Hub@HASOC 2020: Fine-tuning Pre-Trained Transformer Language Models for Hate Speech and Offensive Content Identification in Indo-European Languages

Bo Huang, Yang Bai

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

Abstract

This paper presents the system description of the Hub team participating in HASOC2020: Hate Speech and Offensive Content Identification in Indo-European Languages. The focus of this shared task research is to identify hateful or offensive content in English, German, and Hindi comments posted on Twitter. Each language consists of two tasks, the first of which can be seen as a coarse-grained binary classification task, and the other can be seen as a fine-grained quaternary classification task. We only participated in the English task and the German task. According to our analysis of the task description and data set, we use two fine-tuned pre-trained transformer models ALBERT and BERT for the English task and the German task. In this paper, we will discuss the experiments and results of the English task and the German task.

Keywords

Hate Speech, Offensive Content, pre-trained transformer models, ALBERT, BERT

1. Introduction

Since 2011 was known as the first year of the mobile Internet, the number of online social media and social media users have shown a spurt of growth. In the context of the ever-expanding user community, coupled with the free and interactive characteristics of online social media communication. As a result, many issues worthy of our attention have been exposed in online social media, such as the lack of communication norms and the out-of-control of information dissemination, which makes the dissemination of online social media prone to various negative functions. The negative effects of cyberbullying, online witch hunts, fake news, and privacy violations in social media bring huge risks to individuals, communities, companies, and society as a whole [1]. Therefore, research on how to identify these negative influence speeches in social media has great value and significance. Academia, social media companies, and technology companies have also realized the importance of this issue. They have been investing a lot of technology and funds in identifying offensive languages [2]. This also provides great opportunities and challenges for natural language processing.

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

hublucashb@gmail.com (B. Huang); baiyang.top@gmail.com (Y. Bai)

0000-0002-4203-1935 (B. Huang); 0000-0002-7175-2387 (Y. Bai)

The task goal proposed by the HASOC2020 task organizer team is to identify hate speech and offensive content from the comment data set obtained from the online social media Twitter [3]. The annotated data set provided by the competition organizer team contains three different languages (English, German, and Hindi) [4]. Every language has the same task A and task B. In task A, we need to perform coarse-grained binary classifications of the data set: Non-Hate-Offense (NOT) and Hate and Offensive (HOF). In task B, we need to perform fine-grained quaternary classifications of the same data set: Hate speech (HATE), Offensive (OFFN), Profan (PRFN), and Non-Hate Offensive (NONE). There are many ways to try to complete the requirements of the tasks mentioned above, such as some commonly used methods in machine learning Naive Bayesian Model (NBM), Support vector machine (SVM), K-nearest neighbors (KNN), etc. Besides, there are some methods in deep learning such as Convolutional Neural Networks (CNN), Recurrent Neural Networks(RNN) [5], Long Short-Term Memory(LSTM) [6], etc. After analyzing the data, we learned that the main difficulties and challenges in completing the task are: the amount of data of different categories of labels is not balanced; the distinction between HATE, OFFN, and PRFN in the fine-grained quaternary task is not obvious; and the noise caused by the data itself. However, many recent studies have shown that Pre-Trained Transformer Language Models have achieved state-of-the-art results on many tasks in natural language processing(NLP) [7].

Therefore, combining the excellent performance of Pre-Trained Transformer Language Models on many NLP tasks and the analysis of the best result method model in Semantic Evaluation 2020(SemEval-2020) task 12 by Zampieri et al. [8] In terms of model selection, we use Bidirectional Encoder Representations for Transformers (BERT) [9] and A Lite BERT (ALBERT) [10], and fine-tune the two models to complete the English task and the German task. In terms of data preprocessing, we perform different processing methods according to the different characteristics of task A and task B. In a second step, we used the preprocessed data as input for the fine-tuned model for training. Finally, the trained model is used to predict the result of the test set.

2. Related Work

In recent years, issues related to the identification of hate speech have received increasing attention. In the recently concluded SemEval-2020: Task 12 on Multilingual Offensive Language Identification in Social Media (OffensEval-2020) attracted many participants to participate, making OffensEval-2020 the most popular in SemEval-2020 One of the tasks [8]. The difference between hate speech and other NLP tasks is that hate speech may have strong cultural implications. In other words, in different cultural contexts, the same sentence may or may not be considered offensive [11].

