A Combination of Lexicon-Based and Classified-Based methods for Sentiment Classification based on Bert

Jiaxin Zhang

The University of Sheffield. Western Bank Sheffield S10 2TN, UK

jiaxin.zhang@strath.ac.uk

Abstract. Sentiment classification is a crucial problem in natural language processing and is essential to understand user opinions. There are two main approaches to solve this problem, one is the classified-based method, the other is the lexicon-based method; however, both methods perform not well on the long-sequence methods, and each method has its advantages and disadvantages. This paper introduced a new method called Lexiconed BERT, which cream off the best and filter out the impurities from the above two methods. The evaluation shows that our model achieves excellent results in the long sequence sentence and reduce resource consumption significantly.

Keywords: Sentiment Analysis; BERT; Lexicon-Based Method.

1. Introduction

Sentiment Analysis, drawing the sentiment polarize from the given sentence, is an essential and practical application in Natural Language Processing. Sentiment Analysis has played a crucial role in many applications, such as chatbot, recommendation system, etc.

There are mainly two general methods used for sentiment analysis: the classified-based method which regards the sentiment analysis as a classification problem, and trains a classifier (SVM, RNN, etc.) to make the prediction; the lexicon-based method which used the sentiment lexicons---the words labeled with specific sentiment tag by hand, predicts the sentiment based on an algorithm. As described, the sentiment benefits from both methods: to classified-based method it could be flexible due to the various selected features, but at other times it could be more accurate on some certain corpora domains by lexicon-based methods. Sentiment analysis could be regarded as a typical text-classification task when used the classified-based method, though it holds the state-of-art result in many text-classification tasks, the result in sentiment analysis was not up to the mark.

This paper introduces a new method which is a pastiche of BERT and lexicon-based method. Our contributions include: We put forward a new pre-trained task for BERT, that a task cuts out for the sentiment analysis task, which does not predict the masked words but the polarization of the sentiment lexicons; We also take advantage of the lexicon-based method, which helps us to select useful features and accelerate both the training and inference speeds. For evaluation, we achieved 83% accuracy on the testing data, which is comparable with the traditional BERT method, but with much fewer computation resources. Compared with Lexicon-based methods, our model only needs basic sentiment lexicons to be tagged and easily updated.

2. Related Work

2.1. Sentiment Analysis

Many researchers have tested multiple models on the sentiment classification tasks.[1] Bo Pang et al. [2] trained three models (Naïve Bayes, Maximum Entropy, and SVM) on a movie review corpus, the result surpasses the random-choice baseline of 50%. Many researchers used CNN or RNN to classify the sentiment, dos Santos et al. [3] proposed the CharSCNN model used convolutional neural networks to extract sentiment features from the sentences. Wang et al. [4] used LSTM to predict the sentiment on the Twitter dataset. Later, Wang et al. [5] proposed an architecture which combined the CNN and the RNN for sentiment classification, the model benefited from the coarse-grained features extracted by the CNN and long-distance dependencies learned by RNN. The above methods rely on that the deep neural network could learn the essential features automatically; however, input all the data to the models may consume large resource. Taboada et al. [6] proposed SO-CAL, which makes sentiment classification by pre-annotated sentiment lexicons, although this model replies on the annotated features, it cannot perform well on the newly emerged data.

2.2. Bidirectional Encoder Representations from Transformers

BERT [7] is the short form for Bidirectional Encoder Representations from Transformers. The BERT model has won many benchmarks on multiple NLP basic tasks. Recently, BERT has been the most popular pre-trained model, so many text classification tasks reach the state-of-art result with the help of the BERT pre-trained model. Sun et al. [8] fine-tuned the pre-trained BERT model and evaluated the model on the SentiHood and SemEval-2014 Task 4, which achieved an excellent result. One of the BERT's downstream task is the text classification problem, though the tasks perform extremely good by fine-tuning the original BERT pre-trained model, the ability of it to extract sentiment features is not: instead, it performs not well, and when handling with the long sequence, BERT requires the entire input, however, this will cause much resource consumption.

