electronics

MDPI

Article

Integrated Model Text Classification Based on Multineural Networks

Wenjin Hu [1], Jiawei Xiong [1], Ning Wang [1], Feng Liu [1,*], Yao Kong [2] and Chaozhong Yang [3]

[1] College of Computer Science, Xi'an Polytechnic University, Xi'an 710600, China; huwenjin@stu.xpu.edu.cn (W.H.); jw-xiong@stu.xpu.edu.cn (J.X.); wangning@stu.xpu.edu.cn (N.W.)
[2] College of Electronics and Information, Xi'an Polytechnic University, Xi'an 710600, China; kongyao19@xpu.edu.cn
[3] National Time Service Center, Chinese Academy of Sciences, Xi'an 710600, China; ycz@ntsc.ac.cn
* Correspondence: liufeng@xpu.edu.cn

Abstract: Based on the original deep network architecture, this paper replaces the deep integrated network by integrating shallow FastText, a bidirectional gated recurrent unit (GRU) network and the convolutional neural networks (CNNs). In FastText, word embedding, 2-grams and 3-grams are combined to extract text features. In recurrent neural networks (RNNs), a bidirectional GRU network is used to lessen information loss during the process of transmission. In CNNs, text features are extracted using various convolutional kernel sizes. Additionally, three optimization algorithms are utilized to improve the classification capabilities of each network architecture. The experimental findings using the social network news dataset demonstrate that the integrated model is effective in improving the accuracy of text classification.

Keywords: deep network architecture; FastText; bidirectional gated recurrent unit; text classification

Citation: Hu, W.; Xiong, J.; Wang, N.; Liu, F.; Kong, Y.; Yang, C. Integrated Model Text Classification Based on Multineural Networks. *Electronics* **2024**, *13*, 453. https://doi.org/10.3390/electronics13020453

Academic Editor: Dimitris Apostolou

Received: 11 December 2023
Revised: 16 January 2024
Accepted: 20 January 2024
Published: 22 January 2024

1. Introduction

As the volume of massive text data grows, it has become increasingly important to mine and manage useful text information, and text classification, which is an important branch in the domain of natural language processing (NLP), has made a significant contribution to the assignment of text classification. As text classification techniques have evolved, specialists have used expert systems, traditional machine learning and now deep learning [1–3].

Text classification has seen great progress over the last few years. For example, reference [4] proposes two hybrid deep learning models focusing on the analytical localization of the attention mechanism to obtain high classification accuracies for different datasets. The efficient self-attention-driven text-matching network proposed in reference [5] outperforms existing techniques on the Stanford natural language reasoning and WikiQA datasets with many fewer parameters. Reference [6] proposed a transformer encoder–decoder-based multilabel text categorization algorithm that is able to adaptively extract the dependencies between different labels and texts. Reference [7] aimed to apply an automatic classification model based on BERT to a new energy industry policy, and the model comparison results showed that the BERT model had higher accuracy, recall and F_1 scores and a better classification effect. Reference [8] proposed an XLNet–CNN–GRU dual-channel aspect-level comment text sentiment classification method, which obtained higher accuracy and F_1 values than the five compared neural network structures in the field of NLP.

At present, there are two problems to be solved in text classification: one is the accuracy of multilabel category text data, and the other is the network structure of the classifier models [9]. In general, the deeper the structure of the network is, the better the classification effect is. However, the computational complexity and space complexity of the model increase exponentially. In practical applications, such a deep network model will

lead to a very slow classification speed. To balance the accuracy and depth of the model, shallow network models are frequently employed to address the depth of the model, while integrated models are applied to address the accuracy of text classification. For shallow models, single-network structures based on deep neural networks (DNNs) [10], recurrent neural networks (RNNs) [11] and convolutional neural networks (CNNs) [12] are often used. For integrated models, Schapire R E [13] proposed the boosting classification algorithm, which creates a single strong learner by combining several weak learners, improving classification performance, followed by the bagging algorithm [14]. Combinations of models have also emerged in deep learning, and Lai S [15] used a combination of the CNN and RNN, also known as the recurrent convolutional neural network (RCNN) model, that is capable of not only extracting important features in text but also obtaining contextual information about the text, which leads to excellent categorization effectiveness in text categorization assignments. Additionally, a random multimodel for text categorization based on three deep learning network architectures was proposed by Kowsari [16,17]. It utilizes DNN, RNN and CNN architectures to stochastically generate neurons and hidden layers for each model and derives the prediction results with majority voting to enhance the accuracy of text categorization. However, because the quantities of neurons and hidden layers are generated at random, the structure of the generated network is variable with each iteration, resulting in difficult training of the models and very complicated computation [18]. In addition to the combination of the three network architectures, hierarchical deep learning for text classification was introduced by K. Kowsari [19]. It integrates all deep learning techniques into one hierarchy for document classification with enhanced accuracy over conventional approaches.

In this paper, the primary objective is to improve the classification accuracy for news text by introducing a novel integrated model. The integrated model in this paper uses three different network architectures. On the basis of each network architecture, an improved shallow FastText network, a shallow bidirectional GRU network and a shallow CNN are designed. Additionally, various optimization algorithms are used to handle feature issues for long and sparse text. Furthermore, the learning rate is adjusted by the optimizer to improve the flexibility of the model by adapting the trained model to handle datasets with diverse text features. Ultimately, a robust text categorization model is yielded from the improved network structure through an integrated strategy and parallel training method. Therefore, the integrated model of this paper will provide a new method of text classification for single classifiers and deep integrated models.

2. Optimizer

In conventional integrated models, one of the limitations is the time complexity. The loss and the gradient of the model have difficulty converging, which leads to a decrease in accuracy in predicting text categories [20]. To reduce the problems of loss and gradient convergence during the training of the model, three optimizers are used to improve the model.

2.1. Optimization Algorithm for Nesterov Momentum

For the classification task, the corresponding optimization algorithms are selectively adapted for individual classification models. The principal purpose of these algorithms is to improve the classification accuracy by optimizing the model parameters. A significant indicator of the training model parameters lies in the decision on the learning rate; the learning rate will considerably determine the convergence of the model toward the global optimal solution [21]. In considering the learning rate to accommodate volatile data types, the Nesterov-based root mean square propagation (RMSProp) [22] algorithm with dynamic adjustment of the learning rate during the training iterations is incorporated to make the model more flexible. The principal steps are as follows: initially, the samples for each iteration are collected from the training subset, the gradients of the iterative samples are calculated and averaged based on Formula (1) and an exponential decay parameter

is employed to manipulate the historical information, also defined as the cumulative gradient based on Formula (2). Thus, with parameters g and r, the learning rate based on Formula (3) and model parameters based on Formula (4) are dynamically updated during the iterative process.

$$g = \frac{1}{m}\nabla_{\tilde{\theta}} \sum_i \zeta(f(x^{(i)}; \tilde{\theta}), y^{(i)}) \tag{1}$$

$$r = \rho r + (1-\rho) \cdot g \cdot g \tag{2}$$

$$v = \alpha v - \frac{\varepsilon}{\sqrt{r}} \cdot g \tag{3}$$

$$\theta = \theta + v \tag{4}$$

$$\tilde{\theta} = \theta + \alpha v \tag{5}$$

where g and r denote the gradient and exponential attenuation coefficients, respectively, v represents the learning rate, the update of the parameter is symbolized by θ, and the momentum coefficient is indicated by α.

2.2. Optimization Algorithm of Deviation Correction Based on Adam

Adam [23] is extensively applicable to text classification models, which are ideally qualified to handle text gradient sparsity as well as model parameter issues. When calculating the gradient, Adam handles the sparsity and instability of gradients by incorporating the estimation of first- and second-order moments based on Formulas (6) and (7), respectively, and modifying the deviation based on Formulas (8) and (9), respectively.

In Formula (6), the gradient g and exponential attenuation rate are calculated to update the first moment. In Formula (8), the deviation is corrected to accelerate the convergence of the model. Formula (7) modifies the deviation by introducing the second moment sum (9) to improve the capability of the model to handle nonstationary objectives. Formula (10) updates the value of the parameter in Formula (11). In text classification, this algorithm not only reduces the memory consumption when training the model parameters but also solves the problem of the convex convergence of the model.

$$s = \rho_1 s + (1-\rho_1) g \tag{6}$$

$$r = \rho_2 r + (1-\rho_2) g \cdot g \tag{7}$$

$$\hat{s} = \frac{s}{1-\rho_1^t} \tag{8}$$

$$\hat{r} = \frac{r}{1-\rho_2^t} \tag{9}$$

$$\Delta\theta = -\varepsilon \frac{\hat{s}}{\delta + \sqrt{\hat{r}}} \tag{10}$$

$$\theta = \theta + \Delta\theta \tag{11}$$

where the sample gradient is represented by g, the estimated exponential decay rate is denoted by ρ_1 and ρ_2, a small constant that maintains numerical stability is represented by δ, the step size is represented by $\in$, and $\Delta\theta$ is used to update parameter θ.

