Chapter 13
Short Text Classification of Invoices Based on BERT-TextCNN

Jiuwei Zhang, Li Li, and Bo Yu

Abstract Traditional invoice text classification methods are labor-intensive and inefficient. In order to effectively identify the types of invoices, a Chinese text classification model based on deep learning BERT-TextCNN is designed, and a short text classification dataset of invoices is obtained from a municipal tax bureau to train and test the model, and to compare and analyze the performance of BERT-TextCNN model, BERT model, and TextCNN model. As a result, compared to traditional neural network models, the BERT + TextCNN model can accurately classify Chinese text, effectively prevent excessive fitting, and have good generalization ability. The performance of text classification is improved compared to both BERT model and TextCNN model alone. Draw a conclusion through experiments which show that the BERT-TextCNN model has good classification effect and good stability.

13.1 Introduction

The rapid development of computers, especially driven by online social networking, text data has gradually become a mainstream form of text. Due to the large amount of data and complex text semantics, text classification has become a challenge. Facing such a large and complex text data, it is especially important to classify them accurately and effectively. The length of text can be long or short, so we can classify these text data into short text data and long text data. Short text has the characteristics of short text content, easy to read, and easy to disseminate, and it exists widely in the Internet as a carrier of information dissemination and interaction, such as news headlines, social media information, invoice names, and other texts. Therefore, how to enable computers to classify large amounts of text data is becoming a topic of

J. Zhang (✉) · L. Li
College of Computer Science and Technology, Shenyang University of Chemical Technology, Shenyang, China
e-mail: z1264641108@163.com

B. Yu
Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang, China

R. Kountchev et al. (eds.), *Proceedings of International Conference on Artificial Intelligence and Communication Technologies (ICAICT 2023)*, Smart Innovation, Systems and Technologies 368, https://doi.org/10.1007/978-981-99-6641-7_13

interest to researchers. In general, text classification tasks have only few classes. When the classification task has a large number of classes, traditional Recurrent Neural Network (RNN) [1] (e.g., LSTM and GRU) algorithms perform poorly in terms of accuracy.

Therefore, in this article, we designed a BERT-based model and integrated its output into CNN to deal with classification matters. This method adopts the BERT Chinese model released by Google, which is pre-processed and then gets the word vector characteristics [2]. The feature of the word vector in the obtained sentence is the convolution kernel size from CNN. We combine the above two and use softmax to get the results. The reliability of this model in category task is demonstrated by comparing it with various text classification models.

13.2 Related Research

In the past, text categories were distinguished by plain Bayesian, KNN, decision tree, etc. With the rapid progress of deep learning, natural language processing technology has made rapid development. Deep neural networks are becoming a common method for text classification due to their powerful expressive power. Despite their attractiveness, neural text recognition models lack training data in many applications. In recent years, several Chinese classification methods emerge in an endless stream. Convolutional neural networks and recursive neural networks are applied to image processing and speech processing, and have made achievements. Later, they are applied to text processing technology. The first is to find a way to express words that can be recognized by computers that the computer will understand, making it possible for the computer to perform subsequent computation and analysis. The above is referred to as text representation. Word embedding is a kind of text representation that is often used. Words are put into the space, and these are expressed as vectors. One-hot, bag-of-words model, TF-IDF, etc., are the common text representation ways. But, the above method will result in problems, e.g., higher dimensionality and sparsity. They cannot explain two words with similar meanings very well. This is the reason why Word2vec 1 model emerged later. Word2vec has different models. The sequential grouping model is used to analyze the value at this time through the previous and subsequent articles. The sequential skip-gram model (Skip-gram) uses the value at this time to judge the meaning of the previous and subsequent articles. This way is associated with the previous and subsequent articles. The problem of too much computation and waste of resources is solved. Word2vec is not good at handling polysemy words. Word2vec is a static method, so it cannot be adjusted dynamically to enhance the specified things. Bidirectional Encoder Representations from Transformers (BERT) [3] is a pre-training model, which solves the problem of multiple meanings well by considering contextual information. BERT pays more attention to the early training of words, so it only needs to adjust the model according to different scenarios [4].