2.3. SentiWordNet

SentiWordNet [9] is a dataset that assigns each synset of WordNet three sentiment scores: positive, negative, and neutrality. As described above, our proposed model will predict the polarization of the sentiment lexicons rather than the masked words, using the word from SentiWordNet. SentiWordNet assigns the word “estimable” to the POS score of 0.75 so that the model shall predict the word “estimable” as positive. For predicting words polarized in pre-train step, typical sentiment lexicons with labels are indispensable, SentiWordNet, an authoritative lexical resource, is utilitarian for us. The SentiWordNet provides us with potential sentiment lexicons with high-quality scores, and it is helpful with reducing features dimensions for the model.

3. Methodology

The model is based on the BERT model, however, more applicable to the sentiment analysis task. BERT, a large Transformer model, in which each token’s added positional embeddings are embedded to word embeddings, transforms these embeddings to a sequence vector $h = [h_1, \dots, h_n]$. Each hidden vector from the sequence vectors is transformed into a query q_i, k_i, and v_i through three distinguish linear transformation, then attention head will compute attention weights between all the tokens by calculating the dot-product between each query and key vectors, the final attention weights are obtained through the softmax normalized. The final output o of the attention head should be calculated by multiplying the attention weights with the value vector v and sum the corresponding items of each dimension up. See the formulas below:

$$a_{ij} = \frac{exp\,(q_i^T k_j)}{\sum_{l=1}^{n} exp\,(q_i^T k_l)} \tag{1}$$

$$o_i = \sum_{j=1}^{n} \quad a_{ij} v_j \tag{2}$$

It is crucial to introduce the pre-processing step of the BERT, where our method has nuance compared with it. Two unique tags, $< cls >$ and $< sep >$, are added to the beginning of each input and the ending of each sentence, which constitute the whole input, respectively. The vector at the position of the $< cls >$ tag will be sent to the softmax layer and made a prediction. The slight differences between the original processing method and ours are 1. The $< sep >$ tag is inserted at the beginning of each sentence and could only attach the tokens belonging to the corresponding sentence part; 2. The $< cls >$ tag could only attach the$< sep >$ tags in the whole input rather than the entire input; the above two points are achieved through our new customized mask method, see the following section.

3.1. Feature Selection

This section describes how we select our features based on SentiWordNet. SentiWordNet assigns each word three scores: positive, negative, neutrality, the sum of theses scores is equal to 1.0. As mentioned before, a movie review is quite long compared to the traditional reviews, which could reach even more than 500 words per review; however, not all these words are useful features for our task. In lexicon-based methods, it only pays attention to the lexicons, which contain intrinsic sentiment meanings, such as good, wonderful, terrific, etc. These words in lexicon-based method inspires us that not all the words in an entire movie review are important equally so that we choose the words which contain sentiment meanings as the input features to our model. We select these words according to the SentiWordNet, the word which has either a positive score or a negative score larger than 0.5 could be kept as features. The length of input could cut down to extraordinary size, which leads to fewer resources consuming and rapid inference speed. There are another two circumstances that need to be considered, negative words and subjunctive mood; both of them could change the attitude of the sentence, for example, “good” and “not good” are contradictory; consequently, these kinds of words are considered as features for input too.

3.2. Sentiment Attention Mask

3.2.1. Customized Mask. In original BERT, it hides masked words from the other words through setting the mask tag to 0 and inserts only one $< \text{sep} >$ tag at the beginning of the entire input. As some movie reviews encounter the problem called “hold off before starting up,” for example, the reviewer may lambaste the movie in the previous sentences, then convert to praise the movie in the last sentence, so that the final sentiment would be set to positive. However, the sentiment conveyed by the previous sentences would disturb the final sentiment to negative. For this reason, we want to extract the sentiment from each clause independently and then consider the final sentiment based on the combination of these sentiments, we insert $< \text{sep} >$ tag at the beginning of each clause, like the following:

$< sep >< clause_1 >, < sep >< clause_2 >, \dots , < sep >< clause\_n >$

In order to achieve that the sentiment of each clause is independent at first, we set each clause that could only be seen by its $< sep >$ tag, and $< sep >$ could see other $< sep >$s to share information, like the following:

$< sep > good\ wonderful < sep > fabulous$

[1,	1,	1,	1,	0]
[1,	1,	1,	0,	0]
[1,	1,	1,	0,	0]
[1,	0,	0,	1,	0]
[0,	0,	0,	1,	0]

For the negative words and subjunctive modal verbs, they can only see the word they decorate; the following example is from the Large Movie Review Dataset, the words in the bold format are the features selected, just for the simplicity, the chosen lexicons is more than the example:

I didn't know this came from Canada, but it is not very good however worthy watching.

