# Aggression Detection in Social Media using Deep Neural Networks

**Sreekanth Madisetty**
Department of CSE
IIT Hyderabad
Hyderabad, India
cs15resch11006@iith.ac.in

**Maunendra Sankar Desarkar**
Department of CSE
IIT Hyderabad
Hyderabad, India
maunendra@iith.ac.in

**Abstract**

With the rise of user-generated content in social media coupled with almost non-existent moderation in many such systems, aggressive contents have been observed to rise in such forums. In this paper, we work on the problem of aggression detection in social media. Aggression can sometimes be expressed directly or overtly or it can be hidden or covert in the text. On the other hand, most of the content in social media is non-aggressive in nature. We propose an ensemble based system to classify an input post into one of three classes, namely, *Overtly Aggressive*, *Covertly Aggressive*, and *Non-aggressive*. Our approach uses three deep learning methods, namely, Convolutional Neural Networks (CNN) with five layers (input, convolution, pooling, hidden, and output), Long Short Term Memory networks (LSTM), and Bi-directional Long Short Term Memory networks (Bi-LSTM). A majority voting based ensemble method is used to combine these classifiers (CNN, LSTM, and Bi-LSTM). We trained our method on Facebook comments dataset and tested on Facebook comments (in-domain) and other social media posts (cross-domain). Our system achieves the F1-score (weighted) of 0.604 for Facebook posts and 0.508 for social media posts.

## 1 Introduction

In recent years, a plethora of data is posted on the web. The data is in the form of text, images, audio, video, memes, etc. Facebook, Twitter are some of the popular social networking sites where a lot of data is posted. This data (tweets or posts) contains the opinions, feelings expressed by the users on various topics or incidents that are happening across the world or about their personal life. Along with a large increase of user-generated content in social networks, the amount of aggression related content is also increasing (Kumar et al., 2018b).

Aggression in social media is targeted to a particular person or group to damage the identity and lower their status and prestige (Culpeper, 2011). Aggression is often expressed in two ways: directly expressed or hidden in the posts. According to (Kumar et al., 2018a), aggression in social media can be broadly classified into three classes: *Overtly Aggressive (OAG)*, *Covertly Aggressive (CAG)*, and *Non-aggressive (NAG)*. Aggression is expressed directly in the overtly aggressive text. It can be expressed either through lexical features or lexical items or certain syntactic structure. Any text in which the aggression is not expressed directly is said to be covertly aggressive text. Aggression is hidden here. It is very hard to distinguish between overtly aggressive text and covertly aggressive text (Malmasi and Zampieri, 2018). In this paper, we develop a classifier for the problem of aggression detection in social media. Table 1 contains some examples of *Overtly Aggressive*, *Covertly Aggressive*, and *Non-aggressive* posts, from the dataset (Kumar et al., 2018b).

Aggression also contains hate speech. Hate speech is generally defined as any communication that defames a person or group on the basis of some characteristic such as color, gender, race, sexual orientation, ethnicity, nationality, religion, or other characteristic (Nockleby, 2000). We proposed a voting based ensemble classifier using deep learning methods in the ensemble for detecting the aggression in

Table 1: Example Facebook posts showing different categories of aggression

| Id | Text | Category |
|---|---|---|
| 101 | Pakistan is comprised of fake muslims who does not know the meaning of unity and imposes their thoughts on others.....all the rascals have gathered there... | Overtly Aggressive |
| 102 | Communist parties killed lacks of opponents in WB in 35 years ruling????? ? | Overtly Aggressive |
| 103 | For tht those hv more thn 2 flats or 2 land is legal rest has to surrender properties to govt | Covertly Aggressive |
| 104 | In government office implement the online transactions... also all parties should show off their account transaction for running their elections in RTI. | Covertly Aggressive |
| 105 | Hi Anuj brought divis to days at 710 what to do and Tata motor dvr 275 any target | Non-aggressive |
| 106 | please advice about Bank nifty & how the NPA will benefit for PSU Banks ? | Non-aggressive |

social media posts. The deep learning methods used in the ensemble are CNN, LSTM, and Bi-LSTM. First, we apply CNN on the training data. Next, we apply LSTM and Bi-LSTM on the same training data. Finally, we use majority voting ensemble method to predict the final label for the given post.

