Usage of user hate speech index for improving hate speech detection in Twitter posts

Ehlimana Krupalija
Faculty of Electrical Engineering
University of Sarajevo
Sarajevo, Bosnia and Herzegovina
ekrupalija1@etf.unsa.ba

Dženana Đonko
Faculty of Electrical Engineering
University of Sarajevo
Sarajevo, Bosnia and Herzegovina
ddonko@etf.unsa.ba

Haris Šupić
Faculty of Electrical Engineering
University of Sarajevo
Sarajevo, Bosnia and Herzegovina
hsupic@etf.unsa.ba

2022 XXVIII International Conference on Information, Communication and Automation Technologies (ICAT) | 978-1-6654-6692-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICAT54566.2022.9811159

***Abstract*—Social media is an important source of real-world data for sentiment analysis. Hate speech detection models can be trained on data from Twitter and then utilized for content filtering and removal of posts which contain hate speech. This work proposes a new algorithm for calculating user hate speech index based on user post history. Three available datasets were merged for the purpose of acquiring Twitter posts which contained hate speech. Text preprocessing and tokenization was performed, as well as outlier removal and class balancing. The proposed algorithm was used for determining hate speech index of users who posted tweets from the dataset. The preprocessed dataset was used for training and testing multiple machine learning models: k-means clustering without and with principal component analysis, naïve Bayes, decision tree and random forest. Four different feature subsets of the dataset were used for model training and testing. Anomaly detection, data transformation and parameter tuning were used in an attempt to improve classification accuracy. The highest F1 measure was achieved by training the model using a combination of user hate speech index and other user features. The results show that the usage of user hate speech index, with or without other user features, improves the accuracy of hate speech detection.**

Keywords—data mining, sentiment analysis, natural language processing, hate speech detection, Twitter data analysis

I. INTRODUCTION

Sentiment analysis aims to yield a better understanding of human emotions, attitudes and opinions expressed in different forms of language [1] [2]. The availability of many electronic sources (e.g. blogs, online communities, discussion forums, social networks) [3] makes it possible to collect large amounts of authentic data to classify different forms of affects (opinions, views and beliefs) [4] [5]. Three different approaches [6] based on knowledge, statistical methods or both are used to detect emotions in textual content. In sentiment analysis, the textual content itself is analyzed at sentence, document or aspect level [2].

Social networks are an abundant source of data for sentiment analysis due to the amount of emotions and opinions present in user posts. Twitter, a social network with more than 217 million monetizable daily active usage [7], is especially convenient for data mining because it has a small character per post limit (280 characters [8]) and a tag usage system which can be used as powerful filtering criteria.

Hate speech is defined as content that expresses and encourages intolerance, discrimination or any other negative emotion aimed at any particular object or person [9] [10] [11]. Free speech [12] is closely related to and often mislabeled as hate speech, which makes it hard to distinguish these two terms. Unlike hate speech, free speech is not intended to hurt or degrade the group or individual it is aimed at, although in some cases, it can be hard to determine whether the content represents free or hate speech [9] [12]. Because of this, hate speech detection models need to be trained and specifically aimed at hate speech content.

Two important factors make social networks a common source of hate speech: amplification and anonymousness [10]. Users who spread hate speech can both keep their identity hidden and directly contact those who they wish to harm. Hate speech spread is often event-related on Twitter [10] [13] [14] (e.g. scandals, terrorist attacks, elections, COVID-19 pandemic) which makes this social network a valuable source of hate speech posts in data mining.

Machine and deep learning on Twitter data has been successfully used for hate speech detection targeting multiple groups, such as Black-Lives-Matter (BLM) movement [15], religious groups [16], racial groups [17] and immigrants in Spain [13]. Some approaches use counter speech interaction on Twitter for hate speech detection [18], whereas others use analytical tools such as Tweet Binder to automatically detect hate speech in various Twitter content [19]. In [15], multiple deep learning models were applied for detecting hate speech against the BLM movement, where the best F1 measure achieved was 88.9%. In [20], multiple machine learning models (KNN, logistic regression, SVM and naïve Bayes) as well as multiple deep learning models (LSTM, BERT) were applied for detecting hate speech spreaders. The results show that the machine learning models (accuracy of 66%) outperformed deep learning models (accuracy of 54%). In [21], multiple datasets were used for comparing SVM and CNN classifiers, where CNN classifier and its proposed modification (highest F1 measure of 94%) outperformed SVM model (highest F1 measure of 90%).