Although the feature sets concerned in different methods of hate speech recognition are very different, in general, these methods are mainly focused on supervised learning. Simple Surface Features similar to the bag-of-words model can provide very clear and easy-to-understand information in text classification tasks. Authors using this method usually merge multiple larger n-grams into a feature set [12] [13]. However, to further improve the performance, the combination with additional features is required. The experiment of Nobata et al. verified this

method [14]. Using bag-of-words similar methods usually require specific words to appear in the training set and test set. Therefore, in later research work, people turned their attention to Word Generalization Features. Use the artificial neural network method to train *word embeddings* on the corpus, and use the distance between the vectors to indicate the semantic similarity of different words. For tasks similar to identifying hate speech, we usually classify entire sentences or a complete paragraph. Djuric et al. proposed a method of directly using embedding to represent the entire text and proved its effectiveness [15].

The Lexical Resources method is to construct a list of words related to hate speech. The vocabulary list contains words and phrases of different performance levels. Besides, each vocabulary item will also have an assigned weight to represent the degree of influence on the discrimination result. However, comparing the two methods mentioned above (Simple SurfaceFeatures and WordGeneralization Features), their common problem is that they fail to explore the role of contextual information [14]. Hate speech is highly dependent on contextual information [12]. Methods such as simply using keywords to identify hate speech cannot achieve the desired results. Dinakar et al. used Knowledge-Based Features to identify homosexual and bisexual-related hate speech [16]. This scheme can use context and knowledge feature information to identify hate speech, but it is limited by the knowledge feature domain framework and is only suitable for a small number of specific tasks. Recent research results show that the Pre-Trained Language Model based on the Transformer architecture has great advantages whether it is at the word level or contextual information [7].

As mentioned in the third paragraph of the Introduction(§1), in this paper, we use two pre-trained language models based on the Transformer architecture: BERT and ALBERT to complete the detection of hate speech in two languages: English and German.

Table 1
The label statistics of the initial training set and test set of English and German in subtask A

Language	Sub-Task A	HOF	NOT	Total
English	Training set	1856	1852	3708
English	Test set	423	391	814
German	Training set	673	1700	2373
German	Test set	134	392	526

3. Data and Methods

3.1. Data Description

The annotation data set provided by the task organizer team comes from tweets on Twitter. We only analyze the data sets of the English tasks and German tasks that we participated in. For subtask A, the label distribution of the English training set is very uniform(50%,50%), on the contrary, the label distribution of the German training set is very uneven(28%,72%). For subtask B, the distribution of English training set and German training set labels in the training set is very imbalanced. The common point of the label distribution of the two languages is that the

NONE label ratio is very high(50%,72%), and the HATE and OFFN label ratios are small. Since the text of the training set comes from tweets, the content contains many non-letter symbols. For example, URL, emoticons, some unknown letter combinations, numbers, etc. Some data examples are given in the Data Preprocessing(§4.1). The training set label statistics for English and German can be found in Table 1 and Table 2.

Table 2
The label statistics of the initial training set and test set of English and German in subtask B

Language	Sub-Task B	HATE	OFFN	PRFN	NONE	Total
English	Training set	158	321	1377	1852	3708
English	Test set	25	82	293	414	814
German	Training set	146	140	387	1700	2373
German	Test set	24	36	88	378	526

