YNUWB at SemEval-2019 Task 6: K-max pooling CNN with average meta-embedding for identifying offensive language

Bin Wang, Xiaobing Zhou*, Xuejie Zhang
School of Information Science and Engineering
Yunnan University, Yunnan, P.R. China
*Corresponding author zhouxb@ynu.edu.cn

Abstract

This paper describes the system submitted to SemEval 2019 Task 6: OffensEval 2019. The task aims to identify and categorize offensive language in social media, we only participate in Sub-task A, which aims to identify offensive language. In order to address this task, we propose a system based on a K-max pooling convolutional neural network model, and use an argument for averaging as a valid meta-embedding technique to get a meta-embedding. Finally, we use a cyclic learning rate policy to improve model performance. Our model achieves a Macro F1-score of 0.802 (ranked 9/103) in the Sub-task A.

1 Introduction

In the past ten years, with the popularity of the Internet, social media platforms such as facebook and twitter have gradually become important tools for people's daily communication, and users can publish their own content on these platforms. As the number of people interacting on social media platforms increases, online aggression language behavior also grows, and now it has become a major source of social conflict.

Semeval 2019 Task 6 is proposed for identifying online offensive languages (Zampieri et al., 2019b). Its goal is to use computational methods to identify offense, aggression and hate speech in user-generated content on online social media platforms. We can prevent abuse of offensive language by using this approach in social media platforms. This task gives us some data from the social media platform, and classifies the content through computational analysis.

In this competition, we only participate in Sub-task A: identification of offensive language. For this task, we use a deep learning method to build a K-max pooling convolutional neural network model which uses convolutional neural networks of different filters to extract features and preserves k largest eigenvalues during the pooling phase. We use two different pre-training vector models (fastText and Glove) to obtain a more accurate meta-embedding with a simple averaging technique (Coates and Bollegala, 2018). In addition, we have adopted a Cyclic Learning Rate (CLR) strategy (Smith, 2017), which avoids the process to find the optimal learning rate, and the learning rate varies within a reasonable interval rather than monotonically. Unlike the Adative Learning Rate, the CLR does not require additional calculations.

The rest of the paper is structured as follows: In section 2, we describe some of the relevant research work. In section 3, we describe the task data and how to build the model. In section 4, we describe the experimental results.

2 Related Work

In recent years, offensive language has prevailed in social media, and people are increasingly interested in identifying offensive speech, especially on social media platforms. This topic has attracted the attention of a large number of researchers in industry and academia. The field of Hate Speech Automatic Detection in the text has unquestionable social impact potential, especially in online communities and digital media platforms (Fortuna and Nunes, 2018). In this section, we will review some of the studies and briefly discuss their findings.

Dinakar et al. decomposed the overall detection problem into detection of sensitive topics, incorporated itself into the text classification subquestion, and solved the problem of text cyberbullying detection by constructing a separate topic-sensitive classifier(Dinakar et al., 2011). Burnap et al. used probabilities, based on the combination of rules and space-based classifiers and voting element classifiers to predict the possible spread of network hatred in Twitter data samples (Burnap and Williams, 2015). Kwok et al. used supervised machine learning methods to obtain tagged data from different Twitter accounts in an inexpensive way, and to learn the binary classifiers of the "racist" and "nonracist" tags (Kwok and Wang, 2013). Gambäck et al. built a convolutional neural network model based on word2vec embedding(Gambäck and Sikdar, 2017). And a new method for deep neural networks based on convolution and gated recursive networks was proposed by Zhang et al. (Zhang et al., 2018). In the field of research on hate speech, originally Xu et al. introduced the social study of bullying and formulated it as NLP tasks(Xu et al., 2012), then Ross et al. suggested that the existence of hate speech should not be considered as a binary yes or no decision, and the evaluator needs a more detailed commentary(Ross et al., 2016), and now ElSherief et al. believed that it is necessary to further deepen the understanding of online hate language to determine whether the target is individual or group (ElSherief et al., 2018).

3 Methodology and Data

3.1 Data description

In this task, we only use the official training data set for training and trial data set to verify. The official data provided by OLID is mainly from Twitter (Zampieri et al., 2019a). In Sub-task A, the purpose is to distinguish whether the tweet is offensive, so the data is divided into two categories: Not Offensive (NOT): Posts that do not contain offensive or defamatory; and Offensive (OFF): This category includes insults, threats, and posts that contain defamatory or cursed words. The training data set has a total of 13240 tweets, in which there are 8840 of NOT and 4400 of OFF, the ratio is about 2:1. The data is slightly unbalanced, but we have not dealt with the data imbalance problem.

3.2 K-max pooling CNN model

Our network architecture is shown in Figure 1. It is a variant of the CNN model structure proposed by Yoon Kim (Kim, 2014). Next we explain the details of our system.

- **Input layer:** This layer mainly inputs all the preprocessed text data into the model.