Usage of user features has shown to improve model performance, as demonstrated in [20], where average F1 score of 83% was achieved by using information about the user gender, followers and public lists as additional features. Usage of user post history for labeling users as hate speech propagators has also been proposed [21], where multiple text representations were combined and accuracy of 63% was achieved through the usage of majority voting of SVM and random forest classifiers.

As Twitter becomes more popular, hate speech detection and content filtering become more important. The usage of Twitter user features and history of posted content can be utilized for machine learning model performance improvement. This work aims to determine whether using user hate speech index in combination with other user features along with post content can improve the success of hate speech detection. In Section II, data acquiring and preprocessing is explained. An algorithm which uses the timeline of the user in order to calculate the user hate speech index is proposed. In Section III, various machine learning models are trained on the proposed dataset. Results achieved on the original, balanced, anomaly-free and transformed dataset are shown and compared. In Section IV, an analysis of

model performances is made and user hate speech index importance is determined.

II. METHODOLOGY

In order to prepare the data for machine learning methods, several preprocessing steps were necessary – merging multiple available datasets into a single large dataset, text preprocessing, calculating user hate speech index based on user history, outlier removal and class balancing.

A. Merging multiple datasets

In order to include a variety of tweets from different sources and with different hate speech types, multiple available datasets were merged into a single larger dataset. Table I lists the three chosen datasets with the total number of tweets originally contained in each dataset. Only 21.46% of tweets from all datasets (30,834 tweets) were available. The remaining tweets could not be fetched because tweets or the users who posted them have either been deleted or made private since the dataset was published. All datasets contained different labels which were unified into two new labels: normal (*neither, none, normal*) and hate speech (*hate speech, offensive, racism, sexism, hateful, abusive*).

In order to acquire additional information about all instances from the merged dataset, Python library *tweepy* was used. The following attributes were added to the dataset: tweet text, user screen name, user location, user followers count and user statuses count.

TABLE I. THE STRUCTURE OF CHOSEN AVAILABLE DATASETS FOR HATE SPEECH DETECTION

Dataset	Number of tweets	Labels
T-Davidson [22]	26,953	*hate speech, offensive, neither*
NAACL SRW [23]	16,906	*racism, sexism, none*
ENCASEH2020 [24]	99,799	*hateful, abusive, normal*
Total number of tweets	**143,658**	

B. Calculating user hate speech index

Another column was added to the dataset by using the history of user posts to determine the user hate speech index (labeled as *uhsi*). The pseudocode that describes this process is shown in Algorithm 1. Python library *tweepy* was used for acquiring the 20 newest posts of the desired used by using the username added to the dataset in the previous step. Python library *hatesonar*, which uses logistic regression for hate speech detection [27], was used for automatically determining whether individual posts are a form of hate speech or not.

Because some user data was unavailable (e.g. user had less than 20 Twitter posts), another attempt at calculating user hate speech index was made, in which 2 instead of 20 newest posts were used. If the second attempt failed, user hate speech index of -1 was added to the dataset.

C. Preprocessing data

Fig. 1 shows all steps which were taken in order to preprocess the tweet text column contained in the dataset. Raw textual data from the dataset contained different types of noise which needed to be removed in order to prepare the data as input for machine learning models. This noise included the usage of emoticons, special characters, links, user tags (@username), hashtags (#tag) and retweet tags (RT:). After removing this noise, trailing whitespace was removed and all letters were converted to lowercase. After this step, 109 tweets were empty (they did not contain anything except noise). After their removal, the dataset size was reduced to 30,725 tweets.

