Proceedings of the 2023 International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/27/20230200

# Rumor detection methodology based on sentiment analysis and the transformer model with decision-level fusion

**Zhengqiao Zhong**

Beijing University of Posts and Telecommunications, Beijing 102200, China

1052892871@qq.com

**Abstract.** The use of transformer models in natural language processing (NLP) has gained significant attention in recent years due to their exceptional performance in various language tasks. This paper explores the application of transformer models in rumor detection, the relevant research on rumor detection, the use of transformer models, and the techniques used to boost the model's performance. Ultimately, the purpose of this paper is to provide insight into the potential of transformer models in detecting rumors on social media. Unlike other rumor-detecting models, the author adds a sentiment analysis model as a supplement to rumor detection. Also, to address the issue of insufficient information in early-stage comments on rumors, this paper proposes a decision-level fusion method before the output layer, which effectively utilizes information from different sources and minimizes the negative impact of insufficient data sources. The early-stage rumor detection accuracy of the model is greatly enhanced by this method, therefore, the article's main contributions can be regarded as follows: First, this paper proposes a combination of an aspect level text sentiment analysis method according to syntactic features, gated recurrent units, and a self-attention mechanism. Experimental findings demonstrate that, compared to the original model without taking the sentiment analysis method into account, the proposed network model has advantages in accuracy and Macro F1 evaluation indexes. Second, a cross-text rumor-detecting method based on Decision-level fusion is proposed. Its advantage is that when the cross-text data source is incomplete and a certain text is missing, another text can be used to continue the analysis. Experimental findings show the effectiveness of this method in improving the accuracy of emotion recognition by integrating data from different modes. Third, a comparison is conducted between the performance of the Transformer-sentiment model and other related models, Text-CNN, Bi-LSTM, etc. The result shows that this integrated Transformer-sentiment model can not only solve the rumor detection tasks at higher accuracy, but can also overcome the shortcomings of the lack of datasets, which means that the model is more robust, and is able to detect rumors at the early stage of the rumor spreading process.

**Keywords:** rumor detection, transformer, sentiment analysis, decision level fusion.

## 1. Introduction

Rumors and false information are widespread on social media and online platforms, and they can cause significant harm, including panic, misinformation, and even violence. Therefore, detecting and

DOI: 10.54254/2755-2721/27/20230200

combating rumors is becoming increasingly important, and natural language processing (NLP) has become a critical tool in this area.

Detecting rumors can be a challenging task, as rumors are often ambiguous and difficult to be distinguished from facts. Rumors can be based on incomplete or misleading information, and they can be intentionally spread to manipulate public opinion. Furthermore, the spread of rumors can be amplified by social media algorithms, making it difficult to track their origins and spread. Therefore, detecting rumors requires sophisticated NLP techniques that can identify the linguistic features that distinguish rumors from factual information.

This paper explores the application of transformer models in rumor detection to utilize the overall advantages. Besides, compared with previous rumor detection methods, this method incorporates the emotional features of Weibo comments, uses the Bi-LSTM + Attention model to simulate human emphasis on key information when reading, enhances the model's sensitivity to emotional key information, and extracts the emotional features of Weibo texts. Ultimately, this paper combines the model with the Decision Level Fusion strategy, which can utilize the data from different resources. Even though lacking some of the resources, the strategy can keep analyzing based on the rest of the resources and is more robust to the data.

## 2. Literature review

In the last few years, researchers have proposed multiple methods and techniques to detect rumors in social networks. In this response, an overview of relevant research will be provided on rumor detection in social networks. At present, rumor detection includes two development stages: early rumor detection on a basis of traditional machine learning and rumor detection on a basis of the deep neural network model.

In the early stage of rumor detection approaches, most of the related works are carried out based on classical machine learning. Castillo et al. [1] extracted four types of features manually from data on the Twitter platform. Qazvinian et al. [2] classified and labeled collected data to verify the effectiveness and differentiation of shallow text content features, microblog elements, and behavior features in detecting rumors. They then built Bayes classifier and ensemble classifier models.

Traditional machine learning-based rumor detection methods aims to find a feature set with high discriminative power and to conduct feature processing. Thanks to the fast progress made in deep learning, research on rumor detection has gradually shifted from the rumor events themselves to more granular text attribute features, and model methods have also begun to use deep network learning models that can obtain higher-level feature levels.

