

# Huiping Shi@HASOC 2020: Multi-Top<sub>k</sub> Self-Attention with K-Max pooling for discrimination between Hate , profane and offensive posts

Huiping Shi<sup>a</sup>, Xiaobing Zhou<sup>b</sup>

*School of information Science and Engineering, Yunnan University, Yunnan.P.R china*

## Abstract

This paper describes a system submitted to HASOC2020: Hate speech and offensive content recognition. The purpose of the task is to discriminate offensive language in social media. We participated in a subtask A and B of English and German. The subtasks A are to identify hate speech, The subtasks B are to identify hate speech, offensive speech and profane speech, offensive blasphemy from a fine-grained perspective. To accomplish these subtasks, we proposed a system based on Multi-Top<sub>k</sub> Self-Attention and K-Max pooling model and used  $k$ -fold method for training fitting. We get the macro F1-score of our model is 0.5042 of subtask A in English, 0.2396 of subtask B in English, 0.5121 of subtask A in German, 0.2736 of subtask B in German.

## Keywords

HASOC 2020, offensive language, multi-Top<sub>k</sub> Self-Attention, K-Max pooling

## 1. Introduction

With the development of information technology, social media provides people with more and more convenient forms of communication. People can freely express their opinions on social networks. At the same time, some people will use social networks to release their one-sided emotions to guide the public to attack the innocent, undermine objective discussions, and intensify conflicts. An immeasurable language environment is the foundation of a harmonious atmosphere and the cornerstone of universal progress [1]. To purify the Internet environment, we need to identify the negative emotions on the Internet. Identifying hate speech and insulting, derogatory or obscene content and another negative emotional speech on social networks belongs to the research direction of emotion classification in natural language processing. Sentiment classification [2] is one of the main tasks in classification tasks in natural language processing, and it is also a hot spot in domestic and foreign research in recent years. The task of sentiment classification is to help researchers quickly obtain, organize, and analyze relevant text information, and analyze, summarize, and infer the emotion contained in the text. Traditionally, regular text language is more beneficial to the analysis, processing, induction, and reasoning of natural language processing. This is because the regular text used conforms to specific rules, it is critical for natural language processing to find the rules in the document.

---


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

✉ 2399078861@qq.com (H. Shi); zhoubx@ynu.edu.cn (X. Zhou)

🆔 0000-0002-5059-7031 (H. Shi)



© 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)

However, in emotional classification, the language used is more perceptual rather than rational, and many sensitive texts are somewhat different from regular texts. Most of the languages used on social networks are related to personal life experience, and these texts are even affected by the personal national language in different ways of expression. Although the texts on social network media are in the same language, its meaning is quite different. Besides, the language text on social media is updated very quickly which is because frequently updated hot events on social media make netizens reach a consensus on a certain event. Besides, the language text on social media is updated very quickly. Facing the ultra-fast updating of social media texts, some scholars of natural language processing still insist on studying the characteristics of hate texts. From the initial manual feature extraction to the rule-based feature extraction of machine learning, and now, the feature extraction of neural networks. The feature extraction has experienced a long and complicated text research process.

HASOC competition, which is an evaluation task to identify hate speech and offensive content in Indo-European languages [3, 4]. This year, HASOC provides 2 subtasks with each for Language. Subtask A: Coarse-grained classification, offensive and profanity content. Subtask B: Fine-grained classification, the distinction between profanity and offensive posts.

We participated in subtasks A and B in English and German, and the datasets discussed in this article come from HASOC. Based on the method of deep learning, we have developed an end-to-end neural network model, which takes Multi-Top<sub>k</sub> Self-Attention as the core and joins K-Max pooling. During training, K-Folding that can alleviate data imbalance and data overfitting, and the approach of fitting generation is used for batch training. This model has achieved an excellent result of subtasks A and B in English and German of HASOC 2020.

The structure of this paper is as follows: In Second 2, we introduce the related work of identifying hate speech and offensive language. In Section 3, we describe the data set and the model structure in detail. In Section 4, we describe the experimental result and data analysis.