Algorithm 1: ***Determining user hate speech index***

Input: Twitter username of the desired user ($username$)

Output: User hate speech index ($uhsi$)

$Timeline \leftarrow get\_newest\_tweets(username, 20)$
$uhsi \leftarrow 0$
for $i \leftarrow 0$ to 20
    $label \leftarrow hatesonar(Timeline[i])$
    if $label\ != 'normal'$
        $uhsi \leftarrow uhsi + 1$
    end if
end for
$uhsi \leftarrow \frac{uhsi}{20}$
Return $uhsi$

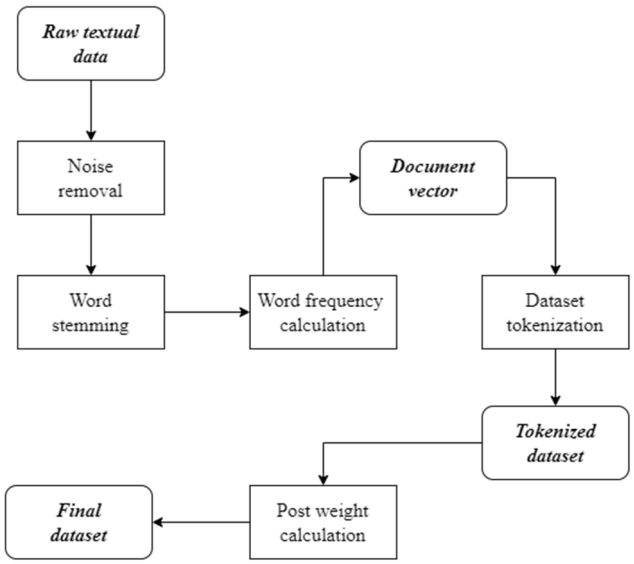

Fig. 1. Steps necessary for performing dataset tweet text preprocessing

Word stemming was applied to the resulting noise-free textual data. For this purpose, Python library *nltk* was used. The resulting dataset contained only word stems, which allowed different forms of the same word (e.g. *do, does, done*) to be encoded as one token. In order to encode the textual data, it was first necessary to create the document vector, which contained all different word stems of the dataset - $s[i]$. The frequency of each word stem $f(s[i])$ was calculated as the sum of occurrences of the stem word $n_{s[i]}$ in every instance of the dataset $dataset[k]$, as shown in (1). The weight of each word stem $w(s[i])$ was calculated as the ratio of instances of the dataset $dataset[k]$ which are labelled as normal text that contain the word stem $s[i]$, as shown in (2).

$$f(s[i]) = \sum_{k=0}^{len(dataset)} n_{s[i]}\,(dataset[k]) \quad (1)$$

$$w(s[i]) = \frac{\sum_{k=0}^{len(dataset)} \begin{matrix} s[i]\ in\ dataset[k]\ and \\ label(dataset[k]) == 'normal' \end{matrix}}{len(dataset)} \quad (2)$$

The dataset tokenization was performed after the document vector of the dataset was created. Every word stem in tweet texts of the dataset was replaced by its row number – token in the document vector, which resulted in tweet text column becoming a list of integer values.

Additionally, tweet post weight $pw(dataset[k])$ was calculated as the sum of all frequency and weight products ($f(s[i])$ and $w(s[i])$) for each stem $s[i]$ contained in the tweet text for every dataset instance $dataset[k]$, as shown in (3). After this step, the dataset preprocessing was finished.

$$pw(dataset[k]) = \frac{\sum_{i=0}^{len(dataset[k])} f(s[i])\,w(s[i])}{len(dataset[k])} \quad (3)$$

D. Outlier removal

In order to reduce the amount of noise in the dataset, outlier removal was performed. All instances with user hate speech index set to -1 (unavailable user history) were removed, which reduced the dataset size by 10,901 instances. Scatterplot graphs for four dataset features (user posts, user followers, post weight and user hate speech index) are shown on Fig. 2. The graphs were used to determine outliers in the dataset. All instances with more than 10^6 followers, more than 30^6 user posts, post weight bigger than 4,000 or user hate speech index bigger than 0.8 were removed from the dataset. This reduced the dataset size by 75 instances. The final dataset size after outlier removal was 19,749 instances.