Liu et al. [5] proposed a multi-layer model construction, which can obtain the contents of previous and subsequent articles from the articles in a sequential manner. And LSTM is used to extract the contextual and sequential features of documents. The architecture of multi-layer models is more complex. It includes recursive neural networks such as LSTM, which require more computing resources and training time. The advantage of FastText text classification model is fast and efficient, but its direct use for distinguishing small text categories is of low accuracy rate. Feng Yong et al. proposed a method that fuses Term Frequency-Inverse Document Frequency (TF-IDF) and Implicit Dirichlet Distribution (IDD). Latent Dirichlet Allocation (LDA) for distinguish different texts [6]. The method performs TF-IDF filtering on the lexicon processed by the n-grammar model in the input stage of the FastText text classification model, performs corpus topic analysis using the LDA model, and complements the feature lexicon based on the obtained results, thus biasing the input word sequence vector mean in favor of highly discriminative entries and making it is more suitable for the environment where short and small texts are distinguished. Comparing the experimental results, it can be seen that the way has a higher accuracy rate in the classification of Chinese short texts. The application of TF-IDF and LDA is based on specific tasks and corpora, and may require adjustments and optimizations to each task and dataset. The generalization ability of this method may be relatively low, making it difficult to adapt to the needs of different tasks and domains.

Qiaohong Chen et al. proposed a novel text representation method to extract high-quality features from the entire training set by applying Gini impurity, information gain, and chi-square test from phrase features [7]. The phrase features extracted from each document must be linearly represented by these high-quality features, and then, after Word2vec word vector representation, advanced features are drawn out using convolutional neural network convolutional and pooling layers, and finally classified using Softmax classifier. This method depends on the feature selection of the whole training set in the feature extraction stage. This may lead to inaccurate feature selection in situations where the dataset is insufficient or unevenly distributed, affecting the final classification performance.

The attention mechanism is added to the text data coding, and the hierarchical structure of text classification is figured out. Attention mechanism is added to sentences and words, which is superior to long-term short-term memory (LSTM), CNN, and other models. Later, the transformer [8] model appeared, which abandoned the previous CNN and RNN, and the attention mechanism formed the entire network. It is a process of encoding and decoding, so this paper uses BERT. The BERT is used as an embedding [9] layer to access to other mainstream models and is trained and validated on the same invoice dataset. And it becomes one of the current mainstream models with good performance. Based on this, a network structure based on BERT-TextCNN is proposed in this paper, and a comparison experiment with BERT model and TextCNN model in the invoice text dataset is conducted.

13.3 Research Methodology

In Chinese text analysis, the model we propose is the BERT-TextCNN model. The model structure is shown in Fig. 13.1. In this pattern, words and phrases are embedded into a model, taking into account the pre prepared BERT model. Extract attachment vector from BERT and use it as input of CNN.

13.3.1 BERT

Contrasted with the ELMO model released by Google in 2018, BERT [3] changes the language pattern between BiLSTM and transformer [10], which truly implements the concept of bidirectional coding. The BERT model consists of a stack of encoders of transformer. This is a two-way encoding pattern. The input is the sum total token flushbonading, segment flushbonading, and location embedding. In addition, the output is a T-vector with feature message. The model composition of BERT is shown in Fig. 13.2.

In Fig. 13.2, E represents the embedding layer, which is the input layer of the BERT model. At this layer, each word or marker in the text sequence is converted into a

