A BERT-Based Ensemble Learning Approach for Sentiment Classification in Twitter*

Shupei Li and Ziyi Xu

LIACS, Leiden University, Leiden, Netherlands
{s3430863, s3649024}@umail.leidenuniv.nl

Abstract. In this project, we present an ensemble learning method for sentiment classification. We use the state-of-the-art BERT as base models and develop a modified soft voting strategy for the meta model. The effectiveness of our proposed approach is verified on SemEval-2017 dataset, which quantifies the sentiment in tweets via a three-point ordinal scale. Experimental results show that the overall performance of our proposed model is better than that of baselines.

Keywords: Sentiment Analysis · BERT · Ensemble Learning

1 Introduction

Sentiment analysis, a growing field today, is the process of analyzing pieces of writing to determine the emotional tone they carry. In simple words, sentiment analysis helps to find the author's attitude. This method can be used by businesses to analyze product reviews and feedback, especially for social media companies with large information streams because of the wealth of data they generate. Researchers also have a special interest in social media data because of their easy availability and rapid change.

SemEval is an International Workshop on Semantic Evaluation, formerly SensEval. It is an ongoing series of evaluations of computational semantic analysis systems. The SemEval-2017 Task 4 focuses on classifying and quantifying the sentiment of tweets. The task was included in this workshop in the previous year [6], and Sentiment Analysis in Twitter has been run yearly since 2013 [7]. The subtask A of this task is the problem we try to unravel in this article. It is about deciding the overall sentiment of tweets and marking them on a three-point ordinal scale.

Our proposed model adopts two strategies with the goal of achieving a higher accuracy. One is choosing more powerful BERT and its variants as base models instead of CNN, LSTM, or SVM. The other is that we design an ensemble learning approach to integrate the advantages of different BERT models, which encourages the learning diversity and might enhance model performance. The classification task is divided into two stages. In the first stage, we train a series of base models to generate corresponding predictions. These predictions are the

* Text Mining, Master CS, Fall 2022, Leiden, the Netherlands

inputs of the meta model in the second stage. The final predictions are the outputs of the meta model.

The remainder of the paper is structured as follows. Section 2 discusses previous work related to sentiment classification. Following that, Section 3 introduces datasets provided by SemEval. We explain the data preparation workflow, our proposed approach, and baselines in Section 4. Section 5 describes the procedure of experiments in detail and reports the experimental results. The paper is concluded with a discussion about results and a brief summary. We also attach the contributions of group members at the end of the paper.

2 Related Work

The prevalence of social media has been driving the growth of sentiment analysis research since early 2000 [11]. Earlier studies mainly focused on the lexicon-based approach, which discovers sentimental polarity based either on a predefined dictionary or on some semantic methods [5]. Previous works have also intensively researched the applications of traditional machine learning methods in sentiment analysis, such as SVM, naive Bayes, etc [10]. However, both lexicon-based and machine learning approaches require manually feature selection, which limits their applicability in large and dynamic user-generated content like tweets.

In recent years, deep learning methods have been introduced into sentiment analysis field and have achieved the remarkable performance on many benchmark datasets [10]. Popular deep learning frameworks for sentiment classification include CNNs and RNNs. Inspired by computer vision field, the CNN architecture first constructs a sentence matrix using tokens and word vectors. Then it treats the sentence matrix as an "image" and applys regular convolution operations on the matrix [12]. The major drawback of CNNs is that CNNs can't capture the hierarchical relationship between local features. Besides, the pooling layer may cause the loss of critical information and harm the classification accuracy. LSTMs and GRUs are two commonly used RNN variants. Nowak et al. [8] compared LSTM, bidirectional LSTM, and GRU with other algorithms on real-world datasets and found RNNs outperformed baselines in sentiment classification task. However, RNNs are not suitable for processing long sequences and the training process is very slow. There were also attempts to combine CNNs and RNNs, aiming at enhancing model performance further [3][2]. But these hybrid models don't overcome the intrinsic weaknesses of CNNs and RNNs.

