Zeus at HASOC 2020: Hate speech detection based on ALBERT-DPCNN

Siyao Zhou[a], Rui Fu[a] and Jie Li[a]

[a]School of Information Science and Engineering, Yunnan University, Yunnan, P.R. China

Abstract

The use of social media has grown rapidly in the past few years. User generated data often contains objectionable content. Identifying hate speech, cyber-attacks and offensive language is a very challenging sentiment analysis task. In this paper, we participated in HASOC's English hate speech and offensive content identification task and proposed the ALBERT-DPCNN model based on emotion analysis, which combined ALBERT and DPCNN obtained richer semantic features, ranking third in the task and achieving good results.

Keywords

ALBERT-DPCNN, Hate speech, Offensive language, Sentiment analysis, HASOC

1. Introduction

In the Internet era, new social networking platforms with high openness have attracted a large number of global netizens, the amount of information in social media is increasing exponentially. However, individuals or groups with questionable motives seize the opportunity to spread extreme hate speech by taking advantage of the fast dissemination of information and large user groups on social networking platforms. Not only that, some people have reflected their xenophobic thoughts in real life through the form of group conflicts and violent attacks.

The current approach to tackle the problem of a large portion of hateful posts is filtering. However, society also needs to ensure that freedom of speech is maintained and social norms are not violated. Social media needs to censor a lot of content, and censoring and removing hateful or offensive words is a fairly cumbersome process. Therefore, people are aware of the importance of this research, Hate Speech and Offensive Content Detection in Indo-European Languages (HASOC 2020 [1]) provides multi-language research with over 10,000 annotated tweets from Twitter. HASOC provides 2 subtasks for each language, such as English, German, and Hindi.

We apply deep learning methods to participate in two subtasks of English, and propose the ALBERT-DPCNN model to complete the text classification of this task, which depends on very little preprocessing and feature engineering compared with other methods.

The rest of the paper is organized into four parts. The second section discusses in detail what we have done. Next, we will detail the method used by ALBERT-DPCNN in section 3. Then,

FIRE '20, Forum for Information Retrieval Evaluation, December 16–20, 2020, Hyderabad, India

973996794@qq.com (S. Zhou); 251248362@qq.com (R. Fu); 1470465334@qq.com (J. Li)
0000-0002-5583-098X (S. Zhou); 0000-0001-6806-1713 (R. Fu); 0000-0002-6590-6693 (J. Li)

the fourth section analyzes our experimental results. Finally, we conclude the paper in section 5 and discuss future work.

2. Related Work

Over the years, as social media has become more popular, abusive language has become more common on these platforms. Waseem used SVM and LR classifiers to detect racist or sexist content[2]. In recent years, there has been a growing body of research on offensive language. The existing technologies for the detection of offensive language and hate speech in social media are found mainly through Bag-of-words (BOW)[3], Recurrent Neural Networks (RNN)[4], and Word embedding[5]. In these years of research, the RNN model has achieved good results in sentiment analysis tasks. Serra et al.[6]achieved good results using character-based RNN to detect hate speech in tweets. Nobata et al.[7] used the regression model and obtained character n-gram features that were the most predictive in detecting offensive remarks through comparison. Vijay et al.[8] designed a classification system based on sentiment analysis from the perspective of machine learning. Badjatiya et al.[9] introduced a deep learning approach, using CNN, LSTM, and other deep learning models to detect offensive remarks in English and Hindi. Self-attention[10] technology has been widely used in text classification in recent years. While BERT[11] model uses the pre-training technology to further increase the generalization ability of the word vector model and fully describe the character-level, word-level, sentence-level, and even inter-sentence relationship characteristics. As mentioned above, there has been a lot of research on sentiment analysis in many different code-mixing types of languages. In this paper, we propose a language model (ALBERT-DPCNN) architecture based on ALBERT to detect hate speech, offensive language, and blasphemy.

