such as semantic analysis[4], information retrieval[5] and other text processing tasks. According to the complexity of the neural network in the text classification task, it can be divided into shallow neural network[3,6] and more complex deep neural network[7]. At the same time, CNNs model also obtained an excellent result in sentiment classification[8]. Because the context information used by Recurrent neural networks (RNNs)[9], they have satisfying results in some datasets. The Recurrent convolutional neural networks (RCNNs)[10] can also exploit the context information, utilize max pooling fusion text feature, and reduce complexity. The model based on attention mechanism are also beginning to appear in various NLP tasks, because it can more directly explain the importance of documents with various features[11].

A single word is usually regarded as a feature of the text in the text classification. However, a single word as a text feature has a low classification ability. Usually, if a single word or phrase frequently appears in one document and rarely appears in others at the same time, the word or phrase is suitable for classification and exhibits excellent performance as a text feature. Nevertheless, for some single words, it is not easy to determine the classification. Therefore, the classification is not easily ascertained when a single word is used as a feature of text. For example, as shown in Fig.1, it is difficult to distinguish whether the word "apple" refers to a "fruit" or "phone", as the probability of it belonging to one of them is the same. If we want to correctly classify the word, it is necessary to understand the single word from the context.

In response to this problem, we propose words in

Manuscript Received Dec. 25, 2018; Accepted Jan. 9, 2020. This work is supported by National Natural Science Foundation of China (No.61772180, No.41201404) and Fundamental Research Funds for the Central Universities of China (No.2042015gf0009).
 DOI:10.1049/cje.2020.03.005

pairs instead of a single word as input to the neural network. Words in pairs are a common combination of single words in natural language which represents a complete concept. In other words, words in pairs include all single word combinations which have a high mutual association. In Fig.1, if we utilize the words in pairs "apple-phone" as a feature of text for text classification, the probability of it belonging to "phone" is much higher than the probability of it belonging to "fruit". Similarly, the probability that the words in the pair "apple-juice" belong to "fruit" is much higher than in other classifications. Therefore, using the words in pairs "apple-phone" or "apple-juice" as features of text will considerably improve correct text classification. In conclusion, the words in pairs have a better classification ability because they have more recognizable semantics than the single word. In other words, the specific meaning of a single word as a feature of text is difficult to classify, and needs to be understood from the context. On the contrary, it is easy to classify the words in pairs as features of text even if there is no contextual information and the specific meaning is easier to obtain. However, although the explicit words in pairs as features of text have a very good performance and are suitable for text classification, their number is lacking in some documents, which may influence the accuracy. In order to fulfil the task of text classification and improve the accuracy, it is necessary to mine the implicit words in pairs.

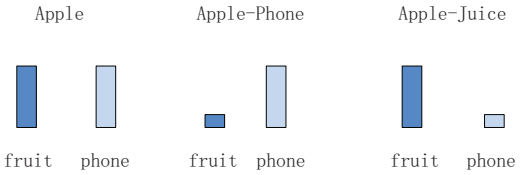

Fig. 1. Classification of the word "Apple"

Motivated by the above mentioned analysis, we intend to utilize the High utility itemset (HUI) algorithm to mine explicit and implicit words in pairs as features of text and as input to the neural network. The main innovations of this study are as follows:

1) For the classification issue of single words as text features, we replace them by words in pairs, which have strong importance and association as input of the neural network, thus obtaining a classifier with a stronger classification ability;

2) For the insufficient number of explicit words in pairs in some documents, we mine implicit words in pairs to improve the classification ability of the model;

3) It is proposed to utilize the HUI to mine words in pairs that are important and frequently appear together in each document. The words in pairs thus determined are used as inputs of the neural network, and tested through different types of datasets, to achieve good results.

II. Related Work

We briefly review the related text classification, and discuss about HUI methods in this section.

1. Text classification

CNNs were originally designed for tasks in computer vision and image processing fields. In recent years, in NLP tasks, deep neural networks such as CNNs and RNNs have been applied in text classification[3], semantic analysis[4], information retrieval[5], and other fields, achieving good experimental results. The main idea of CNNs was to utilize convolution to extract features, and design a convolutional neural network with a 5-layer structure. However, as it was a shallow CNNs[6], some researchers tried to utilize a complex neural network[7] replace shallow CNNs for text classification. In 2015, Tang *et al.*[8] utilized CNNs for sentiment classification.

In addition, RNNs[9] can utilize context information and have also achieved excellent performance in some datasets. RCNNs incorporated the CNNs and RNNs[10]. The structure of the RCNN is similar to that of the RNNs, which can provide the use of context information, and it utilizes the max pooling fusion to improve the accuracy of text classification.But RNNs cannot extract some local features appropriately, as opposed to CNNs that can properly obtain the local features of the text. In 2018, Wang[12] proposed a Disconnected recurrent neural networks model, which introduced the local features of the CNNs model to extract into the RNN model, thus achieving excellent results. Howard *et al.*[13] proposed a universal language model that can be utilized as a pre-training model. Zeng *et al.*[14] proposed a network to encode potential topic representations indicating class labels for solving problems of data sparsity. The attention mechanism has been widely used in various types of deep learning tasks. Because it is very interpretable, it is widely used[11].

In the existing algorithm of text classification, an important step is transforming single words to word embedding for convenient processing by computers. At present, word embedding has gradually become an important tool for the text processing task of NLP. Among the available tools, the Google team published the Word2vec tool in 2013[15], which has been widely applied in various tasks in the field of NLP since it was proposed. The Word2vec is a word embedding tool whose representation is the single words, and it was trained by Google from one billion Google news data. Another widely utilized word embedding tool is Glove[16]. Benefiting from these word embedding tools, Wang *et al.*[17] proposed a joint words and labels embedding method for text classification, which achieved excellent experimental results. Zied *et al.*[18] explored an unsupervised approach to classify documents into

categories simply described by a label.

