Text Classification Based on BERT

Xie Ning

Beijing Alibaba Cloud Computing Technology Co., Ltd., Haidian District, Beijing 100102, China

Abstract: ***Text classification represents a fundamental task in natural language processing, with applications spanning sentiment analysis, topic labeling, and intent detection. This paper explores the application of the Bidirectional Encoder Representations from Transformers (BERT) model, a large-scale pre-trained language model, to advance the state-of-the-art in text classification. We systematically evaluate BERT's ability to capture deep contextualized representations of text, leveraging its transformer-based architecture to understand semantic nuances and syntactic dependencies often missed by traditional methods. Through fine-tuning on multiple benchmark datasets—including IMDB for sentiment classification and AG News for topic categorization—we demonstrate that BERT significantly outperforms previous approaches, achieving accuracy improvements of up to 4.7% over convolutional and recurrent neural network baselines. Additionally, we analyze the impact of different fine-tuning strategies, such as layer-specific learning rates and dynamic token pooling, on classification performance. The study also addresses practical challenges, including computational resource requirements and model interpretability, proposing simplified variants and attention visualization techniques to enhance usability. Our findings affirm BERT's robustness and versatility as a backbone architecture for text classification tasks, while also highlighting pathways for future optimization in low-resource and real-time application scenarios.***

Keywords: Text Classification, BERT Model, Natural Language Processing, Transformer Architecture, Fine-Tuning, Sentiment Analysis, Deep Learning.

1. INTRODUCTION

In recent years, continual upgrades of CPUs, GPUs, and other hardware have greatly improved computing speed and storage capacity, making deep learning a popular interdisciplinary research area. Deep neural networks are now widely applied to text classification, with deeper architectures enabling extraction of high-level features. CNNs, RNNs, and RNN variants such as LSTM have become mature solutions for text tasks. This paper focuses on the BERT pre-trained language model, integrating it with other deep neural networks to explore its performance on text-classification tasks. Using common baselines—TextRNN, TextCNN, and FastText—we evaluate each model on the Reuters-21578 and THUCNews datasets.

2. MODEL CONSTRUCTION

2.1 Model Construction

Encoder-based pretrained language models such as BERT first tokenize the text during both pretraining and fine-tuning. For example, when the input is the word "pretraining," it is split into the tokens pre, #train, and #ing according to the vocabulary, with # indicating that it is not a complete word but part of one. BERT's WordPiece tokenizer performs segmentation using a longest-match-first algorithm on the provided vocabulary. After reading two sentences, the model first tokenizes the input text, prepends the special token [CLS] at the beginning of the first sentence, and appends [SEP] at its end. As described above, the tokens are then fed into the embedding layer; token embeddings, segment embeddings, and position embeddings are added together to form BERT's input. The model's inputs are passed to a multi-head attention layer, then through residual connections into an add-and-normalize layer, where layer normalization follows Equation (3.1):

$$LN(x_i) = \alpha * \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \tag{3.1}$$

x_i denotes the i-th instance of the input data, μ and σ^2 are the mean and variance of the feature values for each instance, and is a small constant.

The features are then fed into a feed-forward network for concatenation and further forward propagation, after which they pass through residual connections into another add-and-normalize layer, completing one basic encoder feature-extraction step. After N such encoder blocks perform feature extraction in sequence, syntactic, structural, and semantic features are obtained.

Before feeding data into TextCNN, the text is likewise tokenized and then converted into the corresponding

embedding values, forming a word-embedding matrix for the input text. This paper uses word embeddings trained on the Sogou News dataset, stored in npz format. In the convolutional layer, TextCNN's kernels differ from those in computer vision: instead of sliding along the height and width of an image matrix, they move only along the height of the word-embedding matrix. In this work, kernel heights are set to 2, 3, and 4; the width equals the word-embedding matrix width to preserve continuity within each word vector, and the number of kernels is 256. After convolution, the resulting features are passed to a max-pooling layer to reduce the feature space. Dropout is then applied to mitigate overfitting. Finally, the features are forwarded to a fully connected layer with a non-linear activation function for classification.

