

Lee@HASOC2020: ALBERT-based Max Ensemble with Self-training for Identifying Hate Speech and Offensive Content in Indo-European Languages

Junyi Li, Tianzi Zhao

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

Abstract

This paper describes the system submitted to HASOC 2020. This task aims to identify hate speech and offensive content in Indo-European languages . We only participate in the English part of subtask A, which aims to identify hate speech and offensive content in English. To solve this problem, we propose an ALBERT-based model , and use the self-training and max ensemble to improve model performance. Our model achieves a macro F1 score of 0.4976 (ranks 20/35) in subtask A.

Keywords

Hate Speech and Offensive Content, Indo-European Languages, Self-training, ALBERT, Max Ensemble,

1. Introduction

In recent years, due to the rapid development of the mobile Internet and social media platforms, people have begun to use various social media to share their lives, such as Facebook and Twitter. People share their views on life on social media, and this behavior may receive good or bad comments. Some bad comments slowly evolved into offensive language. Social media is flooded with a lot of offensive language [1], and these remarks have led to deviations in people's perception of things. Therefore, all major social media platforms urgently need tools to automatically monitor user speech [2].

HASOC 2020 [3] is a data challenge to identify hate speech in multiple languages. Its goal is to use computational methods to identify offensive and hate speech in user-generated content on online social media platforms. This task provides posts from social media platforms and classifies this content. At the same time, the application of multiple languages greatly broadens our recognition range.

In this task, we only participate in subtask A of English language : Identifying hate, offensive and profane content. In the task, the main problem to be solved is how to get the best task performance. In order to solve the problem that affects task performance, this paper proposes a method that combines two effective strategies. First, we introduced an external data set [4] to increase the amount of training data and avoid overfitting of model training. Secondly, We use


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

✉ 18314327187@163.com (J. Li)

🆔 0000-0002-7162-5396 (J. Li); 0000-0002-8908-2537 (T. Zhao)



© 2020 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

an ALBERT-based model with model self-ensemble. Through many experiments, this method can get a good task performance, which can effectively solve the problem.

The rest of our paper is structured as follows. Section 2 describes data preparation. Methods are described in Section 3. Experiments and evaluation are described in Section 4. The conclusions are drawn in Section 5.

2. Data and Data Preparation

2.1. Data

The organizers provided training and test datasets, containing 3708 and 814 sentences respectively. We counted the number and distribution of labels in the dataset. The number of labels in the dataset is shown in Figure 1.

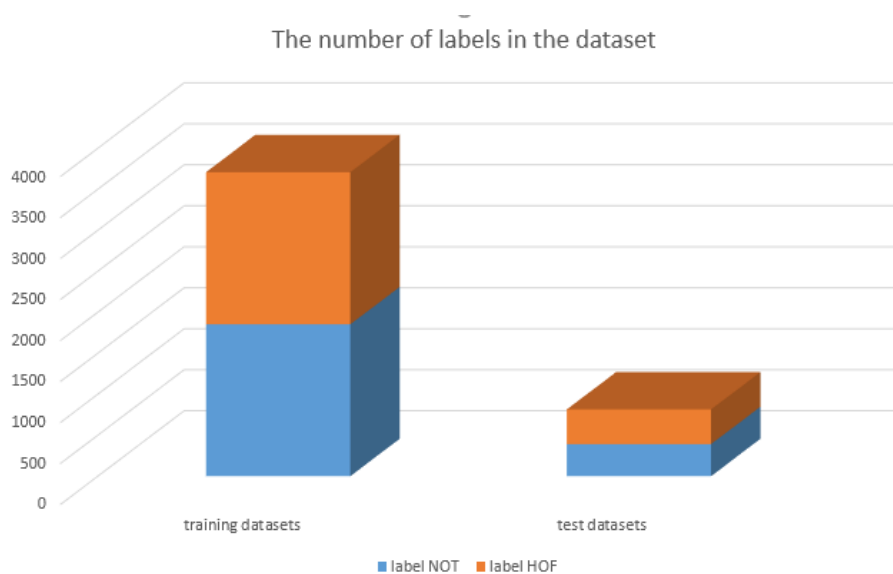


Figure 1: The number of labels in the dataset

where NOT means that This text does not contain any Hate speech, profane, offensive content and HOF means that its contains hate, offensive, and profane content.

2.2. Data Preparation

Some text in the tweets has no effect on the meaning expressed. Tweets are processed using the tweettokenize tool [5]. Cleaning the text before further processing helps to generate better functionality and semantics. We perform the following preprocessing steps.

- We know that some repeated symbols have no meaning. As a result, repeated periods, question marks and exclamation marks are replaced with a single instance with the special mark "repeat" added.