2.3. Improved Optimization Algorithm Based on SGD

The batch gradient descent (BGD) [24] algorithm was the most widely used gradient algorithm before the invention of the stochastic gradient descent (SGD) algorithm. The BGD algorithm is a batch gradient algorithm for the whole dataset. By calculating the direction of the solution gradient for all samples, this approach can acquire the global optimal solution. However, the computational effort is substantial, and the computational speed is relatively slow when the data quantity is large. In this work, the SGD algorithm is used to address the limitations of the BGD approach. SGD [25] is a popular optimization approach that enhances the gradient descent algorithm. The essential idea of the algorithm is that a random sample from all the training data can be taken at each iteration to estimate the objective function's gradient. Hence, it can significantly scale down the time sophistication of the algorithm and be applicable to large-scale text datasets. A batch of text training datasets is fed into the model when using the random gradient descent algorithm. The objective of the algorithm focuses on optimizing the target function, as expressed in Formula (12).

$$\min_{w \in R^n} F(w) = \frac{1}{m} \sum_i \zeta(f(x^{(i)}; w), y^{(i)}) \tag{12}$$

where ζ is the model's experience loss for the training data, a sample is selected at random to calculate the gradient in each iteration, and the parameter value of w is adjusted in reverse. Then, the average of all cumulative parameter values is calculated. The final parameter value of the model is obtained based on Formula (13).

$$w = \frac{1}{m} \sum_i^m w_i - \eta_i \nabla \zeta(f(x^{(i)}; w_i), y^{(i)}) \tag{13}$$

where w_i is the set of parameters of sample i and η represents the learning rate.

The combination of Nesterov momentum and RMSProp enhances the model convergence speed and adaptability to diverse data types by introducing momentum and dynamically adjusting the learning rate. The Adam algorithm excels in addressing sparse gradients and parameter instability in text classification. It achieves this through first- and second-order moment estimates and deviation correction, reducing memory consumption and accelerating model convergence. The improved SGD, incorporating random gradient estimation, is suitable for large-scale text datasets, significantly reducing the computation time and enhancing the speed of the algorithm. In summary, the Nesterov momentum combined with RMSProp improves convergence speed and adaptability. Adam excels in handling sparse gradients, while the enhanced SGD is well suited for large-scale text datasets, reducing the computation time.

3. Model Design and Integration

3.1. FastText Model Design

The FastText [26] model has the function of word vector computation and classification. To enhance the classification capabilities, the objective model is trained by expanding the layers of the network model, while FastText utilizes a shallow network composed of the input, hidden and output layers. Figure 1 shows its basic structure.

Where the input layer is the feature vector after word embedding, the feature vector is composed of words and phrases in the text or sentence, and the model's output is the probability value of the sentence or text belonging to different prediction categories. The hidden layer calculates the mean value of the input vector, calculated based on Formula (14).

$$hidden = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{14}$$

where the mean value of the word vector in the input layer represents the sentence information, and the values calculated in the hidden layer are then fed into the softmax multiclassifier to output the predicted class information.

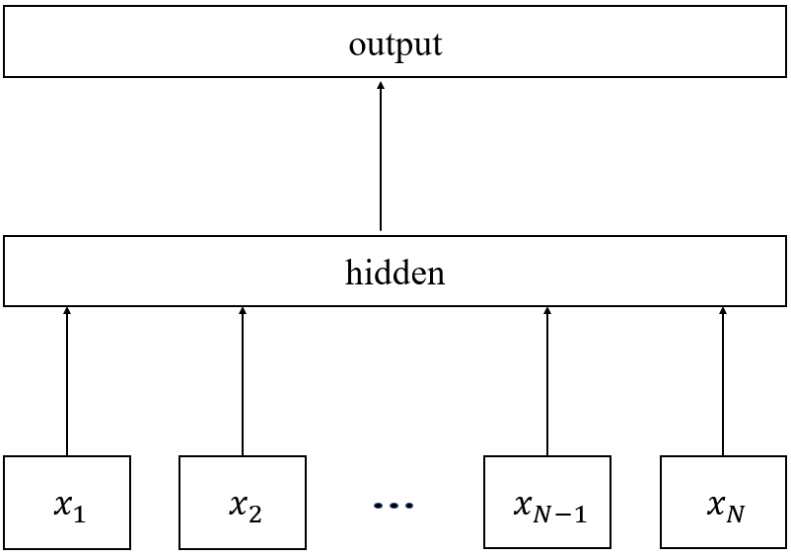

Figure 1. Basic structure of the FastText network.

Additionally, the FastText model boosts the categorization capability of the model by training tricks. There are two main ways: the first is to improve operational efficiency by using the hierarchical softmax layer. The hierarchical softmax uses the Huffman tree structure instead of the flat softmax. It mainly uses the Huffman coding method to encode multiple tags, and the values of all leaf nodes are calculated by the original calculation. It needs to calculate only the value from the root node to one of the leaf nodes, which greatly reduces the complexity of model training and the test time on the test set. The second is to apply N-grams to extract features and use the hashing algorithm to map the 2-gram and 3-gram vocabulary information into two tables. Because the vocabulary of 2-grams and 3-grams is much larger than that of word embedding, the hash bucket method is used to map 2-grams and 3-grams to buckets, and the 2-grams and 3-grams in the same bucket share the same word vector. The matrix composed of word embedding and N-grams is depicted in Figure 2.

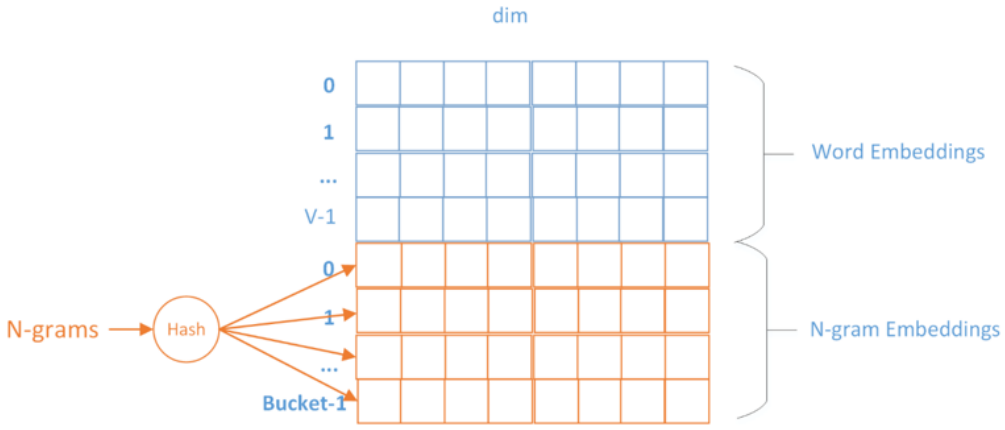

Figure 2. Word embedding and N-gram mapping table.

In the hidden layer, the word embedding, 2-grams and 3-grams of the input sentences are concatenated, and the mean value of each sequence word is obtained. Then, the calculated mean value is sent to the softmax layer through a nonlinear activation function of the full connection layer for normalization processing and finally outputs the probability values of the predicted values of each category. The architecture of FastText is depicted in Figure 3.

