Weibo Text Sentiment Classification Model Based on FastText-BERT-Attention

Jianming Zhang*
School of Management, Xi'an Polytechnic University, Xi'an, Shaanxi, China
**Corresponding Author.*

Abstract: This paper introduces a Weibo text sentiment classification model that integrates FastText, BERT, and an attention mechanism, aiming to overcome the limitations of traditional models in processing social media data. By leveraging FastText's efficient word-level feature extraction capabilities, BERT's deep semantic understanding, and the feature fusion advantage of the attention mechanism, this model significantly enhances the accuracy of Weibo text classification. Experimental results show that compared to Word2Vec, FastText, and BERT models, the FastText-BERT-Attention model proposed in this paper demonstrates higher precision, recall, and F1 scores in the sentiment binary classification task, proving its effectiveness and superiority in handling large-scale short text data from Weibo comments. This study not only presents theoretical innovation but also exhibits superior performance in practical applications, making it particularly suitable for processing short text data from social media platforms like Weibo.

Keywords: FastText, BERT, Attention, Text Sentiment Classification

1. Introduction

Currently, text sentiment classification based on deep learning primarily relies on powerful pretrained models such as BERT, as well as efficient word representation methods like FastText. These methods have achieved significant accomplishments in semantic understanding and context modeling. As a pretrained model based on Transformers, BERT excels in understanding the deep semantics and contextual relationships of texts. However, the main limitation of BERT lies in its high computational complexity and substantial resource requirements, which may become a bottleneck in some resource-constrained application scenarios. FastText is known for its efficient training process and its ability to handle short texts well, especially in capturing local (word-level) features. However, it is relatively weaker in capturing long-distance dependencies and complex semantic information. In tasks like microblog text classification, models that combine BERT and FastText can effectively compensate for the shortcomings of individual models and enhance classification performance.

2. Literature Review

Research on sentiment analysis of Weibo texts typically involves the use of text sentiment analysis techniques to classify sentiments in Weibo comments, followed by an in-depth analysis of the public's emotional characteristics in online public opinion. The commonly used sentiment analysis methods can be categorized into three types.

2.1 Research on Sentiment Classification Based on Sentiment Dictionary

Dictionary-based analysis relies on specific domain sentiment dictionaries. It determines the overall emotional orientation of the text by searching for emotional words within the text and calculating the weighted sum of these words' emotional polarity and intensity. For example, Wan [1] demonstrated the effectiveness of this approach by translating Chinese comments into English and then analyzing the emotional polarity of the translated text using English resources. Yang et al. [2] expanded existing dictionaries by assigning emotional intensities based on word similarity, while Yang et al [3] developed a sentiment dictionary for hotel reviews using an improved PMI algorithm.

2.2 Research on Sentiment Classification

Based on Machine Learning

Machine learning-based analysis does not depend on pre-constructed sentiment dictionaries but instead learns the emotional features of texts through algorithms. This method requires manual annotation of the emotional polarity of training data. Pang et al. [4] showed the application of different machine learning models in film review classification, while Kang et al. [5] and Yang et al. [6] proposed an improved Naive Bayes algorithm and an SVM model utilizing textual features, respectively.

2.3 Research on Sentiment Classification Based on Deep Learning

Deep learning-based analysis can automatically extract text features and identify deeper semantic information. Rhanoui et al. [7]'s model, based on CNN and BiLSTM, is suitable for long text sentiment analysis, while Chen et al. [8] and Shen et al. [9] combined BERT with CNN or BiLSTM for short text processing. Singh et al. [10] used BERT to analyze sentiments towards COVID-19 on Twitter, and Basiri et al. [11]'s model integrated BiGRU, BiLSTM, and attention mechanisms, effectively handling texts of varying lengths.