For the last clause: <sep> not very good however worthy, the mask for "not" should looks like: [1, 1, 1, 1, 0, 0].

3.2.2. Pre-Training. At the pre-training step, we predict every sentiment lexicon's polarization rather than the masked words as original BERT does. Although SentiWordNet will provide each word with three individual scores for positive, negative, and neutrality. We do not do regression prediction here, rather classification task. As we ignore the words which have the highest score in neutrality, we divide the words into two types: positive and negative. At the result, the model will take the output from the last layer and predict the polarization of each sentiment lexicon (see the formula below):

$$P = \sigma(w * h + b) \tag{3}$$

Where h is the hidden outputs from the transformer block, the above probability P is estimated through one fully-connected layer, then activated by the sigmoid function. We use the cross-entropy loss for this step:

$$loss = -(ylog(p) + (1 - y)\ log\ log\ (1 - p)\ ) \tag{4}$$

Where y refers to the prediction from the function above when the prediction is 1, i.e., positive, the loss function is $-ylog(p)$, or vice versa.

This step does not use the powerful pre-trained model from the original BERT, we set both the embedding size and the hidden size to 256, and choose the cross-entropy loss to be updated. We finished our pre-training step with the learning rate of 1e-4 in 100000 steps on a single NVIDIA GTX 1060ti GPU.

3.2.3. DownStream Task – Sentiment Classification. After the pre-train step, we can continue the next step – predict the polarization of the entire review. There is another crucial tag which has not been mentioned above, the $< \text{cls} >$ tag. This tag will be inserted at the beginning of the input when processing the data. The $< \text{cls} >$ tag could only see the $< \text{sep} >$ tag at the beginning of each clause, which could be achieved through the mask. The output of the $< \text{cls} >$ tag will predict the polarization in a non-autoregressive way:

$$f = softmax(W_2 h_z + b_2) \in R^{[V]}) \tag{5}$$

Where h_z is the hidden state of the $< cls >$ tag, and $|V|$ is the polarization size, i.e., 2, f refers to the probabilities of the positive tag and the negative tag. The loss function used for the downstream task is negative log-likelihood loss:

$$-y_{true} * log\ (p) \tag{6}$$

We fine-tuned the model in 100000s with the learning rate of 2e-5 on the same GPU as the pre-train step.

4. Evaluation

4.1. Datasets and Training

We use the Large Movie Review from Stanford University for the evaluation, because the length of the reviews is long enough, which satisfies one of our research goals, i.e., long sequence. Large Movie Review is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets; it consists of 25000 movie reviews with labels of high qualities for training data, 25000 for testing data too. Each review was labeled as positive sentiment or negative sentiment, then leading to our classification results should be either positive or negative. F-1 Score, precision, and recall were used as evaluation metrics.

The hyper-parameters configuration is the same as the BERT-BASE model, except for the final linear layer applied on the encoder outputs from the BERT encoder. The linear layer's hidden size before the softmax (or sigmoid for the pre-train step) layer is identical to the polarization size, i.e., 2. The Adam optimizer updated both for pre-train step with 1e-4 learning rate and for fine-tune step with 2e-5 learning rate; The batch size is 64.

4.2. Result Analysis

We compare our model with two state-of-the-art methods:

- A typical lexicon-based method called SO-CAL, which is the Semantic Orientation CALculator, a tool to extract sentiment from text [6];
- BERT-BASE, we used the existing pre-trained model, and just fine-tuned the model on our dataset.

The model was implemented by TensorFlow [10], and the Transformer part of the code was from the original source BERT code.