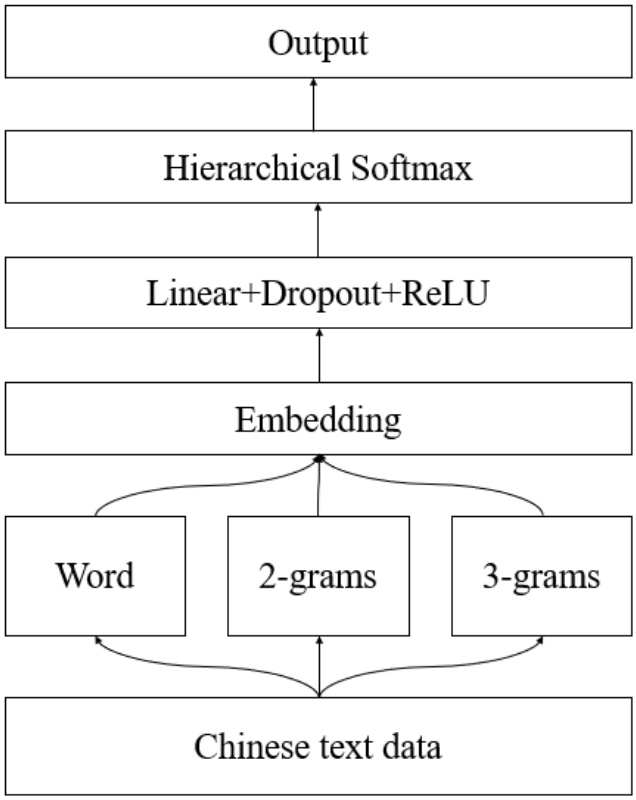

Figure 3. FastText network design structure.

3.2. Shallow Bidirectional GRU Network Design

In the traditional RNN network structure, the RNN can retain only short memory information. If the sequence of text data is very long, the earlier time series information cannot be transmitted to the following time series information, resulting in inaccurate text classification results. In addition, there is a problem of gradient disappearance during backpropagation, mainly because the value of the gradient update neural network weight changes little, resulting in an inability to learn more text series information. To eliminate this issue, the long short-term memory (LSTM) [27] and GRU [28] text classification methods are used. The LSTM approach has a significant quantity of parameters and is computationally sophisticated. The GRU is used instead of LSTM. As shown in Figure 4, this method can achieve the classification effect of LSTM. The gating mechanism is used to update the door and reset the door using Formulas (15) and (16), respectively. Formula (16) dictates whether to reset the current input and the previous h_{t-1} information, and the amount of prior information saved to current moment is determined by the update door. Finally, the output vectors are calculated based on Formulas (17) and (18).

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{15}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{16}$$

$$\tilde{h}_t = \tanh(W \cdot [h_{t-1} \cdot r_t, x_t]) \tag{17}$$

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t \tag{18}$$

where z_t is the update gate vector, σ is the rectified linear unit (ReLU) activation function, x_t is the input text feature vector, W is the parameter matrix, r_t is the reset gate vector, and h_t is the output vector.

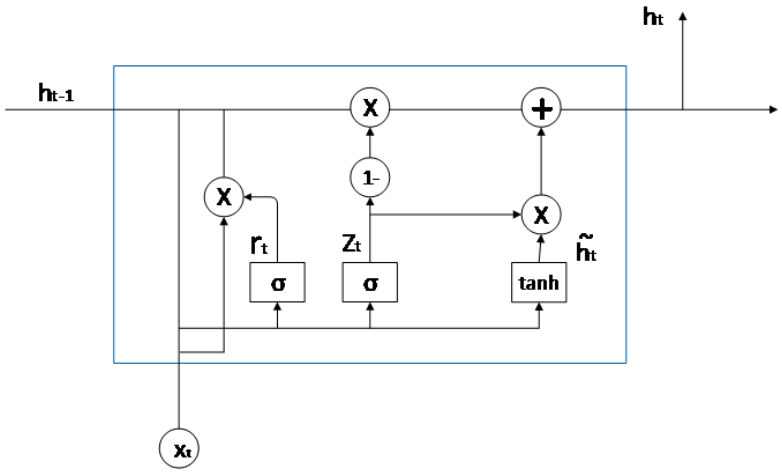

Figure 4. GRU network structure.

In addition, a bidirectional GRU network is designed to address the issue of information loss during the propagation of a single GRU network. In the bidirectional GRU network, there are two main hidden states: the forward-learning GRU network unit and the reverse-learning GRU network unit [29]. Suppose that the hidden state at moment t needs to be calculated. The input of forward learning consists of h_{t-1} and x_t. Reverse learning is composed of h_{t+1} and x_t. Then, the hidden states of forward learning and reverse learning are calculated based on Formulas (19) and (20), respectively.

$$h_{t\rightarrow} = f(W_{\rightarrow} \cdot h_{t-1} + U_{\rightarrow} \cdot x_t + b_{\rightarrow}) \tag{19}$$

$$h_{t\leftarrow} = f(W_{\leftarrow} \cdot h_{t+1} + U_{\leftarrow} \cdot x_t + b_{\leftarrow}) \tag{20}$$

where $W_{\rightarrow}$ and $W_{\leftarrow}$ represent the hidden layer weight matrix in forward learning and the hidden layer weight matrix in reverse learning, respectively, h_{t-1} and h_{t-+1} represent the hidden state at times t-1 and t+1, respectively, $U_{\rightarrow}$ and $U_{\leftarrow}$ represent the weight matrix of forward input and reverse input in the input layer, respectively, x_t represents the input at time t, $b_{\rightarrow}$ and $b_{\leftarrow}$ represent the forward and reverse offset values, respectively, and f represents the activation function.

In the bidirectional GRU network, the forward- and backward-learning GRU do not interfere with each other before the model output, and the weight matrix of the input, hidden layers and bias term are also not shared [30]. At the output, the forward-learning text information and the backward-learning text information are spliced, and the feature vector y_t is output at time t, which is finally normalized using the softmax function and calculated based on Formula (21).

$$y_t = soft\max(f(V \cdot [h_{t\rightarrow}, h_{t\leftarrow}] + b_y)) \tag{21}$$

where f is the sigmoid activation function, V is the weight matrix, and is the offset term of the output layer.

Finally, the maximum value of the probability in the number of categories is taken as the final prediction result. The whole GRU is depicted in Figure 5.

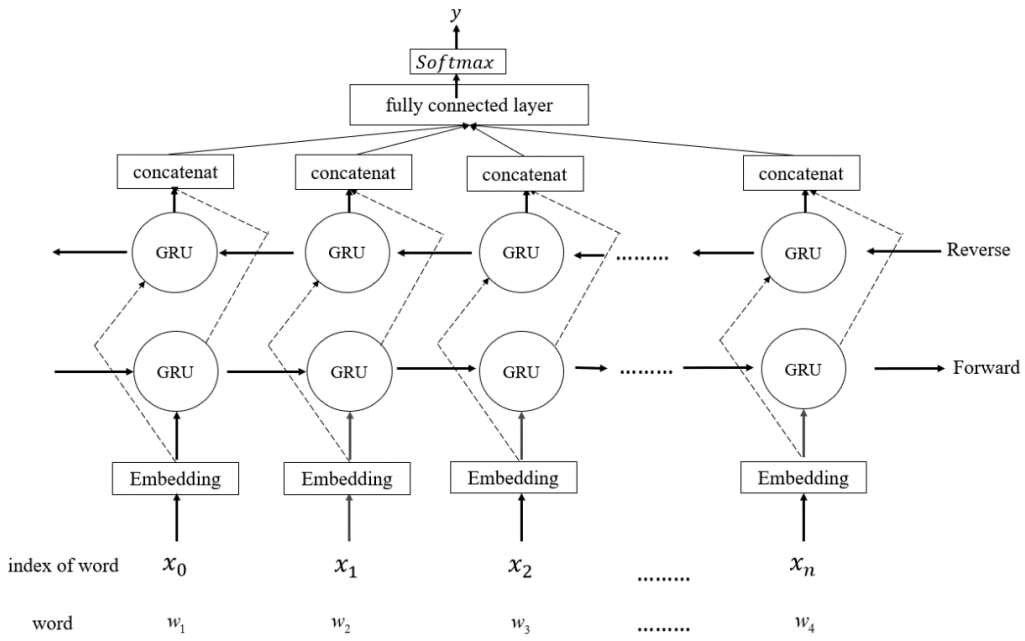

Figure 5. Bidirectional GRU network design structure.

3.3. Shallow TextCNN Model Design

The shallow TextCNN model incorporates multiple layers, and the particular procedure of each layer is as follows:

3.3.1. Input Layer

Before inputting the text data in the input layer, the text data need to be tokenized. Then, the tokenization sentences are mapped into word vectors by embedding, and finally, all the word vectors are spliced together to form a vector matrix.

3.3.2. Convolutional Layer

The principal objective of the convolutional layer is to extract the input features represented by the sentence word vectors. The convolutional kernel mainly extracts the features through the dot product operation of the matrix. In addition, the parameter matrix of the convolutional operation is weight sharing, which can greatly improve the efficiency of sentence features. In this experiment, a total of three sizes of convolutional kernels (2,3,4) are used to extract text features to solve the problem of feature loss.