Although Word2vec can represent the distance relationship between words very well, it cannot obtain the frequency of words appearing in some types of documents; it also cannot exclude shared words that usually appear in all types of documents, as they are useless for classification. However, words belonging to shared or private documents and appearing frequently has an important influence on the experimental result of text classification. For example, words such as "world" or "human" will appear frequently in all types of documents. However, these shared words do not contribute to the classification of the topic and influence the weight of private features, because they appear more frequently, thus influencing the experiment results of text classification. Some researchers have proposed methods to calculate word frequency, such as the traditional Term frequency Inverse document frequency (TF-IDF) method[19]. This method can calculate the frequency of words appearing in certain types of documents and also exclude the shared words that usually appear. However, the deep neural network is not utilized in the TF-IDF method. This method only calculates the word frequency and does not fully extract the local features of the text or utilize the word embedding tools. The experiment results of these methods are not as good as those of some deep learning methods[20,21] in text classification. For instance, Some scholars have tried to use capsule networks for text classification, and have achieved good experimental results[22,23]. Chen *et al.*[24] propose a transfer capsule network model for text classification. Frederick *et al.*[25] proposed an approach integrating feature attributions into the objective function to allow machine learning practitioners to incorporate priors in model building. Tatsuya *et al.*[26] proposed a method to simultaneously learn tokenization and text classification to address the problems of text segmentation. Kazuya *et al.*[27] presented a method for text categorization by leveraging the predominant sense of words depending on the domain-specific senses.

2. High utility itemset

Frequent itemset mining only considers the item that appears or not in the transaction rather than the quantity of items and their weight as unit profits in transaction databases. For this issue, HUI mining emerges as an option, because it considers the quantity and weight of items in the field of data mining. In 2009, Ahmed *et al.*[28] proposed an algorithm named Incremental high utility pattern (IHUP), which mines HUI using a tree structure to maintain the transaction and item utilities information. In 2010, Tseng *et al.* proposed algorithms called Utility pattern Growth (UP-Growth)[29] and UP-Growth+[30]. These two algorithms were both based on IHUP and introduced some pruning strategies to reduce the number of candidate itemsets, thus improving the algorithm performance. In addition, Tseng *et al.*[31,32] also proposed to mine the closed itemset without loss for acquiring the complete HUI and achieved excellent experiment results. The algorithms mentioned were based on a tree structure; they need to maintain the transaction and utility information on this structure and generate the candidate itemsets from the tree structure, finally computing the HUI from the candidate itemsets.

In 2012, Liu *et al.*[33] proposed a new algorithm named High utility itemset miner (HUI-Miner) for mining HUI. HUI-Miner was different from the above mentioned two-stage algorithms because it was based on a vertical data structure without generating a candidate itemset and instead directly generating the HUI. Firstly, a series of data structures named utility lists were generated, which included three fields: transaction, item utility, and remaining utility. Among them, the remaining utility of the itemset was overestimated utility information for pruning the search space. After generating all the utility lists, the HUI was generated by scanning the utility lists without generating any candidate itemset in this process.

In 2014, Philippe *et al.*[34] proposed a Faster high utility itemset (FHM) algorithm based on HUI-Miner and added a new strategy called Estimated utility co-occurrence pruning (EUCP) to reduce the number of connection operations. In 2015, Krishnamoorthy *et al.*[35] proposed the HUI method that employs novel pruning strategies, which was similar to the HUI-Miner and was also based on a vertical data structure without generating a candidate itemset, but it added two new pruning strategies. The performance of this algorithm was slightly better than that of the HUI-Miner. In 2015 and 2016, Sahoo *et al.* and Wu *et al.*, respectively, implemented the algorithms Closed+ high utility itemset miner (CHUM)[36] and Closed+ high utility itemset mining without candidates (CHUIMiner)[37] for mining HUI based on the closed vertical structure. In 2017, Guo *et al.* proposed the High utility itemset transaction weighted utility (HUITWU)[38] algorithm, which combined the vertical structure and tree structure algorithms.

III. Words in Pairs Neural Networks

1. Mining words in pairs

The words in pairs are a common combination of single words in natural language which represent a complete concept. In other words, the words in pairs include all single word combinations which have a high mutual association. In this study, we utilize the algorithm of HUI mining for obtaining the words in pairs whose utility is higher than the threshold. Each item represents a word. Its related definition

is as follows. $D = \{T_1, T_2, \cdots, T_n\}$ is considered to be a classification of documents, that includes a set of sentences. Each sentence $T_d(1 \leq d \leq n)$ contains k words as T_d=$\{i_1, i_2, \cdots, i_k\}$. Each word i_k appears one or more times in the sentence T_d , and the quantity of the word i_k is regarded as the utility value. Let $U(i_k, T_d)$ be the utility value. Table 1 shows the word and quantity of each sentence in document D, which has 4 sentences. Each sentence has several words, each word appears one or more times in the sentence, and they are represented by capital letters.

Table 1. Example of document D

ID	Sentence	Word and quantity
T_1	$ACBECE$	$(A,1)(B,1)(C,2)(E,2)$
T_2	$ADAFEF$	$(A,2)(D,1)(E,1)(F,2)$
T_3	$ABDFD$	$(A,1)(B,1)(D,2)(F,1)$
T_4	$BDEDBE$	$(B,2)(D,2)(E,2)$

Table 1 presents the raw data of a certain document classification and the words and their quantity in the sentence. Evidently, the HUI mining not only considers a word appearing in a sentence but also the quantity of words in the sentence.