- **Embedding layer:** Converting text into word embeddings represents each word of the text with a d dimensional vector by using a pre-trained word vector model.

- **Convolutional layer:** In this layer, the obtained word vectors are subjected to convolution operations to obtain multiple feature maps. The specific operation is: a sentence contains L words, each of which has a dimension of d after the embedding layer, and forms a $L * D$ sentence representation by splicing L words. There are several convolution kernels in the convolutional layer, the size of which is $N * d$, and N is the filter window size. The convolution operation is to apply a convolution kernel to create a new feature in a matrix that is spliced by words. Its formula is as follows:

$$C_l = f(w * x_{(l:l+N-1)} + b) \qquad (1)$$

where l represents the lth word, c_l is the feature, w is the convolution kernel, b is the bias term, and f is a nonlinear function. After the convolution operation of the whole sentence, a feature map is obtained, which is a vector of size $L + N - 1$.

- **Pooling layer:** The main function of this layer is to perform dimensionality reduction on the features of filter to form the final feature with a K-max pooling operation, which takes the value of the scores in Top K among all the feature values, and retains the original order of these feature values. Obviously, K-max Pooling can express the same type of feature multiple times, that is, it can express the intensity of a certain type of feature; In addition, because the relative order of these Top K eigenvalues is preserved, it should be said that it retains part of the position information. However, this location information is only the relative order between features, not absolute location information. For example: "I think the scenery in this place is not bad, but there are too many people." Although the first half reflects the positive emotions, the global text expresses the negative emotions, and K-max pooling can capture such information.

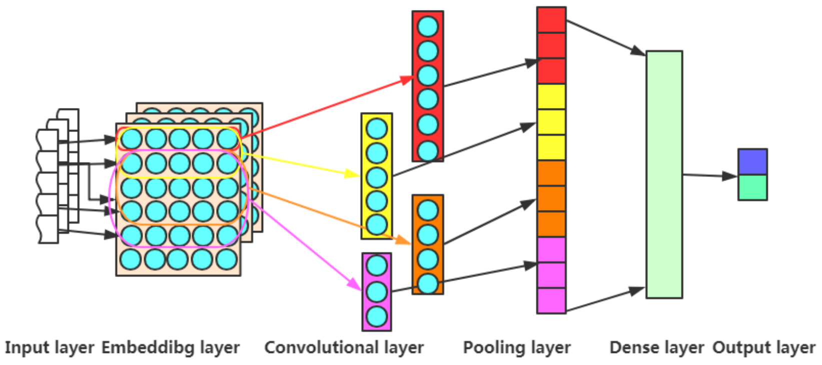

Figure 1: The architecture of K-max pooling CNN model

- **Fully connected layer:** In this model architecture, there are two layers of fully connected layers. The first layer receives the feature vectors obtained by the pooling layer, and the last layer is used for classification and prediction.

3.3 Word embedding

We use two different pre-trained word embeddings, fastText and Glove. FastText is provided by Mikolov et al. (Mikolov et al., 2018), it is a 2 million word vector trained using subword information on Common Crawl with 600B tokens, and its dimension is 300. Glove is provided by Jeffrey Pennington et al. (Pennington et al., 2014), it is a 2.2 million word vector trained using subword information on Common Crawl with 840B tokens, and its dimension is also 300.

We use a mathematical mean of the word vectors by fastText and Glove to produce a high performance word vector. Since the principles between fastText and Glove are different, the word vector representation of the same word is also slightly different, and the average embedding set retains semantic information through preservation of the relative distances between words (Coates and Bollegala, 2018).

3.4 CLR

In this article, we use the cyclic learning rate strategy provided by Leslie N . Smith (Smith, 2017), which is a new method of setting the global learning rate. The advantage is that it avoids a lot of experiments to find the optimal learning rate and can be faster. We set the base learning rate and the maximum learning rate so that the learning rate fluctuates cyclically within this interval. The wave method we use is exp_range, which is a triangle loop that scales the loop magnitude by a factor while keeping the initial learning rate constant.

4 Experiment and results

4.1 Data preprocessing

Text from tweets are inherently noisy. Tweets are processed using tweettokenize tool. Cleaning the text before further processing helps to generate better features and semantics. We perform the following preprocessing steps.

- The “#” symbol is removed and the word itself is retained for hashtags.
- All of “@user” is replaced with username. Username mentions, i.e. words starting with “@”, generally provide no information in terms of sentiment. Hence such terms are removed completely from the tweet.
- Repeated full stops, question marks and exclamation marks are replaced with a single instance with a special token ”repeat” added.
- All contractions are split into two tokens(e.g.: “it’s” is changed to “it” and “is”).
- Emoticons (for example, ‘:(’, ‘:)’, ‘:P’ and emoji etc) are replaced with their own meanings by emotion lexicons.
- Lemmatization, restoring language vocabulary to general form (can express complete semantics) by WordNetLemmatizer.
- Tokens are converted to lower case.