## 2. Related Work

The research of natural language processing on hate speech on social networks can be traced back to 2010. Gries et al. collected and annotated the text on social media(tweets), completed a love hate data set about social media speech, and provided these data sets to scholars who need to do research [5]. Provided these datasets to scholars who needed to do research. The scholars who made the dataset also held regular competitions to improve the dataset regularly to encourage more people to participate in this task. Dhillon et al. proposed the SAS model [6]. SAS provides a direction for studying computer network methods, technologies and systems. SAS uses natural language processing technology to identify sentence components (such as subject, verb, and object), disambiguate and identify entities so that SAS can identify whether there is a basic relationship. SAS provided one or more user interface tools for sentiment analysis. In 1996, Hochreiter et al. proposed Long Short Term Memory (LSTM) [7]. The emergence of LSTM made the technology of natural language processing an epoch-making milestone. Malmaisi et al. combined the method of feature representation (character n-gram, word n-gram and word skip-gram) with the Support Vector Machine (SVM) classifier to distinguish hate speech on social media from general profanity, and the accuracy of 0.78 was obtained [8, 9]. Feature

representation plays an important role in natural language processing. In 2018, Malmaisi et al. learned from the previous research, and based on the feature representation model, they tried to integrate multiple natural language classification models and achieved an accuracy of 0.80 on the three classification task [10].

The study of hate speech is not limited to the study of feature representation [11, 12]. In the negative polarity and emotional intensity [13, 14], the research of hatred classification features [15, 16] has also made great progress. In deep learning, there are some similarities between natural language processing and image processing. Some neural network researchers apply the attention mechanism originally used for image processing to natural language processing, such as [17, 18]. It is found that the attention mechanism is highly adaptive in natural language processing, and can be applied to various tasks, and achieve better than the neural network model originally used for natural language processing. In the attention mechanism, in addition to determining the 'value', the initial values of the 'key' and the 'query' must be manually set. The manual operation has not only increased the experimental error. In 2017, the self-attention mechanism was first proposed by Lin et al [19], and they set the value, key and query to the same value, thereby reducing the external influence and making the model self-optimizing. The self-attention mechanism is used in various tasks. In 2019, based on the study of self-attention mechanism, Child et al. proposed a sparse self-attention mechanism [18]. While reducing the calculation time of the algorithm, the model uses the filtering function to optimize the self-attention research, and the model also maintains a good effect on various tasks.

### **3. Methods**

We preprocess the data set using word representation (Fasttext) to convert the text into vector that the computer can process. Two Multi-Top<sub>k</sub> Self-Attention were applied to refine the representative features of the text. After model using the K-Max pooling and Tanh functions get text characteristics, finally, softmax and tanh function were used to calculate scores of each classification. The model is shown in figure 1

#### **3.1. Input Layer**

The input layer accepts the preprocessed text data. Put the sorted data into the model, which data are in accordance with the format of data processed by the model

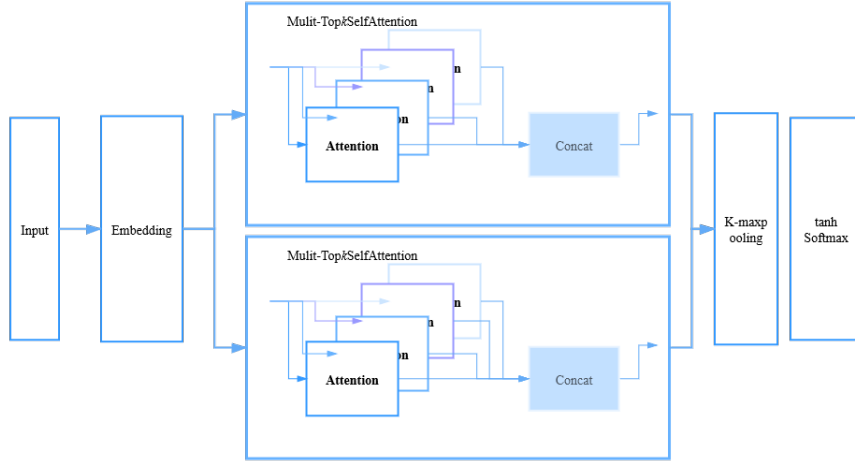
#### **3.2. Embedding Layer**

This layer accepts the input of the Input layer and vectorizes the words in the existing dictionary, which are input by the pre-training vector model.