Several deep learning models for rumor detection have been presented by researchers to address the drawbacks of conventional techniques that require a lot of manual labor. Recurrent neural networks (RNNs) were utilized by Ma et al. [3] to model reposts and extract hidden representations from time-series content data. RNNs and autoencoders were combined to create an unsupervised deep learning model by Chen et al. [4]. Convolutional neural networks (CNNs) were utilized by Yu et al. [5] to create a rumor detection model. The benefits of CNNs, gated recurrent units (GRUs), and vectorized Weibo data were coupled by Li et al. [6].RNNs, however, struggle with parallelization and vanishing gradient issues. Although LSTM and GRU can ease the vanishing gradient problem, they cannot address it fully. The CNN-based rumor detection model lacks a grasp of features, and the Transformer has a number of advantages over the models outlined above that make it particularly useful for NLP applications. Attention mechanism, parallelization, multi-head attention, and lower memory needs are its four main benefits. Attention mechanism means that the model can selectively attend to the most relevant information for a given task, rather than relying on a fixed window of context like CNNs or LSTMs. Parallelization makes the Transformer much faster to train than LSTMs. Multi-head attention allows it to learn multiple representations of the input sequence in parallel. Lower memory requirements mean the Transformer has lower memory requirements than LSTMs because it does not need to store the hidden state for each input position. Instead, it computes the attention weights for each input position on the fly, which allows it to process longer input sequences without running out of memory.

DOI: 10.54254/2755-2721/27/20230200

The use of transformer models in NLP has gained significant attention in recent years due to their exceptional performance in various language tasks. Wu et al. [7] used Context-Aware Self-Attention Networks as the feature extraction layer of the model. The introduction of context-aware self-attention networks has achieved pioneering results on multiple data sets.

## 3. Rumor detection methodology based on sentiment analysis and transformer model with decision-level fusion

In this part, the emotion detection model based on Rumor Detection Method based on Sentiment Analysis and Transformer Model with Decision-Level Fusion will be introduced in detail in terms of data preprocessing, model structure, and objective optimization methods. The specific process of the method in this paper is shown in Figure 1.