The overall process of text classification can be broken down into the following steps: Firstly, preprocess the text data, including test data and training data, filtering out meaningless feature words and tag vocabulary, etc.; Then, proceed with feature selection and extraction to carry out text learning, obtaining an ideal model trained by the classifier; Finally, input the test data into the model, evaluate and feedback on the results to further optimize the model. harmila, Devi, and Devi [12] reviewed the use of deep learning in sentiment analysis, highlighting its strength in feature extraction from texts and sentiment prediction, and its role in understanding social media sentiments for business decisions. Wang, Liu, Fang, and Li [13]explored deep learning in social media sentiment analysis, noting its superiority in extracting complex features for better performance than traditional methods. They also discussed unimodal and multimodal sentiment analysis and future research directions. Chen and Fnu [14] focused on Aspect-Based Sentiment Analysis (ABSA) and deep learning's advantages in capturing textual features without complex feature engineering. Zhao [15] provided an overview of deep learning advancements in sentiment analysis, especially Transformer and pre-trained models, their benefits, limitations, and the challenges and future directions in sentiment analysis research, highlighting the potential for accuracy improvement.

Up to now, sentiment classification algorithms for microblog texts both domestically and internationally have achieved relatively good development, but there are still areas in need of improvement: In the domain of long texts, models trained by FastText contain more redundant information, which can easily affect classification accuracy; A lack of sufficient data can also lead to underfitting in models trained by FastText. Although the BERT model considers contextual semantics, it has drawbacks such as a large number of parameters during training, leading to slow training convergence speed and long training times, as well as the possibility of overfitting when dealing with short texts like microblog comments.

3. Model Framework

The integration of FastText with the deep learning model BERT, renowned for its revolutionary progress in various Natural Language Processing (NLP) tasks, offers a compelling solution for the challenges faced in processing social media data like Weibo comments. BERT's approach of pretraining on a vast corpus to learn rich language representations, followed by fine-tuning for specific tasks, enables it to achieve high performance across different tasks. However, BERT's effectiveness in understanding complex semantic relationships comes with certain limitations when dealing with social media data, such as a limited capacity to handle non-standard language and neologisms, constraints in processing long texts, and issues related to its numerous parameters, slow convergence, and potential overfitting on short texts. The model architecture is shown in Figure 1.

FastText, on the other hand, is particularly suited for large-scale text data due to its rapid training and classification speeds, making it highly efficient for processing vast amounts of Weibo text, especially in resource-constrained environments. Weibo comments often contain

non-standard usage, neologisms, and internet slang. FastText's capability to handle such vocabulary through n-gram features and subword information, even for words not seen during the pretrained stage, complements BERT's context-sensitive features. This combination allows for a more comprehensive understanding and representation of text content. By leveraging n-gram and subword information, FastText effectively addresses rare words and neologisms, which are particularly prevalent in texts like Weibo that feature a large amount of new words and internet slang.

Incorporating the strengths of FastText into our model enhances its ability to process and classify Weibo text sentiment more accurately and efficiently. This synergistic approach combines the depth of BERT's contextual understanding with FastText's agility in handling dynamic, informal text types found in social media, offering a robust framework for sentiment analysis and other NLP tasks within the context of Weibo and similar platforms.

The overall structure of the model for sentiment classification of Weibo texts, integrating FastText and BERT with an attention-based feature fusion, comprises the following modules, each serving a distinct function within the process:

3.1 Preprocessing Module

This module is responsible for the initial handling of input text. It involves data cleaning and the segmentation of text into 2-gram sequences to be processed by the FastText encoding module.

3.2 FastText Encoding Module

By applying 2-gram processing to the input text, this module leverages word vectors and employs an averaging method to accumulate these vectors, resulting in word-level representations. This approach facilitates the capturing of local context and subword information, enhancing the model's ability to handle rare words, neologisms, and short text peculiarities effectively.

3.3 BERT Encoding Module

Utilizing a pretrained BERT model, this module extracts contextually relevant word vectors and sentence vectors from the input text. It provides deep semantic understanding and captures the nuances of language use, including the relationships between words and sentences within texts.

3.4 Feature Filtration Module

Employing a multi-head attention mechanism, this module performs weighted selection and filtering of word-level features generated by both BERT and FastText. The objective is to emphasize features critical to the classification task and reduce the impact of noise, thereby enhancing the model's focus on semantically significant elements.

3.5 Feature Fusion Module

Another layer of multi-head attention mechanism facilitates the deep interaction and integration of word-level and sentence-level features from both FastText and BERT. This process yields a comprehensive hidden layer representation that encapsulates the combined strengths of both encoding methods, ensuring a richer and more nuanced feature set for classification.