3. Data and methodology

In this section, we introduce the method we used. We try traditional machine learning and neural network, as well as pre-training methods. Because of ALBERT's recent success in sentiment analysis and other language-processing tasks, by comparison, We choose the model based on ALBERT for the HASOC task.

3.1. Data description

The data is provided by the HASOC organizer, and we present the statistics of the HASOC dataset in the following table. In the English task, there were 3708 posts in the training set and 814 posts in the test set. The texts of Sub-task A is labeled as HOF (Hate and Offensive) and NOT (Non Hate-Offensive), whereas Sub-task B is denoted as NONE, HATE (Hate speech), OFFN (Offensive), PRFN (Profane) respectively. For Sub-task A, the number of posts in different categories is about the same, but for Sub-task B, the number of posts in different categories is not similar.

Table 1
Statistics of the English Sub-task A set provided by the organizers.

Sub-task A	NOT	HOF	Total
Train	1852	1856	3708
Test	391	423	814

Table 2
Statistics of the English Sub-task B set provided by the organizers.

Sub-task B	NONE	PREF	OFFN	HATE	Total
Train	1852	1377	321	158	3708
Test	414	293	82	25	814

3.2. ALBERT

The architecture of ALBERT and BERT is similar, using the Transformer encoder and GELU nonlinear activation function, but ALBERT has a much smaller number of parameters as compared to the traditional BERT architecture. As with Transformer encoders, the encoder consists of two layers, a self-attention layer, and a feedforward neural network. Self-attention helps the current node to focus on more than just the current word, thus getting the semantics of the context. The decoder also includes the two layers of the network mentioned by the encoder, but in between the two layers is the attention layer, which helps the current node get the key content that needs to be paid attention at present. ALBERT takes word sequence as input and embedding operation of the input data. Embedding finished, the data will be entered into the encoder layer. After self-attention finish handling the data, the data will be delivered to the feedforward neural network, and the computation of the feedforward neural network can be parallel, the result will be output to the next encoder.

3.3. DPCNN

Rie Johnson et al.[12]proposed a deep convolutional neural network called DPCNN (Deep Pyramid Convolutional Neural Networks). This is a wide and effective deep text classification convolutional neural network based on word-level, which can extract long-distance text dependency by deepening the network continuously.

DPCNN is mainly composed of a Region embedding layer (text area embedding layer) and two convolution blocks (each block is composed of two convolution functions with a fixed convolution kernel of 3 and Max-pooling layer), as shown in Figure 1 below. DPCNN adopts the method of pre-activation. During convolution operation, the feature set of input will pass through the Max-pooling layer and then be input into the convolution layer. In other words, The output of the convolution operation linearly activates is $W\sigma(x) + b$, not $\sigma(Wx + b)$.

After passing through a pooling layer with size 3 and stride 2, the length of the sequence is

