CN-HIT-MI.T at SemEval-2019 Task 6: Offensive Language Identification Based on BiLSTM with Double Attention

Yaojie zhang, Bing Xu, Tiejun Zhao
Laboratory of Machine Intelligence and Translation, Harbin Institute of Technology
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
yjzhang@hit-mtlab.net, hitxb@hit.edu.cn, tjzhao@hit.edu.cn

Abstract

Offensive language has become pervasive in social media. In Offensive Language Identification tasks, it may be difficult to predict accurately only according to the surface words. So we try to dig deeper semantic information of text. This paper presents use an attention-based two layers bidirectional long-short memory neural network (BiLSTM) for semantic feature extraction. Additionally, a residual connection mechanism is used to synthesize two different deep features, and an emoji attention mechanism is used to extract semantic information of emojis in text. We participated in three sub-tasks of SemEval 2019 Task 6 as CN-HIT-MI.T team. Our macro-averaged F1-score in sub-task A is 0.768, ranking 28/103. We got 0.638 in sub-task B, ranking 30/75. In sub-task C, we got 0.549, ranking 22/65. We also tried some other methods of not submitting results.

1 Introduction

Recognition of Offensive information has research and application value in many aspects. With the popularity of social media, people's comments on social media has become an important part of public opinion. Although freedom of speech is advocated, there are still some unacceptable words. The study of offensive language has only recently arisen. With the deepening of the research, we need to consider the different sub-tasks of its decomposition.

In OffensEval tasks (Zampieri et al., 2019b), offensive content was divided into three sub-tasks taking the type and target of offenses into account. Sub-task A is offensive language identification. We should identify a short text sentence as offensive or non-offensive. Sub-task B is automatic categorization of offense types. We need to classify a sentence as having an attack target or not, if this is an offensive sentence. Sub-task C is offense target identification. Its purpose is to identify the target of an attack sentence with an attack target. The target is individual, group or other.

User's comments on Twitter are usually cluttered. In order to clean up the data, a series of preprocessing for the data is necessary. Then, pretrained word vectors are helpful to extract semantic features from deep learning model. For classification model, bidirectional long-short memory neural network can catch the contextual information in text, in order to get offensive semantics from text. A residual connection cascade the first layer's output and the second layer's output can get features of text at different levels. Attention mechanism is used for the final output. Besides, referring to ZHANG et al. (2018), emojis in a sentence have a significant impact on sentiment. We assume them may affect the offensive semantics of text too, and double attention mechanism can deal with this semantic relationship.

The rest of this paper is organized as follows. Section 2 introduces some research advances in Aggression Identification and Hate Speech. Section 3 describes the data we use and the detailed introduction of our system. Section 4 shows the performance of our system and comparison with other models. Section 5 describes some of our summaries and future work directions.

2 Related Work

Due to the universality of offensive language in social media, in order to cope with offensive language and prevent abuse in social media, research on related aspects has gradually emerged in recent years.

There have been several seminars on offensive

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 564–570
Minneapolis, Minnesota, USA, June 6–7, 2019. ©2019 Association for Computational Linguistics

language research, such as TRAC[1] which shared task on Aggression Identification summarized in Kumar et al. (2018), need to distinguish open, secret and non-aggressive texts. And ALW[2] which is work for Abusive Language. Fišer et al. (2017) did some work on the legal framework, dataset and annotation schema of socially unacceptable discourse practices on social networking platforms in Slovenia. Gambäck and Sikdar (2017) introduced a deep learning based Twitter Hate Speech text include racism, sexism, both and non-hate-speech classification system. Waseem et al. (2017) put forward a series of subtasks of hate speech, cyberbullying, and online abuse. Su et al. (2017) described a system for detecting and modifying Chinese dirty sentences. And GermEval is also shared related tasks (Wiegand et al., 2018) which initiate and foster research on the identification of offensive content in German language microposts. Additionally, the main work of Malmasi and Zampieri (2017) and Malmasi and Zampieri (2018) is to approach the problem of distinguishing general profanity from hate speech. Zhang et al. (2018) tried to use a Convolution-GRU for detecting hate speech on Twitter.