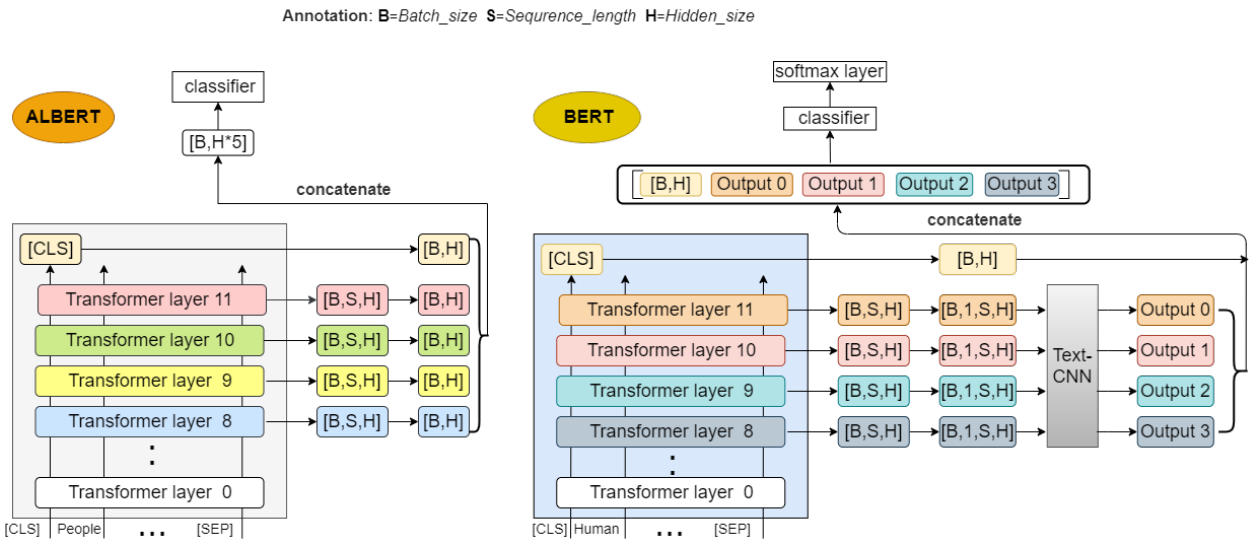

Figure 1: Fine-tuned ALBERT and the fine-tuned BERT

3.2. Fine-tuned of ALBERT and BERT

The architectures of ALBERT base and BERT base are both composed of 12-layer Transformer. Compared with the BERT model, the result of the original embedding parameter P is the product of the vocabulary size V and the hidden layer size H. ALBERT factorizes the Embedding matrix by using a lower-dimensional embedding space of size E and then project it to the hidden space.

$$V * H = P \quad \rightarrow \quad V * E + E * H = P \tag{1}$$

Different from $H=E$ in BERT, when $H \gg E$, the number of parameters of ALBERT has a significant reduction. Another big difference from BERT is that ALBERT's default decision is to share all parameters across layers [10]. Based on these improvements, the training effect of ALBERT is better than that of BERT.

The final result we submitted was obtained using fine-tuned ALBERT and BERT models. For ALBERT, the output matrix shape of the last four layers is the same([*batch_size*, *sequence_length*, *hidden_size*]). we get the 0-dimensional(*batch_size*) and 2-dimensional(*hidden_size*) of the last four layers of matrices, and then splice them with the *pooler_output*(*[CLS]*) output matrix to obtain a new matrix. Finally, input this new matrix to the classifier(*linear classifier*).

For BERT, we get the output matrices of the last four hidden layers. Then, we reshape the output of the last four hidden layers of BERT to obtain four 4-dimensional output matrices ([*batch_size*, *sequence_length*, *hidden_size*] reshape to [*batch_size*, *1*, *sequence_length*, *hidden_size*]). Next, input these four matrices into the TextCNN network[17]. After that, connect the *pooler_output*(*[CLS]*) with the four output matrices of TextCNN. Finally, input the matrix obtained in the previous step into the classifier(*linear classifier*) and a layer of *softmax* [18]. In TextCNN, we use a three-layer network with a convolution kernel size of 3, 4, and 5. The number of convolution kernels is 256. Compared with the sequential structure of LSTM, CNN can better recognize the semantic information of a tweet. Because some orders in tweets will be changed by various noises. The detailed structure of the two models is shown in Figure 1.

4. Experiment and Results

4.1. Data Preprocessing

According to the analysis of the data in the previous chapter (§3.1), we preprocessed special symbol data, numeric data, emoji, character data, stop words, and other symbols. The solution we adopted is to replace some useful information such as names, URLs, numbers, and delete other symbols that need to be preprocessed. Such as all emoticons and all punctuation will be deleted. As is shown in the comparison case below.