In this project, we consider BERT, the state-of-the-art model in several natural language processing tasks. Compared to CNNs and RNNs, BERT leverages the attention mechanism to model the complex relationship between features in short-distance as well as long-distance context. And it is efficient on GPUs and TPUs [1]. We describe our method in detail in Section 4.

3 Data

The dataset consists of 11 files with tweets from 2013 to 2015. Tweets are marked with sentiment labels on a three-point scale {Positive, Neutral, Negative}. Each tweet corresponds to one row in datasets following a fixed format: [`id`, `sentiment label`, `text`].

The criterion of tweet selection is covering popular topics at the time of sending tweets. Datasets are downloaded via the Twitter API and have been preprocessed by workshop in advance, where three kinds of data have been removed: repeated tweets, the bag-of-words cosine similarity exceeded 0.6, and topics with less than 100 tweets. CrowdFlower is used to create all annotations on both training set and test set. Each tweet is annotated by at least five people to ensure the accuracy of annotations. Another main quality control measure is performing hidden tests to filter out unqualified annotations. There are also manual inspections on pilot runs aiming at adjusting the annotation instructions dynamically. Table 1 summarizes the descriptive statistics of 11 files.

Table 1. Descriptive Statistics of Datasets

Dataset	Positive	Neutral	Negative	Total
2013train	3,640	4,586	1,458	9,684
2013dev	575	739	340	1,654
2013test	1,475	1,513	559	3,547
2014sarcasm	20	7	22	49
2014test	982	669	202	1,853
2015train	170	253	66	489
2015test	1,038	987	365	2,390
2016train	3,017	2,001	850	5,868
2016dev	829	746	391	1,966
2016devtest	994	681	325	2,000
2016test	7,059	10,342	3,231	20,632
Total	19,799	22,524	7,809	50,132

4 Methodology

4.1 Data Preparation

We perform data cleaning and dataset division before modeling. Notice that records in 2016test file have a redundant `\t` before `\n`. We remove the unnecessary `\t` to ensure the dataset would be read into Python appropriately. To create training set and test set, we concatenate all 11 files and randomly sample 80% records as training set while treating the others as test set. As a result, there are 40,105 records in training set and 10,027 records in test set. We also create the validation set for hyperparameter tuning by randomly sampling 20% recoreds from the training set. The random seed is set to 42 in our experiments to ensure the reproducibility.

4.2 BERT

BERT [1] is a popular NLP model that has achieved the remarkble performance on various tasks, such as text classification, question answering, etc. Compared to traditional RNN models, it encodes sequences in both directions instead of following a left-to-right or right-to-left routine, which is more closer to how humans understand the meaning of the text. Its bidirectional encoding ability is accomplished by the transformer architecture, which is a stack of encoders using multi-head attention mechanism. Specifically, encoders process the entire sequence at once and use layer-wise tensor operations to learn relationships between words in a sentence. This design not only encodes the inputs bidirectionally but also eliminates the possible local bias, for it gives equal importance to the local context and the long-distance context. It is worth mentioning that the training process of BERT is more efficient than RNN due to the feasibility of parallelization.

BERT requires a special format of input called WordPiece. The WordPiece tokenizer splits words into tokens and adds special tokens at the beginning as well as the end of the sentence. Preprocessed inputs provide three aspects of information for BERT model: tokens, sentence segments, and the absolute position. We perform the tokenization operation on both the training set and the test set before modeling.

We also consider two variants of BERT in our project – RoBERTa [4] and DistilBERT [9]. RoBERTa is an optimized version of BERT model. Authors of RoBERTa find that the original BERT is actually undertrained after reproducing BERT experiments. Their solution is increasing the training epochs of BERT and carefully select hyperparameters, which significantly improves the model performance in practice. On the contrary, DistilBERT is a compressed version of BERT. The main idea of DistilBERT is reducing the model size via distillation technique. Distillation consists of a teacher model and a student model. The teacher model is trained on a large dataset and is fine-tuned to maximize the accuracy. However, many features learned by the teacher model are redundant for a specific task. Therefore, we can train a much smaller student model to focus on key features and imitate the output of the teacher model. Experimental results show that DistilBERT is cheaper to train while maintaining a comparable performance to BERT.

