*Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 87–92
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020


# Spyder: Aggression Detection on Multilingual Tweets

**Anisha Datta[1], Shukrity Si[1], Urbi Chakraborty[2], Sudip kumar Naskar[2]**
Jalpaiguri Govt. Engineering College, India [1]
Jadavpur University, India [2]
{sukriti.si98, dattaanishadatta, urbichakraborty}@gmail.com, sudip.naskar@cse.jdvu.ac.in

**Abstract**

In the last few years, hate speech and aggressive comments have covered almost all the social media platforms like facebook, twitter etc. As a result hatred is increasing. This paper describes our (**Team name: Spyder**) participation in the Shared Task on Aggression Detection organised by TRAC-2, Second Workshop on Trolling, Aggression and Cyberbullying. The Organizers provided datasets in three languages – English, Hindi and Bengali. The task was to classify each instance of the test sets into three categories – "Overtly Aggressive" (OAG), "Covertly Aggressive" (CAG) and "Non-Aggressive" (NAG). In this paper, we propose three different models using Tf-Idf, sentiment polarity and machine learning based classifiers. We obtained f1 score of 43.10%, 59.45% and 44.84% respectively for English, Hindi and Bengali.

**Keywords:** Aggression Detection, Cyberbullying, Tf-Idf, Sentiment polarity, Machine learning

## 1. Introduction

According to data of smartinsights (https://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/), the number of social media users in 2019 was above 3 billion. Due to this huge increase, different types of user generated contents can be seen on social media. Many social media platforms like twitter, facebook, instagram, blogs etc. give users the opportunity to post status, pictures, videos, etc. and anyone can comment and reply to the comments on the posts. The social media posts and comments can be appreciative, affectionate, funny, aggressive, hate-speech or even sarcastic. Due to the huge interaction between people on social media, the incidents of aggression can be seen growing day by day in the form of trolling or hate-speech. The impact of this phenomenon is immense, as it can even lead anyone to commit suicide, two communities to start riot, etc (Phillips, 2015). For this reason, this research topic is of great importance and it has gained popularity among researchers in the last few years. The objective of this research topic is to automatically identify aggressive posts in social media, there by detecting the social media offenders and prevent any undesirable incidents. Research on this topic is very trending and is also a need of the hour.

This workshop focuses on the applications of NLP and Machine Learning to tackle these issues. This includes two shared tasks out of which we have participated on the 1st task as detailed below -

The task was to identify the aggressive posts from the social media texts. The participants were provided with the datasets containing three languages – English, Hindi and Indian-Bengali. People nowadays use multiple languages to write comments or posts on social media. A very important aspect of this task is to handle code-mixing and code-switching in languages since these are abundantly used in social media platforms. The datasets that we were provided with contain three classes "Overtly Aggressive" (OAG), "Covertly Aggressive" (CAG) and "Non-Aggressive" (NAG) where Overtly means totally aggressive, Covertly means bullying or trolling indirectly containing almost no or less aggressive words and the third one is not aggressive at all.

For our experiments we used three different models for three different languages. We used Tf-Idf vectorizer to vectorize the word-tokens. For English dataset, we used the XGBoost classifier followed by the bagging method. For Hindi dataset, we used the Gradient Boosting classifier and many different types of features like aggressive words lexicon, sentiment scores, parts of speech tags etc. Lastly we used the Gradient Boosting Classifier for Bengali dataset.

The rest of the paper is organized as follows. Section-2 gives a brief account of the related works. Section-3 presents a description of the datasets. In section-4, the system architecture and the feature engineering are explained. Section-5 presents the results and comparison. Section 6 concludes the paper and provides avenues for future work.

## 2. Related Work

Although aggression detection in text is a relatively new research topic, quite a few research work have been carried out on this topic (AmirHRazavi and Matwin., 2010; Ritesh Kumar and Chennuru, 2018; Ritesh Kumar and Zampieri, 2020). (Duyu Tang and Qin, 2014) showed how positive and negative emoticons can be used for this work. (Kwok and Wang., 2013) used uni-gram model for this task. (Chikashi Nobata and Chang, 2016) used different types of syntactic features and embedding features for aggression detection in text. (Mohammad, 2012) mapped hashtags like 'yuck', 'joy' into different types of emotions and classified the texts. In (Or˘asan, 2018), they used Support Vector