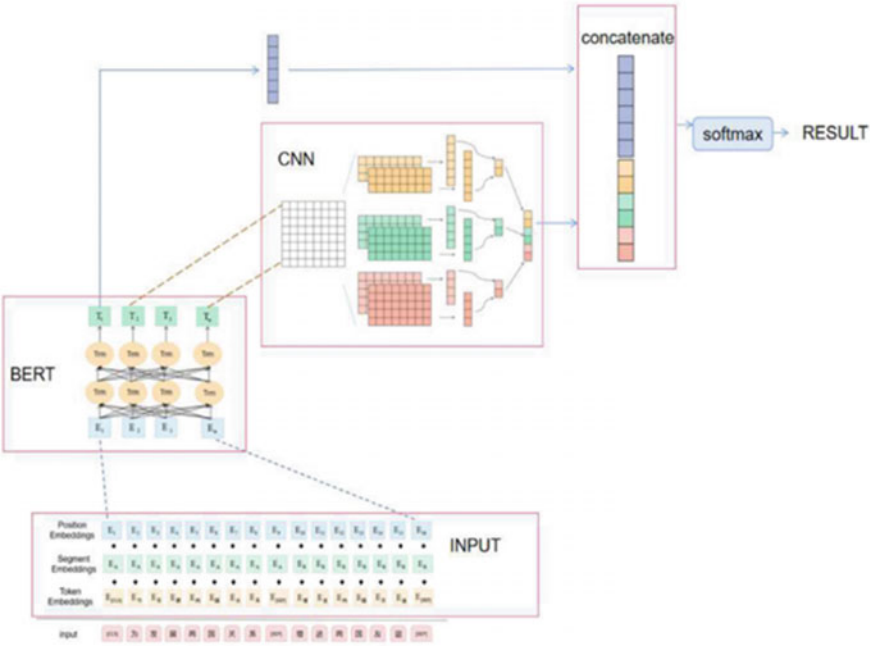

Fig. 13.1 BERT-TextCNN structure

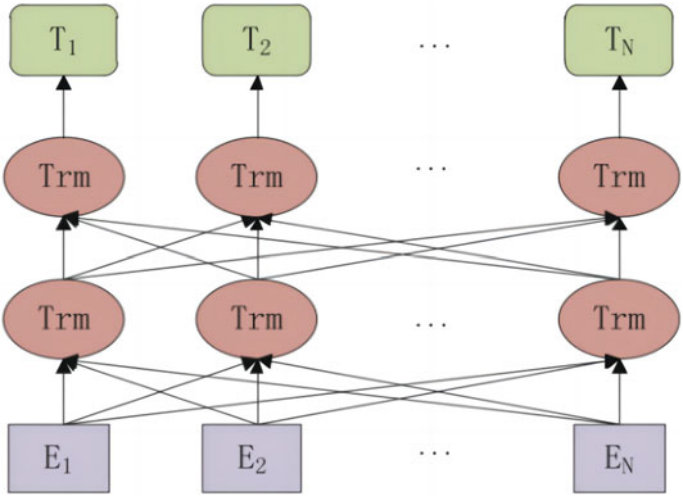

Fig. 13.2 BERT composition

corresponding vector representation, called an embedding vector. These embedding vectors will capture the semantic relationships and contextual information between words as input for subsequent processing. Trm represents a converter, which is a core component of the BERT model. BERT uses a multi-layer converter structure. Through the self-attention mechanism and the feedforward neural network layer, the input embedded vectors are encoded and feature extracted for multiple rounds. The transformer can capture contextual dependencies in text sequences and learn rich semantic feature representations.

The main purpose of the model is to generate a language model, so only multi-layer encoder construction is used. The encoder is mainly composed of feedback network layer and self-attention layer. If we want the computer to focus on some information, the attention mechanism can be implemented. Self-attention mechanism is to add the preceding and following words to the current word. This article is to add the features in the front and back of words to the features of words with different permissions, so that the computer can determine whether words in the sentence are more compactly connected with other words in the sentence.

In Fig. 13.3, multi-head attention is a self-attention mechanism used to capture the correlation between different positions in the input sequence. It maps the input sequence into multiple queries, keys, and values, and then aggregates the values by calculating attention weights. Multi-head attention allows the model to focus on different representation subspaces in the input sequence, thereby improving the model's expressive ability. Dropout is a regularization technique used to reduce model overfitting. During training, dropout randomly discards a portion of the output of neurons, making the model independent of specific features of individual neurons. This helps to improve the generalization ability and robustness of the model. Add represents adding the input to the output of the sub-layer in the residual connection.

In the encoder structure, after the self-attention and feedforward neural network sublayer, the residual connection will add the output of the sublayer to the input. This facilitates the flow of information and facilitates gradient propagation, promoting model training and convergence. Layer normalization is a normalization technique used to adjust the mean and variance of inputs at a hierarchical level. In encoder structures, layer normalization is usually followed closely by addition operations. It helps to alleviate the internal covariate offset problem and improve the stability and rate of convergence of the model. Feedforward is a sub-layer of the encoder structure, which processes the input by applying two linear transformations and nonlinear activation function. The feedforward neural network operates on the representation after position coding to extract higher level feature representation. It usually includes a hidden layer and an activation function, such as ReLU.

As we all know, attention mechanism can make the computer pay attention to the information that we want it to pay attention to. The self-attention mechanism is to integrate the context into the encoding of the current vocabulary. In this paper, the features of a word in a context are added to the features of the word with different weights, so that the computer can judge which words in a sentence are more closely related to the other words in the sentence.