Definition 1 The utility value of a word i_k in the sentence T_d is denoted as $U(i_k, T_\mathrm{d})$ and defined as $\mathrm{U}(i_k, T_\mathrm{d}) = \mathrm{Count}(i_k, T_\mathrm{d})$, where $\mathrm{Count}(\cdot)$ is the quantity of the word. For example, in Table 1, $U(\{C\}, T_1) = 2$.

Definition 2 The utility value of words in pairs X in the sentence T_d is denoted as $U(X, T_\mathrm{d})$ and defined as $U(X, T_\mathrm{d}) = \sum_{i_k \in X \wedge X \subseteq T_\mathrm{d}} U(i_k, T_\mathrm{d})$. For example, in Table 1, $U(\{AC\}, T_1) = U(\{A\}, T_1) + U(\{C\}, T_1) = 3$.

Definition 3 The utility value of words in pairs X in the document D is denoted as $U(X)$ and defined as $U(X) = \sum_{X \subseteq T_\mathrm{d} \wedge T_\mathrm{d} \subseteq D} U(X, T_\mathrm{d})$. For example, in Table 1, $U(\{AE\}) = U(\{AE\}, T_1) + U(\{AE\}, T_2) = 6$.

Problem statement Given a specified threshold. The words in pairs X are called high utility words in pairs if $U(X)$ is no less than the threshold. Otherwise, they are called low utility words in pairs. For example, in Table 1, assume that the $threshold = 4$ and AE represents high utility words in pairs because $U(AE)$=6.

After acquiring all the high utility words in pairs in each classification of the document, a filter Ψ_k is constructed, where Ψ_k represents the filter from the kth classification of all words in pairs whose utility is no less than the threshold. The specific calculation method is as Eq.(1).

$$\Psi_k = \{X | X \in D^k, U(X) \geq threshold\} \tag{1}$$

where the k represents the label of classification and D^k is the kth classification document datasets, X is the words in pairs in the D^k. For the specific method of mining the HUI, please refer to to Ref.[33].

2. Implicit words in pairs

The words in pairs consist of two single words and the higher their utility value, the stronger the association. For the problem of insufficient number of explicit words in pairs in some documents, it is necessary to mine the implicit ones to expand the number of words in pairs.

Table 2. Example of words in pairs

(Implicit) Words in pairs	Utility	Distance
AB	4	0.67
AF	6	1
BF	3.6	0.6

For example, the words in pairs AB frequently appear in the sample and their utility is $U(AB) = 4$ are shown in Table 2. The words in pairs AF also frequently appear in the sample and their utility is $U(AF) = 6$. Assuming a threshold = 4, the words in pairs AB and AF are both explicit, with utility no less than the threshold. These words need to be maintained for the filter Ψ_k. However, the words in pairs BF do not frequently appear in the sample because their utility is less than the threshold. In fact, words B and F have an implicit association because they both have strong association with the word A. By calculating the implicit distance of the words in pairs BF, the implicit utility of the BF is obtained. If the implicit utility of the words in pairs BF is higher than the threshold, it is considered to be maintained for the filter Ψ_k.Otherwise, it does not need to be maintained.

Assume the words in pairs XY with utility $U(XY)$. The distance of the words X and Y can be computed as Eq.(2).

$$\Delta_{XY} = \frac{U(XY)}{\max(U(\cdot))} \tag{2}$$

where $\max(U(\cdot))$ represents the maximum utility of the words in pairs in one classification document. For example, in Table 2, $\max(U(\cdot))$=6.

Assume that $\boldsymbol{x} = \{x_1, x_2, \cdots, x_m\}$ represents all the distance of explicit words in pairs that have the common prefix X, and $\boldsymbol{y} = \{y_1, y_2, \cdots, y_n\}$ represents all the distance of explicit words in pairs that have the common prefix Y. Thus, the implicit distance of the words in pairs XY can be computed as Eq.(3).

$$\Delta_{XY} = \frac{\theta}{2}\sqrt{\frac{1}{M}\sum_{i=1}^{M} {x_i}^2 + \frac{1}{N}\sum_{i=1}^{N} {y_i}^2} \tag{3}$$

where θ represents the implicit weight, which is usually 10% to 100% in this study.

Fig.2 shows the calculation method of the distance of the implicit words in pairs BF. Here, θ is 100%, then $\Delta_{BF} = \frac{1}{2}\sqrt{{\Delta_{AB}}^2 + {\Delta_{AF}}^2} = \frac{1}{2}\sqrt{0.67^2 + 1^2}$=0.6. After

calculating the distance of the implicit words in pairs BF, we calculate its implicit utility by Eq.(4). For example, $U(BF)$=max$(U(\cdot)) \times \Delta_{BF} = 6 \times 0.6 = 3.6$.

$$U(XY)=\max(U(\cdot)) \times \Delta_{XY} \tag{4}$$

If the implicit utility of the words in pairs is no less than the threshold, it needs to be maintained. Otherwise, the words in pairs do not need to be maintained. Regarding the single words in the same classification, the shorter the distance between words, the stronger their association. On the contrary, if the distance between words is longer, the association between them is weaker.