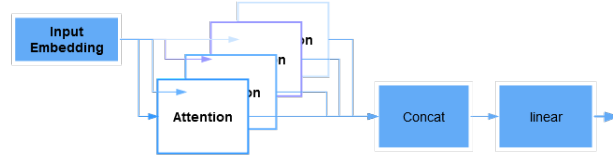
#### **3.3. Encode layer**

##### **3.3.1. Multi-Top<sub>k</sub> Self-Attention**

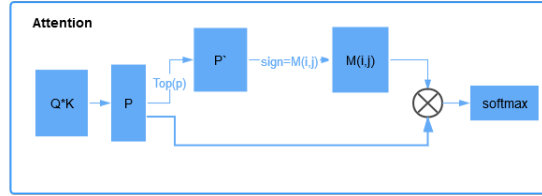
In 2019, the results of Transformer [20] are obvious to all, but the problems of Transformer are also very obvious. The transformer takes a non-discriminatory approach to feature rep-



**Figure 1:** The architecture of *Muilt – Top<sub>k</sub>SelfAttention* with *K – Maxpolling* for Subtask B in English



**Figure 2:** The architecture of Multit-Top<sub>k</sub> Self-Attention in figure 1



**Figure 3:** The architecture of Attention in figure 2

resentation. As a result, Transformer extracts some of the meaningless features, preventing further improvement. To solve this problem, we propose Multi-Top<sub>k</sub> Self-Attention, As shown in figure 2. Multi-Top<sub>k</sub> Self-Attention adds a filtering mechanism based on the Transformer. Multi-Top<sub>k</sub> Self-Attention screens out the *top – n* features that are important to the text and sets the unimportant features to  $-\infty$  [18]. The Attention block is shown in figure 3

As depicted in the figure 3, we initialize  $Q = W_{qx}$ ,  $K = W_{kx}$ ,  $V = W_{vx}$ , then calculate  $P$ ,

$$P = \frac{QK^T}{\sqrt{d}} \quad (1)$$

here  $d$  is the first dimension of Embedding.  $Q$ ,  $K$ , and  $V$  all come from input, but the weights of the matrix of a linear transformation are different. Then we multiply  $Q$  and  $K$  by dot product to get the dependency between the input word and the word. Finally, all values are mapped to a space with dimension  $d$ . We get the score for the attention mechanism, and  $P$  contains all the characteristic merit scores. At this point, our model is the same as that of the Transformer. Then we start to filter the feature score of  $P$ .

$$P' = Top_k - heap(P) \quad (2)$$

The  $P'$  here is the filtered matrix. we use the heap algorithm [21], maintains a small top heap of size  $K$  and puts the data into the heap in turn. When the heap size is full, we only need to compare the top element with the next number. If it is larger than the top element, the current top element is discarded and inserted into the heap. Finally, all of the  $top_k$  are in the heap. But this  $p$  prime right here is the same thing as  $p$ . The purpose of this  $top_k$  operation is simply to mark  $top_k$  elements in  $P$ . We maintain a matrix called  $M_{q \times k}$ .

$$M(i, j) = \begin{cases} 1 & \text{if } P(i, j) \text{ in } top_k \\ -\infty & \text{if } P(i, j) \text{ not in } top_k \end{cases} \quad (3)$$

$$Score = P \times M \quad (4)$$

When  $P(i, j)$  is in the range of  $top_k$ ,  $M(i, j)$  is 1, and the reverse is negative infinity. Then we filter out the values of the  $top_k$ .

$$Top_k Self Attention = Softmax(Score) \times V \quad (5)$$

At this moment, we have completed an Attention block. We use the output of the Attention block as the input for the next Attention block. Finally, all values are combined.

$$Multi - Top_k Self Attention = \sum_{0 < h \leq H}^H Top_k Self Attention_h \quad (6)$$

### 3.3.2. K-Max pooling

In the previous operation, we connected parameters from two Multi- $Top_k$  Self-Attention. At this point, we take advantage of the  $Top_k$  filtering mechanism again. But the difference is, here's  $Top_k$ , we use the algorithm is the merge algorithm [22], take the first  $k$  value. The idea of the integration algorithm is to combine and sort the parameters on the dimension by using the division method. That is.

$$output = K - MaxPooling(Multi - Top_k Self Attention) \quad (7)$$

## 3.4. Output Layer

Concerning the output layer, in order to improve training efficiency, we first use the  $Tanh$  function and finally use  $Tanh$  function and softmax function to calculate the probability output of each class.

**Table 1**  
Result

Tasks	precision	recall	f1-macro	ranking
Task A in English	-	-	0.5042	6
Task B in English	-	-	0.2396	13
Task B in German	-	-	0.5121	7
Task B in German	-	-	0.2736	4

**Table 2**  
Data distribution

Label	English TaskA	German TaskA
HOF	1861	601
NOT	1953	1851
Label	English TaskB	German TaskB
HATE	154	102
OFFN	311	126
PRFN	1374	364
NONE	1953	1860

## 4. Experiment and Result

### 4.1. Result

In this experiment, we took part in subtasks A and B in English languages and German languages provided by HASOC. We have completed a total of 4 subtasks. We submit the result on the platform given by HASOC [4], use F1-macro as the evaluation criteria. In these tasks, We get the macro F1-score of our model is 0.5042 of subtask A in English, 0.2396 of subtask B in English, 0.5121 of subtask A in German, 0.2736 of subtask B in German. The result and ranking as shown in table 1.