Rest of the paper is organized as follows. Related literature for current work is described in Section 2. Next in Section 3, problem statement of our work is defined and details of the proposed method are presented. Experimental evaluation of the method is described in Section 4. We conclude the work by providing directions for future research in Section 5.

## 2 Related Work

In this section, we describe the literature work in the areas of aggression, hate speech, offensive, and abusive language detection. Hate speech and cyberbullying are often considered as same.

Detection of cyberbullying is modeled in (Dinakar et al., 2011). The authors found that cyberbullying in the text can be automatically detected by building topic-sensitive classifiers. Binary classifiers outperform multiclass classifiers. Bullying traces in social media is studied in (Xu et al., 2012). The authors formulated several NLP tasks with the bullying traces in social media. NLP tasks used here are text categorization, role labeling, sentiment analysis, and latent topic modeling. However, only a few baseline solutions were studied in the context of this problem. (Dadvar et al., 2013) proposed a method to improve the cyberbullying detection by taking user context into account. Here user context is the past (six months) comments of each user. Three types of features are used: content based features, cyberbullying features, and user based features. A method is developed to detect the tweets against blacks in (Kwok and Wang, 2013). The authors applied a simple Naive Bayes classifier with unigrams as features. They built a dataset with class labels: racist and non-racist. However, only unigrams may not be sufficient to differentiate the tweets of racist and non-racist.

An approach to detect hate speech with comment embeddings is developed in (Djuric et al., 2015). The authors have used two phase approach. In step 1, *paragraph2vec* is used to learn the embeddings of comments and words. In step 2, a binary classifier is applied on the learned embeddings. They have used Yahoo finance dataset with 56,280 hate speech comments and 895,456 clean comments. A dictionary-based approach to detect racism in Dutch social media is proposed in (Tulkens et al., 2016). The authors have used three dictionaries. The first dictionary was created by taking the racist terms from the training data. The second dictionary was created through automatic expansion using word2vec. The last dictionary was created by manually filtering incorrect expansions. A method is developed for abusive language detection in online user content in (Nobata et al., 2016). The authors have used the following features: N-gram features, linguistic features, syntactic features, distributional semantic

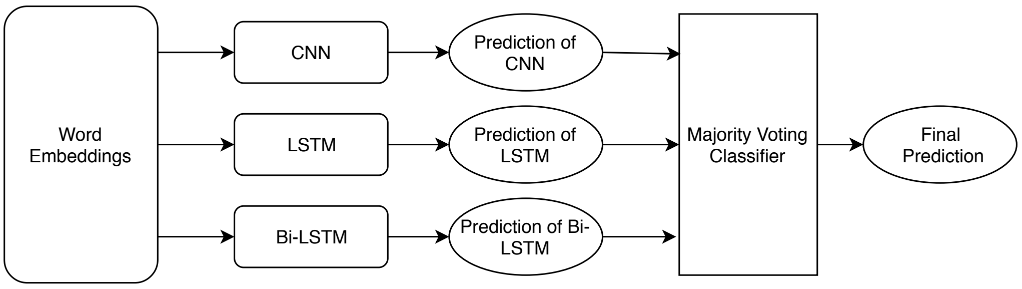


Figure 1: An overview of our proposed method.

features. They have used Vowpal Wabbits regression model [1]. Recently a survey on hate speech detection using natural language processing is done in (Schmidt and Wiegand, 2017). An automated method to detect hate speech and offensive language is developed in (Davidson et al., 2017). The authors have used the following features: TF-IDF (Unigram, Bigram, and Trigram), POS (Unigram, Bigram, and Trigram), Flesch-Kincaid Grade Level, Flesch Reading Ease scores, sentiment lexicon, and some content based features. They have used the classes: hate speech, only offensive language, and neither.