3.3.3. Pooling Layer

The pooling layer needs to further deal with the results of the convolutional operation, and the pooling layer mainly compresses and reduces the dimensionality of the feature mapping results to reduce the parameters of the model. At present, there are two primary pooling layers: the maximum layer and the average layer. The average pooling and maximum pooling are calculated based on Formulas (22) and (23), respectively.

$$AvgPool = \frac{x_1 + x_2 + ... + x_N}{N} \tag{22}$$

$$MaxPool = \max(x_1, x_2, ..., x_N) \tag{23}$$

where x_i represents the first feature after the convolutional operation and N represents the total features after the convolution operation.

3.3.4. Fully Connected and Softmax Layers

The fully connected layer and softmax layer are the last layers. After convolution and pooling of the feature matrix after the input layer, all the feature results are connected. Then, through a probability distribution function of an N-dimensional vector, where N indicates the number of categories, the category with the highest probability is exported by the softmax layer. The final CNN architecture designed in this experiment is shown in Figure 6.

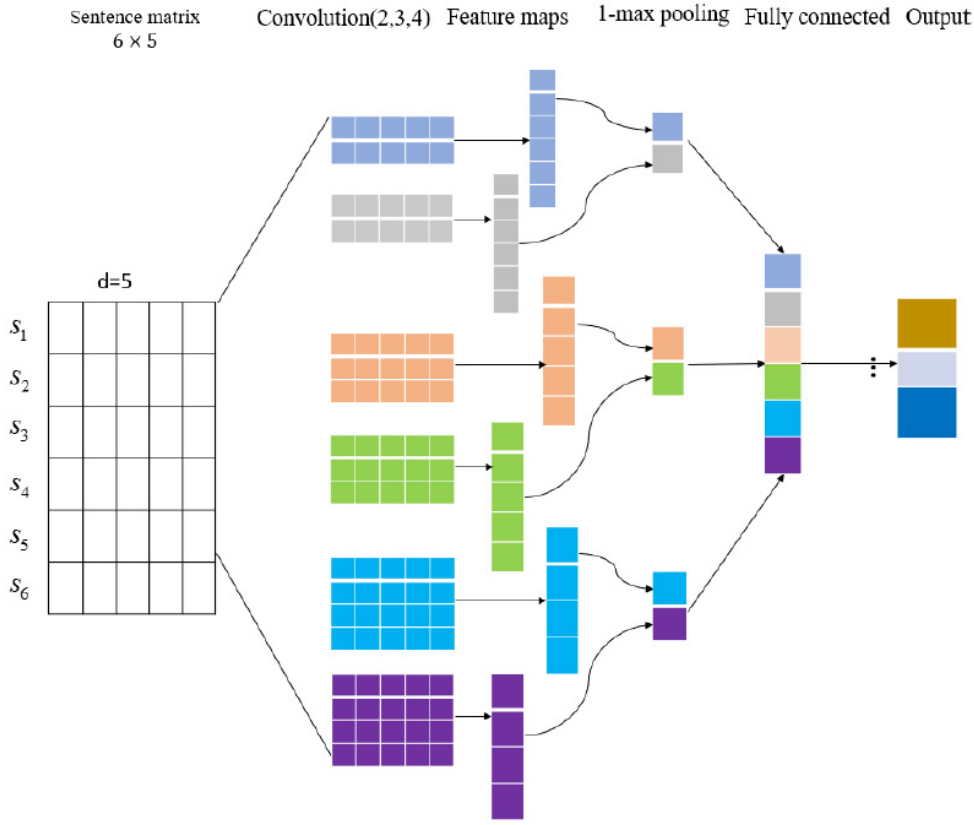

Figure 6. Shallow CNN network architecture.

3.4. Model Integration

In the integrated model, this paper integrates three network architectures: the FastText network, shallow bidirectional GRU network and shallow TextCNN network designed in the first three sections. Figure 7 shows the overall structure of the integrated model. The model is mainly composed of three layers: the input layer, hidden layer and output layer. The input layer represents the input text feature vector, and the output layer represents the classification result statistics of each classifier after applying the softmax function. In the hidden layer, from left to right, FastText, Bidirectional GRU and TextCNN network structures are shown. These network architectures are integrated to classify the input text information at the same time; finally, the classification results of the three network structures are determined, and the maximum value is selected as the final classification result.

By using the integrated strategy of voting, the integrated model has the following characteristics:

(1) Different from the traditional single classifier, the designed network architecture model can effectively extract text features; for example, word embedding, 2-gram and 3-gram feature vectors are used to stitch together in FastText, and convolutional kernels (2, 3, 4) and pooling layers are utilized to extract features in CNNs. For the purpose of minimizing the model's information loss during training, a bidirectional GRU is used for forward and backward learning.

(2) To further boost the training efficiency and accuracy of the model, each network architecture uses the SGD, RMSProp and Adam optimization algorithms at the same time because these three algorithms can solve the problems of large sample data, sparse sample data and the model learning rate. Therefore, for each network architecture, three optimization algorithms are used to train nine network models, which can solve most of the problems in the training process.

(3) General integrated models have the risk of overfitting. Each network model trained in this work uses the dropout method to reduce the overfitting problem in the training process so that each model has a strong learning ability. This paper also uses the ReLU activation function to reduce problems such as training stops and gradient disappearance in the training process.

(4) Traditional integrated learning uses serial training, and the integrated model in this paper uses parallel training to train nine kinds of network models at the same time, which significantly shortened the training time of the model. In addition, another function of parallel training is that when each model makes a prediction, the prediction results of all prediction models are recorded, the final voting is conducted, and the largest number of votes is compared with the actual categories. The model evaluation indicators, such as the accuracy, are calculated.

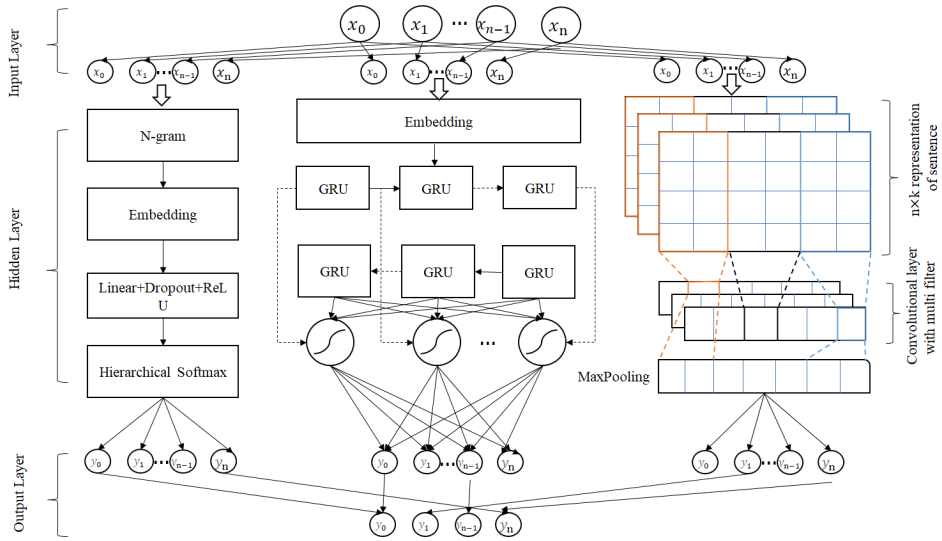

Figure 7. Integrated network architecture.

In summary, the integrated model is designed as illustrated in Figure 8.

The main idea of the integrated model is as follows: the total number of models trained in parallel is k, the number of categories of documents is m, and the classification results of text datum i of each model are counted, in which the datum i with the largest number of votes is considered to belong to the c_{ij} category. Finally, the accuracy of the c_{ij} category is summed to average. The final prediction result is calculated based on Formulas (24)–(26).

$$y_{i,j} = [y_{i1}, y_{i2}, ..., y_{ik}] \tag{24}$$

$$c_{i,j} = \max[c_{i1}, c_{i2}, ..., c_{im}] \tag{25}$$

$$\hat{p}_{i,j} = \frac{\sum_{n=1}^{N} soft\max(y_{in})}{N} \tag{26}$$

where $y_{i,j}$ represents the result of model j classifying text i, $c_{i,m}$ represents the number of votes that text datum i belongs to category m, $\hat{p}_{i,j}$ represents the accuracy of text datum i belonging to category j, and N represents the number of votes.

