Sentence Classification Model Based on Convolutional Neural Networks and Bayesian Classifiers (Postprint)

Authors: Li Wenkuan, Liu Peiyu, Zhenfang Zhu, Liu Wenfeng

Date: 2018-12-13T00:00:00+00:00

Abstract

Traditional sentence classification models suffer from deficiencies such as complex feature extraction processes and low classification accuracy. By leveraging the feature extraction advantages of currently prevalent convolutional neural networks based on deep learning models and combining them with traditional sentence classification approaches, we propose a sentence classification model based on convolutional neural networks and Bayesian classifiers. The model first utilizes convolutional neural networks to extract text features, then employs principal component analysis to reduce the dimensionality of these features, and finally uses a Bayesian classifier for sentence classification. Experimental results on the publicly available movie review dataset from Cornell University and the sentiment classification dataset from Stanford University demonstrate that the proposed method outperforms both deep learning-only models and traditional sentence classification models.

Full Text

Preamble

Vol. 37 No. 2
Application Research of Computers
ChinaXiv Partner Journal

Sentence Classification Model Based on Convolutional Neural Network and Bayesian Classifier

Li Wenkuan[1],[2], Liu Peiyu[1],[2]†, Zhu Zhenfang[3], Liu Wenfeng[1],[2],[4]

1. School of Information Science & Engineering, Shandong Normal University, Jinan 250014, China

2. Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan 250014, China

3. School of Information Science & Electric Engineering, Shandong Jiaotong University, Jinan 250014, China

4. School of Computer Science, Heze University, Heze, Shandong 274015, China

Abstract: Traditional sentence classification models suffer from complex feature extraction processes and low classification accuracy. This paper proposes a sentence classification model based on convolutional neural networks (CNN) and Bayesian classifiers by leveraging the advantages of the popular deep learning-based CNN in feature extraction and combining it with traditional sentence classification methods. The model first uses a CNN to extract text features, then applies principal component analysis (PCA) to reduce the dimensionality of these features, and finally employs a Bayesian classifier for sentence classification. Experimental results demonstrate that on Cornell University' s public movie review dataset and the Stanford Sentiment Treebank dataset, the proposed method outperforms models using only deep learning or traditional sentence classification approaches.

Keywords: deep learning; sentence classification; convolutional neural network; principal component analysis; Bayesian classifier

0 Introduction

Sentence classification [1] is a core task in natural language processing (NLP) that has attracted widespread attention from NLP researchers in recent years, becoming a hot research topic. It involves the computational study of people' s opinions, emotional evaluations, and attitudes toward entities such as products, services, problems, events, topics, and attributes [2].

Currently, commonly used sentence classification methods fall into two categories: traditional classification algorithms and popular deep learning-based classification algorithms. Traditional sentence classification algorithms [3] include Naive Bayes classifiers, Support Vector Machines (SVM), and Maximum Entropy models. The Naive Bayes classifier [4] can obtain model parameters by training on a relatively small number of training texts, offering fast training speed and guaranteed accuracy in many complex real-world scenarios. However, its feature extraction process is complicated due to the need to consider syntactic patterns such as "emoticons + punctuation secondary emotion extraction." The SVM algorithm [5] transforms sentence classification tasks into quadratic optimization problems, obtaining global optimal solutions and effectively addressing the local extremum issues inherent in neural networks. However, it primarily

selects text features based on word frequency during text feature representation, completely ignoring contextual structural information. The Maximum Entropy model [6] offers flexible feature selection without requiring additional independence assumptions or internal constraints, but it heavily depends on the corpus and suffers from long training times due to large computational requirements. These shortcomings reveal that traditional sentence classification methods exhibit deficiencies in text model representation and feature selection stages, including complex feature extraction, extracted features that easily ignore contextual structure information, and lengthy model training times.