Machine and Random Forest as classifiers and emojis and sentiment scores were used as features. (Nemanja Djuric and Bhamidipati, 2015) used word embeddings which worked better than bag of words to detect aggressive text. (Jan Deriu and Jaggi, 2016) also did the work with the help of emotional sentiment.
However, all the research works mentioned above are based on the English language (Jun-Ming Xu and Bellmore, 2012). These days, with the increasing availability of multi-lingual keypads in the mobile devices and the support for multi-lingual contents in the websites, detecting aggression from multi-lingual texts has become a necessity. (Vinay Singh and Shrivastava, 2018) used CNN and LSTM to detect aggression on Hindi-English code-mixed texts. In (Shukrity Si, 2019), an ensembling method were used with the help of Aggression lexicons, sentiment scores, POS tags and, emoticons on English and Hindi-English code-mixed languages. (Kalika Bali and Vyas, 2014) proposed a model for English-Hindi code-mixed comments from facebook. (Yogarshi Vyas and Choudhury, 2014) proposed a model for Hindi-English codemixed language which is based on the feature - parts of speech. There has also been work on aggression detection in other languages like Chinese (Hui-Po Su and Lin, 2017), Arabian (Hamdy Mubarak and Magdy, 2017), Dutch (Stephan´ Tulkens and Daelemans, 2016), etc.
Our work is based on three languages - English, Hindi and Indian Bengali. There are English-Hindi code-mixing cases too in the datasets. We proposed different models for the different languages and the models are based on machine learning algorithms like XGBoost and Gradient Boosting and features like Tf-Idf, sentiment scores, POS tags and aggressive words lexicon. The methodology is described elaborately in Section 4.

## 3. Datasets

The TRAC 2020 Shared Task Organizers (Bhattacharya et al., 2020) provided datasets in 3 languages – English, Hindi and Indian Bengali. The English dataset contains 5,120 texts for training and 1,201 texts for testing. The Indian Bengali dataset contains 4,785 texts for training and 1,188 texts for testing (in both Roman and Bangla script). The Hindi dataset contains 4,981 texts for training and 1,200 texts for testing (in both Roman and Devanagari script). Table 1 presents the statistics of the shared task datasets provided by the Organizers.

Table 1: Dataset statistics

| ***Data*** | ***Training*** | ***Test*** |
|---|---|---|
| English | 5,120 | 1,201 |
| Hindi | 4,981 | 1,200 |
| Bengali | 4,785 | 1,188 |

Some examples are shown in figure 1.

**English and Codemixed :**
It seems like these people want to be famous nothing more they don't care if people die by their hateful talks. *NAG*
Hello mister....kabir singh is a achi cheez...u don't know how many family are destroyed by drinking..... *CAG*
conservatives fuck you all.. *OAG*

**Hindi :**
"यह एक संवेदनशील,कठोर,सनक, बेपरवाह, दर्द देने वाला,प्रेम कहानी है", *NAG*
कितना आवश्यक है उस भाषा में डिबेट करना जो भाषा आपके पेनलिस्ट्स से आती ही नहीं।, *CAG*
इन लिब्रल्स का भोसड़ा मारू, *OAG*

**Bengali :**
পতিতাদের চরিত্র রাজনৈতিক নেতাদের থেকে হাজার গুন ভাল। ওদের সততা আছে। টাকা মেরে দেয় না। ওদের আর ঘাটিওনা ভাই।", *NAG*
রানুদি হাই কোর্টে গিয়ে ভিখা কোরতো.....😂🤗😂....., *CAG*
তুই তো খানকি মাগি টাকার মুখ দেখেছিস তাই জন্য এতো গরম", *OAG*

Figure 1: Examples of given texts with categories

The preprocessing steps and further classification process are described as follows.

## 4. Methodology

Different feature models are used for these 3 different languages in classification process. Though same vectorizing tool is used in all of these three that is Tf-Idf Vectorizer. Three different models are described below.
For English Task-A, we have used Tf-Idf Vectorizer (taking unigram and bigram sequence of words) with 500 maximum words as features. We use XGBoost classifier (with learning rate 0.01 and random state 1) to train the given dataset. Then we have used bagging classifier where the base classifier is also XGBoost (maximum samples=0.5, maximum features=0.5). No extra data is used here for training.
For Bengali dataset, we have used Tf-Idf Vectorizer (max words = 500 , bigram) as feature to vectorize the word tokens. Then we have used Gradient Boosting Classifier for classification. We are using the given dataset and no extra data is used for training here.
For Hindi dataset, we have used Tf-Idf Vectorizer, aggressive word lexicons , sentiment scores(taking compound score from positive and negative scores of individual words) and part of speech tags (as some POS tags are important in classification like-adverbs,adjectives etc.) as features. And we have used Gradient Boosting Classifier for classification. No extra data is used for training here.
Now we describe the vectorizer tool, classification algorithms and other feature models in details.