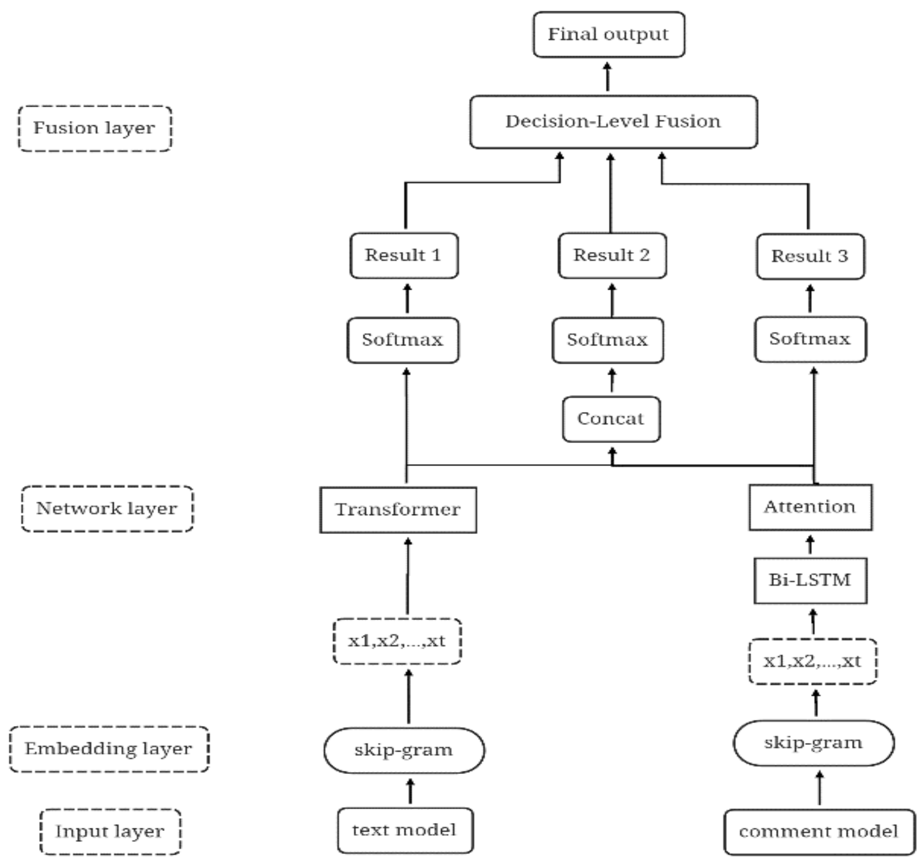


**Figure 1.** The overall structure of the model.

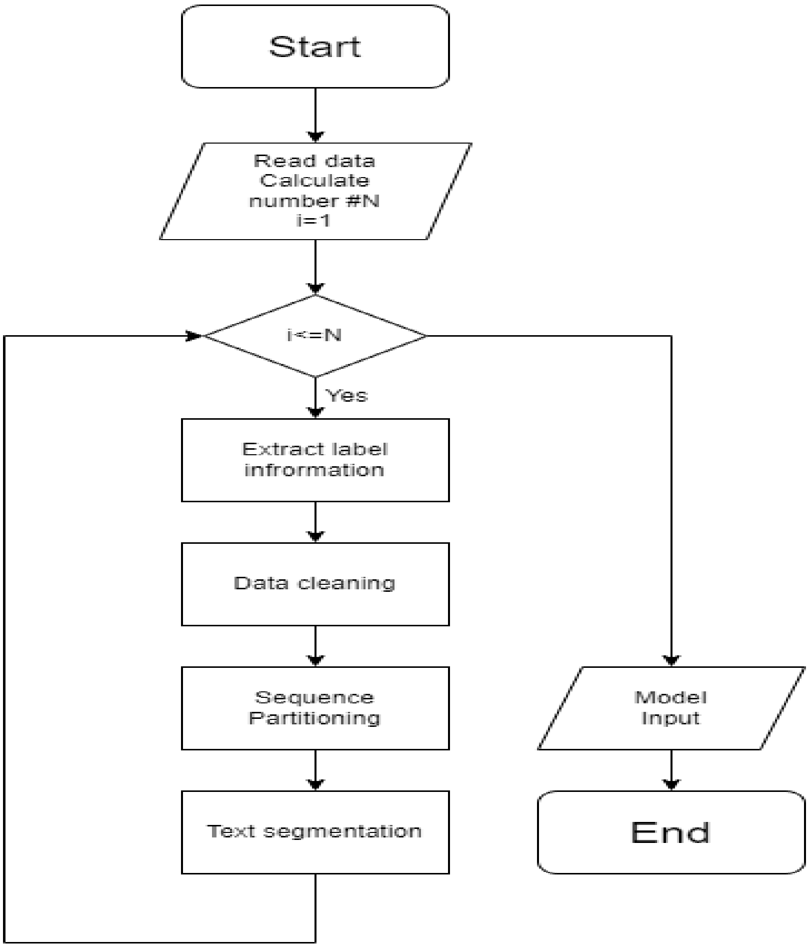


**Figure 2.** The flow chart of data preprocessing.

*3.1. Input layer*

The author chooses Weibo, an online platform which is popular in China, to get sufficient text messages, and complement the Rumor detection task.

Micro-blogs on the Weibo platform are usually short text messages with a certain word limit, so the source blogs contain limited text features. This paper refers to the data processing method of Ma et al. [3], and uses the sequential information of the source blog, such as forwarding and comment, to conduct modeling.

Suppose there is a data set M={M}, and each blog event has a series of related forwarding and comment, denoted as $M=\{(m_i, t_i)\}$, where $m_i$ represents a text message of forwarding or comment; $t_i$indicates the timestamp corresponding to the forward or comment. The objective of rumor detection is to predict the category label $y \in \{0,1\}$ for each blog M based on its forwarding sequence $\{(m_i, t_i)\}$ as well as the blog text {M} itself. In this paper, y=1 is defined as real information and y=0 as rumor.

*3.2. Embedding layer*

The length of the forwarding sequence of different event blogs is different. Therefore, it is essential to preprocess the data, aggregate and divide the text information, and finally process it into vector data suitable for model input.

*3.2.1. Sequence partitioning.* To solve the issue of inconsistent lengths of blog sequences of different events, this paper introduced the degree of aggregation N to represent the degree of aggregation of blog

DOI: 10.54254/2755-2721/27/20230200

sequences. Then transform M = $\{(m_i, t_i)\}$ into F = $\{f_i\}$, and represent $f_i = \{m_{i*N}, \cdots, m_{(i+1)*N}\}$ as a new sequence after text aggregation. The value of N is set appropriately with the model approach adopted.

*3.2.2. Text segmentation.* In this paper, the Chinese text is divided by using the Jieba word segmentation tool. Before the word segmentation, punctuation marks and illegal characters are removed from the text.

*3.2.3. Language model.* After word segmentation, it is necessary to convert the data of text symbol into that of word vector required by model input. This paper adopts the Skip-gram model proposed by Mikolov et al. [8] to learn the distribution representation of words. The Skip-gram model models a posteriori probability from target words to context and predicts the occurrence probability of any word $w_t$ in its context words respectively through the target word $w$.

*3.2.4. Data preprocessing process.* Figure 2 explains the data processing process in detail.