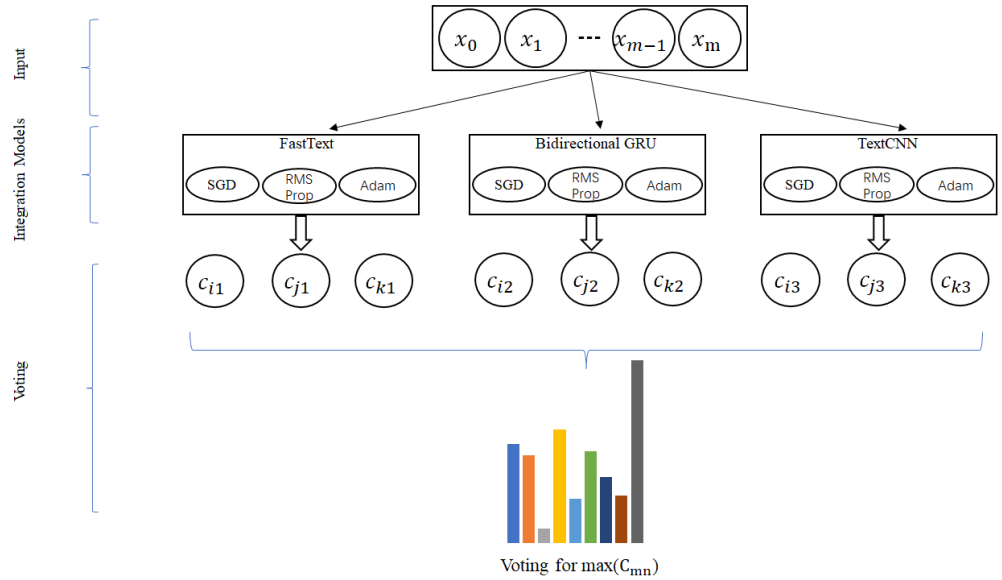

Figure 8. Integrated network voting structure.

4. Experiment

4.1. Dataset

The THUCNews dataset supplied by the NLP group of Tsinghua University is applied in this experiment, and the news headlines in each text datum are extracted as the basis for classification. The length of the text is between 20 and 30, with a total of 10 categories. Each category has 20,000 pieces of text data. For 200,000 pieces of news text data in total, the division of samples in each category is shown in Figure 9. In this experiment, the THUCNews news data are completely separated into three parts, with 90% for model training and 5% each for testing and validation, as shown in Figure 10.

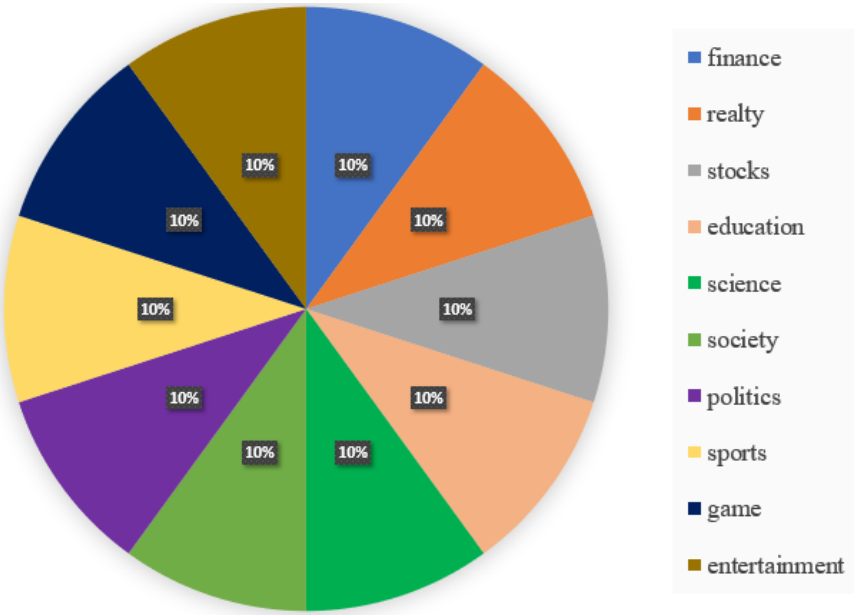

Figure 9. Sample division of THUCNews categories.

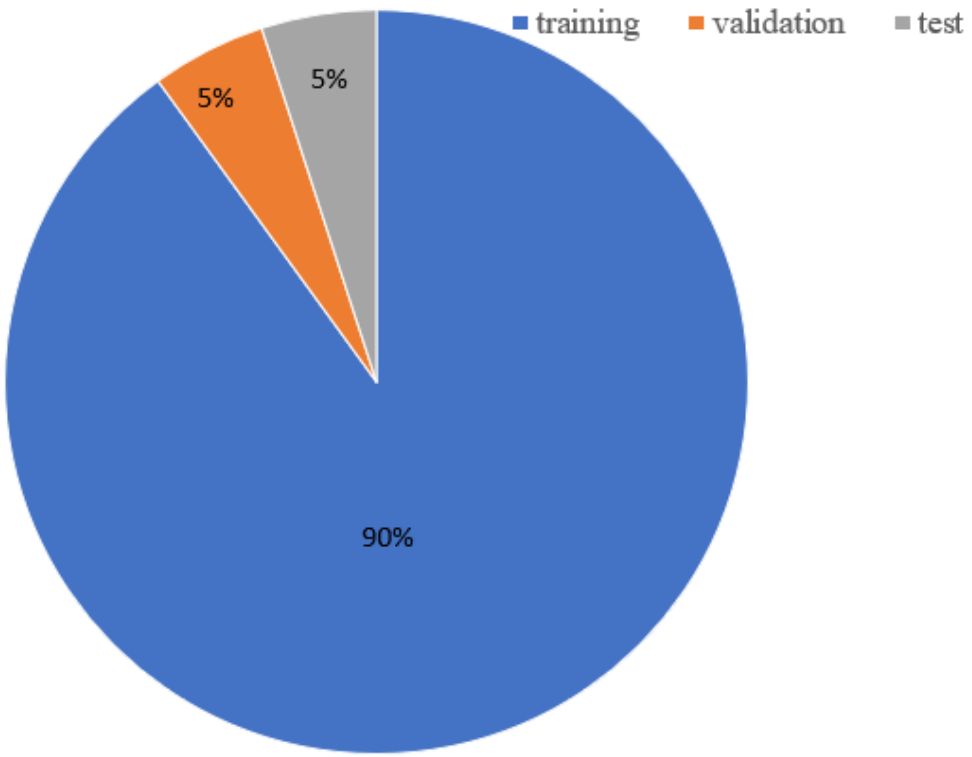

Figure 10. Sample partition of the dataset.

4.2. Text Preprocessing

First, 10 categories of text data are extracted from the downloaded news dataset. Because each text data point is composed of news headlines and body content, the first line of each document is read and written into a new .txt document. At the same time, the format is "the first line of the text title + Tab spacer + category mapping into corresponding numbers". Then, 5% of the data from the 10 categories is selected as the test set and the verification set. The remaining 90% serve as the training set. After extraction, all the training sets, test sets and verification sets are written into train.txt, test.txt and dev.txt text, respectively. Finally, the useless data, such as spaces, need to be removed from the text. Then, each word is divided into segments, and a word list related to it is established. Each word in the word list has an index value corresponding to the mapping relationship between the word lists.

4.3. Parameter Setting

The parameters of FastText, the shallow bidirectional GRU network and the shallow TextCNN are set as shown in Table 1, Table 2 and Table 3, respectively.

Table 1. FastText model parameters.

Parameter	Value
word vector dimension	300
hidden layers	1
activation function	ReLU
hidden layers	1
batch size	128
learning rate	0.001
dropout	0.5
hidden layer size	256
pad size	32

Table 2. Bidirectional GRU model parameters.

Parameter	Value
word vector dimension	300
hidden layers	2
activation function	ReLU
hidden layers	1
batch size	128
learning rate	0.001
dropout	0.5
hidden layer size	128
pad size	32

Table 3. Shallow CNN model parameters.