E. Class balancing

Dataset class distribution was analyzed. The class distribution of the original dataset was not equal – 35.47% of instances were labeled as hate speech, whereas 64.53% of instances were labeled as normal. Oversampling technique was used to balance the class distribution, as shown on Fig. 3, which resulted in dataset size being increased to 25,490 instances by oversampling the hate speech class.

III. Results

A. Applying machine learning models

In order to perform hate speech detection, multiple machine learning models were used – k-means clustering, naïve Bayes, decision tree and random forest. 8-fold validation was performed on all models. Different combinations of dataset features were used as input for all machine learning models:

- one feature (post weight);
- two features (post weight, user hate speech index);
- three features (post weight, user posts, user followers);
- four features (post weight, user hate speech index, user posts, user followers).

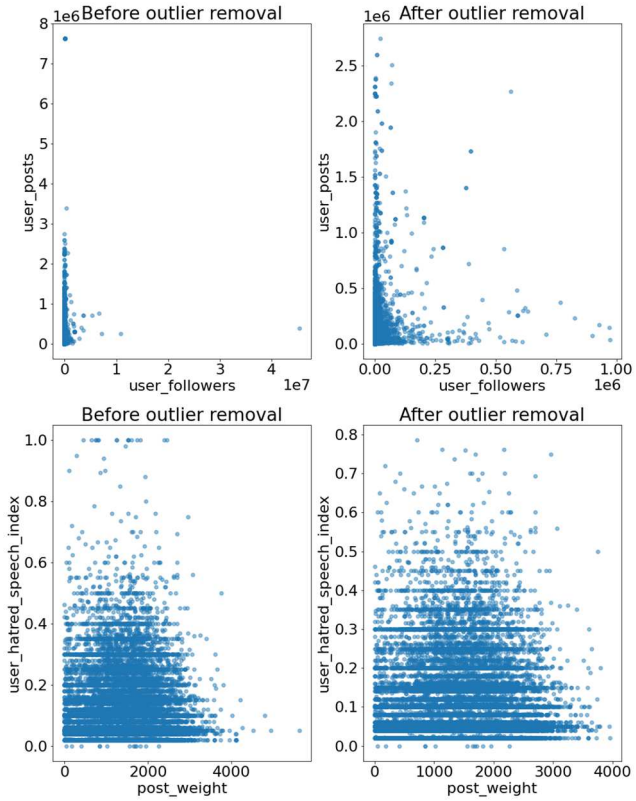

Fig. 2. Scatterplot graphs of dataset features used for determining outliers

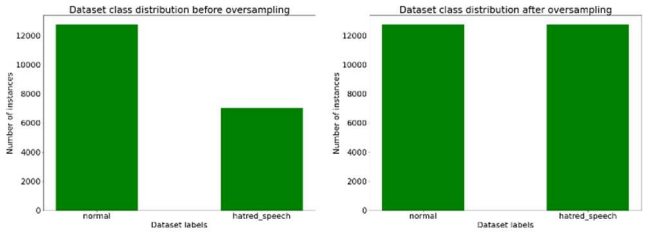

Fig. 3. Dataset class distribution before and after oversampling

K-means clustering unsupervised machine learning model with k = 2 was applied to the train subset in order to determine the clustering tendency of the dataset instances. Principal component analysis (PCA) was also performed to transform the dataset into a form that captures the most valuable information encoded in its attributes. The results of the clustering without and with the usage of principal component analysis are shown on Fig. 4, where it is visible how the dataset attribute range significantly changes after applying PCA and extracting the principal components.

Three different naïve Bayes supervised machine learning models were applied to the dataset in order to perform classification – Gaussian, complement and multinomial models. Decision tree supervised machine learning model was also applied to the dataset in order to perform classification. The visualization of the decision tree model after training is shown on Fig. 5, where it is visible that the tree was very complex and contained a large number of rules which were impossible to distinguish in order to perform tree pruning.