With the advancement of deep learning in NLP [7,8], deep learning models have made significant progress in language modeling and sentence classification. In 2013, Mikolov proposed the Word2Vec model [9], which converts words in language into dense vectors understandable by computers, suitable for processing local sequence data. This effectively addresses the shortcomings of traditional text representation methods, such as extremely high vector dimensionality, excessive sparsity, and lack of correlation between words. In 2014, Kim proposed applying convolutional neural networks to sentence classification tasks [10], where CNNs eliminate the need for manual feature extraction in large datasets, simplifying the complex text feature extraction steps in traditional sentence classification algorithms. In 2017, Ma et al. proposed a multi-layer attention mechanism-based CNN and applied it to sentence modeling [11], enabling better capture of local text features and validating the effectiveness of combining attention mechanisms with CNNs. These developments demonstrate that deep learning models surpass traditional sentence classification methods in text feature extraction and representation.

This paper combines the popular deep learning-based CNN with the traditional Naive Bayes classifier, leveraging the CNN' s advantage of extracting highly discriminative text features while capitalizing on the Naive Bayes classifier' s strengths of fast training speed, high accuracy, and easily obtainable parameters in complex real-world scenarios. This approach accurately extracts text features and effectively solves sentence classification problems.

1.1.1 Convolutional and Pooling Layers

Convolutional Neural Networks (CNN) are a research hotspot in image processing, capable of directly inputting multi-dimensional raw images into the network, avoiding complex preprocessing and extracting highly discriminative image features [12]. In the NLP domain, CNNs can similarly leverage their advantages in image preprocessing to simplify text preprocessing, extract highly discriminative text features, and reduce feature engineering workload. Unlike traditional neural networks, CNNs add convolutional and pooling layers between the input layer and fully connected layers, effectively addressing issues such as numerous parameters and network depth limitations in traditional neural networks.

The convolutional layer [13] serves as a feature extraction layer. It extracts local features by performing convolution operations on the input matrix from the previous layer using different convolution kernels, forming a feature matrix of convolution kernel features. The convolution operation is calculated as shown in Equation (1):

$$c_{i,j}^{l+1} = f\left(\sum_m \sum_n c_{i+m,j+n}^{l} \cdot k_{m,n}^{l+1} + b^{l+1}\right)$$

where f represents the nonlinear activation function, c^l represents the i-th feature map of layer l, $c_{i,j}^l$ represents an element of the feature map, k^{l+1} represents the convolution kernel matrix of the i-th feature map in layer $l+1$, and b^{l+1} represents the corresponding bias term.

The pooling layer [14] performs downsampling operations on the feature vector maps from the convolutional layer. Utilizing the principle of local correlation in feature maps, it conducts aggregation statistics within adjacent small regions to further extract more important feature information. Additionally, pooling layers can generate fixed-dimensional feature vectors for input sentences of different lengths and pass these to fully connected layers for classification. Commonly used downsampling operations include Average Pooling, Max Pooling, and Stochastic Pooling.

The pooling operation calculation expression is shown in Equation (2):

$$p_i^l = \text{down}(c_i^{l-1})$$

where p_i^l represents the feature set output after pooling, h represents convolution kernels of different sizes, and m represents the number of convolution kernels in each group.

1.1.2 Attention Mechanism

The attention mechanism was first applied in image processing, enabling neural networks to focus on certain key information when processing images [15]. The attention mechanism can be understood as selectively filtering important information from a large amount of data and focusing on this important information while ignoring unimportant information. This focusing process is reflected in the calculation of weight coefficients—the larger the weight, the more focus is placed on the corresponding information value.

In NLP tasks, the attention mechanism primarily functions to extract text semantics. Assuming a single word vector in sentence S is represented as w_i, and

the attention given to that word in a specific task is α_i, then the representation of sentence S is as shown in Equation (3):

$$S = \sum_{i=1}^{n} \alpha_i w_i$$

where n represents the number of words input in sequence, and different values of α_i change the focus on sentence S. If the task is represented by semantic vector q, then α_i is expressed as a function $F(w_i, q)$ of w_i and q. Common calculation methods for function F include tensor product of vectors [16] and bilinear functions [17].