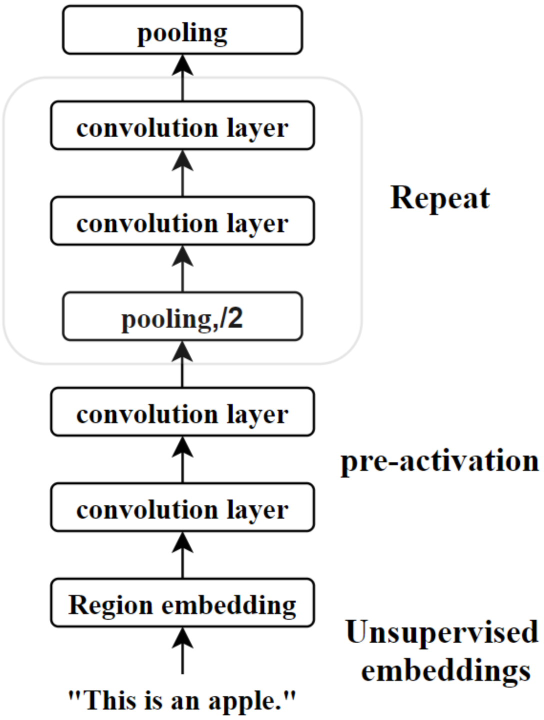

Figure 1: Model DPCNN

reduced to half of the original length. The sequence goes through this pooling operation, with a convolution kernel of size 3, the pieces of text it can perceive are twice as long. Because of the Max-pooling layer described earlier, the length of the text sequence decreases exponentially as the number of blocks increases, causing the sequence length to take on the shape of a pyramid as the network deepens.

3.4. ALBERT-DPCNN

The model makes full use of the content of the whole sentence, vectorize sentences, extract useful linguistic, syntactic, and semantic features. It takes into account the effect of each word in context on the other words and the different meanings of the same word in different contexts.

First, to adapt the model to downstream tasks, the input sequence adds a [CLS] token at the beginning of the sentence, [SEP] token used as a separator between sentences or as a flag at the end of a sentence. For the classification task, ALBERT's output (output of the pooling layer) is obtained by passing the [CLS] token of the sequence through the last Hidden layer, containing the information of the entire sequence. However, output of the pooling layer is usually not the best result for distinguishing input semantic content. So, once we get the output of the pooling layer, the model output the first token ([CLS] token) of the last four Hidden layers to DPCNN. After extracting the text context features, connect them to the classifier. In this way, the advantages of feature extraction of ALBERT and DPCNN can be utilized, simultaneously,

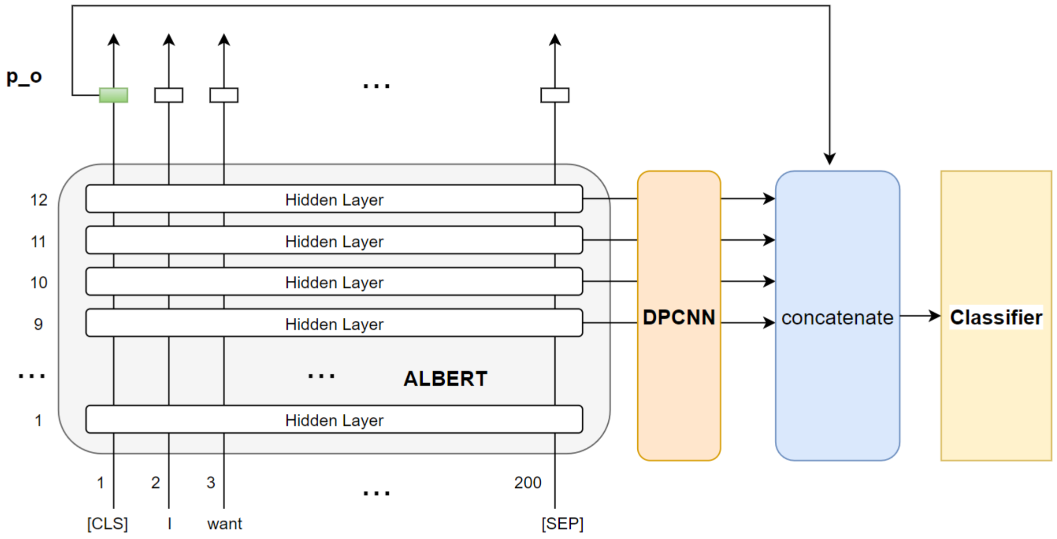

Figure 2: Schematic diagram of ALBERT-DPCNN architecture, p_o is the output of the pooling layer, and the first token of the last four hidden layers ([CLS] token) is output to DPCNN.

and the semantics of the text can be well explained. The model architecture is shown in Figure 2.