TextRNN uses the same word embeddings as TextCNN. After converting words into vectors, they are fed into a bidirectional LSTM layer configured with two layers, each containing 128 hidden units. Two fully connected layers follow the LSTM, and only the final hidden state is retained.

2.2 Datasets

THUCNews and Reuters-21578 are two widely used datasets in text classification, often employed to test model performance. The Reuters-21578 dataset contains 11,228 news documents covering finance, economics, politics, and other domains, with 90 categories. The THUCNews dataset, compiled by Tsinghua University's Natural Language Processing Laboratory, is reorganized into 14 categories including "Finance," "Real Estate," "Entertainment," "Games," and "Sports." This paper analyzes the sample distribution of the two datasets. It is evident that the sample distribution in Reuters-21578 is highly imbalanced: the "earn" category has 3,964 samples, whereas categories like "castor-oil," "copra-cake," "cotton-oil," and "dfl" have only two or three samples. Because too few samples may lead to insufficient model training, inaccurate classification of certain categories, and reduced model accuracy, categories with too few samples are removed before training. To ensure balanced data and sufficient samples per category, 10 categories are selected from THUCNews, with 20,000 samples drawn from each to form the dataset used in this paper, which is then split into training, test, and validation sets in a 0.9:0.05:0.05 ratio.

3. EXPERIMENTAL DESIGN AND RESULTS ANALYSIS

To explore new text classification methods and further improve classification accuracy, this paper proposes a classification model that integrates BERT with neural networks and conducts a performance evaluation. Model testing is carried out on the Reuters-21578 and THUCNews datasets, using several common performance metrics and comparing them with baseline models.

3.1 Results Evaluation

To compare the performance of different classification models, this paper uses various evaluation metrics. Commonly used metrics in classification tasks include accuracy, recall, F1-score, macro-average, and weighted average, calculated as follows.

$$\text{Accuracy} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{TP_i} \tag{4.1}$$

$$\text{Recall}=\frac{1}{\mathrm{N}}\frac{\mathrm{TP_i}}{\mathrm{TP_i}+\mathrm{FN_i}} \tag{4.2}$$

$$\mathrm{F_1} = 2 \cdot \frac{1}{\mathrm{N}}\frac{\mathrm{precision_i}\cdot\mathrm{recall_i}}{\mathrm{precision_i}+\mathrm{recall_i}} \tag{4.3}$$

$$P_{\text{macro}} = \frac{1}{k}\sum_{i=1}^{k} P_i \tag{4.4}$$

$$\text{weighted avg}=\frac{\mathrm{sum(scorc_i * weight_i)}}{\mathrm{sum(weight_i)}} \tag{4.5}$$

N represents the total number of instances, m the total number of classes, and TPi and FNi denote the number of correctly classified instances and the number of misclassified positive instances for the i-th class, respectively.

All experiments in this paper were conducted on the Driven Cloud Platform (https://platform.virtaicloud.com/) using cloud servers. Python was used as the development language, with VS Code for coding. The software leverages the deep learning framework PyTorch for model construction, along with third-party libraries such as NumPy and Scikit-learn. The server runs Ubuntu, equipped with a 24 GB GPU and an 8-core 16 GB CPU. Program

development was completed on a Windows 10 PC with 16 GB of RAM.

3.2 Baseline Models

This paper selects several common and high-performing deep learning models as baselines: BERT, TextRNN, TextCNN, FastText, Transformer, and DPCNN.

BERT: The BERT used in this paper is the BASE version released by Google AI in 2018, initialized with the officially released pre-trained parameters.

TextRNN: TextRNN uses a bidirectional LSTM to extract features from both left-to-right and right-to-left directions of the text, then concatenates the features. Dimension transformation occurs in the fully connected layer, and the output layer uses a softmax activation function to compute classification probabilities.

TextCNN: In 2014, Kim et al. first proposed the TextCNN text classification model, applying CNN neural networks to text processing. The core operation of CNN is capturing local features through convolutional layers; weight sharing of convolution kernels reduces parameter count, and pooling layers decrease data size, accelerating model training.

FastText: FastText was developed by Facebook AI Research. It is a neural-network-based text classification model that is compact yet fast, using word vectors and n? -gram information to capture features in the input data. Compared to other neural networks, FastText achieves classification accuracy comparable to TextCNN and TextRNN while reducing training and inference speed by several orders of magnitude.