A method to detect abusive language on Arabic social media is proposed in (Mubarak et al., 2017). The authors have created a list of obscene words and hashtags. They also expand the list of obscene words using this classification. Obscene, offensive and clean are the classes used. A rule based approach to rephrase profanity in Chinese text is developed in (Su et al., 2017). The authors have framed 29 rephrasing rules. The proposed system will provide rephrased text with less offensive content. However, these rules are handcrafted. Convolutional Neural Networks are used to classify hate speech in (Gambäck and Sikdar, 2017). The authors have used four CNNs: character 4-grams, word vectors based on semantic information built using word2vec, randomly generated word vectors, and word vectors combined with characters.

A recent discussion on the challenges of identifying profanity vs. hate speech can be found in (Malmasi and Zampieri, 2018). The results demonstrated that it can be hard to distinguish between overt and covert aggression in social media.

## 3 Methodology and Data

In this section, we describe the methodology used for the problem of aggression detection in social media. First, we describe the CNN architecture used in our approach. Next, we describe the LSTM and Bi-LSTM architectures. Finally, we explain our voting based ensemble classifier used in this paper. An overview of our proposed method is shown in Figure 1.

### 3.1 Convolutional Neural Networks (CNN)

Recently, CNNs have been achieving excellent results in various NLP tasks. Our CNN model is inspired from (Kim, 2014). There are two channels in CNN architecture: static and non-static. In static channel, the word vectors are static and are not changed whereas in non-static channel, the word vectors are tuned according to the task. Word vectors are low dimensional dense representations of one hot vectors. These word vectors or word embeddings can be treated as universal feature extractors i.e., they can be applied to any task and achieve comparable performance.

Our CNN architecture contains five layers: Input layer, convolution layer, pooling layer, hidden dense layer, and output layer. Let $w_i \in \mathbb{R}^d$ be the d-dimensional word vector corresponding to $i$th word in the sentence. Let the sentence be comprised of sequence of tokens: $\{t_1, t_2, t_3, ...t_n\}$. The sentence vector can be obtained by the following equation.

$$w_{1:n} = w_1 \circ w_2 \circ w_3 ... \circ w_n \tag{1}$$

[1]https://github.com/JohnLangford/vowpal_wabbit

where $\circ$ is concatenation operator, $w_i$ is the corresponding word vector of $t_i$. Sentence matrix of length $s \times d$ is given as input to the input layer, where $s$ is the number of tokens in the sentence and $d$ is the dimension size. As part of convolution operation, a filter $ft$ is applied to a window of $h$ words to produce a new feature. Similarly, all the convolution features are calculated for all the possible windows of $h$ words. Multiple filters can also be used. After finding the convolution features, these are given as input to the pooling layer. The purpose of pooling layer is to get most important activation. In this architecture, we use max pooling for this purpose. Next, features which are obtained by applying max pooling are given as input to hidden dense layer and then finally softmax activation function is used to get the prediction for the given text or sentence. We also use dropout parameter to reduce the problem of overfitting.

## 3.2 Long Short Term Memory networks (LSTM)

In conventional neural networks, we assume that all inputs are independent of each other. However, this assumption may not be valid for several problems where the input is inherently sequential, or there are explicit dependencies between the input segments. For example, to predict a next word in a sentence, it is better to know the previous words in the sentence. Recurrent Neural Networks (RNN) can predict the next word in the sentence based on the previous words which come before it. RNNs work well if the context is short. If the context is long RNNs may not work properly because they can not remember the long dependencies and also because of *vanishing gradient* problem. To overcome the drawbacks of RNNs, Long Short Term Memory networks (LSTM) were introduced. These are special kind of RNNs which are capable of learning long-term dependencies and long contexts.