3 Methodology and Data

3.1 Method

Our method is composed of 6 stages: Preprocessing, Embedding, Bidirectional LSTM, Attention, Double attention and Softmax. The whole architecture of the single model is shown in Figure 1.

3.1.1 Preprocessing

We have done some processing on the original training data and test data. The main purpose is to make the data cleaner, reduce the number of unknown words in the dictionary, and do some processing of error words. The first step is to convert all words into lowercase. Our preliminary view is that uppercase or lowercase has no direct impact on offensive language recognition tasks. The second step is to deal with punctuation symbols. We use space as separator to divide sentences into words. But in many cases in data sets, punctuation symbols and words, as well as punctuation symbols and punctuation symbols are closely linked without space. In this way, the system will not be able to recognize them. Just like ”sloth.” or ”!!!”, and we turned them into ”sloth .” and ”! ! !”. The third step is abbreviation processing. We converted ”don't”, ”you're” and so on into ”do n't” and ”you 're”, because there is no ”don't” or ”you're” in the dictionary. We haven't expanded the abbreviations like ”do not” or ”you are”, because the results may not be unique just like ”I'd” can represent ”I would” or ”I had”. Forth, we also need to separate the emojis refer to the dictionary of emojis. (We will introduce the word dictionary and emoji dictionary in detail in embedding section). In addition, we regard all numbers as one word.

After completing the above preprocessing, if a word is still detected as an unknown word, we use ekphrasis[3] tool (Baziotis et al., 2017) for further processing. The first step is word segmentation. We separate some words together may be a customary expression by spaces. For example, ”Googlearecorrupt” is turned into ”Google are corrupt”. Second, if the word still does not exist in the dictionary, we do an error correction operation. After all operations, words that do not exist in the dictionary are marked as unknown words.

Step-by-step processing instead of direct batch processing without intermediate detection improves reliability and avoids modifying correct expressions to errors.

3.1.2 Embedding Layer

We used a word embedding layer to represent words as vectors. The 200 dimension word vectors[4] is pre-trained by GloVe based on a large corpus of Twitter provided by Jeffrey et al. (2014). It includes 1,193,514 words and their vector representations.

A sentence sequence $S(w_1, w_2, ..., w_l)$ is represented as $C(c_1, c_2, ..., c_l)$, where $w_i (i \in [1, l])$ represents a word, c_i represents the vector corresponding to the word. C is the matrix representation of the sequence S.

We also use emoji vectors provided by Eisner et al. (2016) to represent the emoji that appears in a sequence. It includes 1,661 emojis used in