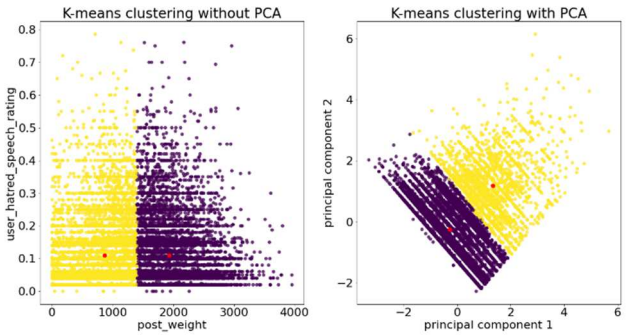

Fig. 4. Resulting k-means clustering models without and with the usage of PCA

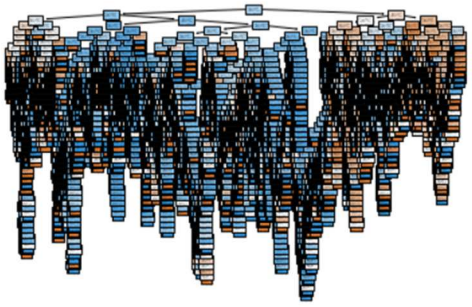

Fig. 5. Visualization of decision tree rules after training the model

The performances of all machine learning models which were applied to the dataset are shown on Fig. 6. All models were applied to original imbalanced and oversampled balanced data. F1 measure was used as criteria because accuracy showed misleading results (e.g. k-means clustering model without PCA with four features on original data achieved 60.87% accuracy, but only 8.82% F1 measure). The graphs show that all models trained on oversampled data achieved higher performances than the models trained on original imbalanced data. In all graphs, models trained by using two or four features (which contain user hate speech rating) achieved better performances than models trained by using one or three features (which do not contain user hate speech rating). The highest F1 measure of 81.18% was achieved for random forest model by using four dataset features.

B. Tuning the models

Python library *sklearn* was used for model tuning. Anomaly detection algorithm *IsolationForest*, which calculates anomaly score by recursively partitioning instances represented in a tree structure, was used to remove 1,966 instances from the original imbalanced dataset and 2,549 instances from the balanced dataset. *PowerTransformer* model was used to stabilize data variance and minimize data skewness. *GridSearchCV* method was used for model parameter tuning. The results of tuning for all machine learning models with the usage of different numbers of features are shown in Table II. PCA model was not included due to the usage of principal component transformation in the initial data. Comparison of best F1 measure without the usage of anomaly detection, with the usage of anomaly detection and with the usage of anomaly detection and data transformation showed F1 measure increase for k-means clustering and Gaussian naïve Bayes models, as well as F1 measure decrease for decision tree and random forest models. The performances of all models trained by using features which contain user hate speech rating achieved better performances than models trained by using features which do not contain user hate speech rating.

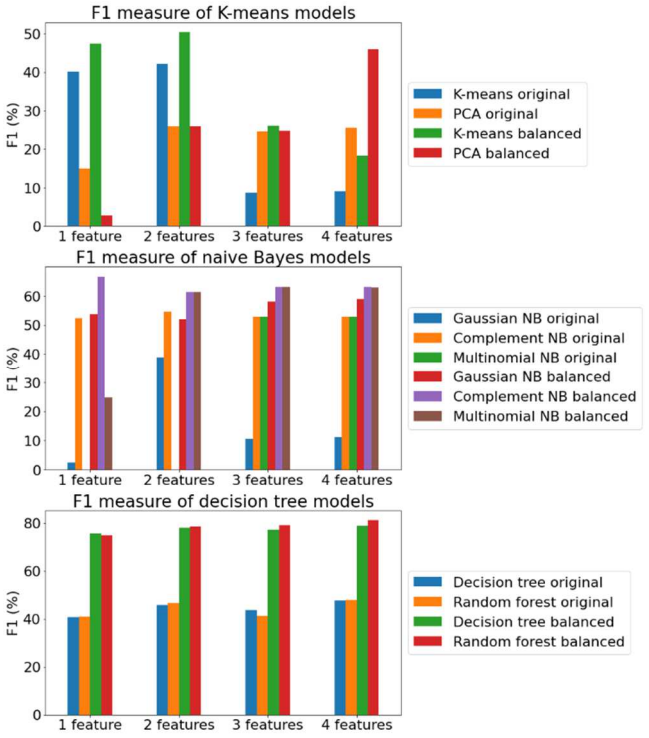

Fig. 6. Graphs showing performances of machine learning models depending on the number of features which were used and class balancing

