Dean&Francis

ISSN 2959-6130

Text Comment Classification and Sentiment Analysis Model Based on LSTM

Chengbin Huang

GuangDong Neusoft university
Email: fei2589693633@163.com

Abstract:

This study addresses the challenge of sentiment analysis and text classification in user-generated comments, an essential task for understanding public opinion in the digital age. With the widespread adoption of social media, the sheer volume of text data has increased exponentially, thereby necessitating advanced analytical tools. The study integrates Long Short-Term Memory (LSTM) networks with attention mechanisms to develop a model adept at both classifying and analyzing the sentiment of textual comments. The research methodology commences with data cleaning, tokenization, and feature extraction, subsequently followed by the implementation of a bidirectional LSTM architecture augmented with an attention mechanism. The model underwent rigorous training and testing on a comprehensive dataset, yielding high accuracy rates in sentiment analysis and text classification. The results underscore the model's proficiency in capturing sentiment nuances and underscore its potential for deployment in e-commerce and content recommendation systems, emphasizing the practical implications of this research for enhancing decision-making processes in businesses and among policymakers.

Keywords: LSTM, Long Short-Term Memory, Sentiment Analysis Text Classification Attention Mechanism Deep Learning Neural Networks Comment Classification Data Cleaning

1. Introduction

In recent years, with the widespread use of the internet and social media, the number of user-generated comments and posts on social media has surged dramatically. These text data contain a vast amount of emotional information. By conducting sentiment analysis on comments in various comment fields, enterprises, governments, scientific research institutions, and others can quickly and accurately understand the public's attitudes and emotional tendencies towards a product, event, or policy, thereby making better decisions. Moreover, sentiment analysis of comments also contributes to the targeted delivery of online advertising and the optimization of personalized recommendation systems.

In China, many research teams are actively engaged in research on comment sentiment analysis, with a particular focus on Chinese comment sentiment analysis. Researchers typically employ methods such as Chinese word segmentation, sentiment dictionaries, and machine learning. In 2018, [1]Li Ming proposed a Chinese comment sentiment analysis method based on convolutional neural networks (CNN). They used the CNN model to extract features from the text and trained the model using a collected set of comment data, employing algorithms like backpropagation and stochastic gradient descent for parameter optimization. Additionally, they integrated sentiment dictionaries for sentiment polarity classification, achieving good results. In mid-2019, [2] Wang Lili conducted research targeting the diversity and ambiguity of language expression in Chinese comment sentiment analysis and proposed a sentiment analysis method based on deep learning. They adopted the CNN model and constructed the training model through multiple layers of convolution and pooling operations. By training on large-scale text data, they improved the model's generalization ability and the effectiveness of sentiment analysis. In foreign research, Kim Y[3] developed a TextCNN (Text Convolutional Neural Network) model for sentence-level classification tasks. The model captures contextual information from the text using windows of varying sizes. However, TextCNN has a fixed field of view, and convolution and pooling operations can render positional information unreliable, making it challenging to identify sentiment in sentences where meaning changes with order. The Transformer architecture rectifies this by introducing positional vectors at the input, which consider the influence of position on semantics.

Zhang X[4] et al. investigated the application of convolutional neural networks in sentiment analysis, studying the CNN model's performance on sentiment classification tasks for English and Chinese text, and proposing several improvement methods to enhance accuracy. Jin W[5] et al. combined the convolutional neural network (CNN), the Long Short-Term Memory Network (LSTM), and the BERT (Bidirectional Encoder Representations from Transformers) model for Chinese long-text classification.

2. Theoretical Basis

To achieve the LSTM-based text classification and sentiment analysis model, the following techniques can support the experiments in this study.

2.1 Data Cleaning in the Dataset

Text comments have a strong complexity and heterogeneity. Without standardizing the processing, it is easy for the model to fail to fit the dataset well. Current research methods for this include text cleaning, tokenization, stopword removal, and feature extraction. Text cleaning primarily involves removing unimportant, heterogeneous characters or emojis, or HTML tags from the text, as these words or emojis do not provide much assistance in training. Tokenization is the process of segmenting sentences into words. In a long sentence, many words are intertwined, and we need to separate them to better conduct comment analysis and sentiment analysis. Stopword removal refers to removing common words like 'and', 'or', 'and' from the text, which do not have much practical meaning. Feature extraction transforms text comments into digital vectors that computers can read and understand, allowing for better prediction analysis and capturing contextual relationships and sentiment tendencies.