### 4.1. Tf-Idf Vectorizer

A machine can't understand raw text data but only number vectors. So the text input must be converted into vector of numbers. There are many tools available in python for this conversion. Bag of Words (BoW), Count Vectorizer, Tf-Idf Vectorizer are some of the examples. Tf-Idf doesn't only count the occurrences of any word in a document, but also gives importance to the words which are more useful in revealing the document's nature. It is the multiplication of Tf (Term Frequency) and Idf (Inverse Document Frequency) which

have the formulae as follow -

$$Tf(t) = \frac{frequency of term t in a sentence}{total no. of terms in that sentence}$$

$$Idf(t) = log \frac{no. of sentences in a document}{total no. of sentences which contain in termt}$$

By taking the log of the inverse count of term t, the value for the words (terms) occurring much frequently in the document (like stopwords, less important words) gets reduced making the classification task easier.

### 4.2. XGBoost

XGBoost stands for Extreme Gradient Boosting. We used XGBoost here for the English dataset. It is a new algorithm and an implementation of Gradient Boosted Decision Tree. It is mainly used for better performance and it reduces the execution time also. It has many features such that system features, model features, algorithm features.

#### 4.2.1. System Features

For better and fast performance this feature is included in the XGBoost library. It has out of core computing, distributed computing, cache optimization and parallelization.

- Out of Core Computing - This is a special feature that works for very large dataset. Large dataset generally does not fit into memory. So this feature can overcome this situation.
- Distributed Computing - This feature is used to run very large models which needs a machine-cluster.
- Cache Optimization - It is used for optimizing the algorithm and data structure.
- Parallelization - It uses all CPU cores parallely during the time of training.

#### 4.2.2. Model Features

Model features include regularization methods and different types of gradient boosting algorithm.

- Stochastic Gradient Boosting - It is a special form of Gradient Boosting Machine that sub-samples the column and row.
- Regularization - It includes L1 and L2 regularization which help to overcome overfitting.

#### 4.2.3. Algorithm Features

This feature is included to increase the efficiency of available resources and computational time. To do this it uses block structure, continued training and sparse aware method.

#### 4.2.4. Bagging with XGBoost

Then we used bagging classifier keeping XGBoost as our base classifier.

Bagging is one type of ensembling method that is used for better prediction. For bagging, the original dataset is divided into many random subsets. Then the base classifier is fitted (here XGBoost) into the subsets. Then the output is given by aggregating (voting or averaging) their individual predictions. This method is known as bagging and with this we can minimize the variance of the model. We used bagging classifer with the help of XGBoost to classify the English task.

### 4.3. Gradient Boosting Machine

Gradient Boosting machine (GBM) was used for Hindi and Bengali dataset. Weak learner by training can become a strong learner - on this assumption GBM works. Gradient Boosting Classifier is mainly consisting of three major components - a loss function, a weak learner and an additive model. On training the loss function is optimized, the weak learner is used to predict on the basis of the task and the additive model is used so that the weak learner can minimize the loss function.

- Loss Function - In supervised learning, error should always be minimized during the training. To calculate the error first we have to take a function, it is called loss function. Loss function is generally taken on the basis of problem statement. The main criteria of a loss function is that it must be differentiable. For classification, we can use logarithmic loss and for regression, squared error can be used. For our task, we used logarithmic loss as our loss function.
- Weak Learner - For Gradient Boosting, Decision Tree is taken into consideration for weak learner. The learner should be greedy and that is why tree is chosen here. Tree are constructed in a greedy way. Trees generally choose the best split points to minimize the scores. And later an additive model is added with this weak learner.
- Additive Model - Additive model is used to minimize the error of loss function. For this algorithm, trees are added but one at a time and for this, the trees should not be changed. Gradient Descent method is also used here and it helps to minimize the error during the addition of trees. After the errors are calculated the weights are updated for minimizing the error. The new output is added to the old output of the existing tree and the process is continued. In this way, Gradient Boosting is heading towards a better result.