LSTMs have the capability to add or remove the information to the cell state $C_t$ with the help of structured gates. The first component of LSTM is *forget gate*. The function of *forget gate* is to decide which information to remember and which information to forget. Sigmoid activation function is used in this layer. So, it will output 1 if the information needs to be remembered otherwise 0. It can be computed by looking at previous state $h_{t-1}$ and current state $x_t$ as follows.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{2}$$

where $[h_{t-1}, x_t]$ is the concatenated vector of $h_{t-1}$ and $x_t$, $b_f$ is the bias term, $W_f$ is a weight vector. Next, LSTM choose what new information to be stored by the cell state. It can be done in two parts. In the first part, *input gate* decides what values to update.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{3}$$

In the second part, a vector of new candidate values $\tilde{C_t}$ that could be added to the cell state is generated by applying $tanh$ activation function.

$$\tilde{C_t} = \sigma(W_C.[h_{t-1}, x_t] + b_C) \tag{4}$$

Now, the new cell state is computed by the following equation.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t} \tag{5}$$

Final gate is *output gate*. It is computed as follows.

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{6}$$

$$h_t = o_t * tanh(C_t) \tag{7}$$

Output is calculated by the $o_t$. $h_t$ is calculated by applying $tanh$ activation to the cell state $C_t$ and by multiplying with $o_t$. This will decide which information should be passed to the next state.

Table 2: The number of instances in the aggression dataset

| Aggression | Train | Dev | Test 1 (Facebook posts) | Test 2 (Social media posts) | All |
|---|---|---|---|---|---|
| Overtly Aggression (OAG) | 2708 | 711 | 144 | 361 | 4532 |
| Covertly Aggression (CAG) | 4240 | 1057 | 142 | 413 | 5852 |
| Non-aggressive (NAG) | 5052 | 1233 | 630 | 483 | 6790 |
| **All** | 12000 | 3001 | 916 | 1257 | 17174 |

### 3.3 Bi-directional Long Short Term Memory networks (Bi-LSTM)

Bi-directional Long Short Term Memory networks (Bi-LSTM) use two LSTMs instead of one LSTM for training. The first LSTM is applied on the normal sentence and the second LSTM is applied on the reversed copy of the given input sentence. This can provide faster learning and additional context for better training. In all the above deep learning methods, we use Glove word embeddings (Pennington et al., 2014) with dimension size 200.

### 3.4 Voting Based Ensemble Classifier

There are different ensemble methods such as bootstrap aggregating (bagging), boosting, stacking, simple averaging, majority voting, etc. to combine the classifiers. In bagging ensemble method, each classifier is trained on random sub-samples which are drawn from the original dataset using bootstrap sampling method. In boosting, each classifier is trained on the same samples but the weights of the instances are adjusted according to the error of the last prediction. In stacking, the classifiers are combined using another machine learning algorithm. In simple averaging, the predictions of classifiers are averaged to get the final predictions. Finally, in majority voting ensemble method, for each test instance, the final prediction is the one which is predicted by the majority of classifiers.

Generally, the performance of the ensemble method is often better than the performances of individual methods in the ensemble. We use majority voting based ensemble method for our problem. The methods used in the ensemble are CNN, LSTM, and Bi-LSTM.

### Data

The number of instances for each of the aggression categories in training and validation set are presented in Table 2. For testing, we use two datasets. One dataset is from Facebook posts with 916 instances and another one is from social media posts with 1257 instances. The data collection methods used to compile the dataset used in the task is described in Kumar et al. (2018b).

## 4 Results

In this section, we describe the results obtained by applying our proposed method.

### Evaluation Metrics

Let $tp$ denotes the true positives, $fn$ denotes the false negatives, $fp$ denotes the false positives, and $tn$ denotes the true negatives. The following are the evaluation metrics used in this paper.

