YUN_DE at HASOC2020 subtask A: Multi-Model Ensemble Learning for Identifying Hate Speech and Offensive Language

Zichen Zhang, Yuhang Wu and Hao Wu*

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

Abstract

This paper describes our system in subtask A of HASOC2020: Hate Speech and Offensive Content Identification in Indo-European Languages. We propose a method of multi-model ensemble learning, which includes BERT, ON-LSTM, and TextCNN models. The multi-model ensemble aims to make better results about text classification than the single model. Our system achieves the Macro average F1-score of 0.5017 and is ranked 11th on the final leader board of the competition among the 36 teams.

Keywords

Text Classification, BERT, ON-LSTM, TextCNN, Ensemble Learning

1. Introduction

With the popularity of social media, much of the world communicates on it, for example, nearly a third of the world's population active on Facebook alone. Meanwhile, anyone can publish content and access content of interest in these platforms [1]. However, it provides space for discourses that include *offensive content and hate speech*, which is used to express hatred towards a targeted group or be derogatory to the members of the group [2]. These words have seriously destroyed social harmony, even led to violence and conflicts. Although many social media platforms confronting the trend have created their provision to against someone's hate speech under laws prohibiting hate speech, they have to need an efficient automatic detection system facing the timely transmission of massive hate speech.

In recent years, lots of researchers in industry and academia have developed some natural language processing (NLP) techniques for detecting hate speech and offensive content. The QutNocturnal team utilized Convolutional Neural Networks (CNN) to identify whether tweets are hate speech [3]. Then, Manolescu et al. proposed a system based on Long Short-Term Memory (LSTM) with an embedding layer, for detecting hate speech against immigrants and women in Twitter [4]. Ordered Neurons LSTM (ON-LSTM) has integrated the hierarchical structure (tree structure) into the LSTM, which allowed the LSTM to automatically learn the hierarchical structure information [5], so Wang et al. proposed ON-LSTM with attention for identifying hate speech and offensive language, and use the K-fold ensemble approach to

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

zczhang@mail.ynu.edu.cn (Z. Zhang); yuhangwu@mail.ynu.edu.cn (Y. Wu); haowu@ynu.edu.cn (H. Wu*)

0000-0001-6716-3339 (Z. Zhang); 0000-0003-0690-5364 (Y. Wu); 0000-0002-3696-9281 (H. Wu*)

enhance the performance [6]. Mozafari et al. investigated the ability of BERT [7] at capturing hateful context within social media content [8]. These methods can effectively detect hate speech, however they just take advantage of themselves. After all the ability of the single model is limited. A good solution is to integrate multiple models to gain the integration advantage.

HASOC2020 aims at identifying hate speech and offensive content in Indo-European Languages [9], which provides a forum and a data challenge for multilingual research on the identification of problematic content. For subtask A, it needs to classify tweets into two categories: *hate and offensive* (HOF), *non-hate and offense* (NOT). We participated in it for the English language and proposed an ensemble learning method that includes BERT, ON-LSTM, and TextCNN models. It can integrate the advantages of each model to enhance the effectiveness of the detection of hate speech and offensive content. The experimental results indicated that our method performs better than the single model and achieves the Macro F1-score of 0.8896.

2. Method

2.1. BERT model

BERT is mainly composed of Bidirectional Transformers blocks, which overcomes the problem of invariance of word vector representation in previous models. It can generate different word vectors of the same word according to different contexts. BERT achieves excellent results in many natural language tasks these years.

In Google's research, a large number of high-quality texts are used to pre-train through self-supervised methods. The language knowledge contained in texts is encoded and transferred to Bi-Transformer for training, and their pre-training model parameters[1] is released. Often we only need to fine-tune the pre-trained model to deal with most text classification tasks.

Because the Bi-Transformer cannot remember the time sequence information, we add the **[CLS]** flag to the head of the input text to indicate whether or not it is used for a classification task. Then, we extract all the first elements of rows from the output of BERT and send it to a simple fully connected layer to output the classification results.

2.2. TextCNN model

TextCNN is a variant of the CNN architecture. Firstly, the sentence is embedded with word vectors represented as $[x_1, x_2, \dots, x_n]$, where x_i is the k-dimensional word vector corresponding to the i-th word in the sentence. A convolution operation involves a filter which is applied to a window of h words to produce a new feature. For example, a feature z_i is generated from a window of words $[x_i : x_{i+h-1}]$ as follows:

$$z_i = f(w[x_i : x_{i+h-1}] + b) \tag{1}$$

where w is a weight matrix, b is a bias vector and $f(\cdot)$ is a non-linear function. This filter is applied to $n - h + 1$ possible windows to produce a feature matrix $Z = [z_1, z_2, \dots, z_{n-h+1}]$. Then it takes the maximum value z_{max} from Z by 1-max pooling layer. Therefore, we adopt m

[1] https://github.com/google-research/bert

filters with different window sizes to achieve the above process, and combine the maximum value of each filter into a vector $V = \left[z_{max}^1, z_{max}^2, \ldots, z_{max}^m\right]$. Finally, we get this model output the probability of each category by a fully connected softmax layer.