### 4.4. Aggressive Word Lexicon

For doing the task, we observed the dataset very carefully and we observed that the texts contain many bad words and slang languages. We considered these as an important feature and named these as aggressive word features. So, we made a lexicon of these aggressive words which can be used to write hate comments and used it to build our model. Here is some examples of aggressive words.

e.g - ”chutiya”, ”jhant”, ”  ” etc.

These types of words are frequently used in texts labelled as 'OAG'. So, this feature is very important to

identify the 'OAG' class in our task. We used this lexicon for Hindi dataset only.

### 4.5. Sentiment Score

Observing the dataset, we can say that aggression is one kind of sentiment and for this, we used sentiment score as one of our features. Generally if the sentiment of a text is very negative, then there is a high chance that the text would be OAG. Because, OAG text contains many slang words which belongs to negative sentiment category. We used this feature for Hindi dataset. Hindi-sentiwordnet is used to get the sentiment score of each word present in the dataset. There are three types of sentiment in sentiwordnet - positive, negative and neutral. We tagged all the tokens accordingly this and used sentiment score as a feature.

### 4.6. POS Tag

POS tag represents part of speech tag. We observed that adjectives and adverbs are highly used in case of OAG and CAG. Higher the present of adjective and adverb higher the chance of the text is to be a OAG or CAG. We used this feature for Hindi dataset and to do this sentiwordnet was used. There are four parts of speech in sentiwordnet - noun, verb, adjective and adverb. We tagged the word-tokens according to their parts of speech and constructed a feature matrix and used it to build our model.

## 5. Results and Discussion

In this section, we will discuss about all our of results in details. Table 2 shows the result of English dataset. We got the weighted F1 score of 43.10% and accuracy of 58% for this model.

The performance of our model is not so good in the shared task competition. The comparison with other models is shown in Table 3.

| System | F1 (weighted) | Accuracy |
|---|---|---|
| Bagging (XGBoost) | 0.4310 | 0.58 |

Table 2: Results for Sub-task EN-A

Table 3: Comparison Table for English Dataset

| | ***Julian*** | ***Sdhanshu*** | ***krishan thvs*** | ***Our Model*** |
|---|---|---|---|---|
| F Score | 80.29 | 75.92 | 44.17 | 43.10 |

From the table, we can clearly see that our performance is very poor. So we need many modifications in our model and we will discuss about the poor performance of our model in the end of this section.

Table 4 shows the result of Hindi dataset. We got the F1 score of 59.45% and accuracy of 62.08%.

The comparison with other models in this dataset is shown in Table 5.

| System | F1 (weighted) | Accuracy |
|---|---|---|
| GBM | 0.5945 | 0.6208 |

Table 4: Results for Sub-task HIN-A

Table 5: Comparison Table for Hindi Dataset

| | ***Julian*** | ***abaruah*** | ***bhanu prakash2708*** | ***Our Model*** |
|---|---|---|---|---|
| F Score | 81.27 | 79.43 | 14.06 | 59.45 |

The performance of our model for Hindi dataset is slightly better than the previous one. But still it needs lot of modification.

The result for Bengali dataset is shown in Table 6. We got the F1 score of 44.84% and accuracy of 59.76%. The comparison with other models for this Bengali dataset is shown in Table 7.

| System | F1 (weighted) | Accuracy |
|---|---|---|
| GBM | 0.4484 | 0.5976 |

Table 6: Results for Sub-task BEN-A

Our performance on Bengali dataset is also not good and we will modify our model for better performance.

Confusion matrices are given to visualize the results for all of the three languages. This matrix gives the actual measurement to test the performance of our model. It compares between the true (actual result) and predicted (model prediction) classes. As for binary classification, the confusion matrix looks like the Table 8.

Here TP means True Positive (predicted as true and actually it is true), FP means False Positive (predicted as true but actually it is false), FN means False Negative (predicted as false but actually it is true) and TN means True Negative (predicted as false and actually it is false).

In our model, this is 3-class classification and so the confusion matrix is of 3*3 matrix. The confusion matrix for the Bengali dataset is shown in figure 2. The confusion matrix for the English dataset is shown in figure 3.

The confusion matrix for the Hindi dataset is shown in figure 4.