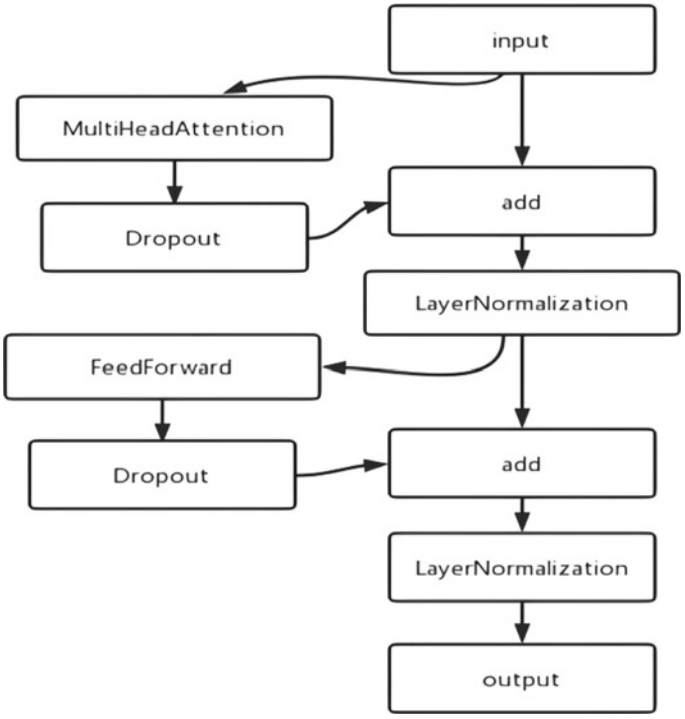

Fig. 13.3 Encoder structure diagram

The calculation of self-attention can be summarized as follows: first, prepare the input vector, and create a query vector, key vector, and value vector for each word. These vectors are obtained by multiplying the word embedding and three transformation matrices (W_Q, W_K, W_V), which are learned in training. Note that the dimensions of these new vectors are smaller than those of the input word vectors ($512 \rightarrow 64$), which is not necessary. This structure is intended to make the computation of the multi-headed attention more stable. Then, calculate the score, and calculate the self-attention of "Thinking" in "Thinking Matches". We need to calculate the score of "Thinking" for each word in the sentence, which determines the degree of attention paid to other parts of the sentence when encoding "Thinking". This score is obtained by calculating the dot product of the query vector of "Thing" and the key vector of other words. Second, divide the score by 8, so that the gradient will be more stable. Then, normalize the score by softmax to make the sum equal to 1. The softmax score determines how much each word pays attention to this position. Multiply the softmax score by the vector corresponding to value (to prepare for subsequent sum ups). The purpose of this is to retain the value of the concerned words and weaken the value of the irrelevant words (e.g., multiply by a small value of 0.001). Accumulate all weighted value vectors to produce the output result of self-attention at the location. Calculate the matrix of query, key and value, combine all input word vectors into matrix X, and multiply them by the trained weight matrix (WQ, WK, WV). The matrix is calculated as follows:

$$x \times w^q = q \tag{13.1}$$

$$x \times w^k = k \tag{13.2}$$

$$x \times w^v = v \tag{13.3}$$

$$Z = \text{soft}\max\left(\frac{q \times k^t}{\sqrt{d_k}}\right) \times v \tag{13.4}$$

In (13.4), the calculation result of matrix Q, K inner product shows the matching degree of the two vectors. After softmax function, the influence degree (weighted result) of the current word to the coding position can be obtained. Dividing by the root sign d_k is to prevent the score from expanding with the increase of dimensions. Without this step, softmax will get a smooth and indistinguishable result. Then, multiply the value matrix to get the self attention score of the current word. Finally, calculate all the words according to the above steps.

A set of Q, K, and V matrices can get a current word's eigenvalue through calculation. The multi-attention mechanism is like a filter in convolutional neural network, which can help us to extract a variety of features. As shown in the figure below, multiple feature expressions are obtained through different heads, all features are

spliced together, and finally, dimension reduction is carried out through the full connection layer. This algorithm uses 8 heads for feature stitching.

13.3.2 TextCNN

Convolutional neural network CNN [11] is used for graphics processing. As its variant model, text convolutional neural network (TextCNN) extracts local features of different sizes in text sequences by setting filters of different sizes. The convolution layer is more important in TextCNN model. It requires less parameters than other deep learning models. Different features of input information can be extracted by convolution. The convolution layer is composed of several convolution kernel modules. The fully connected layer is shown in Fig. 13.4.