4.3 Ensemble Learning

Ensemble learning refers to methods that combine multiple models to achieve better performance in machine learning. It encourages base models to learn different aspects of the data to reduce errors and avoid being entrapped in local optima. In the project, sentiment analysis in Twitter is a multi-class classification problem. And we develop a classification voting ensemble model integrated BERT and its variants. Figure 1 illustrates the architecture of our model.

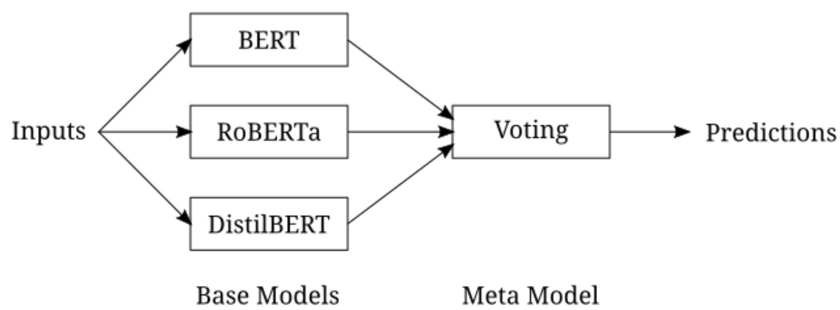

Fig. 1. The Architecture of the Proposed Model

Each base model outputs the class prediction for each record. Here we consider two strategies for the meta model — hard voting and soft voting. Hard voting means the voting classifier adopts the plurality voting strategy to generate the final prediction. In other words, the final prediction is the class label received the majority votes from base models. If all classes have the same votes, the voting classifier will choose a class label randomly as the output. However, hard voting ignores the probability information related to each model's prediction and is likely to be affected by relatively weaker base models. Thus, we set the hard voting ensemble learning as one of our baselines and propose a modified version of soft voting.

Traditional soft voting strategy averages the output probabilities of base models and selects the class with the highest probability. However, the probabilistic prediction is ill-defined in Hugging Face's implementation of BERT, i.e., the outputs may contain negative numbers and are not normalized. We think the negative value is a signal of the unlikely class and retain the negative numbers when aggregating predictions. Our modified soft voting is scaling each model's prediction array to unit norm, calculating the element-wise average, and output the class with the biggest value. Therefore, our proposed model is the ensemble of three BERT base learners and a modified soft voting classifier.

4.4 Baselines

We compare the proposed method against four baselines. The first three benchmarks are BERT, RoBERTa, and DistilBERT alone. We set the ensemble BERT model with hard voting strategy as the fourth baseline, as mentioned in Section 4.3. Accuracy, macro recall, and macro F1 are selected as metrics in experiments.

5 Experiments

5.1 Experimental Setup

Before feeding data to the model, we first preprocess all the input data to make the text more readable and accurate. Three main changings are made: the first is to remove all the @*user_name* at the beginning of the text, the second is to use & replacing &*amp*, and the last is to replace multiple spaces with one space. After accomplishing the above procedures, we transfer the text label into numerical id, setting negative as 0, neutral as 1, and positive as 2.

For the experiments, we build up three individual baselines, BERT, RoBERTa, and DistilBERT, and then use them to generate the other two ensemble models.

The architecture of BERT and its variants is rarely similar, and the process can be concluded as tokenization, training, and tuning. Since the transformer needs the WordPiece tokens as input, we take the pre-trained vocabulary to tokenize the text with the help of `Data Collator`. Then comes the training part. We load the model from pre-trained data and set up the number of labels as 3 according to the task. The batch, the epoch, and the learning rate are set as 16, 50, and 10^{-5}, respectively. Then we create a basic optimizer with the learning rate to compile the model. To fit the model, there are three various callbacks. The `metric_callback` can compute metrics at the end of every epoch without compiling by TF, `early_stopping_callback` stops the model when accuracy remains the same among three computing, and `model_checkpoint_callback` saves the best performance of the model. With these callbacks, we tune the learning rate from 10^{-5} to 10^{-2} to find the best-performing one and use the weights saved by `model_checkpoint_callback` to make the prediction.