This model gives good results but these could be better if we could modify our model in some ways more. There are some modifications that can be done as follows-

(1) We can use extra resources like aggressive words lexicon for Bengali and English datasets as well. It will help to distinguish the aggressive texts from others like in Hindi dataset.

(2) We have used bagging classifier (ensembling

Table 7: Comparison Table for Bengali Dataset

| | ***Julian*** | ***abaruah*** | ***saikesav 564*** | ***Our Model*** |
|---|---|---|---|---|
| F Score | 82.18 | 80.82 | 46.84 | 44.84 |

| **Predicted(row)/ True(col)** | **Positive** | **Negative** |
|---|---|---|
| Pos | TP | FP |
| Neg | FN | TN |

Table 8: Confusion Matrix of Binary Classification

method) in case of English data only with base classifier as XGBoost. But this method can be applied to other two datasets as well for improvement.
(3) We have used only machine learning classifiers for this 3-class classification. But we can implement deep learning models also. Although the datasets are not very large and it might not give good results, but we can try this in future for more exploration.

## 6. Conclusion

After observing the results we can come to a conclusion. The performances of our models is poor and all models need many modifications for better performance. We observed that deep learning methods like LSTM, RNN or CNN-LSTM with pre-trained word embedding methods like glove gave good results for some researches. As we did not use any deep learning technique in our work we can use it and results can be better for this. We will work on this task in future to modify the models and a general model have to be made which can work fine for datasets of any language.

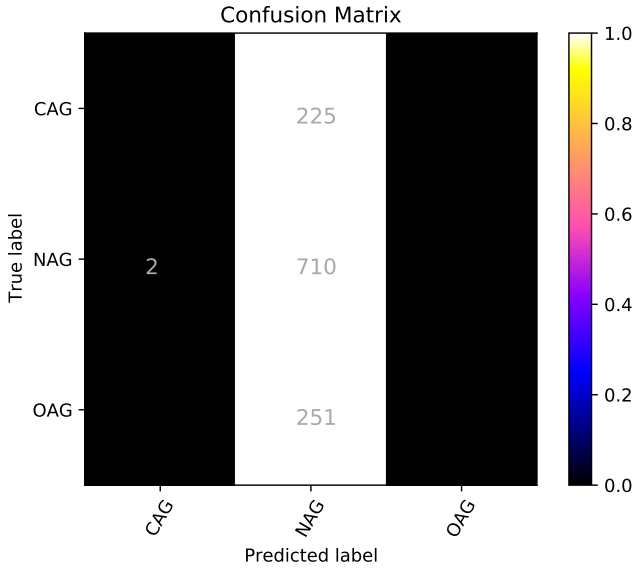


Figure 2: Confusion Matrix for Sub-task BEN-A)