1.2 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction method based on statistical thinking [18]. PCA transforms original data into a set of linearly independent representations through linear transformation, extracting the main feature components of data to achieve dimensionality reduction of high-dimensional data. The basic principles of PCA are as follows:

Let the original dataset be represented as $X_{m\times n}$. First, zero-mean each row of matrix $X_{m\times n}$. The calculation expression is shown in Equation (4):

$$x'_{ij} = \frac{x_{ij} - \bar{x}_i}{S_i}$$

where x_{ij} represents the element in the i-th column of matrix $X_{m\times n}$, $\bar{x}_i$ represents the mean of the i-th row of matrix $X_{m\times n}$, and S_i represents the standard deviation of the i-th row of matrix $X_{m\times n}$.

Next, calculate the covariance matrix C according to Equation (5), where m represents the number of samples. Simultaneously, calculate the eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_k$ of covariance matrix C and their corresponding eigenvectors $d_1, d_2, ..., d_k$:

$$C = \frac{1}{m} \sum_{j=1}^{m} (x_j - \bar{x})(x_j - \bar{x})^T$$

Then, calculate the feature contribution rate according to Equation (6), and arrange the eigenvectors into matrix P from top to bottom in descending order of their corresponding eigenvalues. $Y = PX$ represents the data after dimensionality reduction to k dimensions:

$$e_i = \frac{\lambda_i}{\sum_{k=1}^{n} \lambda_k}$$

1.3 Naive Bayes Classifier

The Naive Bayes classifier is a classification method based on Bayes' theorem and the feature conditional independence assumption [19]. It primarily predicts the posterior probability of a sample belonging to a certain category based on prior probability distribution, selecting the category with the highest probability as the predicted category.

The classification process of the Naive Bayes classifier is as follows: For a given training dataset, first learn the joint probability distribution of input/output based on the feature conditional independence assumption. Then, based on this probability distribution, for a given input x, use Bayes' theorem to find the output y with the maximum posterior probability. The mathematical model of the Naive Bayes classifier can be expressed as follows:

Assume the input feature vector $X(x_1, x_2, \ldots, x_n)$ is the sample to be classified, and the output space is the class label set $Y = \{c_1, c_2, \ldots, c_m\}$. To classify sample X, we need to calculate the conditional probability $P(c_k|X)$ that X belongs to each class c_k. The predicted class expression for X is shown in Equation (7), where c_k is the predicted class for the sample to be classified:

$$c_k = \arg\max_{c_k} P(c_k) \prod_{i=1}^{n} P(x_i|c_k)$$

The steps to calculate the conditional probability in Equation (7) are as follows:

a) Construct a training sample set with known class labels;

b) Count the conditional probability of each feature in each class in the training set, as shown in Equation (8);

c) Assuming that each feature attribute is independent, the conditional probability expression according to Bayes' theorem is shown in Equation (8):

$$P(c_k|X) = \frac{P(X|c_k)P(c_k)}{P(X)} = \frac{P(c_k)\prod_{i=1}^{n} P(x_i|c_k)}{P(X)}$$

d) In Equation (8), the denominator $P(X)$ is the same for all classes, so only the numerator needs to be maximized. The simplified conditional probability expression is:

$$c_k = \arg\max_{c_k} P(c_k) \prod_{i=1}^{n} P(x_i|c_k)$$

2 Model Construction

This paper constructs a part-of-speech attention feature matrix for sentiment words in word vectors. Based on the fundamental structure of CNNs, we propose a dual-channel CNN incorporating an attention mechanism, which is combined with PCA and a Naive Bayes classifier to achieve sentence classification. This section focuses on the overall framework of the model, the specific implementation details of the attention mechanism-based CNN, and the classification process of the Naive Bayes classifier for text features.

2.1 Model Framework

The main idea of the sentence classification model based on CNN and Bayesian classifier is to extract text features through CNN, reduce the dimensionality of the extracted text features using PCA, and finally perform sentence classification using a Naive Bayes classifier. The model framework is shown in Figure 1 [Figure 1: see original paper].