In the traditional neural network, each neuron is connected to each neuron in the next layer, which is called full connection. In CNN, the input layer is convoluted to get the output, which is not all connected but becomes local connection, that is, the local area of the input is connected to a neuron, and each layer uses different convolution kernels, and then combines them. The pooling layer is an important structure in convolutional neural networks. It is applied after the convolution layer. The pooling layer downsamples its input. The most common method is to retain the maximum information, which is generally the maximum pooling through windowing.

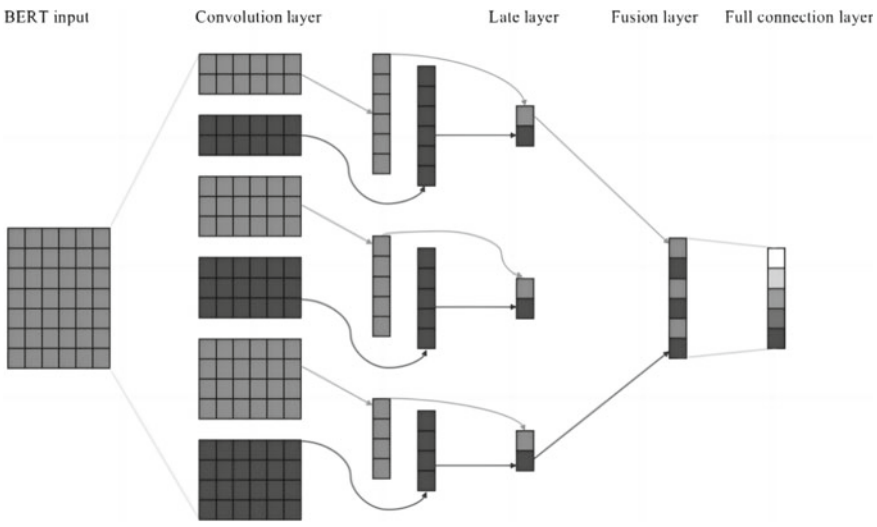

Fig. 13.4 TextCNN model

13.4 Experiment

13.4.1 Experimental Environment and Data Set

The experiment in this paper is implemented under the deep learning TensorFlow. The Python version is 3.7. The operating system is Windows 10 (64 bit). As for experimental hardware, the CPU is i3-9100f.

In supervised learning, the performance of the model is largely dependent on the dataset. The learning of neural networks also depends on datasets, and if the number of datasets is small, the learning will be insufficient. In order to provide suitable datasets for model training and model result evaluation, this paper selects real data from tax offices. Among them, 200,000 invoices are selected, and there are ten categories: tea, pet supplies, textile supplies, clothing, handicrafts, goods, furniture, wine, toys, and jewelry. Each category has 20,000 items with an average text character length of 15–30. 180,000 of them are used as the training set, 10,000 are used as the validation set, and the remaining 10,000 are used as the test set.

13.4.2 Experimental Setup

In training TextCNN, BERT, and BERT-TextCNN, we use cross-entropy as the loss function. TextCNN uses ADAM as the optimizer with a learning rate of 0.001. Meanwhile, in the model, BERT acts as the encoder of comment text and uses the embedding function of BERT language model to encode each comment into a sentence formed by stacking word vectors. As a new feature, it is used as the input of the CNN layer. In order to prevent overfitting, a dropout layer with a discarding rate of 0.5 is added in front of the full connection layer. The hyperparameter settings in this paper are given in Table 13.1.

Table 13.1 Hyperparameter

Parameter	Value
Embedding	64
Learning rate	1e−3
Train epoch	100
Dropout	0.3
Batch size	128
Epoch	20

13.4.3 Evaluation Indicators

The commonly used evaluation indicators for classification tasks include precision, recall, and F1 score. The calculation formula is as follows.

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{TF}} \tag{13.5}$$

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{13.6}$$

$$\mathrm{F1} = \frac{2 \cdot P \cdot R}{P + R} \tag{13.7}$$

13.4.4 Analysis of Experimental Results

In the experiment, we compared the effects of different convolution kernel sizes on the model.

As given in Table 13.2, the best effect is obtained when using convolution kernels of (3, 4, 5) sizes. Therefore, we chose convolution kernel sizes of 3, 4, and 5. Model comparison was performed with the same dataset.