3.6 Classification Prediction Module

The final hidden layer representation is fed into a fully connected layer, which maps the integrated features onto the predicted categories. During the training phase, this module also computes the loss, guiding the optimization and adjustment of model parameters to improve classification performance.

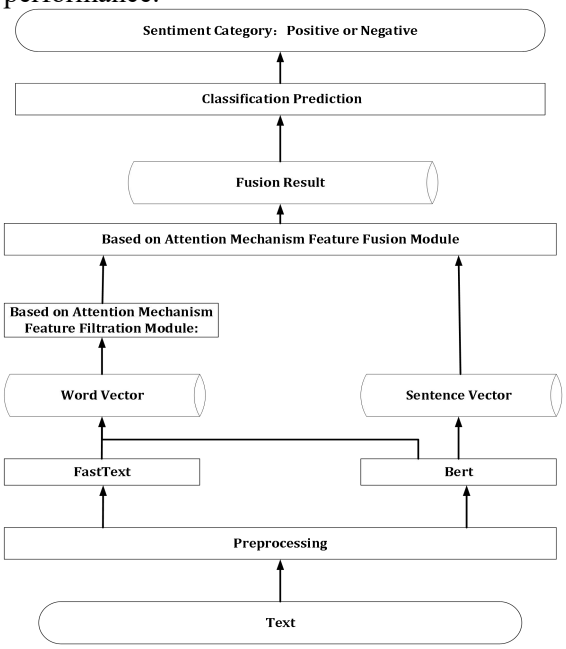

Figure 1. Model Framework

4. Model Implementation

4.1 Data Preprocessing

4.1.1 Data collection

Weibo is one of the main platforms for expressing opinions among Chinese netizens, with a wide user base. Since people mostly express their views in text form on Weibo, data is relatively easy to obtain. Therefore, this study takes the "Double Reduction" policy event as an example, selecting comments from Weibo for research. According to the characteristics of the event, this paper used Octopus software, with "Double Reduction Policy" as the keyword, to crawl original Weibo posts and user information from July 23, 2021, to March 27, 2022. After filtering out posts unrelated to the "Double Reduction" policy, a total of 52,393 posts were collected.

4.1.2 Data cleaning

After collecting the data, the next step involves data cleaning, which is crucial for ensuring the data inputted into the model is as clean and standardized as possible. This process enhances the learning efficiency and performance of the model. The cleaning process includes several steps: first, deleting invalid Weibo posts; second, removing duplicate content posted by the same user; and third, eliminating HTML tags, URLs, @username mentions, and emoticons from the Weibo text. After completing the data cleaning process, the remaining samples are compiled to construct the dataset, which contains a total of 41,993 sample entries.To add n-gram features, it is necessary to split the input text into 2-gram form, meaning each pair of adjacent Chinese characters forms one token:

$$x_{2-gram} = \{t_{12},\ t_{23},\ \ldots,\ t_{(n-1)n}\} \quad (1)$$

4.2 FastText

The FastText encoding module uses the features of n-grams from sentences as input (with n set to 2 in the experiments), and its model layers are relatively shallow. Thus, it focuses on learning the local features of individual characters, while its capability to model contextual representations is weaker. Specifically, for the input 2-gram tokens, this module first obtains the word vector for each token through word vector embedding:

$$w_{i(i+1)} = Embed(t_{i(i+1)}) \in R^d \quad (2)$$

Where d represents the dimension of the embedded word vector. For two adjacent 2-gram word vectors, their average word vector is calculated, serving as the word vector for the intermediate character:

$$v_i = \frac{w_{i(i-1)}+w_{i(i+1)}}{2}, i \in (1, n) \quad (3)$$

Finally, the word vector sequence obtained from FastText is:

$$V_{word-fasttext} = \{v_1,\ v_2,\ \ldots,\ v_n\} \quad (4)$$

4.3 Bert

The BERT module's text processing workflow encompasses the following steps:

Embedding Layer: Tokens are transformed into vector form through the embedding layer. This step includes not only word vector embeddings but also position embeddings and segment embeddings to capture the order of words and the relationships between different sentences.