1. Label information of "text" and "time" corresponding to each event in the blog data set is extracted.
2. The "text-time" of each event is processed separately and data cleaning is completed.
3. The cleaned text is divided into words, and the text sentences are divided into word sequences by the tool of "word segmentation".
4. The word sequence is transformed into the corresponding word vector sequence.
5. The divided new sequences are aggregated, and words in the same sequence are embedded for sum and then average. In this way, word vectors can be transformed into sentence vectors, and the final input data of the model is obtained.

According to the above design, the corresponding data preprocessing flow chart is obtained, as shown in Figure 2.

*3.3. Text semantic features extraction based on transformer*

In this part, the Transformer model is used to identify whether the blog text is a rumor or not. The key idea behind the transformer is self-attention, which allows the model to weigh the importance of different parts of the input sequence when generating output representations.

After the word-embedding process, the input tokens are embedded and a series of transformer layers are passed through to generate a final representation for the input sequence. Every transformer layer consists of two sub-layers, namely multi-head self-attention and position-wise fully connected feed-forward networks. These sub-layers are connected by residual connections and layer normalization, allowing the model to learn complex interactions between the input tokens.

The multi-head self-attention sub-layer is the key component of the transformer. Given an input sequence of token embeddings X = [x1, x2, ..., xt], the self-attention mechanism first generates a query matrix Q, a key matrix K, and a value matrix V by applying learned linear transformations to X:

$$\begin{aligned} Q &= XW_q \\ K &= XW_k \\ V &= XW_v \end{aligned} \tag{1}$$

where Wq, Wk, and Wv are learned weight matrices. The query, key, and value matrices are then used to compute an attention score between each token in the sequence and all other tokens:

$$\text{Attention}\ (Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{dk}}\right)V \tag{2}$$

where dk is the dimensionality of the key vectors, and Softmax is applied row-wise to ensure that the attention weights sum to 1. The resulting attention scores are used to weight the value vectors, which are then summed to produce the output of the self-attention sub-layer:

$$\text{Multihead (Q,K,V)= Concat (head1, head 2,..., headh) WO} \tag{3}$$

DOI: 10.54254/2755-2721/27/20230200

where h is the number of attention heads.

$$\text{headi} = \text{Attention (QWiQi,KWik,VWiv)} \quad (4)$$

where WO is another learned weight matrix.

The position-wise fully connected feed-forward network sub-layer applies a non-linear activation function to each element in the output of the self-attention sub-layer, followed by another linear transformation:

$$\text{FFN}(\text{x}) = \max(0, \text{xW1} + \text{b1})\text{W2} + \text{b2} \quad (5)$$

The output of the position-wise feed-forward sub-layer is then combined with the output of the self-attention sub-layer using residual connections and layer normalization:

$$\text{LayerNorm (x+ Sublayer (x))} \quad (6)$$

where Sublayer(x) represents either the self-attention or the position-wise feed-forward sub-layer.

The input sequence can be represented in increasingly complicated and abstract ways by piling additional transformer layers on top of one another. The widely used Transformer encoder [9] is made up of 6 similar layers. Every layer has two separate sub-layers: a multi-head attention layer and a feed-forward neural network layer. This study continues to employ the 6-layer encoder arrangement.

### *3.4. Comments sentiment features extraction based on Bi GRU+Attention*

Weibo data includes not only the Weibo post text, but also the important data of Weibo comments. Compared to non-rumors, rumors contain more questioning and refuting information in the comments. Therefore, learning the emotional features of Weibo comments can help improve the accuracy of rumor detection models. The attention mechanism is introduced to give more attention to emotional words. In Weibo comments, not all words play an equally important role in expressing emotions, so important emotional words should be focused on. Adding an attention mechanism enables the model focus more on emotional information in comments.

This article extracts emotional features through the Bi GRU + Attention model. By utilizing the emotional preferences of commenters towards Weibo events, this method improves the accuracy of rumor detection.

First, transform the vectors which have been formed in the word embedding process as a 3D tensor of shape (batch_size, time_steps, embedding_size), which represents a feature sequence of comments that are used to compute the attention weights.

Next, apply a Bidirectional GRU layer to the input tensor:

$$h_{\text{it}} = \left[\overline{h_{\text{it}} h_{\text{it}}}\right] \quad (7)$$

$$\overrightarrow{h_{\text{it}}} = \overrightarrow{\text{GRU}}(\text{x}_{\text{it}}), \overleftarrow{h_{\text{it}}} = \overleftarrow{\text{GRU}}(\text{x}_{\text{it}}) \quad (8)$$

This layer returns a 3D tensor gru_out of shape (batch_size, time_steps, hidden_size), where hidden_size is the number of units in the GRU layer. The bidirectional aspect of this layer means that it processes the input sequence in both forward and backward directions and concatenates the outputs.