As shown in Figure 1, the algorithm consists of two main phases: the training phase and the testing phase. In the training phase, the CNN model is trained using the training set text collection represented by Word2Vec. The trained network model is then applied to extract features from all texts. After performing PCA dimensionality reduction on the extracted high-dimensional text features, the reduced text feature collection is used to train the Naive Bayes classifier. In the testing phase, the test set text collection is preprocessed, features are extracted through CNN, dimensionality is reduced via PCA, and finally the reduced text features are fed into the Naive Bayes classifier to obtain sentence classification results.

2.2 Dual-Channel Convolutional Neural Network Based on Attention Mechanism

In the proposed sentence classification model based on CNN and Bayesian classifier, the CNN model is a dual-channel CNN based on the attention mechanism, as shown in Figure 2. This model has five layers. In the text preprocessing stage, sentences in the dataset are converted into word vector matrices using the Word2Vec tool, and these word vector matrices are combined with sentiment word vectors to form part-of-speech attention matrices. These two matrices constitute the dual channels of the CNN.

In the second layer of the CNN (convolutional layer), convolution windows slide across the sentence length, with each window generating one value to produce

a vector of length $sentence\_length + window\_size - 1$, thereby obtaining local features of the input text. In the third layer (pooling layer), average pooling is performed in a sliding window manner on the text feature matrix from the convolutional layer while maintaining the relative positions of features. In the fourth layer (convolutional layer), convolution windows similarly slide across the sentence length on the pooling layer feature matrix from the previous layer, enabling deeper extraction of text features. In the fifth layer (pooling layer), average pooling is performed on the convolutional layer feature matrix from the previous layer along the sentence length dimension to obtain a column vector representing the input sentence features as the output layer of the CNN. Finally, the output layer of the CNN is processed by PCA dimensionality reduction, and the reduced sentence features are used to train the classifier.

The attention mechanism weights inputs or outputs at various parts of the neural network, using word vectors themselves or other external vectors to emphasize relatively important information in the feature matrix. This paper adopts a part-of-speech attention mechanism that combines word vectors to form a part-of-speech attention feature matrix, which together with the word vector input matrix forms dual channels as the input layer of the CNN.

Relying solely on text structure information for sentence classification results in low accuracy. To address this issue, this paper extracts sentiment words from the original sentence by tokenization and combining them with sentiment words from a sentiment lexicon to form a part-of-speech attention feature matrix. For a sentence of length n, we extract the word vectors $a_i \in \mathbb{R}^l$ of sentiment words in the sentence $s = \{w_1, w_2, \ldots, w_n\}$, where l represents the dimension of sentiment word vectors, which is the same as the Word2Vec dimension. By performing an inner product operation between the sentiment word vectors and the word vector matrix of sentence s, we obtain a diagonal matrix A, with the calculation process shown in Equations (10) and (11):

$$A_i = \text{tensor\_product}(a_i, s)$$

$$A_i = \exp\left(\frac{A_i}{\sum_{j=1}^{n} \exp(A_j)}\right)$$

Then, by performing a dot product operation between the diagonal matrix A and the word vector matrix of sentence s, we obtain the part-of-speech attention matrix z_i, with the calculation process shown in Equation (12):

$$z_i = \text{dot\_product}(c_i, A_i)$$

Convolution operations are crucial for CNNs to obtain input features, which are stored in the network structure in the form of feature matrices. Each unit

of the feature matrix is associated with local features from the previous layer. Convolution kernels perform convolution operations on local features, and the values of the feature matrix are obtained through activation functions. This paper adopts generalized convolution operations, also known as wide convolution, which allows convolution kernels to scan the entire input sentence vector without restricting the relationship between input layer sentence length s and convolution kernel size m, and ensures that the convolution operation output is not an empty vector. The calculation expression is shown in Equation (13), where the convolution kernel output is $c' \in \mathbb{R}^{s+m-1}$, c is the word vector matrix of the input sentence, and the convolution kernel $k \in \mathbb{R}^m$. This generalized convolution operation expands the coverage of convolution kernels, removes restrictions on the relationship between input layer sentence length s and convolution kernel size m, considers all possible features, maximally ensures the diversity of text feature extraction, and effectively captures complete sentence information when sentence lengths are inconsistent.