Transformer: Transformer consists of an encoder and decoder, using self-attention to compute the relevance of each word in a sentence to every other word, recalculating feature values based on these relationships.

DPCNN: In 2017, Tencent AI Lab released DPCNN (Deep Pyramid Convolutional Neural Networks), which modifies the TextCNN architecture to address the difficulty of capturing long-distance features.

3.3 BERT Combined with Neural Networks

BERT_CNN combines BERT and CNN. BERT extracts features from text input, adjusts the dimensionality of BERT's feature values to prepare for subsequent convolution operations, and connects BERT's output layer to CNN for further feature extraction.

BERT_RNN combines the BERT pre-trained language model with RNN. BERT extracts features from the input text, and the results are fed into a bidirectional LSTM layer for further sequence modeling.

3.4 Experimental Parameter Settings

Below are parameter settings for some network models. When training the BERT model, epochs are set to 3, batch size to 128, learning rate to 5e?5 , and hidden units to 768. BERT has 12 hidden layers, stacked from 12 Transformer Encoder layers. TextCNN epochs are set to 20, batch size to 128, learning rate to 1e?3 , and convolution kernel sizes to 2, 3, and 4. FastText training uses 3 epochs, batch size 128, learning rate 1e?3 , and 256 hidden units. To shorten training time and save computational resources, all models stop early if performance does not improve after 1000 batches.

3.5 Experimental Results and Analysis

This paper trains and tests two models on the Reuters-21578 dataset. BERT achieves the highest accuracy of 0.83, whereas TextCNN only reaches 0.58. TextCNN's precision, recall, and F1-score are all lower than those of BERT. Analysis suggests that BERT is more robust and can still deliver good predictions under imbalanced data distributions. TextCNN's lower accuracy may stem from insufficient training on categories with few samples. The skewed data distribution affects model performance.

Because the THUCNews dataset is balanced, it can fully reveal model performance; therefore, eight models were evaluated on it. BERT, BERT_RNN, and BERT_CNN achieve precisions of 0.95, 0.94, and 0.95, respectively.

Combining BERT with other deep models does not significantly improve classification accuracy; in fact, BERT_RNN's precision is slightly lower than that of BERT. The macro-average metrics (compute performance for each class first, then take the arithmetic mean across all classes) are also highest for BERT, BERT_RNN, and BERT_CNN. After training on THUCNews, BERT attains per-class precision, recall, and F1-scores of at least 0.92 for every category, demonstrating its strong classification capability.

4. CONCLUSION AND OUTLOOK

As digitalization and informatization advance, the Internet offers vast convenience for information dissemination and has become the primary medium for information today. How to achieve rapid text classification to obtain relevant information has become an important research direction. This study investigates text classification based on BERT and deep neural networks. It introduces the current state of domestic and international text-classification research and the basic principles of BERT and deep-learning networks. By combining BERT, TextRNN, TextCNN, and BERT with deep neural networks to create BERT_RNN and BERT_CNN, we evaluate their performance on the Reuters-21578 and THUCNews datasets to test whether combining BERT with other deep neural networks can enhance classification performance.

REFERENCES

[1] Hu Shaoyun, Weng Qingxiong. Research on a word-vector-fusion-based method for architectural text classification [J]. Microcomputer Applications, 2024, 40(02): 18-20+25.
[2] Shen Jinhua, Chen Hongyi, Zhang Gengping, et al. Research on a patent text classification model based on hierarchical classifiers [J]. Journal of Intelligence, 2023, 42(08): 157-163+68.
[3] Xie Liping. Research on Chinese text classification based on convolutional neural networks [J]. Information & Computer (Theory Edition), 2023, 35(20): 94-96.
[4] Jin Gang. Research on automatic long-text classification using a convolutional recurrent neural network based on distributed word-embedding features [J]. Electronic Technology, 2022, 51(06): 52-54.
[5] Wang Daokang, Zhang Wubo. Research on short-text classification based on MacBERT-BiLSTM and attention mechanism [J]. Modern Electronics Technique, 2023, 46(21): 123-128.