Table 1. Results of comparison between different models on Large Movie Review

Scores	TP / FP	FN / TN	Precision	Recall	F-1
SENT_BERT	10231	2269	0.83	0.81	0.82
	1990	10510			
BERT_BASE	9981	2519	0.78	0.79	0.78
	2778	9722			
SO-CAL	8949	3551	0.50	0.71	0.58
	9003	3491			

As described in Table 1, Our method beats the other two methods, especially outperforms the SO-CAL significantly. The intuition is clear: Each movie review's length is quite long, so the features selected by SO-CAL are too many, and SO-CAL could not handle the relationship between sentences (especially one review consists of many clauses).

Our model used $< sep >$ tags to collect the sentiment information from each clause and combine them through the $< cls >$ tag to make the prediction, which made our model more suitable for the long sequence with many sentiment features. As the movie review in the dataset is quite long, our method could capture the essential features for sentiment analysis without less noisy disturbance, our model not only has much faster speed than the BERT-BASE model in training and inference speeds but also get higher F-1 score than the BERT-BASE.

One of the advantages of our model is less computation consumption, the average length of the reviews in Large Movie Review Dataset is approximately 300, which would hinder the model from fast train speed and require lots of resources. To fix this problem, the model does not take the entire sentence as input, but rather the sentiment lexicons in the sentence. Only the words with either positive score or negative score will be regarded as features; then the model would mask some words with dynamic percentage; finally, the model should predict the sentiment polarization of these masked words. The

model after the pre-trained step could know the sentiment of each word, which is particularly crucial for the downstream task.

5. Conclusion and Future Work

This article describes a new sentiment classification method, which combines the traditional lexicon-based and the BERT. We optimized the feature selection to accelerate the train and inference speeds on the long sequence data and consume fewer resources. We also proposed a new data processing method that was suitable for the sentiment classification method. The model forces the clause of the entire input to attach the words of its own words, and the $< cls >$ tag which attaches the sentiment from each $< sep >$ tag will be used for prediction, these are achieved by the special mask which hides the unnecessary information from the tags. The model reaches the average 82% accuracy on both positive and negative data, which is reasonable.

In the future, we plan to address the following: 1) Choose more features that could influence the sentiment, such as emoji; 2) The exponential increase in the number of information makes sentiment classification hard to handle all kinds of domains. Reviews can span so many different domains that it not easy to gather annotated training data for all of them [11]. It will be helpful to explore the domain-adaptation in our model, remove the domain-specific features and keep shard features among domains, as this will make the model perform well one the unseen data and newly emerged data.

References

[1] Liu, B., 2012. Sentiment analysis and opinion mining. Synthesis lectures on human language technologies, 5(1), (pp.1-167).

[2] Pang, B., Lee, L. and Vaithyanathan, S., 2002, July. Thumbsup?: sentiment classification using machine learning tech-niques. In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, (pp.79-86).

[3] Dos Santos, C. and Gatti, M., 2014, August. Deep convolutional neural networks for sentiment analysis of shorttexts. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers pp. (69-78).

[4] Wang, X., Liu, Y., Sun, C.J., Wang, B. and Wang, X., 2015, July. Predicting polarities of tweets by composing word embeddings with long short-term memory. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp.1343-1353).

[5] Wang, X., Jiang, W. and Luo, Z., 2016, December. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In Proceedings of COLING2016, the 26th international conference on computational linguistics: Technical papers (pp. 2428-2437).

[6] Taboada, M., Brooke, J., Tofiloski, M., Voll, K. and Stede, M., 2011. Lexicon-based methods for sentiment analysis. (Computational linguistics, 37(2), pp.267-307).

[7] Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018.Bert: Pre-training of deep bidirectional transformers for language understanding. (arXiv preprint arXiv:1810.04805).

[8] Sun, C., Huang, L. and Qiu, X., 2019. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. (arXiv preprint arXiv:1903.09588).

[9] Esuli, A. and Sebastiani, F., 2006, May. Sentiwordnet: Apublicly available lexical resource for opinion mining. InLREC (Vol. 6, pp. 417-422).

[10] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean,J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur,M., 2016. Tensorflow: A system for large-scale machinelearning. In 12th USENIX Symposium on Operating SystemsDesign and Implementation (OSDI 16) (pp. 265-283).

[11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011.Domain adaptation for large-scale sentiment classification: a deep learning approach. In Proceedings of the 28th In-ternational Conference on International Conference on Machine Learning (ICML'11). Omnipress, Madison, WI, USA, (513–520).