Next, define a trainable weight vector W of shape (hidden_size,) and apply a hyperbolic tangent activation function to the output of the Bidirectional GRU layer.

$$\text{u}_{\text{it}} = \tanh\left(\text{W}_{\text{w}}\text{h}_{\text{it}} + \text{b}_{\text{w}}\right) \quad (9)$$

Then compute a tensor new u by performing a matrix multiplication between the reshaped u tensor and the reshaped W tensor. This multiplication reshapes u to have shape (batch_size*time_steps, hidden_size) and W to have shape (hidden_size, 1), and the result is a tensor of shape (batch_size*time_steps, 1).

Reshape new u to have shape (batch_size, time_steps), and apply a softmax function to restore u to obtain a tensor alpha of the same shape.

$$\alpha_{it} = \frac{\exp(\boldsymbol{u}_{it}^{\mathrm{T}}\boldsymbol{u}_w)}{\sum_t \exp(\boldsymbol{u}_{it}^{\mathrm{T}}\boldsymbol{u}_w)} \tag{10}$$

This alpha tensor represents the attention weights for each time step of the input sequence. Then compute a weighted sum of the output of the Bidirectional GRU layer using alpha.

The result is a tensor of shape (batch_size, hidden_size, 1).

$$\boldsymbol{H}_i = \sum_t \alpha_{it}\boldsymbol{h}_{it} \tag{11}$$

Then the tensor is reshaped to have shape (batch_size, hidden_size) and a dropout layer is applied. This outputs tensor represents the final output of the attention layer, which will be passed to subsequent layers in the model.

*3.5. Cross-text rumor detecting method based on decision-level fusion*

In previous parts, the text semantic feature extraction and comment sentiment feature extraction have been implanted, then it is time to combine the 2 parts to implement the rumor detection task. Decision-level fusion, also known as late fusion, involves learning and training different sources of data separately to obtain recognition results for each source, and then combining them to make a final decision. Even if one modality is missing, the analysis can continue to utilize data from other modalities, making it highly adaptable. Compared to directly concatenating feature vectors, decision-level fusion can select more suitable feature extractors and classifiers for different sources to obtain better local decision results. Unlike simply concatenating the output vectors in 3.3 and 3.4, the author uses the decision-level fusion method, which is not only on a basis of the result of single content. The structure of the Decision-level fusion network can be defined in Figure 3.

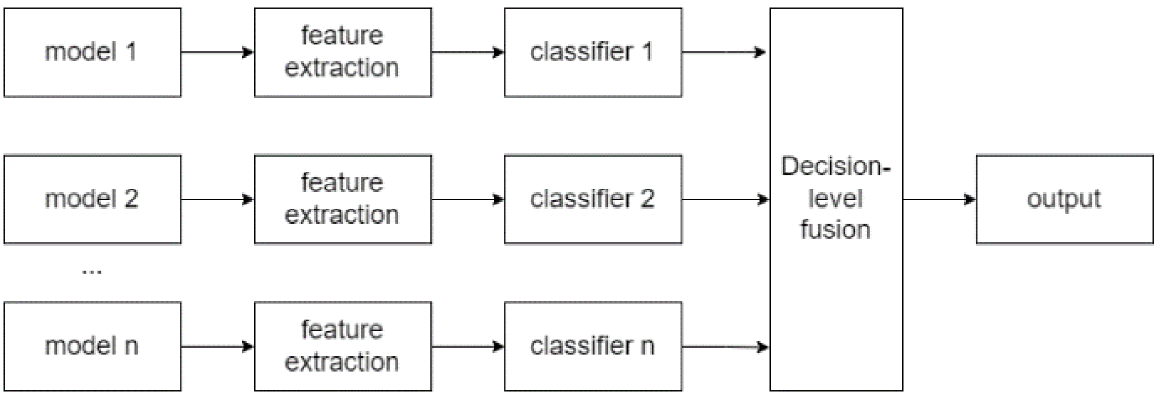


**Figure 3.** The structure of the Decision-level fusion network.

As mentioned earlier, the decision-level fusion approach can choose more suitable feature extractors and classifiers for different modal data. Considering that different sources of data have different contributions to the final discrimination, a weighted fusion method is adopted to assign different weights. Another major advantage of decision-level fusion is that multi-modal data sources are not necessarily complete.

$$\mathrm{P} = \begin{cases} \mathrm{P^t} & \text{input: text} \\ \mathrm{P^i} & \text{input: comment} \\ \mathrm{w_1P^i + w_2P^t + (1 - w_1 - w_2)P^c} & \text{input: (text, comment)} \end{cases} \tag{12}$$

DOI: 10.54254/2755-2721/27/20230200

## 4. Experiment, results and analysis

### 4.1. Dataset of the experiment

The experimental data in this article comes from the Sina Weibo platform, and all the rumor and non-rumor event data are in Chinese. The dataset contains 4,664 events, of which 2,313 are marked as rumor events and 2,351 are non-rumor events. Some statistical data of the dataset are shown in Table 1.