- All contractions were changed to complete parts. This helps the machine understand the meaning of words (for example: "there're" changed to "there" and "are").
- Twitter data contains a lot of emojis. Emojis can cause the number of unknown words to rise, which can lead to poor pre-training effects. Emoticons (for example, ":(", ":", "": P" and emoticons, etc.) are replaced by emotional words with their own meaning. This will improve the pre-training effect.
- Generally, words have different forms according to the change of context. However, different forms of words will cause ambiguity in pre-training and affect the effect of pre-training. Lexicalization, through WordNetLemmatizer to restore language vocabulary to the general form (can express complete semantics).
- Tokens are converted to lower case.

3. Methods

3.1. Self-training

In Natural Language Processing (NLP), using different data in the same domain to train a model is a form of model training. This training method is called self-training [6], and aims to establish a broad semantic understanding to promote performance improvement for training and test tasks.

In this paper, we use self-training to train the model. The self-training process of our model is shown in Figure 2. Our self-training method uses the idea of "Teacher and student". "Teacher and student" refers to the same training process. The beginning of student training is the end of teacher training, which can deepen the learning of the model. We use the "Teacher-Student" method to design model self-training. The "Teacher" part uses an additional external dataset, which is the task dataset [7] of task12 from SemEval2020. This dataset has the same purpose as the HASOC2020 task, but the content of the data is different. We only randomly used 10,000 sentences. The student part uses the training dataset of the task. Experiments show that our design is an effective method.

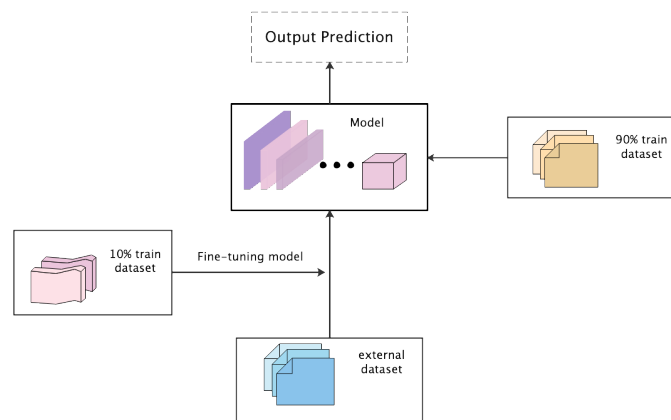


Figure 2: The architecture of self-training

3.2. ALBERT

Google has introduced a new language representation model called BERT [8], which stands for Bidirectional Encoder Representations from Transformers. However, the large-scale parameters of the BERT lead to an exponential increase in training time and a shortage of computing resources. Meanwhile, too much parameter ratio will cause the performance of the model to decrease. Therefore, Google and Toyota Institute of Technology proposed a new model, the ALBERT model.

The ALBERT [9] model combines two parameter reduction techniques, which eliminate the main obstacles to scaling pre-trained models. The first is decomposed embedded parameterization. This separation makes it easier to increase the hidden size without significantly increasing the parameter size of the vocabulary embedding. The second technique is cross-layer parameter sharing. This technique prevents the parameters from increasing with the depth of the network. Through these two technologies, the ALBERT model performs better when the number of parameters decreases.

In this task, we use the ALBERT model to get good performance. Our model is shown in Figure 3.