Parameter	Value
word vector dimension	300
fully connected layers	1
convolutional layers	1
pooling layers	1
number of convolutional kernels	256
batch size	128
learning rate	0.001
dropout	0.5
convolutional kernels size	(2,3,4)
pad size	32

4.4. Evaluation and Experimental Analysis

To describe the performance of the integrated model, the evaluation indicators are calculated based on Formulas (27)–(30).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{27}$$

$$Precision = \frac{TP}{TP + FP} \tag{28}$$

$$Recall = \frac{TP}{TP + FN} \tag{29}$$

$$F_{1-score} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{30}$$

To prove the superiority of this experimental model, the RCNN model, pre-trained transformer model, deep pyramid convolutional neural network (DPCNN) model and integrated model are used for comparative experiments. The transformer model, through the use of residual connections, a self-attention mechanism and positional encoding, can effectively address issues related to temporal and textual sequence positions. Given the superiority of the transformer architecture model in text classification, we compare the transformer model in this paper with our model, observing their performance on a news dataset. Each group of comparative experiments uses the same dataset and corpus. Figure 11 displays the classification results of all comparison models under the verification set. The classification result of the integrated model under ten categories is better than that of any group of comparative experiments.

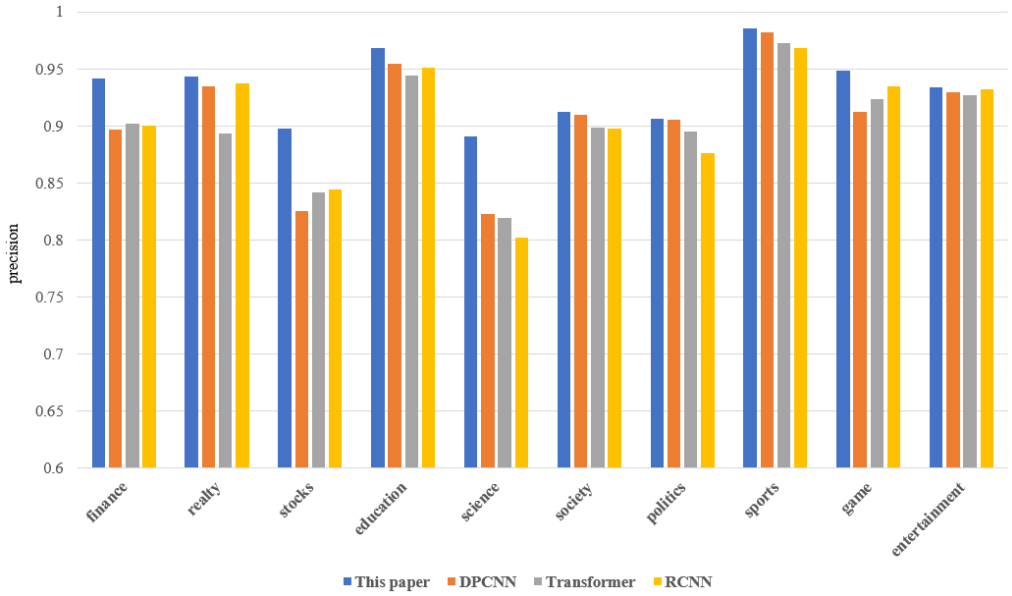

Figure 11. Precision comparison bar chart.

Table 4 displays the classification precision of each model in each category. It can be calculated that the precision of the integrated model exceeds that of all comparative models. The model in this paper is 4.03%, 0.68%, 5.29%, 1.41%, 6.78%, 0.22%, 0.12%, 0.39%, 1.35% and 0.23% higher than the highest comparative model. The integrated model has a good classification effect. Similarly, the table quantifies the accuracy of the three base classification models.

Table 4. Statistical table of model precision (unit %).

	This Paper	DPCNN	Transformer	RCNN	TextCNN	Bidirectional GRU	FastText
finance	94.18	89.68	90.15	90.15	92.65	91.52	93.65
realty	94.36	93.49	89.33	93.68	91.84	93.33	94.63
stocks	89.74	82.56	84.14	84.45	85.40	84.84	86.23
education	96.85	95.44	94.39	95.06	94.76	95.65	95.60
science	89.05	82.27	81.95	80.20	87.58	84.37	87.86
society	91.19	90.97	89.86	89.78	89.40	89.99	91.19
politics	90.64	90.52	89.53	87.64	89.87	87.05	90.64
sports	98.57	98.18	97.21	96.83	93.96	97.28	98.38
games	94.83	91.21	92.36	93.48	94.77	94.21	94.38
entertainment	93.41	92.94	92.71	93.18	92.65	91.09	93.41

Figure 12 displays the accuracy of the integrated and comparative models. As shown in the graph, the performance of the integrated model is superior to that of the comparative models. The final quantitative relationship among accuracy, recall and $F_{1-score}$ is shown in Table 5.

Table 5. Comparison of the experimental results (unit %).

Model	Accuracy	Recall	F_1
RCNN	90.93	90.73	90.83
Transformer	90.14	90.14	90.15
DPCNN	90.90	90.90	90.96
TextCNN	91.28	91.27	91.26
Bidirectional GRU	92.62	92.57	92.58
FastText	92.60	92.57	92.58
This paper	92.57	92.88	93.08

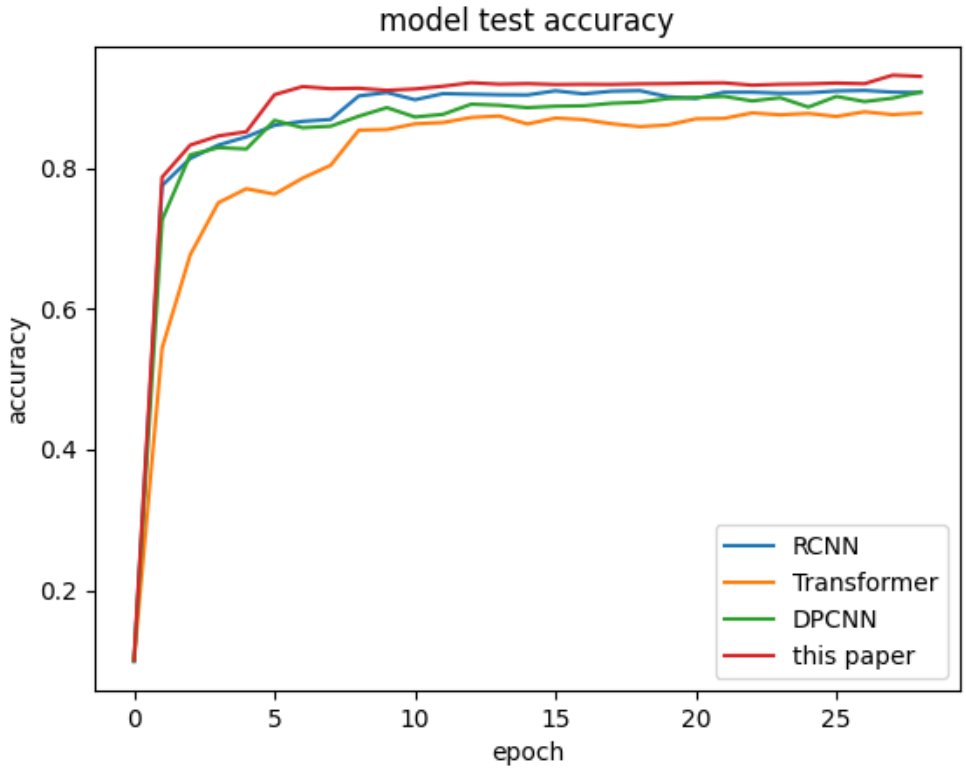

Figure 12. Variation curve for model accuracy.

From Table 5, this paper demonstrates that the proposed model implements an excellent classification effect on the THUCNews dataset, reaching 92.57%. The transformer model has the smallest such effect, with an accuracy of 90.14%. Among the other models, the RCNN and DPCNN reach 90.93% and 90.90%, respectively. Regarding recall and $F_{1\text{-score}}$, the model of this paper is higher than the other models, reaching 92.88% and 93.08%, respectively. The result of the final comparison is shown in Figure 13. To further capture the loss variation of the model during the iterative process, the loss changes of different models are shown in Figure 14. Figure 14 illustrates that after using the optimization algorithm, the loss value of the integrated model steadily decreases and ultimately converges to a stable numerical interval, while the loss of the other models is overall larger than that of the integrated model. This result shows the superiority of the optimization algorithm used in this paper.