### 4.2. Data distribution

The distribution of the data is shown in table 2. All data is sourced from the data provided by the HASOC platform. Subtask A in English and German, identification hate or none hate positions, Hate speech and offensive posts in Subtask A fall into two categories: hate speech(hate), normal text(NOT). Subtask B in English and German, identification hate, profanity and offensive positions. This subtask is a fine classification of English, German, and Hindi datasets. Hate speech and offensive posts in Subtask B fall into four categories: hate speech(hateful), (OFFN) offensive, (PRFN) profanity, (NONE) normal text [4].

We can see that the total amount of data is small and unbalanced distributed, which can easily lead to model overfitting. Due to the small amount of data, we can obtain too few representative features during the training process. So that the model to learn that weak representation or only individual text has characteristics. These features are often irrelevant to the expected

features that need to be extracted. To address the irregular amount and distribution of data, we use cross-validation. We messed up the data set into five parts, validate five times, each time taking 1/5 of the dataset as the validation set.

### 4.3. Conclusion

In this paper, we describe the attention mechanism model based on Multi-Top<sub>k</sub> Self-Attention and K-Max pooling, which used to subtask A and B in English and German of HASOC 2020. And in these subtasks, we have achieved a good result. Unbalanced data distribution will lead to poor model generalization and easy over-fitting. Therefore, in the future, we will focus on research on data imbalance processing.

## References

- [1] 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.
- [2] M. Kanakaraj, R. M. R. Guddeti, Nlp based sentiment analysis on twitter data using ensemble classifiers, in: 2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN), IEEE, 2015, pp. 1–5.
- [3] P. Majumder, D. Patel, Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages (2019).
- [4] 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.
- [5] S. T. Gries, Corpus linguistics and theoretical linguistics: A love–hate relationship? not necessarily, *International Journal of Corpus Linguistics* 15 (2010) 327–343.
- [6] N. S. Dhillon, J. Liang, Nlp-based sentiment analysis, 2014. US Patent 8,838,633.
- [7] S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural computation* 9 (1997) 1735–1780.
- [8] S. Malmasi, M. Zampieri, Detecting hate speech in social media, *arXiv preprint arXiv:1712.06427* (2017).
- [9] J. Li, X. Chen, E. Hovy, D. Jurafsky, Visualizing and understanding neural models in nlp, *arXiv preprint arXiv:1506.01066* (2015).
- [10] S. Malmasi, M. Zampieri, Challenges in discriminating profanity from hate speech, *Journal of Experimental & Theoretical Artificial Intelligence* 30 (2018) 187–202.
- [11] Y. Mehdad, J. Tetreault, Do characters abuse more than words?, in: *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2016, pp. 299–303.
- [12] 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.

- [13] 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.
- [14] S. O. Sood, E. F. Churchill, J. Antin, Automatic identification of personal insults on social news sites, *Journal of the American Society for Information Science and Technology* 63 (2012) 270–285.
- [15] C. Van Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. De Pauw, W. Daelemans, V. Hoste, Detection and fine-grained classification of cyberbullying events, in: *International Conference Recent Advances in Natural Language Processing (RANLP)*, 2015, pp. 672–680.
- [16] P. Burnap, O. F. Rana, N. Avis, M. Williams, W. Housley, A. Edwards, J. Morgan, L. Sloan, Detecting tension in online communities with computational twitter analysis, *Technological Forecasting and Social Change* 95 (2015) 96–108.
- [17] C. Li, C. Quan, L. Peng, Y. Qi, Y. Deng, L. Wu, A capsule network for recommendation and explaining what you like and dislike, in: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 275–284.
- [18] R. Child, S. Gray, A. Radford, I. Sutskever, Generating long sequences with sparse transformers, *arXiv preprint arXiv:1904.10509* (2019).
- [19] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding, *arXiv preprint arXiv:1703.03130* (2017).
- [20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [21] T. Kurita, An efficient agglomerative clustering algorithm using a heap, *Pattern Recognition* 24 (1991) 205–209.
- [22] P. C. Chen, T. Pavlidis, Segmentation by texture using a co-occurrence matrix and a split-and-merge algorithm, *Computer graphics and image processing* 10 (1979) 172–182.