The pooling layer is placed below the convolutional layer, sampling or aggregating local data from the convolutional layer to reduce the dimensionality of the convolutional layer feature matrix and prevent overfitting. This paper adopts Average Pooling, with the pooling layer feature matrix calculation expression shown in Equation (14):

$$x_i^l = f\left(\text{down}(x_i^{l-1}) \cdot \beta^l + b^l\right)$$

where down$(\cdot)$ represents the pooling function, β^l represents the multiplicative bias, and b^l represents the additive bias.

2.3 Text Feature Classifier

After text data passes through the CNN and undergoes PCA dimensionality reduction, the main feature attributes of sentences are obtained. This paper uses a Naive Bayes classifier to classify sentence features. For a given sentence feature vector, Bayes' conditional probability formula is first used, as shown in Equation (15), where $P(x_j|c)$ is the probability of feature attribute x_j given sentence category c. Then, Bayes' formula is used to calculate the posterior probability that the known sentence attribute belongs to different sentence categories, as shown in Equation (16). Finally, according to the maximum a posteriori probability, the sentence is classified into the category with the highest posterior probability, as shown in Equation (17):

$$P(x_j|c) = \frac{P(x_j, c)}{P(c)}$$

$$P(c|x_1, x_2, \ldots, x_n) = \frac{P(c)\prod_{j=1}^{n} P(x_j|c)}{P(x_1, x_2, \ldots, x_n)}$$

$$\hat{c} = \arg\max_{c} P(c) \prod_{j=1}^{n} P(x_j|c)$$

The text features extracted by the CNN are 240-dimensional, which is excessively high. This paper uses PCA to reduce the dimensionality of these features. To study the impact of the cumulative contribution rate of principal component eigenvalues on classification performance, we analyzed the classification accuracy of the proposed model under different cumulative contribution rates. The relationship between classification accuracy and eigenvalue cumulative contribution rate is shown in Figure 3 [Figure 3: see original paper].

As shown in Figure 3, let the eigenvalue cumulative contribution rate be α. When α decreases from 100% to 95%, redundant information in the text features extracted by the CNN is gradually eliminated through PCA dimensionality reduction, and classification accuracy gradually increases. When $\alpha = 95\%$, redundant information is sufficiently removed, achieving the highest classification accuracy. When α continues to decrease from 95%, some useful text features are eliminated, causing classification accuracy to decline accordingly. This demonstrates that the dimensionality of principal components critically affects classification accuracy. In our experiments, selecting $\alpha = 95\%$ reduces the 240-dimensional features extracted by the CNN to 80 dimensions through PCA, achieving significant dimensionality reduction.

3.1 Experimental Data

To verify the effectiveness of the proposed model, experiments were conducted using the Cornell Movie Review Data (MRD) dataset (https://www.cs.cornell.edu/people/pabo/movie-review-data/) and the Stanford Sentiment Treebank (SST) dataset (https://nlp.stanford.edu/sentiment/). MRD is used for binary classification (negative, positive) of sentences, while SST is used for five-category classification (very negative, negative, neutral, positive, very positive). MRD consists of movie review data, containing 1,000 positive reviews and 1,000 negative reviews, with 5,331 sentences labeled for sentiment polarity and 5,000 sentences labeled for subjective/objective tags. In our experiments, 1,400 reviews were randomly selected as the training set, 400 as the test set, and 200 as the validation set. The SST dataset is an extension of MRD, containing 11,855 sentences with manually labeled categories: 8,544 for training, 2,210 for testing, and 1,101 for validation. The statistics of the experimental data are shown in Table 1 , with a training:test:validation ratio of 7:2:1.

Table 1 Statistic of the datasets

3.2 Hyperparameter Settings

The word vector dimension is $d = 300$. We used a pre-trained 300-dimensional Google Word2Vec file for mapping, with words not appearing in Word2Vec mapped to 300-dimensional random vectors in $[-1, 1]$ using a random function. The experiments used dual channels for convolution operations on the input matrix, with ReLU (Rectified Linear Units) as the convolution kernel function. Training was performed using SGD (Stochastic Gradient Descent) with the Adadelta optimizer proposed by Zeiler [20]. Other experimental parameters are shown in Table 2 . Initial parameters were set empirically, then the model was trained for 100 iterations on the validation set while observing changes in cross-entropy loss for parameter tuning. Finally, the parameter set that performed best on the validation set was selected as the output for training the model.