[1] https://sites.google.com/view/trac1/home

[2] https://sites.google.com/site/abusivelanguageworkshop2017/

[3] https://www.github.com/cbaziotis/ekphrasis

[4] https://nlp.stanford.edu/projects/glove/

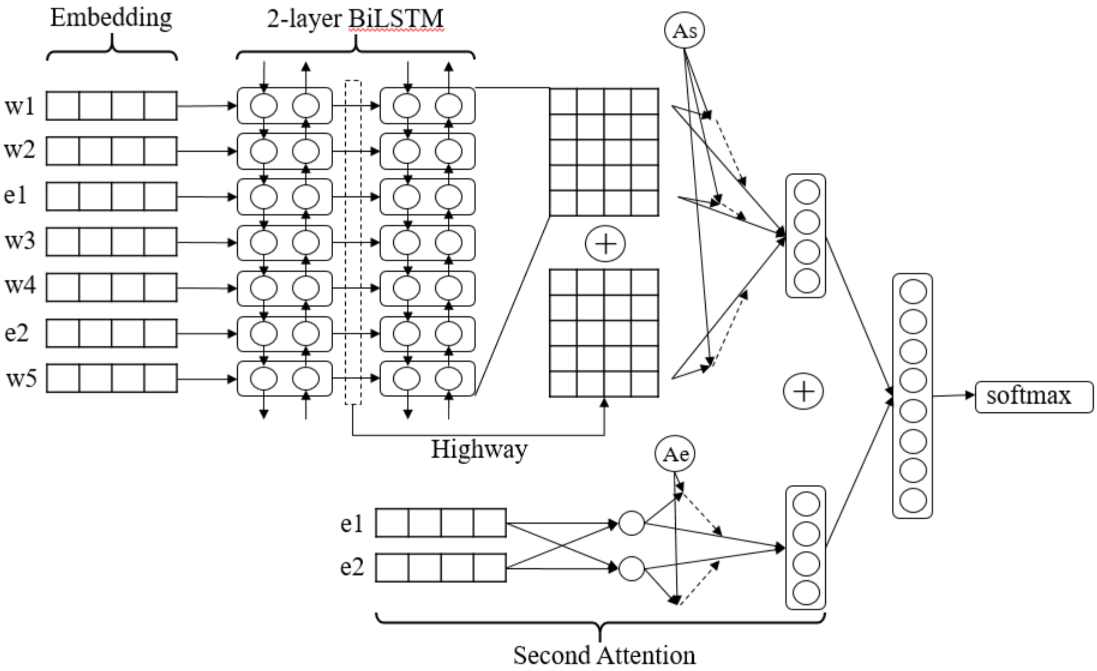

Figure 1: The whole architecture of 2-layer BiLSTM with Double Attention Based Offensive Language Identification. Where w is word, and e is emoji

Twitter and their 300 dimension vector representations. We have made a PCA dimension reduction on emoji vectors, making 300 dimensions into 200 dimensions to suit word vectors. In addition, all bottom feature representation vector are normalized.

3.1.3 Bidirectional LSTM Layer

The structure of one cell in LSTM is shown in Figure 2. Three gated mechanisms of LSTM allows LSTM memory unit to store and access information for a long time. i_t express input gate, f_t express forget gate and o_t express output gate. c_t express memory cell, s_t express memory state and h_t express hidden state. Where t denotes at time t. The forward pass of LSTM is shown in (1)-(6).

$$i_t = f(W_i x_t + U_i h_{t-1} + V_i c_{t-1}) \quad (1)$$

$$f_t = f(W_f x_t + U_f h_{t-1} + V_f c_{t-1}) \quad (2)$$

$$o_t = f(W_o x_t + U_o h_{t-1} + V_o c_t) \quad (3)$$

$$c_t = g(W_c x_t + U_C h_{t-1}) \quad (4)$$

$$s_t = f_t \odot c_{t-1} + i_t \odot c_t \quad (5)$$

$$h_t = o_t \odot g(c_t) \quad (6)$$

Where x_t is the input at time t, $f(.)$ is the sigmoid function, $g(.)$ is the hyperbolic function. W, U and V are trainable weight parameters. In order to extract the semantic relationship features of sentences before and after, we use Bidirectional LSTM to process a sentence. Forward and backward LSTM obtains hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$:

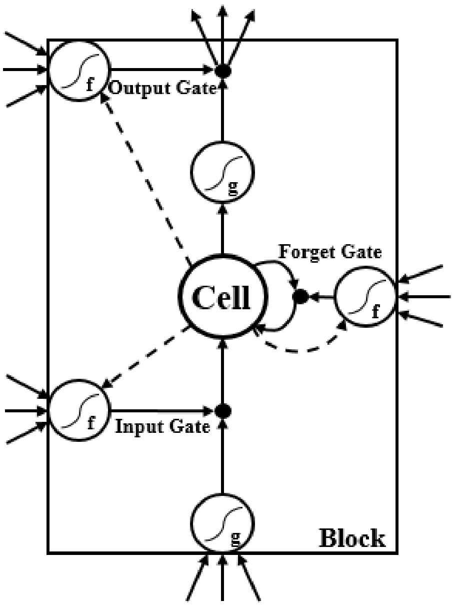

Figure 2: The structure of one cell in LSTM

$$\overrightarrow{h_t} = \overrightarrow{LSTM}\ (w_t, \overrightarrow{h_{t-1}}) \quad (7)$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}\ (w_t, \overleftarrow{h_{t-1}}) \quad (8)$$