Through this method we can get three sentiment classifications corresponding to three models. With these results and the weights of the three best models, two meta-models are generated by hard voting and soft voting. The hard voting ensemble model counts and compares the absolute number of votes to select the final sentiments. Meanwhile, soft voting considers the sentiment distribution after uniform.

5.2 Results

Implementing the experiments with the set-up, we collect the precision, recall, and f1-score of the five models.

Table 2 shows that the soft-voting meta model, with 0.753 precision and 0.732 f1-score, gets the best performance compared to the four baselines. The base-cased BERT model already has a good predictive ability. And the base-cased RoBERTa has a relatively outstanding performance, even outperforms the hard voting meta-model. Using the modified soft voting model, the precision and f1-score are better than RoBERTa, while the recall is not, still a bit lower than RoBERTa.

Table 2. The evalutions of baselines and the soft voting model

Model	Precision	Recall	F1-score
BERT	0.734	0.717	0.718
RoBERTa	0.747	0.724	0.731
DistilBERT	0.722	0.692	0.700
Hard voting	0.746	0.723	0.729
Soft voting	0.753	0.719	0.732

6 Discussion

The soft-voting meta model outperforms the four baselines, but there are still some improvements that could be done.

First, the recall of the soft voting model is relatively low, and the recall of the negative sentiment is only 0.59. Besides, the running of the BERT is very time-consuming, and the efficiency can be improved through parallel. Third, the ensemble learning model is influenced by the classifier. Constrained by computational resources, all of our baseline models use the base-cased classifier. And it may observe better results implementing the large and uncased classifier.

7 Conclusion

We build an ensemble BERT model using soft voting to classify the sentiments of the Twitter text. And this model is quite powerful compared to three BERT base models and the soft voting ensemble model with an accuracy of 0.752.

8 Contributions

Authors contribute equally to the programming and the report.

References

1. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. ArXiv **abs/1810.04805** (2019)
2. Hassan, A., Mahmood, A.: Deep learning approach for sentiment analysis of short texts. 2017 3rd International Conference on Control, Automation and Robotics (ICCAR) pp. 705–710 (2017)
3. Huang, Q., Chen, R., Zheng, X., Dong, Z.: Deep sentiment representation based on cnn and lstm. 2017 International Conference on Green Informatics (ICGI) pp. 30–33 (2017)
4. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. ArXiv **abs/1907.11692** (2019)

5. Medhat, W., Hassan, A., Korashy, H.: Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal **5**(4), 1093–1113 (2014)
6. Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., Stoyanov, V.: SemEval-2016 task 4: Sentiment analysis in Twitter. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 1–18. Association for Computational Linguistics, San Diego, California (Jun 2016)
7. Nakov, P., Rosenthal, S., Kozareva, Z., Stoyanov, V., Ritter, A., Wilson, T.: SemEval-2013 task 2: Sentiment analysis in Twitter. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). pp. 312–320. Association for Computational Linguistics, Atlanta, Georgia, USA (Jun 2013)
8. Nowak, J., Taspinar, A., Scherer, R.: Lstm recurrent neural networks for short text and sentiment classification. In: Artificial Intelligence and Soft Computing. pp. 553–562. Springer International Publishing, Cham (2017)
9. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. ArXiv **abs/1910.01108** (2019)
10. Yadav, A., Vishwakarma, D.K.: Sentiment analysis using deep learning architectures: A review. Artif. Intell. Rev. **53**(6), 4335–4385 (Aug 2020)
11. Zhang, L., Wang, S., Liu, B.: Deep learning for sentiment analysis: A survey. WIREs Data Mining and Knowledge Discovery **8**(4), e1253 (2018)
12. Zhang, Y., Wallace, B.: A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification (2015)

Appendix