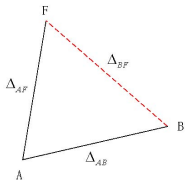

Fig. 2. Distance of implicit words in pairs

3. Neural network framework

In this study, the mining of original document datasets of each classification with the high utility words in pairs, was performed. For each classification of datasets, we split the datasets into training data and test data. We mined the high utility words in pairs on the training data and obtained all the words in pairs with utility of no less than the threshold. Then the words in pairs with high utility were utilized as the text feature and received a word vector representation. The text feature after the word embedding was utilized as the input of the neural network, and the output was obtained through a connected layer. Fig.3 shows the specific framework of the model of words in pairs neural networks.

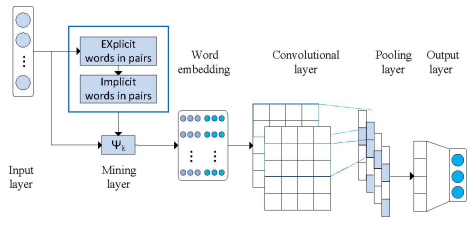

Fig. 3. Model of the Words in pairs neural networks

The model of words in pairs neural networks is divided into 6 layers. The first is the input layer. The input data are the original document datasets of each classification. We mine the high utility words in pairs and obtain all the words in pairs with utility no less than the threshold, which are maintained. In the second layer, all the high utility words in pairs with utility no less than the threshold from the kth classification document constitute the filter Ψ_k as the mining layer. The third layer is the word embedding layer. We utilize the word embedding tools for the word vector representation of these features, which have been passed through Ψ_k, obtaining the word vector matrix. Each line of the matrix represents a words in pairs. The fourth layer of the model is the convolution layer, and utilizes the convolution filter of different sizes with convolution operation to obtain the word vector matrix; we further extract the high-level features of the text for constituting the feature map. The fifth layer is the pooling layer of the model and the max pooling is utilized for maintaining the most significant features of the feature map, reducing the network complexity. The last layer is the output layer which is connected to the pooling layer by a fully connected method,and the output is the classification label.

Fig.3 is the framework of the proposed model. Let I_k be the m-dimensional word vector corresponding to the kth words in pairs. A sentence T=$\{I_1, I_2, \cdots, I_k\}$ includes k words in pairs. A convolution filter $\boldsymbol{w}_n \in \mathbf{R}^{hk}$ is represented through a window of h words to produce a new feature.

It should be noted that the filter $\boldsymbol{w}$ only moves downwards when performing the convolution operation, because each of the sentences represents an indivisible words in pairs. Z_n is the high-level feature obtained by the convolution filter $\boldsymbol{w}$ through the convolution operation involving the sentence T, as Eq.(5).

$$Z_n = f(\boldsymbol{w} \cdot T + b) \tag{5}$$

where $b \in \mathbf{R}$ is the bias term, and the $f(\cdot)$ is a nonlinear hyperbolic tangent function. We obtain multiple high-level features through a convolution operation that involves sentence T by utilizing n filters of different sizes and producing a set of feature maps, as Eq.(6).

$$\boldsymbol{Z} = [Z_1, Z_2, \cdots, Z_n] \tag{6}$$

Meanwhile, to reduce the complexity of the network and maintain the most significant feature, a maximum pooling operation, as Eq.(7).

$$z = \max(\boldsymbol{Z}) \tag{7}$$

After the max pooling operation, these feature maps constitute the sixth layer of the network, and utilize a fully connected network to transmit to the output layer the probability distribution of the output label. In the output layer, we utilize the $Softmax$ function as the classifier that predicts the probability distribution over classes, as Eq.(8).

$$softmax_j = \frac{\exp(z)}{\sum_{j=1}^{C} \exp(Z_j)} \tag{8}$$

where $softmax_j$ are prediction probabilities, C is the class label, z is the input of the fully connected layer from the pooling layer. The last step is to obtain the maximum probability through $Softmax$ function, which is the obtained classification result.

IV. Experiments

In this study, according to the different methods of word vector representation, three experiments were performed utilizing One-hot, Word2vec, and Glove. In addition, different thresholds were set to filter features with weak classification abilities or considered to be meaningless.

1. Experimental datasets

To evaluate the performance of the WPNN, we performed extensive experiments on various databases, as follows.

MR：This dataset is some movie reviews and involves two classifications, positive and negative.

MPQA：Opinion polarity dataset that uses the opinion as a task of classification, including positive and negative.

SUBJ：Subjective dataset involves two classifications, the sentence is subjective or objective.

SST1：This dataset from Stanford. It is divided into five classifications according to the sentimental level of the sentence: very positive, positive, neutral, negative, and very negative.

SST2：The dataset is same as SST-1, but it removes the neutral review and then divides the reviews into two classifications:positive and negative.

TREC：TREC problem dataset classifies the problem into six question types.

IMDB：The movie review dataset from IMDB is divided into 10 classifications utilizing a rating score range from one to ten.

YELP13：Yelp reviews from the Yelp challenge dataset, which is the review data of various restaurants, hotels, and other merchants on the famous American review' s website Yelp. Each example consists of several review sentences and classifies the score into five ratings.

YELP14：The same as YELP13 and divided into five classifications.

2. Baselines algorithm and parameter settings

In this section, we compare our approaches with several types of methods including CNN [3], FAST [39], HAN [11], RNN [9], and RCNN [10]. The experiments were conducted on an Intel i7-8700 machine, 3.2GHz CPU and 11G NVIDIA GeForce GTX 1080 Ti GPU with 32 GB of memory, running on a Windows 10. The WPNN algorithm was implemented in python. For all datasets, 10% of the data was split as the test data. Relevant hyper-parameters are as follows: the dimension was set as 128/300, being dimension 128 on the One-hot encoding and 300 on the Word2vec or Glove encoding, we utilized the convolution windows size of 2, 3, 4, 5, and the number of each convolution kernel was 128; the dropout ratio was set as 0.5, the L2 regularized lambda value was 1.0 and the batch size was 64.