- **Before**: RT @airjunebug: When you're from the Bay but you're really a NY nigga at heart♡!!!!!!!!!!!. W/ @supportcaleon https://t.co/mZ8BAYlnlf;
  After : username when you are from the bay but you are really newyork nigga at heat username url
- **Before**: "Gutschein ""-50% REBELITA Fleecepulover für Frauen!"" https://t.co/EC0VX6LWYj;
  After : guts che number rebel fleece pul over fr frauen url

4.2. Experiment setting

Refer to the analysis of Zampieri et al. in the OffensEval-2020 competition [8]. For the English subtask A, we use the fine-tuned ALBERT, and the English subtask B and the German task we use the fine-tuned BERT. For the training data set, we use preprocessed data in English

Table 3
Comparison of the results of subtask A and subtask B on the validation set when the data is preprocessed and without preprocessing

Task type/score	Unprocessed data set		Processed data set	
Subtask A F1-score	English subtask A 0.8750	German subtask A 0.8031	English subtask A 0.8932	German subtask A 0.8141
Subtask B F1-score	English subtask B 0.6142	German subtask B 0.6070	English subtask B 0.6041	German subtask B 0.5925

and German subtask A, while in English and German subtask B we use data that has not been preprocessed. Because in the course of our experiments, we found that the preprocessed data can have a higher score in subtask A, on the contrary, the data set that is not preprocessed in subtask B has a higher score. We analyze this result because subtask B is a more fine-grained classification task, and the symbols preprocessed by us may be helpful to the classification result. For example, information about the frequency of URL mentions and punctuation, comments and token length, capital letters, words not found in English dictionaries, and the number of non-alphanumeric characters in the token [11]. In the process of using BERT for training, we unfreeze the first four layers in BERT-base.

1. **English subtask A:** We choose the ALBERT-base-v2[1] English pre-training model for fine-tuning (§3.2), and then perform five-fold cross-validation on the training set. The loss function chooses BCEWithLogitsLoss (Binary Cross Entropy With Logits Loss) function provided by PyTorch. The epoch, batch size, maximum sequence length, and learning rate for the subtask are 5, 32, 50, and 4e-5, respectively.
2. **English subtask B:** In this subtask, we use the English BERT-base pre-training model. For the data set, split the original unprocessed training set into a new training set and a validation set in proportion (8:2). The loss function selects the CrossEntropyLoss function provided by PyTorch. The activation function in TextCNN is Mish. The epoch, batch size, maximum sequence length, and learning rate for the subtask are 5, 32, 150, and 5e-5, respectively.
3. **German subtask A:** For the German subtask A, we choose the German BERT[2] pre-training model. The loss function chooses BCEWithLogitsLoss (Binary Cross Entropy With Logits Loss) function. After the original training set is preprocessed, it is divided into a new training set and a validation set in proportion(8:2). The epoch, batch size, maximum sequence length, and learning rate for the subtask are 5, 32, 50, and 4e-4, respectively.
4. **German subtask B:** Different from German subtask A, the German subtask B chooses the CrossEntropyLoss function in the loss function. The epoch, batch size, maximum sequence length, and learning rate for the subtask are 5, 32, 150, and 3e-5, respectively.

[1]https://huggingface.co/albert-base-v2
[2]https://huggingface.co/bert-base-german-cased

4.3. Analysis of Results

In this competition, the evaluation index given by the task organizer is macro-average F1-score. The final published results are verified by the competition organizers on a private data set. In subtask A and subtask B, we use preprocessed and unprocessed data for training respectively. In our experiment, subtask A uses preprocessed data to get better results, and subtask B has poor results on the preprocessed data. Their comparison results when tuned to the best effect are shown in Table 3.

In the English subtask A, the training data is relatively balanced, so the prediction result of the model is not very bad. But in the German subtask A, due to the imbalance of the training set, and the results we submitted did not use k-fold cross-validation. As a result, the generalization ability of the model is not very strong, and the result is not ideal. For the two subtasks B, the training data is not only unbalanced, but also the emotional tendency of the labels is very similar, so the scores obtained by the model are generally low. Our score and the best result score in the final results announced by the competition organizer team can be obtained in Table 4.