**Table 1.** Specific content of the Weibo dataset.

| Weibo Dataset | Rumors | Non-rumors |
|---|---|---|
| Quantity | 2,313 | 2,351 |
| Number of retweets | 2,090,743 | 1,661,716 |
| Minimum number of retweets | 10 | 10 |
| Maximum number of retweets | 59,318 | 52,157 |
| Average number of retweets | 805 | 708 |
| Minimum time span for retweets (min) | 3 | 1 |
| Maximum time span for retweets (h) | 28,095 | 27,682 |
| Average time span for retweets (h) | 2,344 | 1,028 |

### 4.2. Experimental environment and parameter settings

In the dataset, the training set to test set ratio is 3:1. The experiment is carried out using the Adam optimizer and a learning rate of 0.01, both of which are based on the Tensor Flow framework. Table 2 displays the particular model parameters.

**Table 2.** Specific parameter settings of the model.

| Model parameters | Value |
|---|---|
| embedding_size | 300 |
| time_steps | 1000 |
| Number of Transformer encoder | 1 |
| Number of "heads" in a multi-head self-attention mechanism | 6 |
| hidden_num of GRU units | 128 |
| dropout | 0.3 |
| epoch | 20 |

### 4.3. Results and analysis

*4.3.1. Results of the Transformer-Bi GRU+Attention Decision Level Fusion Model.* Plot the curve of the model accuracy during the training and testing process.

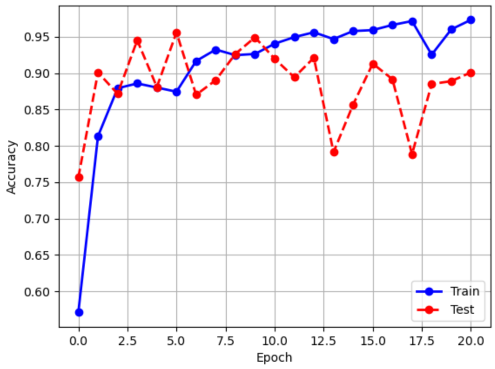


**Figure 4.** The model's training and testing performances.

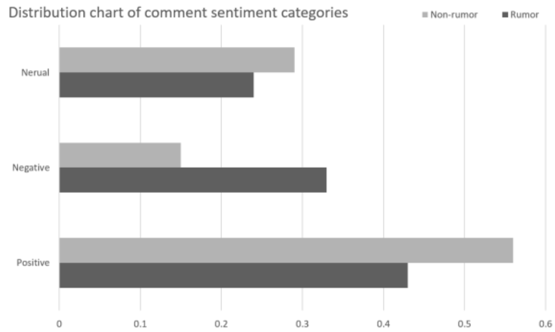


**Figure 5.** The distribution of comment sentiment categories.

According to the experimental result analysis, this model can identify rumors on Weibo effectively, converge from a small number of iterative trainings, and achieve high accuracy. As can be seen in Figure 4, before 10 epochs, the accuracy on the test dataset peaks at 95%, with a minimum value of 87%. After 10 epochs, although the model seems to be overfitting, the paper tries the early-stop method to get the model with better performance in the early epoch. After evaluating the accuracy, precision, recall, and F1 score, the author saves the model as well as its parameters during the 8th epoch, where the model has an accuracy of 92.5% in the training set and 92.7% in the testing set.

*4.3.2. The influence of comment sentiment features on rumor detection results.* To verify the effect of comment sentiment features on rumor detection results, the Bi GRU+Attention model is used as a sentiment classifier to label the sentiment tendency in rumor, non-rumor, positive, negative, and neural. The result is shown in Figure 5. From the data in Table 3, it is evident that the negative emotions in rumors are much higher than those in non-rumors. In addition, the proportion of positive emotions in rumors is lower than those in non-rumors. The research results indicate that, compared with non-rumors, comments on rumors tend to express more negative emotions, such as anger, sadness, and suspicion. This is the comparison of results between the Transformer model and the Transformer-Bi GRU+Attention model. The Transformer-Bi GRU+Attention model adds the sentiment features of Weibo comments based on the Transformer model. The accuracy of this model reached 92.7%, which is 1.1% higher than the Transformer model. This indicates that the sentiment features of Weibo comments are an important indicator that significantly improves the performance of rumor detection.