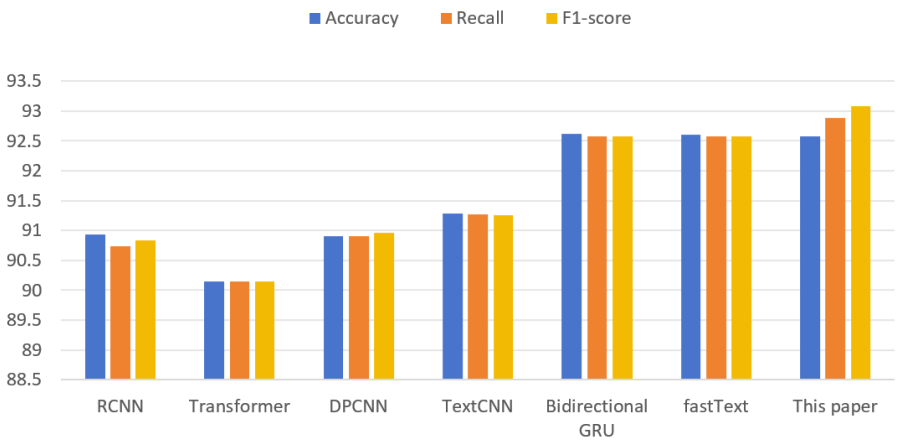

Figure 13. Model comparison bar chart.

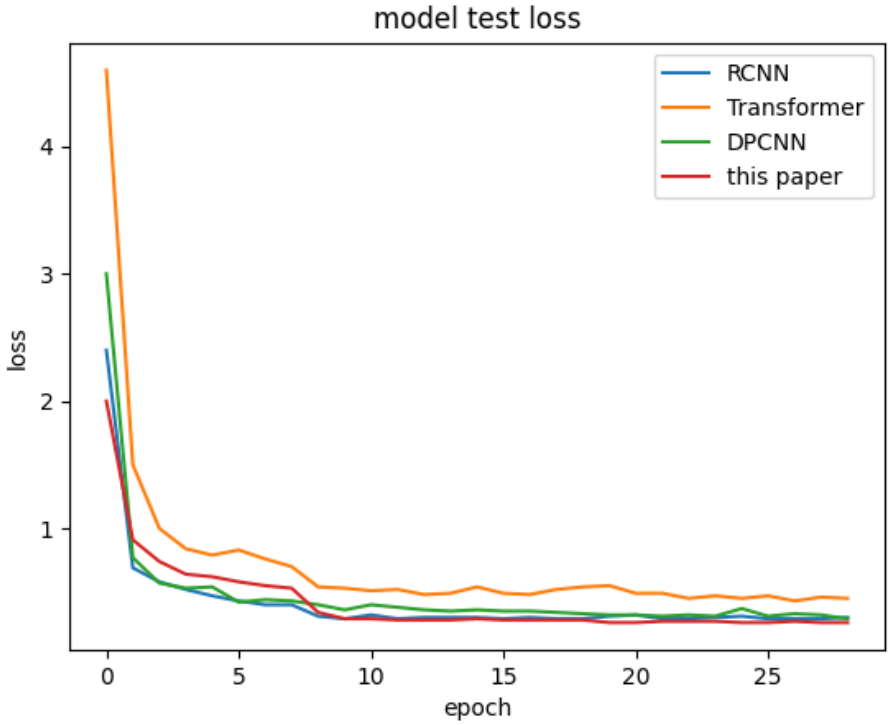

Figure 14. Change curve for model loss.

From Table 6, the impact of the three optimization algorithms on each model can be seen. Table 6 shows the accuracy results of the three optimization algorithms (Adam, RMSprop and SGD) on the three models (TextCNN, bi-directional GRU and FastText). It can be seen that RMSprop achieves the highest accuracy on the TextCNN and FastText models, while it is slightly lower than Adam on the TextRNN model, and SGD performs the worst. These results indicate that the RMSprop optimization algorithm plays a very crucial role in this particular text classification task, while the other two optimization algorithms perform more ordinarily.

Table 6. Results of three optimizer algorithms on the model (unit %).

Precision	TextCNN	TextRNN	FastText
Adam	90.92	90.78	91.91
RMSprop	91.27	90.50	92.57
SGD	86.72	83.13	80.58

Using the aforementioned experimental findings, this study concludes that the classification performance of the integrated model is superior to that of the RCNN, transformer and DPCNN models. They also prove the correctness of the three network architectures and optimization algorithms designed in this paper.

4.5. Model Generalization Study

To further validate the generalization performance of the integrated model, we extend the experimental validation to the TNEWS dataset, which contains news data from 15 different categories. We filtered the TNEWS Chinese news dataset using 10 of the 15 categories. The integrated model shows excellent performance on this dataset, as shown in Table 7, and we can clearly see that the present model still performs well on the TNEWS dataset. The accuracy reaches 90.25%, which is much higher than the other models. In addition, the recall and F_1 score of the integrated model are also significantly better than the other models, reaching 90.44% and 90.34%, respectively. The visualization results are given in

Figure 15. These results fully demonstrate the generalization performance of the integrated model on different datasets and its effectiveness in improving text classification accuracy. Through the validation on different datasets, we believe that the integrated model has high potential application value in improving text categorization accuracy.

Table 7. TNEWS dataset classification results (unit %).

Model	Accuracy	Recall	F_1
RCNN	88.64	89.68	89.16
Transformer	88.17	89.14	88.65
DPCNN	88.75	89.23	88.99
TextCNN	89.46	88.86	89.16
Bidirectional GRU	89.95	89.14	89.54
fastText	89.97	89.73	89.84
This paper	90.25	90.44	90.34

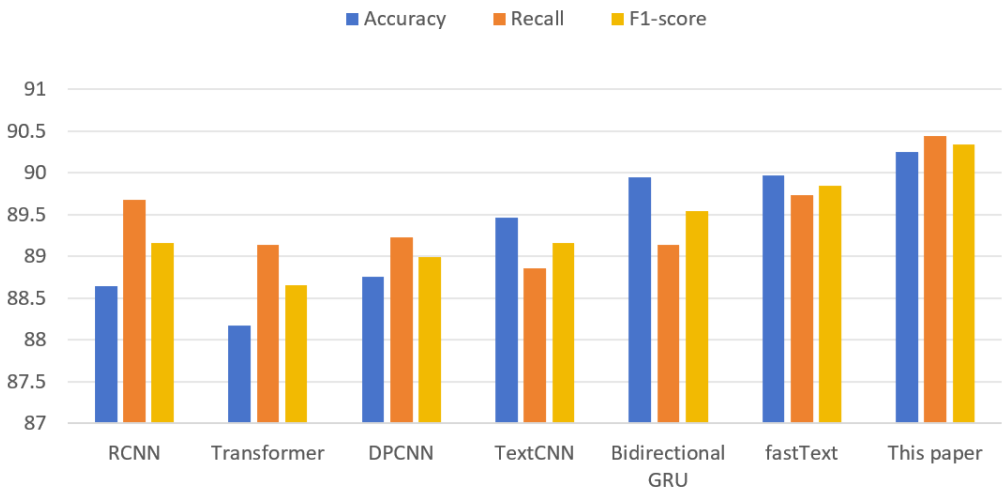

Figure 15. TNEWS dataset visualization results.

4.6. Discussion

The proposed comprehensive model, which integrates an improved shallow FastText network, a shallow bi-directional GRU network and a shallow CNN design, is critically evaluated on the THUCNews text dataset, which consists of 200,000 pieces of news text data in 10 categories. The model achieves an impressive 92.57% accuracy on this dataset, outperforming established models such as DPCNN, RCNN and transformer models. In addition, the combined model outperforms the other models in terms of recall and F_1 score, reaching 92.88% and 93.08%, respectively. This strong performance highlights the effectiveness of the optimization algorithm used to reduce the loss values and enhance the convergence of the model to the global optimal solution.

In addition, extending the experimental validation to the TNEWS dataset, which contains diverse news articles from 15 different categories, further demonstrates the model's superior generalization ability. On the TNEWS dataset, the synthesized model demonstrates an impressive 90.25% accuracy, outperforming other models. The high recall and F_1 scores further validate the effectiveness of the model in improving text categorization accuracy on different datasets. The use of parallel training methods and integration strategies in a comprehensive model provides a promising solution in the field of text categorization, offering the possibility of superior classification accuracy and strong generalization capabilities on different datasets. However, it is important to note that the computational and spatial complexity of the model may have limitations in practical applications. Future

research efforts could focus on optimizing the efficiency of the model without sacrificing accuracy, thus enhancing its practical utility.