Table 2 Hyper parameters of experiment

3.3 Experimental Results

To verify the effectiveness of the proposed model, four baseline methods were established: Naive Bayes Classifier (NBC), Support Vector Machine (SVM), Convolutional Neural Network (CNN), and CNN+SVM. Comparative experiments were conducted on the test sets of MRD and SST datasets. The experimental results of the proposed model and other models on the test sets are shown in Table 3 .

Table 3 Model classification correct rate comparison result

Experimental results show that on the MRD dataset, the accuracies of deep learning-based models CNN and CNN+SVM are 81.1% and 82.3%, respectively, which are 3.7% and 4.9% higher than the traditional sentence classification model SVM, and 4.8% and 6.0% higher than NBC. On the SST dataset, the accuracies of CNN and CNN+SVM are 47.4% and 48.3%, respectively, which are 3.5% and 4.4% higher than SVM, and 4.6% and 5.5% higher than NBC. Traditional sentence classification models NBC and SVM focus on syntactic patterns such as "emoticons + punctuation secondary emotion manual annotation" and word frequency features during data preprocessing, but ignore contextual text structure information, complicating the text sentiment feature representation process and resulting in low text sentiment classification accuracy. Deep learning-based models CNN and CNN+SVM use word vectors to simplify text sentiment feature representation and employ CNNs to capture high-quality text features containing contextual structure information, making their sentence classification accuracy significantly higher than traditional models.

The proposed model achieves an accuracy of 83.7% on the MRD dataset, which is 2.6% and 1.4% higher than deep learning-based models CNN and CNN+SVM, respectively. On the SST dataset, its accuracy is 49.8%, which is 2.6% and 1.4%

higher than CNN and CNN+SVM, respectively. Compared with traditional CNN structures, the proposed CNN architecture incorporates a part-of-speech attention mechanism in the input layer, enabling the network to focus more on sentiment polarity target words during text sentiment feature learning. The text sentiment features extracted by the CNN are processed by PCA to eliminate redundant information. Leveraging the fast training speed and accuracy robustness of the Naive Bayes classifier in multi-dimensional feature spaces, the classifier categorizes the dimensionality-reduced text sentiment features, further improving sentence classification accuracy compared with deep learning-based CNN and CNN+SVM models.

Figure 4 [Figure 4: see original paper] more intuitively shows the sentence classification accuracy of different models on different datasets. It is evident that when the number of category labels for sentence classification increases, the accuracy of both traditional and deep learning models decreases, indicating that existing sentence classification models still cannot accurately extract highly discriminative text features for multi-classification tasks.

The proposed model improves sentence classification accuracy and addresses the interpretability defects of deep learning-based CNNs in black-box testing, enhancing the explainability of deep learning models during classification.

Experimental results show that the proposed sentence classification model based on CNN and Bayesian classifier is not ideal for multi-classification of text sentiment orientation. Therefore, future work will focus on improving the CNN structure to address this issue.

---

4 Conclusion

In text sentiment orientation analysis tasks, previous research primarily focused on improving text sentiment feature representation methods and word frequency statistics in traditional sentence classification algorithms. These methods suffer from complex text sentiment feature representation and extracted features that ignore contextual structure information, increasing feature engineering workload and resulting in low sentence sentiment classification accuracy. This paper combines deep learning-based CNNs with traditional sentence classification algorithms, proposing a sentence classification model that integrates an improved CNN with a Bayesian classifier. The model significantly reduces text sentiment feature representation workload by introducing word vectors. Additionally, incorporating a part-of-speech attention mechanism in the CNN input layer helps improve the quality of extracted text sentiment features and increase sentence sentiment classification accuracy. Furthermore, by replacing the traditional CNN output layer with a Bayesian classifier and leveraging the classifier' s fast training speed, accuracy robustness in multi-dimensional feature spaces, and the CNN' s ability to extract targeted contextual structure information, comparative experiments verify that combining the improved CNN with a Bayesian

classifier effectively improves sentence classification accuracy.