Transformer Encoder: Next, these embedded vectors are fed into BERT's Transformer encoder layers. Through multiple layers of self-attention mechanisms and feed-forward neural networks, the model is capable of capturing complex dependencies between tokens.

Inter-layer Connections and Normalization: Each Transformer layer also incorporates residual connections and layer normalization, which help prevent gradient vanishing or exploding problems during the training process.

Output: The model outputs word vectors and sentence vectors.

4.4 Feature Filtering Module Based on Multi-head Attention Mechanism

The Attention mechanism is inspired by the human ability to automatically focus on certain aspects of observation, calculating attention scores to automatically enhance certain important features while diminishing less significant ones. This process aids in extracting deeper representations or facilitating the interaction and integration of features. In the previously mentioned FastText encoding module and BERT encoding module, the models compute word vectors from FastText, focusing on local representations, and BERT word vectors and sentence vectors, emphasizing sentence context and medium to long-range dependencies. In this module, the attention mechanism is utilized to combine the

n-gram features of FastText with the deep semantics of BERT.

Firstly, the word vectors generated by FastText are subjected to a linear transformation to adjust their dimensions to match the output size of BERT.

A linear transformation is typically represented by matrix multiplication and the addition of a bias term. For the word vector embedding $x_{1-\text{gram}}$ the linear transformation can be specifically expressed as:

$$x_{transformed} = W_{trans} \cdot x_{1-gram} + b_{trans} \quad (5)$$

In this, W_{trans} is the weight matrix of the linear layer, and b_{trans} is the bias term. The purpose of this transformation is to adjust the word vectors to match the dimensionality of the BERT layer outputs.

Secondly, the multi-head attention mechanism is used to combine the linearly transformed FastText embeddings (serving as Q and V) with the outputs of BERT (serving as K and V). Through this method, the model is able to integrate two types of information: the local textual features provided by FastText and the global semantic understanding provided by BERT.

The basic formula calculated by an attention "head" is:

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{softmax}(\frac{\text{QK}^{\text{T}}}{\sqrt{\text{d}_{\text{k}}}})\text{V} \quad (6)$$

For multi-head attention, we split this computation into multiple heads:

$$\text{MultiHead}(\text{Q}, \text{K}, \text{V}) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots \text{head}_{\text{n}})\text{W}^{\text{O}} \quad (7)$$

Where each head_{i} is:

$$\text{head}_{\text{i}} = \text{Attention}(\text{QW}_{\text{i}}^{\text{Q}}, \text{K}W_i^K, VW_i^V) \quad (8)$$

Where $\text{W}_{\text{i}}^{\text{Q}}$, $\text{W}_{\text{i}}^{\text{K}}$, and $\text{W}_{\text{i}}^{\text{V}}$ are dedicated linear transformations for each head, and there's another linear transformation applied to the concatenated output of all heads.

4.5 Attention-based Feature Fusion Module

First, take the sentence vector obtained from BERT as the Queryq, and use the word vectors obtained from FastText and BERT after encoding and weighting through the multi-head attention mechanism as KeyK and Value. The attention scores are calculated as follows:

$$\alpha = softmax(\frac{q \cdot K}{\sqrt{d}}) \quad (9)$$

$$q = v_{sent-bert} \quad (10)$$

$$K = [V^*_{word-fasttext}; V^*_{word-bert}] \quad (11)$$

Where d represents the dimension of the model's hidden layer, and [;] denotes the vector concatenation operation. After obtaining the attention scores, use these weights to perform a weighted summation of the word vectors, and then concatenate the original BERT sentence vector to obtain the fused feature vector f:

$$out_{attention} = \alpha \cdot K \quad (12)$$

$$f = [out_{attention}; V_{word-bert}] \quad (13)$$

4.6 Sentiment Classification and Loss Function

The module first passes the fused feature vector through a fully connected layer (also known as a linear layer), where the output dimension matches the number of target categories. Next, an activation function (Softmax) is applied to convert the output of the linear layer into the predicted probability for each category.