In conclusion, the integrated model proposed in this study exhibits excellent performance on the THUCNews and TNEWS datasets, highlighting its robustness and generalization ability across different datasets

5. Conclusions

This study aims to improve the accuracy of news text classification by introducing an innovative ensemble model. The ensemble model employs three different network architectures—an improved shallow FastText network, a shallow bi-directional GRU network and a shallow CNN design—to cope with the challenges posed by long texts and sparse text features. In addition, various optimization algorithms are employed and the learning rate is dynamically adjusted through optimizer tuning to improve the flexibility of the model. Ultimately, an improved network structure formed by combining the synthesis strategy and parallel training methods produces a robust text classification model that provides a new approach to single classifier and deep ensemble models. The proposed optimization strategy combines parallel training of multiple shallow networks with an ensemble strategy to effectively address the depth and accuracy issues of using a single model. Experimental evaluations on the THUCNews text dataset demonstrate the superior performance of the ensemble model, with accuracy, recall and F_1 scores reaching 92.57%, 92.88% and 93.08%, respectively. It outperforms other models such as DPCNN, RCNN and the transformer model, highlighting the potential of shallow and deep network architectures to enhance text categorization capabilities and provide classification flexibility. On the TNEWS dataset, the synthetic model achieves an impressive accuracy of 90.25%, outperforming the other models. The high recall and F_1 scores further validate the effectiveness of the model in improving text categorization accuracy on different datasets.And excellent results are obtained on the TNEWS dataset, which fully validates the generalization performance of the model. In conclusion, the integrated model is applicable to a variety of news text datasets and provides an effective solution in the field of text categorization.

Although the proposed integrated model performs well in text classification, future research should focus on optimizing its efficiency without compromising accuracy. Exploring techniques to reduce resource requirements and adapting the model to handle various text data types including multimedia and social media generated content are important research directions. In addition, we will complete an application that enables automated management and extraction of useful information, providing solutions to problems related to the accuracy of text data with multiple labeled categories. Overall, research efforts should prioritize optimizing efficiency, expanding applicability to diverse text data and integrating advanced architectures to improve performance in real-world applications.

Author Contributions: Conceptualization, W.H., F.L. and J.X.; data curation, W.H. and N.W.; methodology, W.H., J.X., F.L. and Y.K.; validation, F.L., Y.K. and C.Y.; investigation, W.H., N.W. and C.Y.; writing—original draft preparation, W.H.; writing—review and editing, W.H. and F.L.; visualization, J.X., N.W. and Y.K. All authors have read and agreed to the published version of the manuscript.

Funding: Natural Science Foundation of Shaanxi Province, China (2021JQ-656).

Data Availability Statement: The data is available at http://thuctc.thunlp.org/, https://github.com/yongzhuo/Pytorch-NLU/tree/main/pytorch_nlu/pytorch_textclassification/(accessed on 10 January 2024) and the model code can be provided upon request from the corresponding author, Feng Liu.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

References

1. He, L.; Tan, S.; Xiang, F.; Wu, J.; Tan, L. Research and development of deep learning based text classification. *Comput. Eng.* **2021**, *47*, 1–11.
2. Zhou, N.; Yao, N.; Hu, N.; Zhao, J.; Zhang, Y. CDGAN-BERT: Adversarial constraint and diversity discriminator for semi-supervised text classification. *Knowl.-Based Syst.* **2024**, *284*, 111291. [CrossRef]
3. Nakajima, H.; Sasaki, M. Text Classification Based on the Heterogeneous Graph Considering the Relationships between Documents. *Big Data Cogn. Comput.* **2023**, *7*, 181. [CrossRef]
4. Prabhakar, S.K.; Rajaguru, H.; Won, D.O. Performance Analysis of Hybrid Deep Learning Models with Attention Mechanism Positioning and Focal Loss for Text Classification. *Sci. Program.* **2021**, *2021*, 2420254. [CrossRef]
5. Tiwari, P.; Jaiswal, A.K.; Garg, S.; You, I. SANTM: Efficient Self-attention-driven Network for Text Matching. *ACM Trans. Internet Technol. (TOIT)* **2021**, *22*, 1–21. [CrossRef]
6. Duan, L.; You, Q.; Wu, X.; Sun, J. Multilabel Text Classification Algorithm Based on Fusion of Two-Stream Transformer. *Electronics* **2022**, *11*, 2138. [CrossRef]
7. Li, Q.; Xiao, Z.; Zhao, Y. Research on the Classification of New Energy Industry Policy Texts Based on BERT Model. *Sustainability* **2023**, *15*, 11186. [CrossRef]
8. Wu, D.; Wang, Z.; Zhao, W. XLNet-CNN-GRU dual-channel aspect-level review text sentiment classification method. *Multimed. Tools Appl.* **2024**, *83*, 5871–5892. [CrossRef]
9. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Chenaghlu, M.; Gao, J. Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–40. [CrossRef]
10. Li, Q.; Peng, H.; Li, J.; Xia, C.; Yang, R.; Sun, L.; Yu, P.S.; He, L. A survey on text classification: From traditional to deep learning. *ACM Trans. Intell. Syst. Technol. (TIST)* **2022**, *13*, 1–41. [CrossRef]
11. Fu, T.; Liu, H. Research on Chinese Text Classification Based on Improved RNN. In Proceedings of the 2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 26–28 May 2023; pp. 1554–1558.
12. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. A convolutional neural network for modelling sentences. *arXiv* **2014**, arXiv:1404.2188.
13. Schapire, R.E. The strength of weak learnability. *Mach. Learn.* **1990**, *5*, 197–227. [CrossRef]
14. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
15. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; Volume 29.
16. Kowsari, K.; Heidarysafa, M.; Brown, D.E.; Meimandi, K.J.; Barnes, L.E. Rmdl: Random Multimodel Deep Learning for Classification. In Proceedings of the 2nd International Conference on Information System and Data Mining, Lakeland, FL, USA, 9–11 April 2018; pp. 19–28.
17. Heidarysafa, M.; Kowsari, K.; Brown, D.E.; Meimandi, K.J.; Barnes, L.E. An improvement of data classification using random multimodel deep learning (rmdl). *arXiv* **2018**, arXiv:1808.08121.
18. Real, E.; Aggarwal, A.; Huang, Y.; Le, Q.V. Regularized evolution for image classifier architecture search. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 29–31 January 2019; Volume 33, pp. 4780–4789.
19. Kowsari, K.; Brown, D.E.; Heidarysafa, M.; Meimandi, K.J.; Gerber, M.S.; Barnes, L.E. Hdltex: Hierarchical deep learning for text classification. In Proceedings of the 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 18–21 December 2017; pp. 364–371.
20. Kadhim, A.I. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **2019**, *52*, 273–292. [CrossRef]
21. Smith, L.N. A disciplined approach to neural network hyper-parameters: Part 1—Learning rate, batch size, momentum, and weight decay. *arXiv* **2018**, arXiv:1803.09820.
22. Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA Neural Netw. Mach. Learn.* **2012**, *4*, 26–31.
23. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
24. Ray, S. A quick review of machine learning algorithms. In Proceedings of the 2019 IEEE International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 14–16 February 2019; pp. 35–39.
25. Woodworth, B.; Patel, K.K.; Stich, S.; Dai, Z.; Bullins, B.; Mcmahan, B.; Shamir, O.; Srebro, N. Is local SGD better than minibatch SGD? In Proceedings of the International Conference on Machine Learning PMLR, Virtual, 13–18 July 2020; pp. 10334–10343.
26. Joulin, A.; Grave, E.; Bojanowski, P.; Douze, M.; Jégou, H.; Mikolov, T. Fasttext. zip: Compressing text classification models. *arXiv* **2016**, arXiv:1612.03651.
27. Singh, A.; Dargar, S.K.; Gupta, A.; Kumar, A.; Srivastava, A.K.; Srivastava, M.; Kumar Tiwari, P.; Ullah, M.A. Evolving long short-term memory network-based text classification. *Comput. Intell. Neurosci.* **2022**, *2022*, 4725639. [CrossRef]
28. Huang, Y.; Dai, X.; Yu, J.; Huang, Z. SA-SGRU: Combining Improved Self-Attention and Skip-GRU for Text Classification. *Appl. Sci.* **2023**, *13*, 1296. [CrossRef]

29. Liu, Y.; Song, Z.; Xu, X.; Rafique, W.; Zhang, X.; Shen, J.; Khosravi, M.R.; Qi, L. Bidirectional GRU networks-based next POI category prediction for healthcare. *Int. J. Intell. Syst.* **2022**, *37*, 4020–4040. [CrossRef]
30. Cheng, Y.; Yao, L.; Xiang, G.; Zhang, G.; Tang, T.; Zhong, L. Text sentiment orientation analysis based on multi-channel CNN and bidirectional GRU with attention mechanism. *IEEE Access* **2020**, *8*, 134964–134975. [CrossRef]