- *Precision*: Precision is computed as follows.

$$Precision(P) = \frac{tp}{tp + fp} \tag{8}$$

- *Recall*: Recall is computed as follows.

$$Recall(R) = \frac{tp}{tp + fn} \tag{9}$$

Table 3: Evaluation results of Facebook posts

| **Aggression** | **Precision** | **Recall** | **F1-score** | **#Samples** |
|---|---|---|---|---|
| OAG | 0.4938 | 0.5556 | 0.5229 | 144 |
| CAG | 0.2360 | 0.5634 | 0.3326 | 142 |
| NAG | 0.8602 | 0.5667 | 0.6833 | 630 |
| **Avg/Total** | 0.7059 | 0.5644 | 0.6037 | 916 |

Table 4: Evaluation results of Social Media posts

| **Aggression** | **Precision** | **Recall** | **F1-score** | **#Samples** |
|---|---|---|---|---|
| OAG | 0.5032 | 0.4294 | 0.4634 | 361 |
| CAG | 0.3953 | 0.4116 | 0.4033 | 413 |
| NAG | 0.6089 | 0.6542 | 0.6307 | 483 |
| **Avg/Total** | 0.5084 | 0.5099 | 0.5080 | 1257 |

- *F1-score*: F1-score (weighted) is calculated as follows.

$$F1\text{-}score = \frac{2PR}{P+R} \tag{10}$$

  Weighted F1-score calculate metrics for each class label, and find their average, weighted by support (the number of true instances for each label).

The evaluation results with Facebook posts by applying our proposed method are shown in Table 3. It can be observed that precision, recall, and F1-score of *Non-aggressive* class are higher than other two classes. Similarly, for the *Covertly Aggressive* class all these metrics are lower than other classes. This is because aggression is hidden in *CAG* class posts and it is very difficult to identify that type of posts. For *CAG* class our proposed method is not doing well. However, overall precision, recall, and F1-score values are good.

The evaluation results with social media posts by applying our proposed method are shown in Table 4. The evaluation metrics are behaving in a similar manner as the results of applying our method to Facebook posts i.e., *NAG* class values are better and *CAG* class values are worse than remaining two classes. However, overall values are less compared to the results of Facebook posts. This is because, in the first case, the training and testing are done from the same domain (Facebook) whereas in the second case, the training is done on one domain (Facebook) and testing is done on other domain (other social networking site).

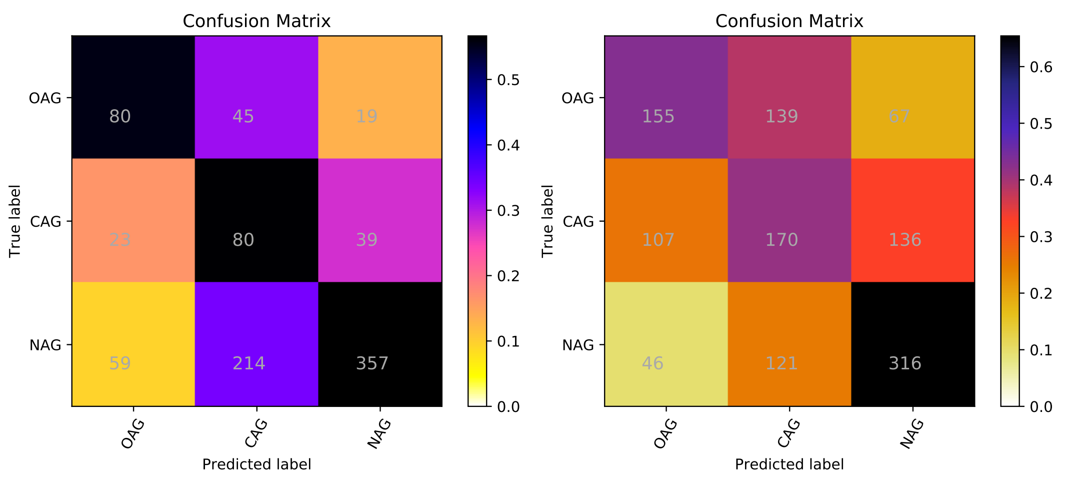
Table 5: Comparison of our proposed method results for Facebook posts.