Table 4
In the English task and the German task, our score and the best result score

Language	subtask A	Subtask B
English	Best F1-score:0.5152 Our F1-score:0.4917	Best F1-score:0.2652 Our F1-score:0.2649
Geman	Best F1-score:0.5235 Our F1-score:0.4953	Best F1-score:0.2831 Our F1-score:0.2567

5. Conclusion

The work described in this paper is submitted to HASOC (2020): Hate Speech and Offensive Content Identification in Indo-European Languages. In our work, we have conducted experiments and researches on all tasks belonging to the two languages of English and German. We use fine-tuning ALBERT and fine-tuning BERT+TextCNN to complete this task. In English subtask A, we use ALBERT and 5-fold cross-validation, and our result is only 0.0235 lower than the first place. In the other three subtasks, we use the BERT-based model to complete the experiment. In the experiment, we also explore the impact of data preprocessing on the results of different tasks. In the binary classification task, the result obtained by using the preprocessed training set data is better than the result obtained by using the original training set data. In the fine-grained quaternary task, using the training set data without preprocessing has better results than using the processed training set data. In future work, we hope that better models and methods can be used to improve the effect of identifying hate speech.

References

[1] C. V. Baccarella, T. F. Wagner, J. H. Kietzmann, I. P. McCarthy, Social media? it's serious! understanding the dark side of social media, European Management Journal 36 (2018) 431–438. doi:10.1016/j.emj.2018.07.002.

[2] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, R. Kumar, Predicting the type and target of offensive posts in social media, arXiv preprint arXiv:1902.09666 (2019). doi:10.18653/v1/n19-1144.

[3] 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.

[4] B. r. Chakravarthi, Leveraging orthographic information to improve machine translation of under-resourced languages, Ph.D. thesis, NUI Galway, 2020. URL: http://hdl.handle.net/10379/16100.

[5] A. Karpathy, The unreasonable effectiveness of recurrent neural networks, Andrej Karpathy blog 21 (2015) 23. doi:10.5220/0006922101420153.

[6] C. Olah, Understanding lstm networks, 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[7] C. Wang, M. Li, A. J. Smola, Language models with transformers, arXiv preprint arXiv:1904.09408 (2019).

[8] M. Zampieri, P. Nakov, S. Rosenthal, P. Atanasova, G. Karadzhov, H. Mubarak, L. Derczynski, Z. Pitenis, Ç. Çöltekin, Semeval-2020 task 12: Multilingual offensive language identification in social media (offenseval 2020), arXiv preprint arXiv:2006.07235 (2020).

[9] 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).

[10] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, Albert: A lite bert for self-supervised learning of language representations, arXiv preprint arXiv:1909.11942 (2019).

[11] A. Schmidt, M. Wiegand, A survey on hate speech detection using natural language processing, in: Proceedings of the Fifth International workshop on natural language processing for social media, 2017, pp. 1–10. doi:10.18653/v1/w17-1101.

[12] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, S. Mishra, Detection of cyberbullying incidents on the instagram social network, arXiv preprint arXiv:1503.03909 (2015).

[13] Z. Waseem, D. Hovy, Hateful symbols or hateful people? predictive features for hate speech detection on twitter, in: Proceedings of the NAACL student research workshop, 2016, pp. 88–93. doi:10.18653/v1/n16-2013.

[14] 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. doi:10.1145/2872427.2883062.

[15] 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. doi:10.1145/2740908.2742760.

[16] K. Dinakar, B. Jones, C. Havasi, H. Lieberman, R. Picard, Common sense reasoning for detection, prevention, and mitigation of cyberbullying, ACM Transactions on Interactive Intelligent Systems (TiiS) 2 (2012) 1–30. doi:10.1145/2362394.2362400.

[17] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882 (2014). doi:10.3115/v1/d14-1181.

[18] C. Sun, X. Qiu, Y. Xu, X. Huang, How to fine-tune bert for text classification?, in: China National Conference on Chinese Computational Linguistics, Springer, 2019, pp. 194–206.