Cascade the results of bidirectional hidden state as the result of BiLSTM:

$$H_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{9}$$

We stack two layers of BiLSTMs. The output of the first layer BiLSTM is used as the input of the second layer. Meanwhile, we consider that the first layer can collect low-level semantic information such as lexical or grammatical information, while the second layer can collect high-level semantic information such as sentiment or offensiveness information. So we added a residual connection between the two layers:

$$H_t^{final} = H_t^{layer_1} \oplus H_t^{layer_2} \tag{10}$$

3.1.4 Attention Layer

We input the representation of hidden state obtained by Bidirectional LSTM layer into attention layer to get sentence coding:

$$s_i = \sum_t \alpha_{it} H_{it}^{final} \tag{11}$$

Where a is the weight value:

$$\alpha_{it} = \frac{exp(u_{it}^{\mathsf{T}} u_w)}{\sum_t exp(u_{it}^{\mathsf{T}} u_w)} \tag{12}$$

$$u_{it} = tanh(W_w H_{it}^{final} + b_w) \tag{13}$$

3.1.5 Double Attention Mechanism

We use another attention mechanism similar to that of encoding sentences to encode emojis in sentences, referring to (ZHANG et al., 2018). If there are emojis in a sentence, we get the corresponding vector representation from the emojis dictionary mentioned in Section 3.2.2. Then the coding is obtained through the attention mechanism:

$$s_i^e = \sum_t \alpha_{it}^e E_i \tag{14}$$

Where E is the vector representation of emojis in a sentence.

3.1.6 Softmax Layer

Finally, we concatenate sentence coding and emojis coding through the full connection layer. We use the softmax classifier to construct scoring vectors for each category and convert them into probabilities:

$$r = s \oplus s_e \tag{15}$$

$$\hat{y} = \frac{exp(Wr + b)}{\sum_{i \in [1,l]} exp(W_i r + b_i)} \tag{16}$$

Where W and b is the layer's weights and biases. We use cross-entropy loss function with L2 regularization term:

$$L(\hat{y}, y) = -\sum_{i=1}^{N} \sum_{j=1}^{C} y_i^j log(\hat{y_i^j}) + \lambda(\sum_{\theta \in \Theta} \theta^2) \tag{17}$$

3.2 Data

We only use the training dataset which contains 13,240 tweets provided by SemEval 2019 Task 6 (Zampieri et al., 2019a). Table 1 shows three different levels of tasks and their corresponding amount of data.

A	B	C	Train	Test	Total
OFF	TIN	IND	2,407	100	2,507
OFF	TIN	OTH	395	35	430
OFF	TIN	GRP	1,074	78	1,152
OFF	UNT	-	524	27	551
NOT	-	-	8,840	620	9,460
All			13,240	860	14,100

Table 1: Distribution of label combinations in the data provided by OLID

We have 13,240 tweets to train subtask A. Of these, 4,400 are offensive and 8,840 are non-offensive. There are 4,400 tweets for subtask B. Of these, 3,876 are targeted insult and threats and 524 are untargeted. In 3,876 targeted tweets, 2,407 of them are individual, 1,074 of them are group, and 395 of them are other.

4 Results

In this part, some experimental settings are briefly described. Additionally, we list the experimental results we submitted and compared them with baseline. We use 5-fold cross validation to get 5 same models. Using the probability predicted by these 5 models to do a soft-vote to get the final probability distribution. The highest probability is the predictive class of our system. What's more, we compare performance on offensive identification detection of different models by 5-fold cross validation.

The number of hidden units in 2-layer of BiLSTM is 150 each layer for sub-task A, where it is 100 for sub-task B and 80 for sub-task C. The size of randomly initialized attention query vector is 256 dimensions. We chose the adam optimizer, and the learning rate starts at $7e-4$, decreases to 0.9 times per 20 epoch, always greater than $1e-4$. The λ of L2 regularization is $1e-5$

Tables 2, 3 and 4 show the performance of our system on test dataset in subtasks A, B and C, respectively. The first 2 or 3 rows of the table show the baseline of the officially provided subtasks.