| **System** | **F1 (weighted)** |
|---|---|
| Random Baseline | 0.3535 |
| Proposed method | **0.6037** |

Table 6: Comparison of our proposed method results for social media posts.

| **System** | **F1 (weighted)** |
|---|---|
| Random Baseline | 0.3477 |
| Proposed method | **0.5080** |

The comparison of our proposed method with random baseline method for Facebook posts is shown in Table 5. In random baseline method, class labels are assigned randomly to each post and F1-score is calculated. It can be observed that our proposed ensemble method outperforms the random baseline



(a) Facebook posts (b) Social media posts

Figure 2: Confusion matrix of the proposed ensemble model for the three classes. The heat map represents the fraction of correctly classified examples in each class. The raw numbers are also presented in each cell.

method. This shows the effectiveness of our proposed method. Similarly, the comparison of our proposed approach for social media posts is shown in Table 6. Our proposed approach is performing better than the random baseline method.

The confusion matrix of our proposed method for Facebook posts is shown in Figure 2a. This shows there is the maximum confusion between *NAG* and *CAG*, with *Non-aggressive* frequently being confused for *Covertly Aggressive* content. A considerable amount of *Non-aggressive* content is misclassified as being *Covertly Aggressive*. The *Non-aggressive* class achieves the best result, with a huge majority of instances being correctly classified. The confusion matrix of our proposed method for social media posts is shown in Figure 2b. This figure shows there is a large amount of confusion between *OAG* and *CAG*, with *Overtly Aggressive* being confused for *Covertly Aggressive*. Similarly, there is a substantial amount of confusion between *CAG* and *NAG*, with *Covertly Aggressive* content is being misclassified as *Non-aggressive* and vice-versa. In this case also, the *Non-aggressive* class achieves best result, with a large number of instances being correctly classified.

## 5 Conclusion

We developed an ensemble method for aggression detection in social media with three deep learning methods in the ensemble. Ensemble method performance is better than the performance of individual methods in the ensemble. We observed that our approach is achieving good performance on cross-domain also. Our proposed method outperforms the random baseline method. It is very difficult to differentiate between *Covertly Aggressive* posts and *Non-aggressive* posts in social media. There is a scope to improve our method because some of the classes (*CAG* and *OAG*) have lower precision values. For future work, we want to include both feature based and deep learning methods in the ensemble.

## References

Jonathan Culpeper. 2011. *Impoliteness: Using language to cause offence*, volume 28. Cambridge University Press.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of ICWSM*.

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

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web Companion*, pages 29–30. International World Wide Web Conferences Steering Committee.

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*.

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

Ritesh Kumar, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018b. Aggression-annotated Corpus of Hindi-English Code-mixed Data. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan.

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

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

Hamdy Mubarak, Darwish Kareem, and Magdy Walid. 2017. Abusive Language Detection on Arabic Social Media. In *Proceedings of the Workshop on Abusive Language Online (ALW)*, Vancouver, Canada.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web*, pages 145–153. International World Wide Web Conferences Steering Committee.

John T Nockleby. 2000. Hate speech. *Encyclopedia of the American constitution*, 3:1277–79.

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.

Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection Using Natural Language Processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media. Association for Computational Linguistics*, pages 1–10, Valencia, Spain.

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.

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, and Walter Daelemans. 2016. A Dictionary-based Approach to Racism Detection in Dutch Social Media. In *Proceedings of the Workshop Text Analytics for Cybersecurity and Online Safety (TA-COS)*, Portoroz, Slovenia.

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.