**Table 3.** The comparison of results between 2 models.

| Model | Accuracy | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| Transformer | 0.9156 | 0.8963 | 0.9211 | 0.9148 |
| Transformer-Bi GRU+Attention | 0.9269 | 0.9301 | 0.9212 | 0.9256 |

*4.3.3. The influence of decision level fusion method on early rumor detection.* As mentioned above, Decision level fusion can greatly enhance the accuracy when lacking datasets, in other words, it satisfies the background of a rumor's early propagating process. In this experiment, the model input data was set to a proportion of the total data for early rumor detection from the perspective of the spread process of Weibo events. It should be noted that, in order to better simulate the spread of Weibo, all the text datasets and the comment datasets are retained t in increasing proportions. Based on the experimental results, Table 4 is organized and a curve is plotted. Results demonstrate that the model can achieve an 85% identification accuracy by using only 15% of the entire event propagation data of Weibo, indicating that

DOI: 10.54254/2755-2721/27/20230200

the model constructed in this study can quickly detect and identify relevant Weibo rumors after the event outbreak.

**Table 4.** The model's performances with different inputs.

| Percents | Accuracy | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| 5% | 0.7838 | 0.7914 | 0.8541 | 0.8216 |
| 15% | 0.8535 | 0.8517 | 0.8675 | 0.8595 |
| 25% | 0.8744 | 0.8936 | 0.8907 | 0.8921 |
| 35% | 0.8875 | 0.9002 | 0.9125 | 0.9063 |
| 45% | 0.8889 | 0.8973 | 0.9230 | 0.9100 |
| 55% | 0.8950 | 0.9143 | 0.9257 | 0.9200 |
| 65% | 0.9125 | 0.9216 | 0.8759 | 0.8982 |
| 75% | 0.9183 | 0.9329 | 0.9148 | 0.9238 |
| 85% | 0.9245 | 0.9310 | 0.9089 | 0.9198 |
| 95% | 0.9253 | 0.9230 | 0.9356 | 0.9293 |

### *4.4. Model comparison*

Several representative rumor detection models and deep learning models were selected for comparison with the model proposed in this study. As machine learning rumor-detecting models, the DTC model and the SVM-RBF model are representative. The DTC model is proposed by Castillo et al. and they constructed a J48 decision tree classifier [1]. The SVM-RBF model is proposed by Yang et al. [10]. Table 5 shows the comparison of the results. The performance of conventional machine learning techniques is subpar when compared to approaches on a basis of deep neural networks. As a deep leaning rumor-detecting model, the experiment reproduces some of the typical composite models, and compares their performances with the current one. The Text CNN-GRU composite model and the GRU-2 model are selected and the parameters are optimized to feed them with the current dataset.

**Table 5.** The comparison with classical machine learning models.

| Model | Accuracy | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| DTC | 0.8312 | 0.8475 | 0.8156 | 0.8311 |
| SVM-RBF | 0.8185 | 0.8224 | 0.8123 | 0.8179 |
| Transformer-Bi GRU+Attention | 0.9269 | 0.9301 | 0.9212 | 0.9256 |

Specifically, the GRU-2 model [3] uses the TF-IDF method to represent each time period's text as a vector. Besides, the accuracy results of the Text CNN-GRU model can be concluded in Figure 5:

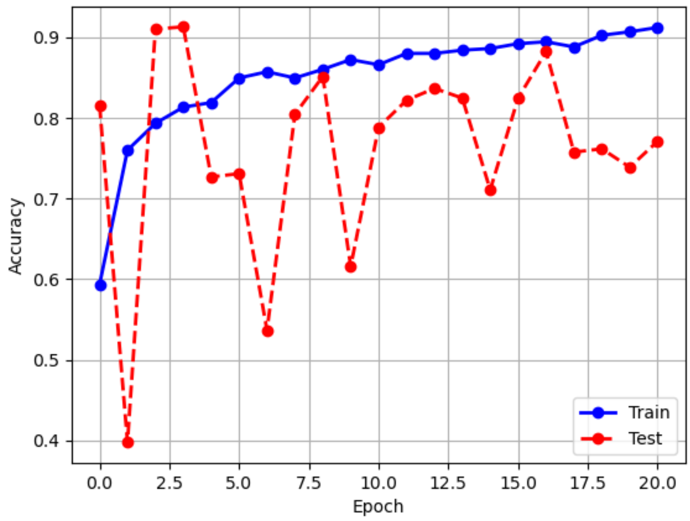


**Figure 5.** The Text CNN-GRU model's training and testing process.

**Table 6.** The comparison with deep learning models.

| Model | Accuracy | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| GRU-2 | 0.9103 | 0.8762 | 0.9561 | 0.9142 |
| Text CNN-GRU | 0.8992 | 0.8895 | 0.9103 | 0.8999 |
| Transformer-Bi GRU+Attention | 0.9269 | 0.9301 | 0.9212 | 0.9256 |

In summary, in comparison with the best model in the baseline methods, the method proposed in this paper improved the accuracy by 1.6 percentage points. The experimental findings demonstrate that the rumor detection method proposed in this study has the following characteristics: for rumor identification, the Transformer model is superior to convolutional neural networks and recurrent neural networks. The model with the addition of emotional features from Weibo comments is superior to the model that only focuses on the text of Weibo posts. By adding a decision layer fusion method, the model can achieve better detection results with less data, and can more effectively deal with the early detection problem of rumors.