4.2 Experiment setting

We use 4-fold cross validation on the training data, because in this experiment we experimented with cross-validation of different k values, and found that the 4-fold cross-validation effect is the best. In addition the batch size is to 512 and the epoch to 20. In our model, the dimension of embeding is 300. Between the embedding layer and the convolution layer we add the SpatialDropout1D layer with a value of 0.2. In the convolution layer, we set up four convolution kernels of different window sizes, which are 1, 2, 3 and 4, the number of filters is 180, the kernel initializer is $normal$, the activation function is $relu$; in the pooling layer we set the k value to 3. Before the fully connected layer, we add a dropout layer, and the rate is 0.6. The activation function of the final output layer is $sigmoid$ for binary classification. The loss function of this model is $binary\ crossentropy$, and the optimizer is $adam$.

For the cyclical learning rate, we set the base learning rate to 0.001, the maximum learning rate to 0.002, the step size to 300, and the scaling factor gamma to 0.99994.

4.3 Result analysis

This Sub-task A is to evaluate the classification system by calculating the marco F1 score. According to the official ranking of Sub-task A, our model has a marco F1 score of 0.8024, ranked 9th, and our result is much higher than the official baseline. The results of the official baseline and our model are shown in Table 1.

System	F1 (macro)	Accuracy
All NOT baseline	0.4189	0.7209
All OFF baseline	0.2182	0.2790
Our model	**0.8024**	**0.8453**

Table 1: Results for Sub-task A

The confusion matrix of our model prediction results in Sub-task A is shown in Figure 2. There are 620 NOT tags in the test dataset, and 240 OFF tags. As can be seen from the confusion matrix, our model has a lot of OFF prediction errors into NOT, about 32% of OFF are predicted to be NOT, while only about 9% of NOT are predicted to be OFF.

4.4 Influence of Word Embedding

The effect of the average meta-embedding technique is shown in Table 2. In this table, the results of fastText, Glove, the word vector generated by concatenating fastText and Glove and the vector generated by the average meta-embedding technique are compared, and the result is obtained in the trial data set. The results show that this average meta-embedding technique can improve the performance of the model.

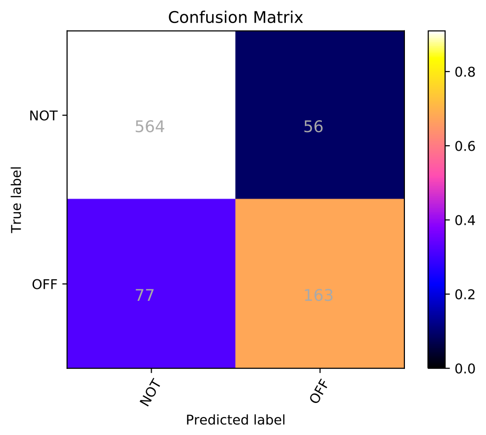

Figure 2: Confusion matrix of K-max pooling CNN model for Sub-task A

word vector	F1 (macro)
fastText	0.8138
Glove	0.7895
concatenated	0.8165
average meta-embedding	**0.8242**

Table 2: Influence of word embedding in trial data set for Sub-task A

As can be seen from Table 2, the concatenated can also improve the performance of the model, but increases the dimension of word embeddings to 600. While the meta-embedded dimension will not exceed the maximum dimension existing in the source embedding.

5 Conclusion

In this paper, we present a K-max pooling convolutional neural network model based on average meta-embedding technology for offensive language detection, which relies solely on the data sets provided to generate competitive results. However, the data imbalance problem will affect the performance of the model, which makes the model prediction tend to be biased towards a high amount of data. In the future work, we will strengthen the processing of data imbalance problems, and try to extract some NER features from the data to further improve the performance of the model.

Acknowledgments

This work was supported by the Natural Science Foundations of China under Grant Nos. 61463050, 61702443 and 61762091, and the Project of Innovative Research Team of Yunnan Province under Grant No. 2018HC019.

References

Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242.

Joshua Coates and Danushka Bollegala. 2018. Frustratingly easy meta-embedding computing meta-embeddings by averaging source word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, pages 11–17.

Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, and Elizabeth Belding. 2018. Hate Lingo: A Target-based Linguistic Analysis of Hate Speech in Social Media. *arXiv preprint arXiv:1804.04257*.

Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. *ACM Computing Surveys (CSUR)*, 51(4):85.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using Convolutional Neural Networks to Classify Hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Irene Kwok and Yuzhou Wang. 2013. Locate the hate: Detecting Tweets Against Blacks. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2016. Measuring the Reliability of Hate Speech Annotations: The Case of the European Refugee Crisis. In *Proceedings of the Workshop on Natural Language Processing for Computer-Mediated Communication (NLP4CMC)*, Bochum, Germany.

Leslie N Smith. 2017. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval)*.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *Lecture Notes in Computer Science*. Springer Verlag.