2.2 RNN

RNN, as a commonly used model in neural networks, has significant advantages in processing sequential data. The most prominent application is in natural language processing. RNN, with its loop structure, can better handle data of a certain length and pass information between sequences, allowing the model to better understand the meaning of sentences. Therefore, in paragraph-level analysis applications, RNN has significant advantages. In addition, RNN also benefits from parameter sharing and temporal variability. Compared to traditional feedforward neural networks, its advantage lies in its ability to store past information and achieve a deep understanding of dynamic inputs. This capability to process historical information in sequences represents RNN's significant advantage in such applications. RNN excels in capturing temporal features in data, especially in short-term contextual relationships. However, when dealing with long-span sequence information, its performance is often limited by the inherent transient memory constraints. This structure is inadequate in capturing long-term dependencies, leading to unsatisfactory performance in long sequence processing tasks. During training, RNN models are prone to gradient vanishing or gradient explosion problems, which further exacerbates the performance bottleneck in long-term memory construction. The RNN network structure is illustrated in Figure 1.

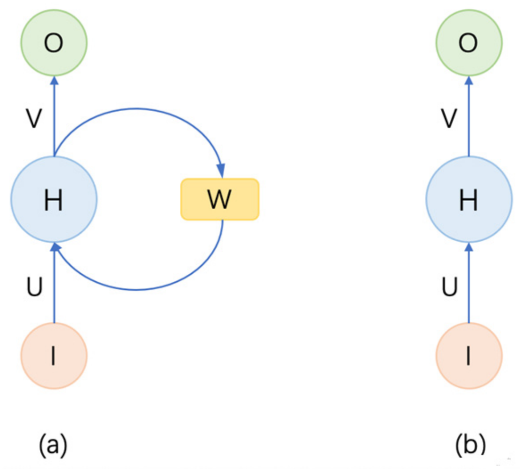


Figure 1 Illustration of a Recurrent Neural Network (RNN) Structure

2.3 LSTM

To address the limitations of RNN in handling long-span text, the Long Short-Term Memory (LSTM) network was developed. LSTM is an improved variant of the recurrent neural network designed to overcome the challenges that regular RNNs face when processing sequences of long durations, such as gradient vanishing and gradient exploding. The most significant difference between LSTM and regular RNN is the introduction of a gating mechanism. This mechanism significantly enhances the network's ability to capture long-distance, long-term dependencies in text.The gating mechanism in LSTM involves three gates: the forget gate, the input gate, and the output gate. These gates are the key to LSTM's efficiency compared to regular RNNs. The collaboration of these gate units allows LSTM to selectively maintain or update the information flow at each time step, enabling strategic information management.

Forget Gate: The forget gate evaluates which existing information should be forgotten at the current time step. It does this by analyzing the combination of the current input and the previous hidden state, producing a coefficient between 0 and 1 to control the retention of previous memories.

Input Gate: The input gate determines how the current input should be incorporated into the new memory. It updates the memory cell based on the instructions from the forget gate and the input gate, in conjunction with the memory content from the previous time step, to determine the current state of the memory cell.

Output Gate: The output gate determines the output at the currenttime step based on the state of the current memory cell and the input information.

These ingenious gating structures ensure that LSTM performs exceptionally well when handling various sequence data, especially in scenarios involving long-term dependencies. In contrast to traditional RNN models, which are limited by their inability to effectively learn long-distance dependencies due to time span, LSTM provides an effective solution. This is why LSTM has achieved remarkable success in many practical applications, such as natural language processing, speech recognition, and time series prediction.

The specific LSTM network structure is illustrated in the following diagram:

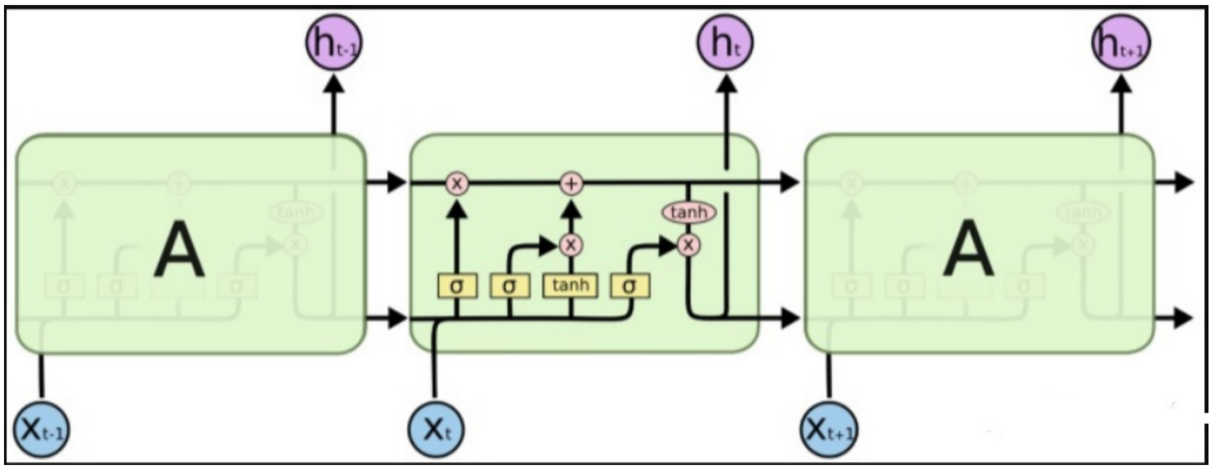


Figure 2 Illustration of the LSTM Network Structure.

3. Experimental Results and Analysis

3.1 Experimental Environment and Model Parameter Configuration

The experiment was conducted on a Windows 10 operating system, utilizing an Intel® Core™ i7-11800H processor, 16 GB of memory, and an NVIDIA Geforce RTX 3060 graphics card. Code was written using Pycharm, with Python 3.8.0.19 as the programming language. The deep learning framework pytorch 2.0 with CUDA 11.8 was employed for model construction and training. The embedding layer was initialized with pre-trained word vectors (dimension of 50), and the option to update the word vectors during training was available. A bidirectional LSTM structure was used, with a hidden layer dimension of 100 and 2 layers. Dropout (retention probability of 0.8) was implemented to prevent overfitting. The LSTM_attention model employed attention mechanisms. The text comments were used for classification, while a regular LSTM with the same configuration was used for sentiment analysis, utilizing a two-layer fully connected network for final classification, outputting a probability distribution for 10 categories. Parameter tuning was performed using the Adam algorithm, which is renowned for its efficient convergence and outstanding computational efficiency. This algorithm intelligently adjusts the parameter update mechanism to accurately guide the loss function towards its minimum, greatly enhancing the model's fitness to the training data and aiming to optimize the core goal of the model.

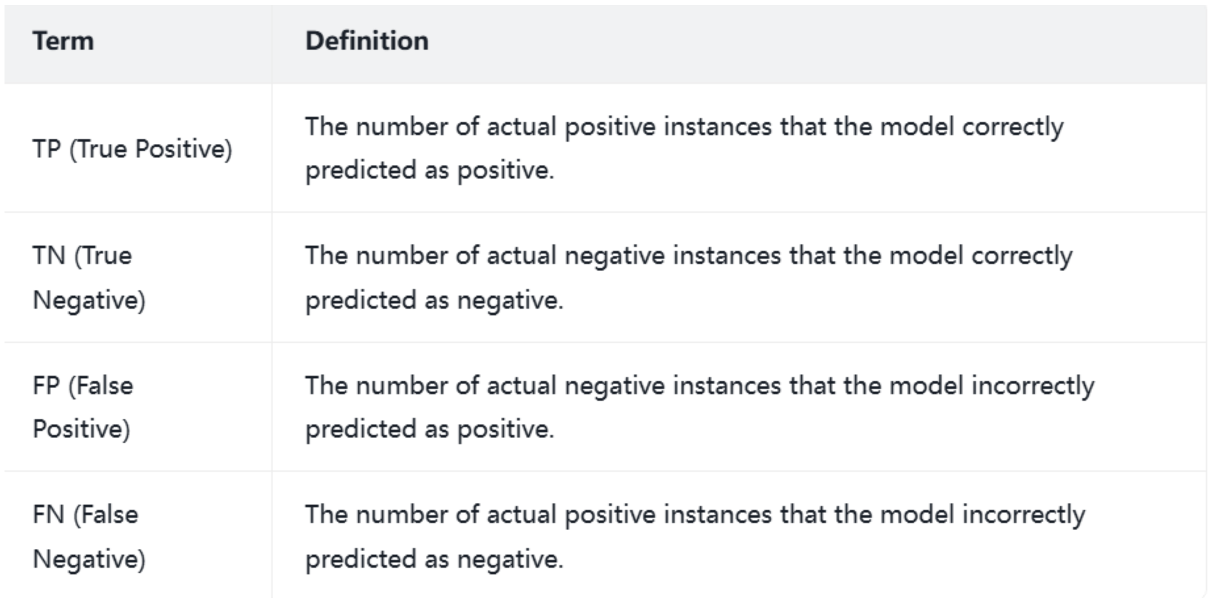
3.2 Model Evaluation Metrics

In general, the evaluation metrics for NLP tasks are ACC (Accuracy) and Loss value, as these two values provide a straightforward and convenient way to assess the model. The ACC formula is as shown below.