3. Evaluation using One-hot encoding

Both the experimental results of the WPNN and the comparison algorithm under One-hot coding are shown in Table 3. The WPNN algorithm achieves the best results on the nine datasets compared with the other five algorithms. Among them, on the MR dataset, the accuracy of WPNN was 79.5%, the accuracy of RCNN was 76.1%, and for the other 4 algorithms it was lower than 76%. The advantages of the WPNN are evident in the experimental results.

On the MPQA dataset, the accuracy of WPNN was 84.7%, the accuracy of RNN was 84.2%, and the accuracy of the other four benchmark algorithms was less than 84%. On the SUBJ dataset, WPNN had an accuracy of 92.7%, both RNN and RCNN had an accuracy of 90.6%, and the other three benchmark algorithms had accuracies that were below 91%. Therefore, on the MPQA and SUBJ datasets, the experimental results of the WPNN algorithm were also better than those of the other benchmark algorithms.

On the SST1 dataset, the experimental results of the current algorithms are not excellent because this is a movie review dataset whose classification is based on the sentiment level of the review. Very positive and positive are difficult to distinguish, and the negative and very negative are also very difficult to distinguish. The accuracy of WPNN algorithm on this dataset was 41.7%, while for the other benchmark algorithms it was below 40%. The SST2 dataset removes the neutral review and divides the dataset into only two classifications: positive and negative. Hence, on the SST2 dataset, the accuracy of WPNN increased to 81.8%, and the other algorithms correspondingly improved, but not to more than 80%. On the TREC dataset, the accuracy of WPNN was 85.3%, the accuracies of CNN and RCNN were 84.4% and 84.3%, respectively, and for the other 3 benchmark algorithms, it was less than 80%. Therefore, the experimental results of WPNN algorithm were better than other algorithms on SST1, SST2, and TREC datasets under One-hot encoding.

In sequence, on the large dataset IMDB, the accuracy of WPNN reached 43.5% and the accuracy of other algorithms on this dataset was less than 41%. YELP2013 and YELP2014 are user reviews datasets with five classifications. WPNN achieved 60.4% experimental accuracy on the YELP2013 dataset, while the accuracy of the other algorithms was less than 60%. For the

YELP2014 dataset, the experimental accuracy of the WPNN was 61.5%, the experimental accuracy of CNN on this dataset reached 61.5%, and the other four benchmark algorithms reached less than 61%.

Table 3. Evaluation using One-hot encoding

Dataset	CNN	FAST	HAN	RNN	RCNN	WPNN
MR	74.5	65.2	74.9	75.1	**76.1**	**79.5**
MPQA	81.3	69.6	80.4	**84.2**	82.6	**84.7**
SUBJ	90.4	83.3	90.1	**90.6**	**90.6**	**92.7**
SST1	38.8	26.5	36.4	38.1	**39.1**	**41.7**
SST2	**79.5**	68.7	77.2	75.8	78.9	**81.8**
TREC	**84.4**	46	78.6	79.7	84.3	**85.3**
IMDB	39.7	22.4	29.6	38.8	**40.9**	**43.5**
YELP13	57.2	41.3	49.3	51.4	**59.3**	**60.4**
YELP14	**61.5**	41.6	56.4	58.3	**60.3**	**61.5**

4. Evaluation using Word2vec encoding

In the case of utilizing the Word2vec tool, the experimental results of the WPNN algorithm and the other five algorithms are shown in Table 4. Utilizing the Word2vec tool can significantly improve the accuracy of the classification. In Table 4, WPNN achieved 79.8% accuracy on the MR dataset utilizing the Word2vec tool, thus increased 0.3 percentage points compared to One-hot encoding. The other five algorithms also achieved corresponding performance improvement, but the accuracy rate did not exceed 80%; thus, on this dataset, the WPNN algorithm had a better performance than the other five benchmark algorithms. On the MPQA dataset, the accuracy of WPNN was 88.4%, the accuracies of RNN and CNN were 88.3% and 88.1%, respectively. The experimental results of the other algorithms are close, except for the FAST algorithm. On the SUBJ dataset, the accuracy of WPNN was 94.2%, the accuracy of RCNN was 94.2%, and for the other four algorithms it was less than 93%.

Table 4. Evaluation using Word2vec encoding

Dataset	CNN	FAST	HAN	RNN	RCNN	WPNN
MR	77.6	70.3	71.3	78.5	**79.3**	**79.8**
MPQA	88.1	68.4	85.7	**88.3**	87.6	**88.4**
SUBJ	**92.7**	86.4	91.7	92.5	**94.2**	**94.2**
SST1	42.3	34.1	42.1	32.1	**44**	**44.8**
SST2	**84.4**	73.1	83.5	79.8	**84.5**	**84.5**
TREC	88.2	39.9	55.1	**88.4**	88.2	**88.6**
IMDB	**43.2**	28.3	41	21.3	43	**45.6**
YELP13	60	46.9	61.2	**62.5**	59.1	**63**
YELP14	**63.6**	44.9	63.3	63	60.8	**64.8**