The final model's predicted distribution is obtained as follows:

$$Y^{\sim} = softmax(Wf + b) \quad (14)$$

$W \in \text{R}^{\text{d}*\text{k}}$, $b \in \text{R}^{\text{k}}$ represents trainable parameters, and K denotes the number of categories. The category with the highest probability will be selected as the final predicted category by the model:

$$y^{\wedge} = argmaxY_i^{\sim} \quad (15)$$

Finally, cross-entropy is chosen as the loss function, as shown in equation (16). Additionally, a regularization coefficient is included during training to apply regularization constraints and thus prevent overfitting.

$$L = -\sum_{i=1}^{l} y_i \, log\left(y_i^{\wedge}\right) + \lambda \|\theta\|^2 \quad (16)$$

Here, l represents the number of sentiment categories in the training set, yi is the true sentiment distribution for the ith category, $\hat{y}i$ is the predicted sentiment for the ith category, λ represents the regularization coefficient, and θ denotes the regularization parameters.

5. Dataset and Parameter Settings

5.1 Dataset Annotation

Next, the corpus is annotated into two categories: non-negative and negative. For this experiment, two postgraduate students from the School of Foreign Studies at Xi'an Jiaotong University were invited, with their main research focus on language and semantic recognition. Within 14 working days, each comment in the corpus, categorized under negative and non-negative emotional polarities, was annotated. Comments with consistent

annotation results were formally included in the corpus. For comments with inconsistent results, a university teacher holding a certificate in psychological counseling was invited for a second round of annotation. Based on the annotation results, the majority (2:1) determined the final emotional polarity, and comments with disputed results were excluded. Finally, 18,876 comments with valid annotations were obtained, including 11,431 positive sentiment comments and 7,445 negative sentiment comments. Of these, 80% of the annotated datasets were used as the training set, and the remaining 20% as the test set.

5.2 Parameter Settings

The experimental setup for this paper is as follows:

The parameter settings for the model in this paper are: learning rate (lr) = 0.001, maximum epochs = 1000, with an early stopping mechanism implemented. Training will automatically stop when there is no improvement in test results for 10 consecutive epochs. This is to prevent overfitting and deterioration in generalization performance in the later stages of training. A smaller learning rate is chosen to allow the model to thoroughly learn textual features; a higher number of epochs is selected to prevent premature termination of iterations due to insufficient data, which could lead to biased experimental data.

Learning Rate and Training Cycles: The learning rate (lr) is set to 1e-6. This smaller learning rate helps the model adjust weights more precisely during iterations, allowing for a deeper learning of textual features. The training is set for 100 epochs to ensure that, despite limited data, the model can fully learn and adapt to the training data.

Early Stopping Mechanism: To prevent overfitting and maintain good generalization ability, an early stopping mechanism is set. If the model's performance does not significantly improve over 10 consecutive batches, training will automatically stop. This helps avoid ineffective training and overfitting.

Word Embedding Dimension: The dimension of word embeddings (embed) is determined by the pretrained word embeddings. If there are no pretrained embeddings, the default dimension is set to 300. This dimension aims to balance the feature expression capability and model complexity.

Hidden Layer Configuration: The size of the intermediate hidden layers in the model is set to 768 dimensions, used for linear transformation layers to match the output of the BERT model. Additionally, the model includes 8 attention heads, with a dropout rate of 0.4 for the multi-head attention mechanism.

Use of BERT Model: For Chinese text, the pretrained bert-base-chinese model is used, which has a hidden layer dimension of 768 and contains 12 attention heads, suitable for a deep understanding of the Chinese context.

5.3 Comparative Experimental Analysis

In this paper, commonly used metrics in text classification tasks, such as precision, recall, and the harmonic mean (F1-Score), are employed to measure the effectiveness of the model. Precision is the proportion of samples correctly identified as positive by the model among all samples identified as positive; recall is the proportion of samples correctly identified as positive by the model among all actual positive samples; and the F1 score is the harmonic mean of precision and recall, used to consider both precision and recall comprehensively.