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

3.2.1 The formula for Accuracy.

The meanings of TP, TN, FP, and FN are represented as follows.



Term	Definition
TP (True Positive)	The number of actual positive instances that the model correctly predicted as positive.
TN (True Negative)	The number of actual negative instances that the model correctly predicted as negative.
FP (False Positive)	The number of actual negative instances that the model incorrectly predicted as positive.
FN (False Negative)	The number of actual positive instances that the model incorrectly predicted as negative.

Figure 3 The meanings of the parameters

The Loss value is also an important numerical indicator for evaluating our model. In this study, the loss function used is the cross-entropy loss function. By observing the trend of the loss value and the loss value itself, we can intuitively judge the training effect of our model and whether the model has been trained to completion. The cross-entropy function calculates the loss value by accumulating the cross-entropy between the output values and the true values, as shown in the formula below.

$$J = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} log\left(\widehat{y^{(i)}}\right) + \left(1 - y^{(i)}\right) log\left(1 - \widehat{y^{(i)}}\right)\right]$$

3.2.2 Cross-Entropy Loss Function

Where X represents the predicted results, Class represents the labels, and the number of predicted results should match the number of data points. The value of Class should correspond to the label, for example, if the label is 2, then Class is 2. In natural language processing, this function is a very commonly used loss function. It provides the optimization objective for model training and also helps us better evaluate and optimize the model by comparing the model's output with the true labels.。

3.3 Model Training Results

Through multiple modifications of the epoch and learning rate parameters, this study found that on this dataset, when the epoch, or the number of training iterations, is set to 10 and the learning rate is 0.001, the model achieves the highest accuracy. The sentiment analysis model achieves an accuracy of 95% when analyzing negative reviews, but only 65% when analyzing positive reviews. The reason for this discrepancy is speculated to be due to the imbalanced distribution of the dataset, leading the model to learn too much on one category and fail to well fit the sample distribution of the other category.

In another model, the attention mechanism was used to classify text comments. This model achieves an accuracy of over 95% in categories such as finance, real estate, technology, sports, politics, and society, but only 60% in the remaining four categories of games and education. The possible reasons for the final performance of this model are also analyzed as being due to the imbalanced distribution of the dataset.

Next, we discuss the evaluation metrics of the model.