On the SST1 dataset, the performance of the algorithm generally improved due to the use of the Word2vec tool. The accuracy of WPNN was 44.8%, the accuracy of RCNN was 44%, and the other algorithms reached less than 43%. On the SST2 dataset, the accuracy of WPNN increased to 84.5%, and the other algorithms correspondingly improved. On the TREC dataset, the accuracy of WPNN was 88.6%, and the accuracy of RNN reached 88.4%, CNN and RNN had the same accuracy rate of 88.2%. Therefore, on the TREC dataset, the WPNN performance was slightly better than that of the other algorithms. On the IMDB dataset, the WPNN accuracy rate reached 45.6%. The other algorithms had an accuracy rate lower than 44%. The WPNN achieved 63% experimental accuracy on the YELP2013 dataset, while the other algorithms achieved less than 63%. The experimental accuracy of WPNN was 64.8% on the YELP2014 dataset and the accuracy of the other five benchmark algorithms was less than 64%. Therefore, when utilizing the Word2vec tool, the performance of the WPNN was superior to that of the other five benchmark algorithms in the YELP dataset.

5. Evaluation using Glove encoding

The experimental results of the WPNN and other algorithms utilizing the Glove tool are shown in Table 5. Although Glove and Word2vec are differently implemented, the experimental results are closely good and they are better to improve the accuracy of classification compared to the One-hot coding. Unlike Word2vec, the Glove tool utilize the global information of the entire corpus.

Table 5. Evaluation using Glove encoding

Dataset	CNN	FAST	HAN	RNN	RCNN	WPNN
MR	78.9	71.1	77.2	79.2	**80.5**	**81.1**
MPQA	**89.6**	69.6	87.6	88.6	88.9	**89.5**
SUBJ	93.2	88.6	91.5	**93.6**	**94.1**	**94.1**
SST1	43.8	30	35.8	43.8	**44.7**	**44.5**
SST2	**85.6**	71.8	79.4	83.9	85.3	**85.4**
TREC	88.2	40	67.9	86.9	**89.4**	**89.6**
IMDB	40.5	27	40.1	39.7	**42.2**	**45.6**
YELP13	54.4	41.3	47.7	41.3	**56.8**	**64**
YELP14	59.9	41.6	58.1	56.3	**61.2**	**65.5**

In Table 5, the advantage of the WPNN algorithm MR dataset is represented, having accuracy of 81.1%; the accuracy of RCNN was 80.5%, and other algorithms also achieved good results. In the MPQA dataset, the accuracy of WPNN was 89.5%, the accuracy of CNN was 89.6%, and the other four algorithms achieved less than 89%. In the SUBJ dataset, the accuracies of WPNN and RCNN were both 94.1%, and the other four algorithms achieved more than 88%. Thus, the performance of these algorithms was closely in the SUBJ dataset. On the SST1 dataset, the accuracy of WPNN was 44.5%, while the accuracy of RCNN was 44.7%. On the SST2 dataset, CNN achieved the best results, with an accuracy of 85.6%, while the accuracy of WPNN increased to 85.4%, and the RCNN reached 85.3%. On this dataset, the performance of these algorithms was also closely.

On the TREC dataset, the accuracy of the WPNN was 89.6% and the experimental results were better

than other benchmark algorithms results. On the larger dataset IMDB, the accuracy of WPNN reached 45.6%, the same obtained with Word2vec. Several algorithms had an accuracy of less than 43% on this dataset. In YELP2013 and YELP2014 datasets, the WPNN achieved 64% and 65.5% experimental accuracy, respectively, while the other datasets achieved less than 62%. On YELP datasets, WPNN achieved the best performance.

6. Different thresholds

For each different type of document, defining thresholds of different sizes mines a different number of words in pairs, resulting in a different number of text features. This has an impact on the experimental results. Fig.4 shows WPNN algorithm at different thresholds of experimental results utilizing One-hot coding.

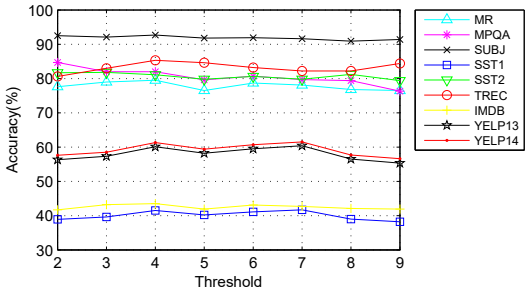

Fig. 4. WPNN evaluation using One-hot encoding at different thresholds

As can be seen from Fig.4, the user-defined threshold that is too high or too low affects the classification accuracy, because too thresholds that are too high may filter too many important features and thresholds that are too low will maintain many text features with low utility values, leading to a reduction in the classification accuracy. Fig.4 shows the experimental results of the WPNN algorithm at different thresholds utilizing One-hot encoding. Among the datasets, on the MR, SUBJ, TREC and IMDB, the best experimental results were obtained when the threshold was 4. On the MPQA dataset, the best result was obtained with the threshold of 2; on the SST2 dataset, the best result was obtained with the threshold of 3; on SST1, YELP2013 and YELP2014 datasets, the best performance was obtained with a threshold of 6. Therefore, based on the analyses, it was confirmed that a too low or too high threshold affects the classification accuracy.

Fig.5 shows the experimental results of the WPNN algorithm at different thresholds on nine different datasets utilizing the Word2vec tool. As can be seen from this figure, the experimental results for the WPNN were the best on the datasets MR, MPQA, SUBJ, SST2, and TREC, when the threshold was set to 2 and as the threshold increased, the accuracy descended. On the SST1 dataset, when the threshold was set to 4, the experimental results were also the best. On three large datasets such as IMDB, YELP2013 and YELP2014, the threshold was set to 6, and the best performance was obtained.