References

[1] Yu Jianfei, Jiang Jing. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification [C]//Proc of International Conference on Empirical Methods in Natural Language Processing. 2016: 236-246.

[2] Zhang Lei, Wang Shuai, Liu Bing. Deep learning for sentiment analysis: a survey [J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018, 8(4): 1253-1256.

[3] Wang Sida, Manning C D. Baselines and bigrams: simple, good sentiment and topic classification [C]//Proc of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Stroudsburg, PA: Association for Computational Linguistics, 2012: 90-94.

[4] Jiang Qiaowei, Wang Wen, Han Xu, et al. Deep feature weighting in naive Bayes for Chinese text classification [C]//Proc of IEEE International Conference on Cloud Computing and Intelligence Systems. 2016: 160-164.

[5] Joseph L, Zhu Yun, Zhang Yanqing. Support vector machines and Word2vec for text classification with semantic features [C]//Proc of IEEE International Conference on Cognitive Informatics & Cognitive Computing. 2015: 136-140.

[6] Hosseini M, Kerachian R. A Bayesian maximum entropy-based methodology for optimal spatiotemporal design of groundwater monitoring networks [J]. Environmental Monitoring and Assessment, 2017, 189(9): 433.

[7] Kumar A, Irsoy O, Ondruska P, et al. Ask me anything: dynamic memory networks for natural language processing [C]//Proc of the 33rd International Conference on Machine Learning. 2016: 1378-1387.

[8] Gatt A, Krahmer E. Survey of the state of the art in natural language generation: core tasks, applications and evaluation [J]. Journal of Ophthalmology, 2018, 45(45): 1-16.

[9] Mikolov T, Chen Kai, Corrado G, et al. Efficient estimation of word representations in vector space [EB/OL]. [2014-02-10]. http://arxiv.org/pdf/1301.3781.pdf.

[10] Kim Y. Convolutional neural networks for sentence classification [C]//Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA: ACL, 2014: 1746-1751.

[11] Ma Dehong, Li Sujian, Zhang Xiaodong, et al. Interactive attention networks for aspect-level sentiment classification [C]//Proc of the 26th International Joint Conference on Artificial Intelligence. 2017: 4068-4074.

[12] LeCun Y, Bengio Y, Hinton G. Deep learning [J]. Nature, 2015, 521(7553): 436.

[13] Jégou S, Drozdzal M, Vazquez D, et al. The one hundred layers tiramisu: fully convolutional densenets for semantic segmentation [C]//Proc of IEEE Computer Vision and Pattern Recognition Workshops. 2017: 1175-1183.

[14] Fernando B, Gavves E, Oramas M J, et al. Rank pooling for action recognition [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(4): 773-787.

[15] Firat O, Cho Kyunghyun, Bengio Y. Multi-way, multilingual neural machine translation with a shared attention mechanism [C]//Proc of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016: 866-875.

[16] Kadlec R, Schmid M, Bajgar O, et al. Text understanding with the attention sum reader network [C]//Proc of the Meeting of the Association for Computational Linguistics. 2016: 908-918.

[17] Pre-Algebra Group, Department of Mathematics, Peking University. Higher algebra [M]. 4th ed. Beijing: Higher Education Press, 2013.

[18] Li Jianlin. A PCA-based combined feature extraction text classification method [J]. Application Research of Computers, 2013, 30(8): 2398-2401.

[19] Duan Liguo, Di Peng, Li Aiping. A new naive Bayes text classification algorithm [C]//Proc of IEEE International Conference on Data Mining. 2014: 51-58.

[20] Zeiler M D. ADADELTA: an adaptive learning rate method [EB/OL]. [2012-12-22]. http://arxiv.org/pdf/1212.5701.pdf.

Note: Figure translations are in progress. See original paper for figures.