2.3. ON-LSTM model

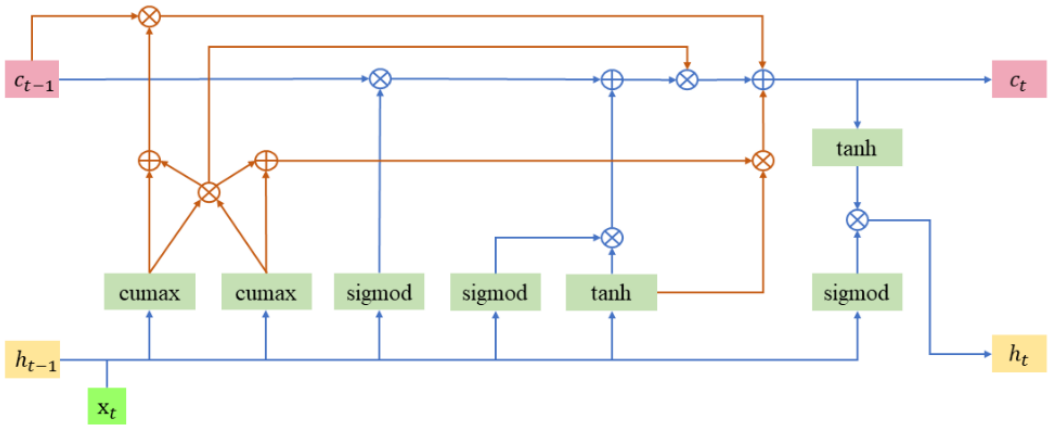

Figure 1: The unit of ON-LSTM model

ON-LSTM is a new variant of LSTM, whose unit architecture is shown in Figure1. It uses an architecture similar to the standard LSTM, so is also composed of forget gate f_t, input gate i_t and output gate o_t. The formula is as follows:

$$\begin{aligned} f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right), \\ i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right), \\ o_t &= \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \end{aligned} \tag{2}$$

where x_t is the input variable at time t, h_{t-1} corresponds to the hidden layer at time $t-1$, U and W are weight matrices, b is the bias vector, and $\sigma(\cdot)$ is a sigmod function.

However, the different between ON-LSTM and LSTM is to enforce an order to the update frequency, we introduce a new activation function, the specific formula is as follows:

$$\begin{aligned} \tilde{f}_t &= \vec{cs}\left(softmax\left(W_{\tilde{f}} x_t + U_{\tilde{f}} h_{t-1} + b_{\tilde{f}}\right)\right), \\ \tilde{i}_t &= 1 - \vec{cs}\left(softmax\left(W_{\tilde{i}} x_t + U_{\tilde{i}} h_{t-1} + b_{\tilde{i}}\right)\right), \\ \omega_t &= \tilde{f}_t \circ \tilde{i}_t, \\ \hat{c}_t &= tanh\left(W_c x_t + U_c h_{t-1} + b_c\right), \\ c &= \omega_t \circ \left(f_t \circ c_{t-1} + i_t \circ \hat{c}_t\right) + \left(\tilde{f}_t - \omega_t\right) \circ c_{t-1} + \left(\tilde{i}_t - \omega_t\right) \circ \hat{c}_t \end{aligned} \tag{3}$$

where $\vec{cs}$ is an accumulative function, which represents the influence of historical information and current information. the $\tilde{f}_t$ and $\tilde{i}_t$ are called master forget gate and master input gate respectively. ω_t gives the vector where the intersection is 1 and the rest is 0. In this way,

restructuring the output of long-term memory c, the high-level information may be stored for a long time, but the low-level information may be updated at each step of input, so the hierarchical structure is embedded by information hierarchy.

2.4. Ensemble learning

We use the multi-model ensemble learning approach to get a stable system that performs well in all aspects. We further use *hard voting* to determine the final category, whose main idea is to vote for a speech by the classification results of each model and the minority obeys the majority. Thus, the system integrates the models of TextCNN, BERT, and ON-LSTM by ensemble learning, as showed in Figure 2.