4. Experiments and Results

4.1. Experiments Setup

For this work, we use the ALBERT-DPCNN model, which is implemented based on Pytorch. We set up stratified 5-fold cross-validation with 42 random seeds for training (StratifiedKFold[1]), and use the form of stratified grouping so that the ratio of each class in each group is as close as possible to that of each class in the overall data.

We use Adam optimizer with a learning rate of 2e-5 and CrossEntropy Loss. The epochs and max sentence length are 3 and 120, respectively. And the batch size is set to 32, and the gradient steps are set to 4.

4.2. Results

In this work, we will introduce the evaluation results we submitted. Evaluation is carried out by HASOC task organizer, the results are shown in Table 3. Both Sub-task A and Sub-task B are evaluated by F1 Macro-Average. Finally, the model we submitted ranked 25th with an F1 score of 0.4954 in English Sub-task A, and 3rd with an F1 score of 0.2619 in English Sub-task B.

We used the hidden layer state of ALBERT to obtain richer semantic features. As can be seen from Table 3, our model has achieved good results in English Sub-task B.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html

Table 3
The results of our model on the official HASOC test sets.

Language English	F1-macro	Rank
Sub-task A	0.4954	25
Sub-task B	0.2619	3

5. Conclusion

With the increasing popularity and influence of social media texts, analyzing the emotions attached to texts becomes more and more important. In this paper, we present the ALBERT-DPCNN model to handle HASOC tasks. Our model achieved remarkable performance and came in second place in English Sub-task B. The research results provide a strong basis for further research on hate speech in multiple languages.

Acknowledgments

I would like to thank the organizers for their hard work and Rui Fu for her help. Her advice has been very helpful to me.

References

[1] T. Mandl, S. Modha, G. K. Shahi, A. K. Jaiswal, D. Nandini, D. Patel, P. Majumder, J. Schäfer, Overview of the HASOC track at FIRE 2020: Hate Speech and Offensive Content Identification in Indo-European Languages), in: Working Notes of FIRE 2020 - Forum for Information Retrieval Evaluation, CEUR, 2020.

[2] Z. Waseem, Are you a racist or am i seeing things? Annotator influence on hate speech detection on twitter, in: Proceedings of the first workshop on NLP and computational social science, 2016, pp. 138–142.

[3] P. Badjatiya, S. Gupta, M. Gupta, V. Varma, Deep learning for hate speech detection in tweets, in: Proceedings of the 26th International Conference on World Wide Web Companion, 2017, pp. 759–760.

[4] S. Agrawal, A. Awekar, Deep learning for detecting cyberbullying across multiple social media platforms, in: European Conference on Information Retrieval, Springer, 2018, pp. 141–153.

[5] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, N. Bhamidipati, Hate speech detection with comment embeddings, in: Proceedings of the 24th international conference on world wide web, 2015, pp. 29–30.

[6] J. Serra, I. Leontiadis, D. Spathis, G. Stringhini, J. Blackburn, A. Vakali, Class-based prediction errors to detect hate speech with out-of-vocabulary words, in: Proceedings of the First Workshop on Abusive Language Online, 2017, pp. 36–40.

[7] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, Y. Chang, Abusive language detection in online user content, in: Proceedings of the 25th international conference on world wide web, 2016, pp. 145–153.

[8] D. Vijay, A. Bohra, V. Singh, S. S. Akhtar, M. Shrivastava, Corpus creation and emotion prediction for hindi-english code-mixed social media text, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop, 2018, pp. 128–135.

[9] P. Badjatiya, S. Gupta, M. Gupta, V. Varma, Deep learning for hate speech detection in tweets, in: Proceedings of the 26th International Conference on World Wide Web Companion, 2017, pp. 759–760.

[10] M. Choi, H. Kim, B. Han, N. Xu, K. M. Lee, Channel Attention Is All You Need for Video Frame Interpolation., in: AAAI, 2020, pp. 10663–10671.

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

[12] R. Johnson, T. Zhang, Deep pyramid convolutional neural networks for text categorization, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 562–570.