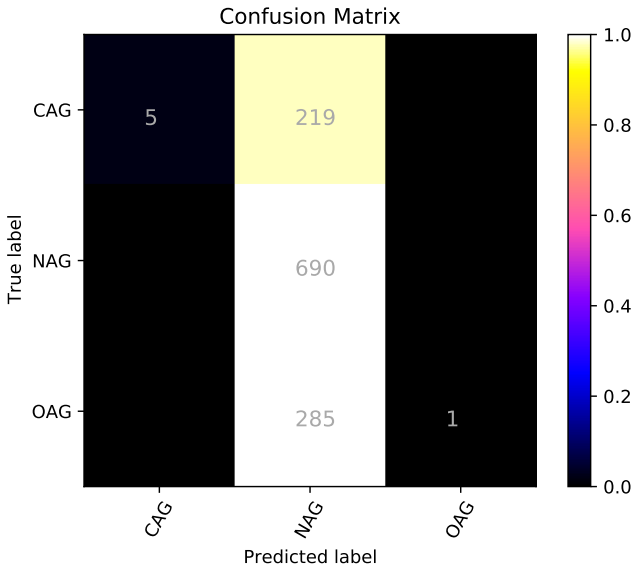


Figure 3: Confusion Matrix for Sub-task EN-A

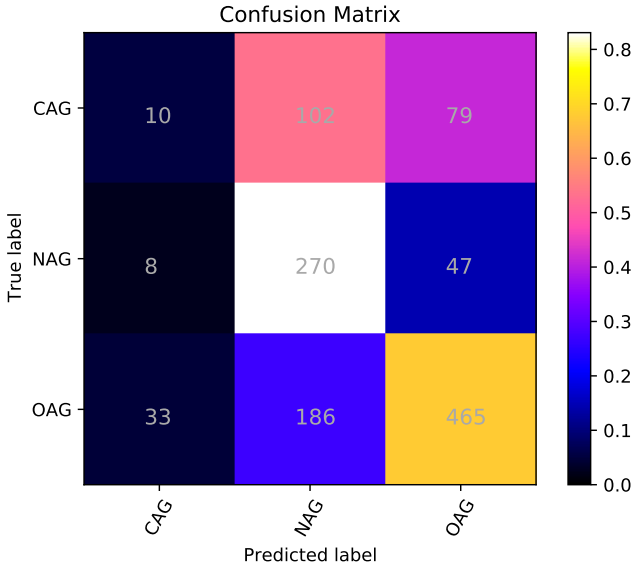


Figure 4: Confusion Matrix for Sub-task HIN-A

This can be done in future.

## 7. Bibliographical References

AmirHRazavi, DianaInkpen, S. and Matwin., S. (2010). Offensive language detection using multi-level classification. In Canadian Conference on Artificial Intelligence, pages 16–27. Springer.

Bhattacharya, S., Singh, S., Kumar, R., Bansal, A., Bhagat, A., Dawer, Y., Lahiri, B., and Ojha, A. K. (2020). Developing a multilingual annotated corpus of misogyny and aggression.

Chikashi Nobata, Joel Tetreault, A. T. Y. M. and Chang, Y. (2016). Abu-sive 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.

Duyu Tang, Furu Wei, N. Y. M. Z. T. L. and Qin, B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 1555–1565.

Hamdy Mubarak, K. D. and Magdy, W. (2017). Abusive language detection on arabic social media. In Proceedings of the First Workshop on Abusive Language Online, pages 52–56.

Hui-Po Su, Zhen-Jie Huang, H.-T. C. and Lin, C.-J. (2017). Rephrasing profanity in chinese text. In Proceedings of the First Workshop on Abusive Language Online, pages 18–24..

Jan Deriu, Maurice Gonzenbach, F. U. A. L. V. D. L. and Jaggi, M. (2016). Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. In Proceedings of the 10th International Workshop on Semantic Evaluation, number EPFL-CONF-229234, pages 1124–1128.

Jun-Ming Xu, Kwang-Sung Jun, X. Z. and Bellmore, A. (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.

Kalika Bali, Jatin Sharma, M. C. and Vyas, Y. (2014). i am borrowing ya mixing?" an analysis of english-hindi code mixing in facebook. In Proceedings of the First Workshop on Computational Approaches to Code Switching, pages 116–126.

Kwok, I. and Wang., Y. (2013). Locate the hate: Detecting tweets against blacks. In Twenty-Seventh AAAI Conference on Artificial Intelligence.

Mohammad, S. M. (2012). emotional tweets. In Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, pages 246–255. Associa-tion for Computational Linguistics.

Nemanja Djuric, Jing Zhou, R. M. M. G. V. R. and Bhamidipati, N. (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 Con-ferences Steering Committee.

Or˘asan, C. (2018). Aggressive language identification using word embeddings and sentiment features. Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 113–119.

Phillips, W. (2015). This is why we can't have nice things: Mapping the relationship between online trolling and mainstream culture. Mit Press.

Ritesh Kumar, Guggilla Bhanodai, R. P. and Chennuru, M. R. (2018). Trac-1 shared task on aggression identification: Iit(ism)@coling'18.

Ritesh Kumar, Atul Kr. Ojha, S. M. and Zampieri, M. (2020). Evaluating aggression identification in social media. In Ritesh Kumar, et al., editors, *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying (TRAC-2020)*, Paris, France, may. European Language Resources Association (ELRA).

Shukrity Si, Anisha Datta, S. B. S. K. N. (2019). Aggression detection on multilingual social media text. 10th ICCCNT - 2019.

Stephan´ Tulkens, Lisa Hilte, E. L. B. V. and Daelemans, W. (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.

Vinay Singh, Aman Varshney, S. S. A. D. V. and Shrivastava, M. (2018). Aggression detection on social media text using deep neural networks. Empirical Methods in Natural Language Processing (EMNLP-2018).

Yogarshi Vyas, Spandana Gella, J. S. K. B. and Choudhury, M. (2014). Pos tagging of english-hindi code-mixed social media content. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 974–979.