System	F1 (macro)	Accuracy
All NOT baseline	0.4189	0.7209
All OFF baseline	0.2182	0.2790
Submitted System	**0.7684**	**0.8244**

Table 2: The results of Sub-task A we submitted. The system is a 2-layer BiLSTM with Double Attention which is described in Section 3.2

System	F1 (macro)	Accuracy
All TIN baseline	0.4702	**0.8875**
All UNT baseline	0.1011	0.1125
Submitted System	**0.6381**	0.8417

Table 3: The results of Sub-task B we submitted. The system is a 2-layer BiLSTM with Double Attention which is described in Section 3.2. It has different hyperparameter settings from Sub-task A

System	F1 (macro)	Accuracy
All GRP baseline	0.1787	0.3662
All IND baseline	0.2130	0.4695
All OTH baseline	0.0941	0.1643
Submitted System	**0.5488**	**0.6338**

Table 4: The results of Sub-task C we submitted. The system is a 2-layer BiLSTM with Double Attention which is described in Section 3.2. It has different hyperparameter settings from Sub-task A and Sub-task B

Figures 3, 4 and 5 show the confusion matrix of the classification results of our system on the test dataset in subtasks A, B and C, respectively.

We can clearly see from figures that our system performs better in category has more training data. This may be because more data in this category makes the system more inclined to classify these samples correctly in training, not because some categories are easier to identify and others are harder. We used the over-sampling method, but the effect is not obvious. Categories with fewer data quickly reach the state of over-fitting, while those with more data are still under-fitting.

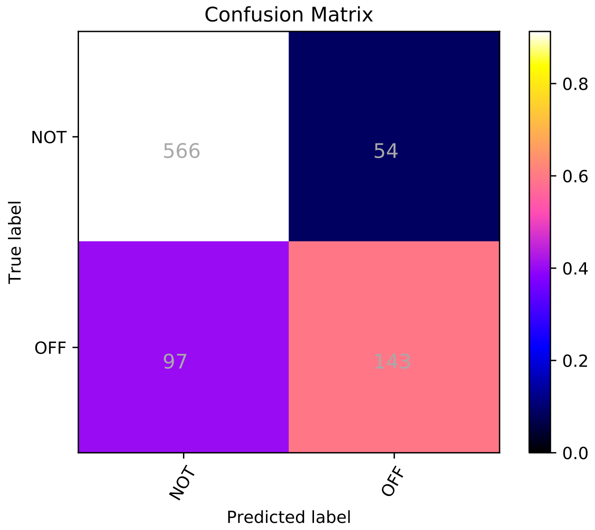

Figure 3: The confusion matrix of the classification results in Sub-task A

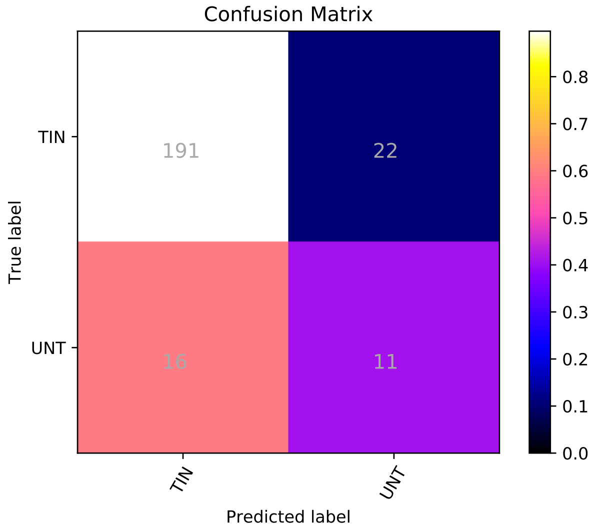

Figure 4: The confusion matrix of the classification results in Sub-task B

Table 5 shows some comparative experiments on closed validation sets. We use 5-fold cross validation, and each data has the same category ratio.

We can see that if we only use the second layer's features, the performance is worse than using the first layer's. When using two layers features at the same time, the effect is the best. In addition, Word Attention can significantly improve system performance, but Emoji Attention only improve system performance a little. This may be because emojis have little impact on text offensive semantics. It is worth mentioning that the system performance is improved obviously after data preprocessing, and the F1 score has increased by 3.12%.