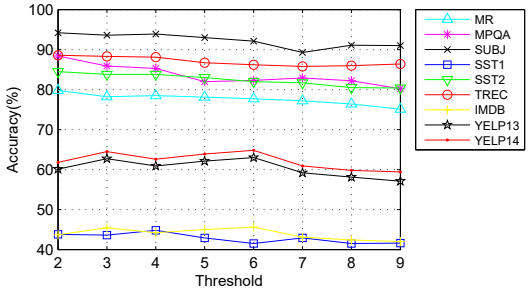

Fig. 5. WPNN evaluation using Word2vec encoding at different thresholds

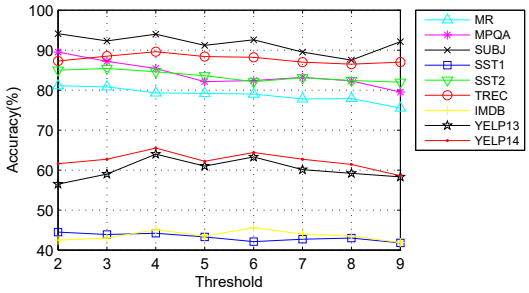

Fig. 6. WPNN evaluation using Glove encoding at different thresholds

Fig.6 shows the experimental results of the WPNN algorithm at different thresholds on nine different datasets using the Glove tool. From this figure, we can see that on the datasets MR, MPQA, SUBJ and SST1, when the threshold was 2, the experimental results were the best, and as the threshold increased, the accuracy descended. On the SST2 dataset, when the threshold was set as 3, and in the IMDB dataset, when the threshold was 6, the WPNN obtained the best performance. This is the same result as the obtained utilizing the Word2vec tool. On the datasets TREC, YELP2013 and YELP2014, when the threshold was set to 4, the results were also the best.

V. Conclusions

In this study, we proposed a framework based on WPNN for text classification. Firstly, we proposed a calculation implicit distance method to obtain the implicit distance between words for mining the implicit words in pairs based on the HUI. In sequence, we elected the words in pairs as text features with strong importance and association as inputs of the neural network and obtained a classifier with a stronger classification ability. After mining all the words in pairs, we used a threshold to extract the most critical words in pairs, which were used as inputs of the neural network; then, we utilized convolution and pooling operations to extract higher level text features to improve the accuracy of

text classification. We performed experiments on nine benchmark datasets and five benchmark text classification algorithms for comparison. According to the different methods of word vector representation, three experiments were performed utilizing One-hot, Word2vec, and Glove. The proposed algorithm significantly outperforms the benchmark algorithm on One-hot vector encoding. It also achieves good experimental results on Word2vec and Glove vector encoding.

References

[1] WU Lianwei,RAO Yuan,YU Hualei,*et al.*, "A multi-semantics classification method based on deep learning for incredible messages on social media", *Chinese Journal of Electronics*, Vol.28, No.4, pp.754–763, 2019.

[2] MENG Deyu,SUN Lina, "Some new trends of deep learning research", *Chinese Journal of Electronics*, Vol.28, No.6, pp.1087–1091, 2019.

[3] Y. Kim, "Convolutional neural networks for sentence classification", *Proc. of Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp.1746–1751,2014.

[4] W. Yih, X. He and C. Meek, "Semantic parsing for single-relation question answering", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, USA, pp.643–648,2014.

[5] Y. Shen, X. He, J. Gao, *et al.*, "Learning semantic representations using convolutional neural networks for web search", *Proc. of International Conference on World Wide Web*, Seoul, Korea, pp.373–374, 2014.

[6] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks", *Proc. of Conference on North American Chapter of the Association for Computational Linguistics – Human Language Technologies*, Denver, Colorado, USAA, pp.103–112,2015.

[7] X. Zhang, J. Zhao and Y. LeCun, "Character-level convolutional networks for text classification", *Proc. of Conference on Advances in Neural Information Processing Systems*, Montreal, Canada, pp.649–657,2015.

[8] D. Tang, B. Qin and T. Liu, "Learning semantic representations of users and products for document level sentiment classification", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Beijing, China, pp.1014–1023,2015.

[9] P. Liu, X. Qiu and X. Huang, "Recurrent neural network for text classification with multi-task learning", *Proc. of Conference on International Joint Conference on Artificial Intelligence*, Melbourne, Australia, pp.1480–1489,2017.

[10] S. Lai, L. Xu, K. Liu, *et al.*, "Recurrent convolutional neural networks for text classification", *Proc. of Conference on Association for the Advancement of Artificial Intelligence*,Austin, Texas, USA, pp.2267–2273, 2015.

[11] Z. Yang, D. Yang, C. Dyer, *et al.*, "Hierarchical attention networks for document classification", *Proc. of Conference on North American Chapter of the Association for Computational Linguistics – Human Language Technologies*, San Diego, California, USA, pp.1480–1489, 2016.

[12] B. Wang, "Disconnected recurrent neural networks for text categorization", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, pp.2311–2320,2018.

[13] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, pp.328–339,2018.

[14] J. Zeng, J. Li, Y. Song, *et al.*, "Topic memory networks for short text classification", *Proc. of Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp.3120–3131, 2018.

[15] T. Mikolov, I. Sutskever, K. Chen, *et al.*, "Distributed representations of words and phrases and their compositionality", *Proc. of Conference on Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, USA, pp.3111–3119, 2013.

[16] J. Pennington, R. Socher and C. Manning, "Glove: Global vectors for word representation", *Proc. of Conference on Empirical Methods in Natural Language Processing*, Doha, Qatar, pp.1532–1543,2014.