## 5. Conclusion

Using sentiment analysis, the Transformer model, and the decision-level fusion model, this article suggests a technique for Weibo rumor detection. The approach fully relies on the attention mechanism to model the global dependency relationship between inputs and outputs, using the Transformer model to learn deep semantic information from Weibo content. For the important feature of the stance, viewpoint, and personal sentiment expressed by commenters in Weibo comments, the Bi GRU + Attention model is used to learn and explore the user's sentiment preference with bidirectional information flow. At the fusion layer before the output layer, instead of simply concatenating feature vectors, this article designs a decision-level fusion model that automatically assigns weights according to the input. Results demonstrate that the rumor detection model proposed in this article achieves higher accuracy compared to the currently best benchmark method on Weibo dataset, demonstrating the feasibility and effectiveness of the model for rumor detection on social media. Additionally, the model proposed achieves good results in early detection of Weibo rumors.

The shortcomings and improvements of this article can be summarized as follows: (1) In terms of word embedding model design, the static skip-gram model used in this article is a relatively traditional approach. In the future, dynamic pre-training models such as BERT and XLNET can be used, and parameters can be modified based on downstream task objectives to improve model performance. (2) In terms of sequence partitioning and input data processing, this article only performs simple time or interval partitioning. Future research can focus on designing more scientific and reasonable sequence partitioning algorithms. Additionally, to meet the input data requirements, this article only calculates the weighted sum of word vectors in the same sentence and takes the average to obtain a sentence vector. (3) This article uses the traditional cross-entropy loss function for classification task optimization in terms of optimization goals. Based on the design of the model structure, future research can examine more reasonable and multi-level optimization objectives. (4) To assess the effectiveness of the model in early rumor detection, more exacting and reasonable evaluation criteria can be sought after. Furthermore, it is not accurate to model the early dissemination of rumors as an increase in the input data volume. The early rumor detection model can be improved by taking into account the time stream by extracting the time information from reposts and comments.

**References**

[1] Castillo, C., Mendoza, M. and Poblete, B. Information credibility on Twitter [C]. Proceedings of the 20th International Conference on World Wide Web. New York: ACL, 675-684 (2011).

[2] Qazvinian, V., Rosengren, E. and Radev, D. R., et al. Rumor has it: Identifying misinformation in microblogs [C]. Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. Edinburgh: ACL, 1589-1599 (2011).

[3] Ma, J., Gao, W. and Mitra, P., et al. Detecting rumors from microblogs with recurrent neural networks [C]. International Joint Conference on Artificial Intelligence, 3818-3824 (2016).

[4] Chen, T., Li, X. and Yin, H. Z., et al. Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection [C]. Proceedings of the 2018 Pacific-Asia Conference on Knowledge Discovery and Data Mining, 40-52 (2018).

[5] Yu, F., Liu, Q. and Wu, S., et al. A convolutional approach for misinformation identification [C]. International Joint Conference on Artificial Intelligence, 3901-3907 (2017).

[6] Li, L. Z., Cai, G. Y. and Pan, J. The method of rumor detection on Weibo based on C-GRU [J]. Journal of Shandong University (Engineering Edition) 49(02), 102-106+115 (2019).

[7] Wu, Z. and Ong, D. C. Context-guided bert for targeted aspect-based sentiment analysis [C]. AAAI Conference on Artificial Intelligence [Online] 35(16), 14094-14102 (2021).

[8] Mikolov, T., Chen, K., Corrado, G. and Dean. J. Efficient estimation of word representations in vector space. International Conference on Learning Representations (ICLR) (2013).

[9] Vaswani, A., Shazeer, N. and Parmar, N., et al. Attention is all you need [C]. Proceedings of the 31st International Conference on Neural Information Processing Systems, 6000-6010 (2017).

[10] Yang, F., Liu, Y. and Yu, X. H., et al. Automatic detection of rumor on Sina Weibo [C]. Proceedings of the 2012 ACMSIGKDD Workshop on Mining Data Semantics (2012).