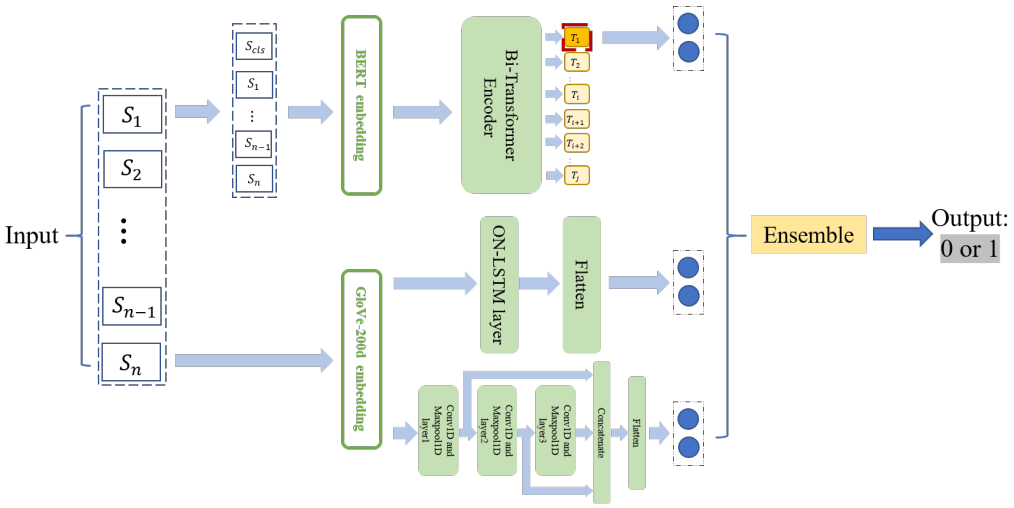

Figure 2: Multi-model ensemble learning

With input of speech $[S_1, S_2, \ldots, S_n]$, the output value of each model is 0 or 1 which is represented as a vote p_i. Adding all the values together to average, we will get the final score of a candidate, as follows:

$$score = \frac{1}{num} \sum_{i=0}^{num} p_i \tag{4}$$

where $score \geq 0.5$ shows that the speech is NOT, otherwise HOF. num in as the number of models.

3. Experiment

3.1. Dataset

We divided the training set and valid set from the HASCO2020 data in English. Statistics of the dataset are shown in Table 1, where data is relatively balanced and does not need us to do distribution processing.

Table 1
Statistics of the dataset.

Dataset	NOT	HOF	Total
Train set	6200	4327	10527
Dev set	499	501	1000

3.2. Implementation details

To achieve better results, we clean and preprocess the texts, mainly including the following steps:

- Using regular expressions to remove users and topics.
- Convert emoji into language expression.
- Check the spelling of words.
- Restore abbreviations.
- Remove URL link.

To ensure the objectivity and fairness of all experiments, we set all models according to the hyperparameters: Adam optimizer, Learning rate=5e-6, epoch=15, batch size=32. And for TextCNN and ON-LSTM model, we use glove.twitter.27B.200d[2] for word embedding.

As for all model training, we adopt the classical cross-entropy loss function, which is used to measure the approximate degree between the predicted data distribution and the real data distribution. And the model learns quickly to achieve the best performance by it. The formula as it follows:

$$CE(p, Y) = \frac{1}{n}\sum_{i=1}^{n} -Y_i log(p_i) - (1 - Y_i) log(1 - p_i) \tag{5}$$

where Y_i is value of label (0 or 1), p_i is probability of Y_i.

3.3. Result analysis

To optimize the prediction results, we first trained each model to obtain the parameters under the best F1-score. The loss of each iteration in the training process is shown in Figure 3. We found that the best performance of each model was not when the loss value reach the lowest, especially ON-LSTM can achieve the best performance in the second epoch. Meanwhile, we save the model parameters when each model has the best performance for making ensemble learning more effective.