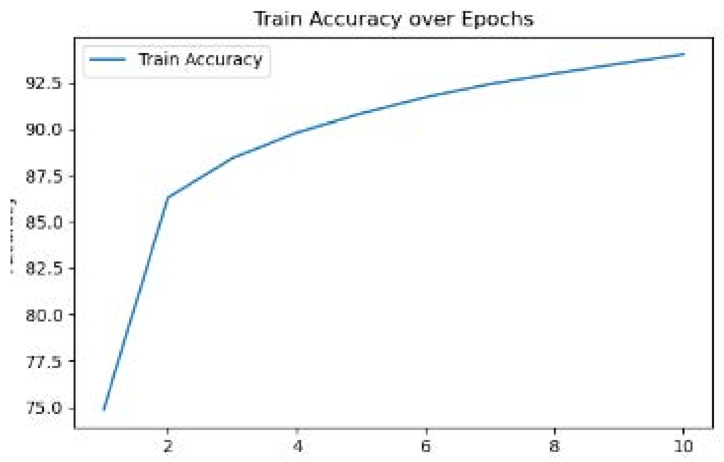


Figure 4 Sentiment Analysis Model Accuracy

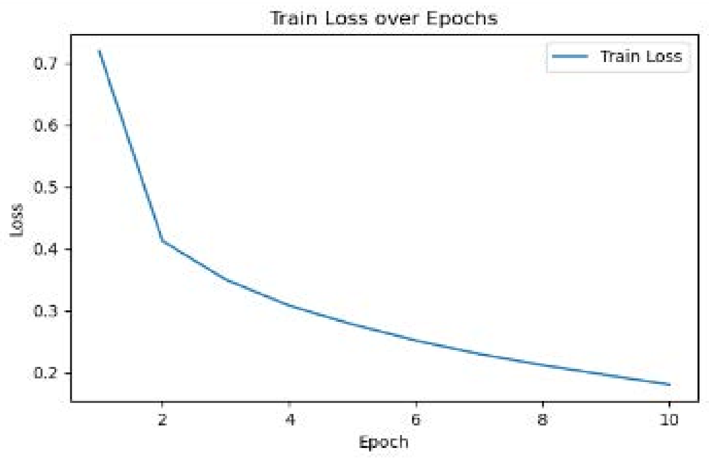


Figure 5 Sentiment Analysis Model Loss Value

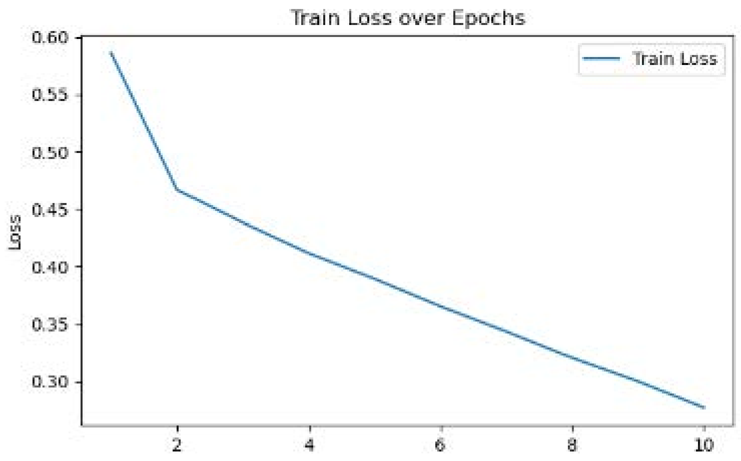


Figure 6 Loss Value of Text Classification Model

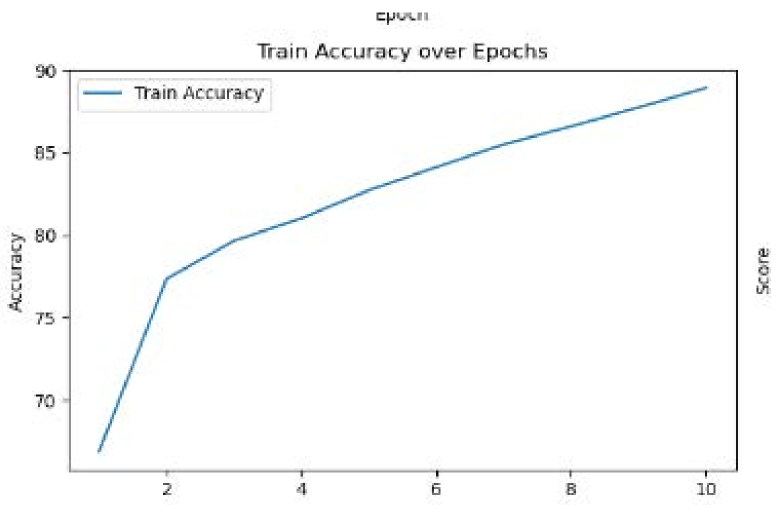


Figure 7 Text Classification Model Accuracy

4. Conclusion

This study integrates text classification with text sentiment analysis and combines the LSTM and attention mechanisms. By integrating existing models, the study effectively enhances the accuracy of text classification and the discriminative ability of fine-grained text sentiment analysis. The results demonstrate that the model performs well in achieving the research objectives and provides a certain degree of assistance in product recommendation on e-commerce platforms and content recommendation on social media, thus possessing practical commercial value.

Compared to traditional text sentiment analysis methods, the integration of the advanced properties of the long short-term memory (LSTM) network within this research demonstrates significant superiority in analyzing sequence data. The attention mechanism, originally devised for the image domain and grounded in human visual attention, has increasingly found application in the text domain. Specifically, in the realm of text sentiment analysis, sentiment-laden words are pivotal to the task. While a standalone LSTM may have limitations in this regard, the marriage of the attention mechanism effectively distills text sentiment features and leverages textual keywords with precision. Consequently, the judicious amalgamation of diverse deep learning methodologies holds immense promise for advancing text sentiment analysis.

Future research and analysis should focus on broader applications and diversify text sentiment analysis to improve the model's generalization ability and accuracy. Exploring the combination with other methods, seeking more effective methods for extracting text sentiment features, and investigating the effect of combining text and image features in sentiment analysis tasks are all worth continuous effort and development.

References

[1] Li Ming, et al. (2018). "Chinese Comment Sentiment Analysis Based on Convolutional Neural Networks." Journal of Computer Research and Development, 55(1), 147-155.

[2] Wang Lili, et al. (2019). "A Sentiment Analysis Method Based on Deep Learning for Chinese Text." IEEE Access, 7, 1179-1188.

[3] Kim Y. (2014). "Convolutional Neural Networks for Sentence Classification." EMNLP, 1746-1751.

[4] Zhang X., et al. (2016). "Deep Learning for Sentiment Analysis: A Survey." Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 6(4), 261-277.

[5] Jin W., et al. (2020). "Combining CNN, LSTM, and BERT for Chinese Long-Text Classification." Expert Systems with Applications, 155, 113516.