In this paper, we perform comparison experiments on invoice text dataset classification using different models of BERT-TextCNN, BERT, TextCNN, and CNN + Attention. The experiments measure the average accuracy (P), average recall (R), and average F1 value for ten labels. BERT model [14]: word vectors are trained by BERT model, and CLS flag bit feature vectors are used directly for downstream classification task. TextCNN [12] is implemented with Word2vec. CNN + Attention [13] obtains important local information from CNN and then calculates the score through attention.

As given in Table 13.3, the accuracy of BERT-TextCNN is 3.16% and 6.15% higher than that of BERT and TextCNN, respectively. It shows that this model is better than other models in invoice classification. BERT-TextCNN model has a high accuracy rate in invoice classification, which shows that the model has a good effect in invoice text classification. Its fine-tuning based on pre-training can effectively

Table 13.2 Comparison of convolutional kernel size

Size	Acc	Pre	Rec	F1
(2, 3, 4)	93.28	93.36	93.28	93.25
(3, 4, 5)	93.97	94.04	93.97	93.93
(4, 5, 6)	93.32	93.46	93.32	93.29

Table 13.3 Model performance comparison

Model	Acc	Pre	Recall	F1
TextCNN	87.57	85.52	83.02	82.45
CNN + Attention	89.96	90.73	90.17	90.21
BERT	90.56	92.89	92.50	90.78
BERT + TextCNN	93.72	94.83	92.96	92.73

Table 13.4 Comparison between test set and validation set

Data	Acc	Pre	Rcc	F1
Validation	95.96	96.03	95.96	95.92
Test set	96.34	96.40	96.44	96.40

solve the problem of polysemy of traditional word vectors, which is the key to obtain high accuracy of the model (Table 13.4).

There is hardly any difference between the test set and the verification set, so the model has good generalization.

13.5 Conclusion

In this paper, an improved BERT-TextCNN classification model based on deep learning algorithm is proposed for invoice short text data. The model uses BERT pre-training to generate word vectors and embeds words into convolutional neural networks. The test results show that the model performs well in all aspects, with high efficiency and accuracy. However, the data used in this paper is not enough, and a large number of data are needed to better train, so it may perform better with the increase of samples.

Although BERT-TextCNN has a significant improvement over TextCNN and BERT in classification, there are still some problems that need to be improved. The number of model parameters is large, and it takes a lot of time for training and loading, so it is an important research work to study the compression of BERT model and reduce the complexity of the model without suffering a large loss of model accuracy.

References

1. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent Neural Network Regularization. arXiv preprint arXiv:1409.2329 (2014)

2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 (2018)
3. Chen, Q., Zhuo, Z., Wang, W.: Bert for Joint Intent Classification and Slot Filling. arXiv preprint arXiv:1902.10909 (2019)
4. Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune bert for text classification? In: Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20 (2019)
5. Liu, J., Xia, C., Yan, H., Xie, Z., Sun, J.: Hierarchical comprehensive context modeling for Chinese text classification. IEEE Access **7**, 154546–154559 (2019)
6. Chawla, S., Kaur, R., Aggarwal, P.: Text classification framework for short text based on TFIDF-FastText. In: Multimedia Tools and Applications, pp. 1–14 (2023)
7. Wang, J., Wang, Z., Zhang, D., Yan, J.: Combining knowledge with deep convolutional neural networks for short text classification. IJCAI **350**, 3172077–3172295 (2017)
8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
9. Goldberg, Y., Levy, O.: word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method. arXiv preprint arXiv:1402.3722 (2014)
10. Sarzynska-Wawer, J., Wawer, A., Pawlak, A., Szymanowska, J., Stefaniak, I., Jarkiewicz, M., Okruszek, L.: Detecting formal thought disorder by deep contextualized word representations. Psychiatry Res. **304**, 114135 (2021)
11. Kim, Y.: Convolutional Neural Networks for Sentence Classification. arXiv preprint arXiv (2014)
12. Song, P., Geng, C., Li, Z.: Research on text classification based on convolutional neural network. In: 2019 International Conference on Computer Network, Electronic and Automation (ICCNEA), pp. 229–232. IEEE (2019)
13. Chen, Z., Tang, Y., Zhang, Z., Zhang, C., Wang, L.: Sentiment-aware short text classification based on convolutional neural network and attention. In: 2019 IEEE 31st International Conference on Tools with Artificial Intelligence, pp. 1172–1179 (2019)
14. Jing, W., Bailong, Y.: News text classification and recommendation technology based on wide & deep-bert model. In: 2021 IEEE International Conference on Information Communication and Software Engineering, pp.209–216 (2021)