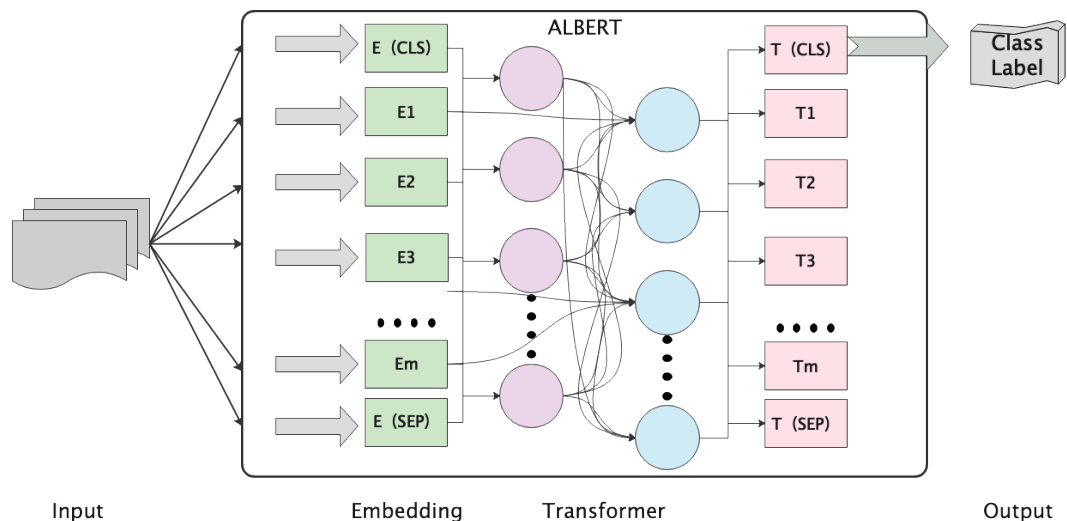


Figure 3: The architecture of the model

3.3. Max Ensemble

In this paper, we hope to make full use of the ALBERT model through better fine-tuning strategies, so as to achieve the best task performance. In fact, fine-tuning the performance of ALBERT is usually sensitive to different random seeds and orders of the training data. In order to alleviate this situation, ensemble methods can be used to reduce overfitting and improve model generalization ability. Therefore, ensemble [10] methods are widely used to combine multiple fine-tuned models. The ensemble ALBERT model usually has higher performance than

a single ALBERT model.

We know that the common ensemble method is based on voting [11]. In this paper, we fine-tune multiple ALBERT models with different random seeds. For each input, we will output the best prediction and probability derived from the fine-tuned ALBERT, and summarize the predicted probability of each model. The output of the ensemble model is the prediction with the highest probability. This ensemble method is called the max ensemble. The formula for the max ensemble [12] we used is shown below

$$ALBERT_{vote}(x; s) = \text{Max}(\sum_{n=1}^s ALBERT(x_s)) \quad (1)$$

where $ALBERT(x_s)$ represents a fine-tuning of the ALBERT model.

4. Experiments and Evaluation

In this task, we use self-training and max ensemble ALBERT-based model. For the ALBERT model, the main hyper-parameters we focused on are the training step size, batch size, warm steps, and learning rate. After learning the hyper-parameter adjustment for similar tasks, we fine-tune the model hyper-parameters. As is shown in Table 1.

Table 1

Details of the hyper-parameters.

train step	learning rate	batch size	warm steps
23800	5e-6	32	1256

This task mainly uses F1 macro-average score for performance evaluation. To test the effectiveness of our method, we conduct ablation experiments for our method. Our experimental results based on test dataset are shown in Table 2.

Table 2

Performance with our methods on test dataset.

Method	F1 macro-average
ALBERT(self-training/o)	0.83
ALBERT(max ensemble/o)	0.85
ALBERT(our methods)	0.90

where $ALBERT(self-training/o)$ means that only self-training is not used. $ALBERT(maxensemble/o)$ means that only max ensemble is not used. $ALBERT(ourmethods)$ means that we use self-training and max ensemble ALBERT-based model.

From this table, we can see that the self-training and the max ensemble method can effectively optimize the effectiveness of ALBERT model. So, for this task, our method can get a good performance.

5. Conclusion

In this task, our main consideration is how to get a good task performance. In other words, we need to adopt methods to optimize the performance of our models. We mainly use the ALBERT model. Based on the model, we also adopt the method of self-training and max ensemble. Experiments prove that our method can achieve the best performance.

However, in the ranking, the performance of our model is still not satisfying. In the future, we will improve our methods by adjusting our model and trying more ensemble methods.

Acknowledgements

This work was supported by the Science Foundation of Yunnan Education Department under Grant 2020Y0011.

References

- [1] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, R. Kumar, Predicting the Type and Target of Offensive Posts in Social Media, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2019, pp. 1415–1420.
- [2] T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandlia, A. Patel, Overview of the HASOC track at FIRE 2019: Hate speech and offensive content identification in indo-european languages, in: Proceedings of the 11th Forum for Information Retrieval Evaluation, 2019, pp. 14–17.
- [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] S. Rosenthal, P. Atanasova, G. Karadzhov, M. Zampieri, P. Nakov, A Large-Scale Weakly Supervised Dataset for Offensive Language Identification, in: arxiv, 2020.
- [5] R. K. Bakshi, N. Kaur, R. Kaur, G. Kaur, Opinion mining and sentiment analysis, in: 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), IEEE, 2016, pp. 452–455.
- [6] I. Z. Yalniz, H. Jégou, K. Chen, M. Paluri, D. Mahajan, Billion-scale semi-supervised learning for image classification, arXiv preprint arXiv:1905.00546 (2019).
- [7] H. Mubarak, A. Rashed, K. Darwish, Y. Samih, A. Abdelali, Arabic offensive language on twitter: Analysis and experiments, arXiv preprint arXiv:2004.02192 (2020).
- [8] 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).
- [9] 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).

- [10] S. Avidan, Ensemble tracking, *IEEE transactions on pattern analysis and machine intelligence* 29 (2007) 261–271.
- [11] A. Onan, S. Korukoğlu, H. Bulut, A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification, *Expert Systems with Applications* 62 (2016) 1–16.
- [12] Y. Xu, X. Qiu, L. Zhou, X. Huang, Improving bert fine-tuning via self-ensemble and self-distillation, *arXiv preprint arXiv:2002.10345* (2020).