[17] G. Wang, C. Li, W. Wang, *et al.*, "Joint embedding of words and labels for text classification", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, pp.2321–2331, 2018.

[18] Z. H. Yahia, A. Sieg, L. A. Deleris, "Towards unsupervised text classification leveraging experts and word embeddings", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.371–379,2019.

[19] B. Trstenjak, S. Mikac and D. Donko, "KNN with TF-IDF based framework for text categorization", *Procedia Engineering*, Vol.69, No.1, pp.1356–1364, 2014.

[20] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, pp.562–570,2017.

[21] L. Zhang, S. Wang, B. Liu, "Deep learning for sentiment analysis: A survey", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol.8, No.4, pp.1–34, 2018.

[22] M. Yang, W. Zhao, L. Chen,*et al.*, "Investigating the transferring capability of capsule networks for text classification", *Neural Networks*, Vol.118, No.6, pp.247–261, 2019.

[23] W. Zhao, H. Peng, S. Eger, *et al.*, "Towards scalable and reliable capsule networks for challenging NLP applications", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.1549–1559, 2019.

[24] Z. Chen and T. Qian, "Transfer capsule network for aspect level sentiment classification,", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.547–556, 2019.

[25] F. Liu and B. Avci, "Incorporating priors with feature attribution on text classification,", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.6274–6283, 2019.

[26] T. Hiraoka, H. Shindo and Y. Matsumoto, "Stochastic tokenization with a language model for neural text classification", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.1620–1629, 2019.

[27] K. Shimura, J. Li, F. Fukumoto, "Text categorization by learning predominant sense of words as auxiliary task,", *Proc. of Conference on Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, pp.1109–1119, 2019.

[28] C. F. Ahmed, S. K. Tanbeer, B.S. Jeong,*et al.*, "Efficient tree structures for high utility pattern mining in incremental databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol.21, No.12, pp.1708–1721, 2009.

[29] V. S. Tseng, C.W. Wu, B.E. Shie, *et al.*, "Up-growth: An efficient algorithm for high utility itemset mining", *Proc. of Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, pp.253–262, 2010.

[30] V. S. Tseng, B.E. Shie, C.W. Wu, *et al.*, "Efficient algorithms for mining high utility item-sets from transactional databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol.25, No.8, pp.1772–1786, 2013.

[31] C.W Wu, P. Fournier-Viger, P. S. Yu, *et al.*, "Efficient mining of a concise and lossless representation of high utility itemsets", *Proc. of Conference on IEEE International Conference on Data Mining*, Vancouver, Canada, pp.824–833, 2011.

[32] V. S. Tseng, C.W Wu, P. Fournier-Viger, *et al.*, "Efficient algorithms for mining the concise and lossless representation of high utility itemsets", *IEEE Transactions on Knowledge and Data Engineering*, Vol.27, No.3, pp.726–739, 2015.

[33] M.C. Liu and J.F. Qu, "Mining high utility itemsets without candidate generation", *Proc. of Conference on EMNLP*,Maui, Hawaii, USA, pp.55–64,2012.

[34] P.Fournier-Viger, C. W.Wu, S.Zida, *et al.*, "FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning", *Proc. of Conference on International Symposium on Methodologies for Intelligent Systems*, Roskilde, Danmark, pp.83–92, 2014.

[35] S. Krishnamoorthy, "Pruning strategies for mining high utility itemsets", *Expert Systems with Applications*, Vol.42, No.5, pp.2371–2381, 2015.

[36] J. Sahoo, A.K. Das and A. Goswami, "An efficient fast algorithm for discovering closed+ high utility itemsets", *Applied Intelligence*, Vol.45, No.1, pp.44–74, 2016.

[37] C.W. Wu , P. Fournier-Viger , J.Y. Gu, *et al.*, "Mining closed+ high utility itemsets without candidate generation", *Proc. of Conference on Technologies and Applications of Artificial Intelligence*, Taiwan, China, pp.1–8, 2015.

[38] S.Guo and H.Gao, "HUITWU: An efficient algorithm for high-utility itemset mining in transaction databases", *Journal of Computer Science and Technology*, Vol.31, No.4, pp.776–786, 2016.

[39] A. Joulin, E. Grave, P. Bojanowski, *et al.*, "Bag of tricks for efficient text classification", *Proc. of Conference on European Chapter of the Association for Computational Linguistics*, Valencia, Spain, pp.427–431, 2017.

WU Yujia was born in 1986. He is a Ph.D. candidate in School of Computer Science, Wuhan University, Wuhan, China. His main research interests include data mining and natural language processing.
(Email: wuyujia@whu.edu.cn)

LI Jing (corresponding author) was born in 1967. He received the Ph.D. degree from Wuhan University, Wuhan, China, in 2006. He is currently a Professor in Computer School of Wuhan University, Wuhan, China. His research interests include data mining and multimedia technology.
(Email: leejingcn@whu.edu.cn)

SONG Chengfang was born in 1978. He received the Ph.D. degree from Zhejiang University in 2007, Hangzhou, China. He is currently an Assistant Professor in Computer School of Wuhan University, Wuhan, China. His current research interests include visualization analysis and location service.
(Email: songchf@whu.edu.cn)

CHANG Jun was born in 1972. He received the Ph.D. degree from Wuhan University in 2011, Wuhan, China. He is currently an Assistant Professor in School of Computer Science, Wuhan University, Wuhan, China. His current research interests include computer vision, large-scale machine learning and stream data mining.
(Email: chang.jun@whu.edu.cn)