Then we compared the multi-model ensemble learning approach to the single model about prediction results, which used Macro F1-score, Precision, and Accuracy to evaluate the prediction

[2]https://github.com/stanfordnlp/GloVe

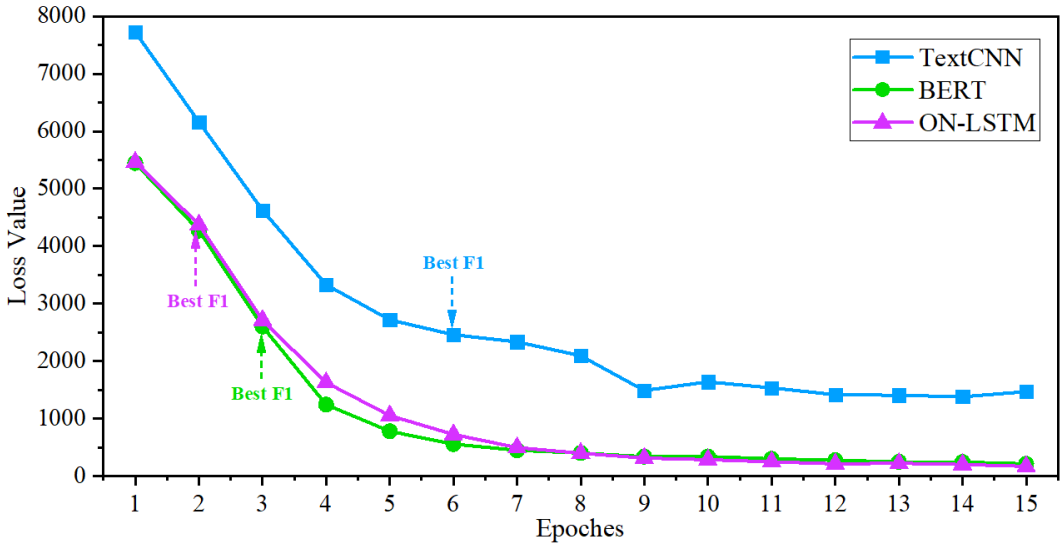

Figure 3: Training process under different model

results, as shown in Table 2. We can see our multi-model ensemble method has improved over the single models on three indicators, especially the Macro F1-score is improved by 5.4% compared with TextCNN. It demonstrated that multi-model ensemble learning can integrate the strengths of each model and reduce the negative impact of the single model on the results.

Table 2
Prediction results under different methods

Method	Macro F1-score	Precision	Accuracy
TextCNN	0.8355	0.7935	0.826
BERT	0.8789	0.9002	0.884
ON-LSTM	0.8761	0.8992	0.879
Ensemble_soft_voting	0.8864	0.9010	0.888
Ensemble_hard_voting	**0.8896**	**0.9119**	**0.892**

Besides, we also compared the different ensemble approaches, using our method based on *hard voting*(Ensemble_hard_voting) versus it based on *soft voting*(Ensemble_soft_voting). We found that *hard voting* is better because it is only to obey most of the model and do not need to synthesize all the model opinions compared to *soft voting*. This can reduce the impact of individual models on prediction results.

4. Conclusions

In this paper, we presented a system to detect hate speech and offensive language for the English language, which uses a method of multi-model ensemble learning for identifying such content. We achieved better results than the single model in subtask A for the English language. In

future research, we will consider a more efficient ensemble method to further enhance the performance of the model.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61962061), partially supported by the Yunnan Provincial Foundation for Leaders of Disciplines in Science and Technology, Top Young Talents of "Ten Thousand Plan" in Yunnan Province, the Program for Excellent Young Talents of Yunnan University.

References

[1] M. Mondal, L. A. Silva, F. Benevenuto, A measurement study of hate speech in social media, in: Proceedings of the 28th acm conference on hypertext and social media, 2017, pp. 85–94.

[2] T. Davidson, D. Warmsley, M. Macy, I. Weber, Automated hate speech detection and the problem of offensive language, arXiv preprint arXiv:1703.04009 (2017).

[3] M. A. Bashar, R. Nayak, Qutnocturnal@ hasoc'19: Cnn for hate speech and offensive content identification in hindi language, arXiv preprint arXiv:2008.12448 (2020).

[4] M. Manolescu, D. Löfflad, A. N. M. Saber, M. M. Tari, Tueval at semeval-2019 task 5: Lstm approach to hate speech detection in english and spanish, in: Proceedings of the 13th International Workshop on Semantic Evaluation, 2019, pp. 498–502.

[5] Y. Shen, S. Tan, A. Sordoni, A. Courville, Ordered neurons: Integrating tree structures into recurrent neural networks, arXiv preprint arXiv:1810.09536 (2018).

[6] B. Wang, Y. Ding, S. Liu, X. Zhou, Ynu_wb at hasoc 2019: Ordered neurons lstm with attention for identifying hate speech and offensive language., in: FIRE (Working Notes), 2019, pp. 191–198.

[7] 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).

[8] M. Mozafari, R. Farahbakhsh, N. Crespi, A bert-based transfer learning approach for hate speech detection in online social media, in: International Conference on Complex Networks and Their Applications, Springer, 2019, pp. 928–940.

[9] 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.