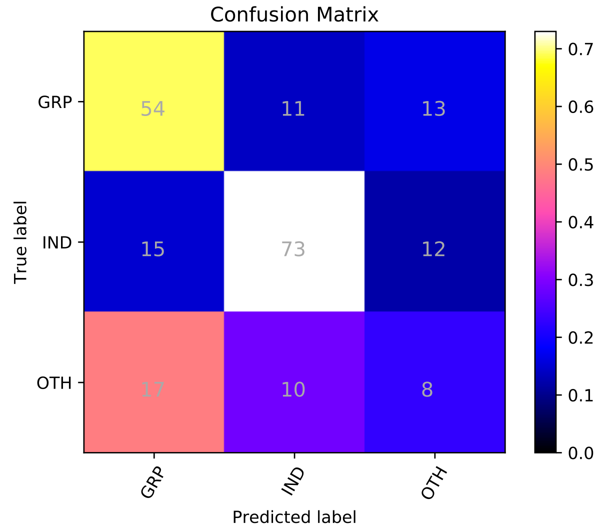

Figure 5: The confusion matrix of the classification results in Sub-task C

Model	A	B	C
1-layer BiLSTM	0.749	0.610	0.527
1-layer BiLSTM with Attention	0.757	0.632	0.544
2-layer BiLSTM with Attention	0.752	0.626	0.513
2-layer BiLSTM (residual connected) with Attention	0.764	0.643	0.549
2-layer BiLSTM (residual connected) with Double Attention (Emojis)	**0.766**	**0.643**	**0.550**

Table 5: Some comparative experiments on closed validation sets

5 Conclusion

This paper introduces a deep learning method Attention-based residual connected BiLSTM with Emojis Attention for SemEval 2019 Task 6: Identifying and Categorizing Offensive Language in Social Media. Our system didn't get the leading score in the competition. Maybe there are the following reasons. Except for the corpus used for pre-training word vectors, we do not use other data for training. Some machine learning models may achieve better results with fewer data. Additionally, we think that hyperparameter adjustment in our system is not perfect. Most importantly, we do not associate the characteristics of offensive language recognition tasks with our model.

The next step of this paper is to associate tasks with models. We will try to constructing a dictionary of offensive words for tasks. We will also try other ways of using external dictionaries except double attention mechanism such as Position Embedding. In addition, we will try some language models that are currently leading the NLP task such as BERT proposed by Devlin et al. (2018).

References

C Baziotis, N Pelekis, and C Doulkeridis. 2017. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*, page arXiv:1810.04805.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning Emoji Representations from their Description. In Proceedings of the 4th International Workshop on Natural Language. In *Proceedings of the 4th International Workshop on Natural Language Processing for Social Media at EMNLP 2016 (SocialNLP at EMNLP 2016)*.

Darja Fišer, Tomaž Erjavec, and Nikola Ljubešić. 2017. Legal Framework, Dataset and Annotation Schema for Socially Unacceptable On-line Discourse Practices in Slovene. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

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

Pennington Jeffrey, Socher Richard, and D. Manning Christopher. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking Aggression Identification in Social Media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbulling (TRAC)*, Santa Fe, USA.

Shervin Malmasi and Marcos Zampieri. 2017. Detecting Hate Speech in Social Media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP)*, pages 467–472.

Shervin Malmasi and Marcos Zampieri. 2018. Challenges in Discriminating Profanity from Hate Speech. *Journal of Experimental & Theoretical Artificial Intelligence*, 30:1–16.

Huei-Po Su, Chen-Jie Huang, Hao-Tsung Chang, and Chuan-Jie Lin. 2017. Rephrasing Profanity in Chinese Text. In *Proceedings of the Workshop Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the First Workshop on Abusive Langauge Online*.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the GermEval 2018 Shared Task on the Identification of Offensive Language. In *Proceedings of GermEval*.

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)*.

Yangsen ZHANG, Jia ZHENG, Gaijuan HUANG, and et al. 2018. Microblog sentiment analysis method based on a double attention model. In *Proceedings of Tsinghua University(Science and Technology)*, volume 58, pages 122–130.

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.