To demonstrate the effectiveness of the model proposed in this paper, a comparative analysis was conducted with the Word2Vec, FastText, and Bert models. The comparative experimental results are as follows:

Table 1. Sentiment Binary Classification Experimental Results

Model	Precision	Recall	F1-score
Word2veb	0.6736	0.6466	0.6598
FastText	0.8736	0.7736	0.8206
Bert	0.9012	0.8056	0.8507
FastText-Bert-Attention	0.9783	0.886	0.9299

From the Table 1, it can be observed that compared to other models, the FastText-BERT-Attention model proposed in this paper shows some improvement in sentiment classification effectiveness.

In the sentiment binary classification task, compared to the Word2Vec, FastText, and Bert models, the accuracy of the model proposed in this paper, trained on the online public opinion dataset, improved by 30.47%, 10.47%, and 7.71%, respectively; the recall

improved by 23.94%, 11.24%, and 8.04%, respectively; and the F1-Score improved by 20.7%, 10.93%, and 7.92%, respectively.

In this study, the attention mechanism effectively integrates the features of the FastText and Bert models, enhancing the classification effect.

6. Conclusions

The FastText-BERT-Attention model, by integrating FastText's efficient word-level feature extraction with BERT's deep semantics, and utilizing the attention mechanism for effective feature capture and fusion, significantly improves the accuracy of Weibo text classification. This model is innovative not only theoretically but also demonstrates superior performance in practical applications compared to other models, making it well-suited for handling large-scale short text data from Weibo comments.

References

[1] Wan X. Using bilingual knowledge and ensemble techniques for unsupervised Chinese sentiment analysis. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. 2008.553-561.

[2] Yang Chao, Feng Shi, Wang Daling, et al. Analysis of Online Public Opinion Orientation Based on Sentiment Dictionary Expansion Technology. Journal of Mini & Micro Computer Systems, 2010, 31(04): 691-695.

[3] Yang A M, Lin J H, Zhou Y M, et al. Research on Building a Chinese Sentiment Lexicon Based on SO-PMI. Applied Mechanics & Materials, 2013, 263-266:1688-1693.

[4] Pang B, Lee L, Vaithyanathan S .Thumbs up? Sentiment Classification using Machine Learning Techniques.2002 [2024-03-13] .

[5] Kang H, Yoo S J, Han D. Senti-lexicon and improved Nave Bayes algorithms for sentiment analysis of restaurant reviews. Expert Systems with Applications, 2012, 39(5): 6000-6010.

[6] Yang Shuang, Chen Fen. Research on Multi-Level Sentiment Classification of Weibo Based on SVM and Multi-Feature Fusion. Data Analysis and Knowledge Discovery, 2017, 1(02): 73-79.

[7] Daeli N O F, Adiwijaya A. Sentiment analysis on movie reviews using information gain and K-nearest neighbor. Journal of Data Science and Its Applications, 2020, 3(1): 1-7.

[8] Chen Tao, An Junxiu. Research on Weibo Short Text Sentiment Classification Based on Feature Fusion. Frontiers of Data and Computing, 2020, 2(06): 21-29.

[9] Chen Zhiqun, Ju Ting. Research on the Orientation Analysis of Weibo Comments Based on BERT and Bidirectional LSTM. Information Theory and Practice, 2020, 43(08): 173-177.

[10] Singh M, Jakhar A K, Pandey S. Sentiment analysis on the impact of coronavirus in social life using the BERT model. Social Network Analysis and Mining, 2021, 11(1): 1-11.

[11] Basiri M E,Nemati S, Abdar M et al. ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. Future Generation Computer Systems, 2021, 115: 279-294.

[12] K. Sharmila, N. S. Devi and R. Devi, "Survey on Sentiment Analysis using Deep Learning," 2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART), Moradabad, India, 2022, pp. 1434-1438

[13] Chandrasekaran G, Hemanth J .Deep Learning and TextBlob Based Sentiment Analysis for Coronavirus (COVID-19) Using Twitter Data. International Journal of Artificial Intelligence Tools: Architectures, Languages, Algorithms, 2022(1):31.

[14] S. Chen and G. Fnu, "Deep Learning Techniques for Aspect Based Sentiment Analysis," 2022 14th International Conference on Computer Research and Development (ICCRD), Shenzhen, China, 2022, pp. 69-73

[15] Zhao, Y. (2023). Overview of Deep Learning Methods for Sentiment Analysis. Advances in Engineering Technology Research