TABLE II. F1 MEASURE COMPARISON FOR MACHINE LEARNING MODELS WITHOUT AND WITH DATASET ANOMALY DETECTION

Model	Features	Without anomaly detection	With anomaly detection	With data transfor-mation
K-means clustering without PCA	1	47.31%	48.41%	51.42%
	2	50.38%	50.07%	58.84%
	3	26.09%	30.55%	52.45%
	4	18.27%	37.82%	51.16%
Gaussian naïve Bayes	1	53.74%	49.20%	56.41%
	2	51.93%	51.96%	66.94%
	3	58.05%	65.27%	58.58%
	4	58.86%	65.23%	65.43%
Decision tree	1	75.62%	75.82%	55.08%
	2	77.97%	78.18%	65.25%
	3	77.28%	77.42%	59.83%
	4	78.95%	78.44%	65.97%
Random forest	1	74.95%	74.91%	68.27%
	2	78.52%	78.21%	72.68%
	3	79.10%	78.58%	73.93%
	4	81.18%	81.03%	75.78%

The overall best results achieved for all machine learning models are shown on Fig. 7. Decision tree and random forest models achieved the best performances (F1 measure of

81.18% for model trained by using four features). The heatmap shows that for all models, achieved performances were better if the models were trained by using user hate speech index. The average improvement of F1 measure with the usage of two features compared to one feature was 4.07% and compared to three features was 3.20%. The average improvement of F1 measure with the usage of four features compared to one feature was 4.70% and compared to three features was 3.83%. The overall average improvement of F1 measure when using user hate speech index (two or four features) was 3.95%.

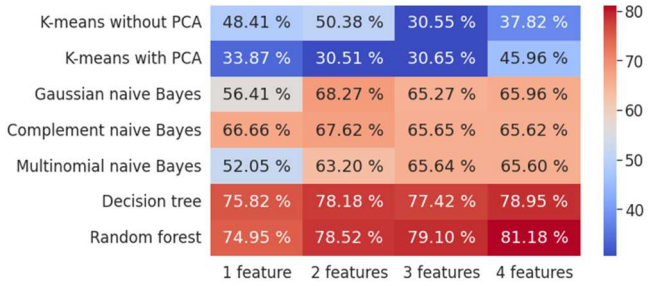

Fig. 7. Heatmap showing highest F1 measures of machine learning models for different number of dataset features used

IV. Conclusion

Social media posts offer valuable information for sentiment analysis. A big problem arises due to the instable availability of media, which makes dataset merging necessary for retaining large dataset size. Careful preprocessing of data needs to be done in order to transform the data into a form compliant with machine learning model training. This process includes acquiring information about the tweets and the users who posted them, preprocessing textual information and calculating user speech index.

The achieved results are comparable with state-of-the-art models when taking into consideration that different datasets and combinations of user features were used for hate speech detection. Determining user hate speech index requires additional time, but its usage improves the classification accuracy of used machine learning models for 3.95% on average. Usage of additional features such as the number of user posts and followers can further improve the model performances. Decision tree and random forest models are better generalized for hate speech detection, achieving the highest F1 measures of all models which were applied to the proposed data (78.95% for decision tree model, 81.18% for random forest model). These models do not require anomaly detection or data transformation, which makes the classification process faster and less expensive.

The models can be further improved by using more than 20 user posts and manual labeling instead of using *hatesonar* library for user hate speech index calculation, making the user post history even more important for classification. GPU parallelization techniques can be used to make the process faster, and other techniques can be used for dataset tokenization (e.g. one-hot vector, TF-IDF). Other features such as user friend count or images and links contained in the tweets can be used to evaluate whether the classification accuracy can be further improved. Other datasets also need to be utilized because a high number (around 80%) of tweets from the original datasets were unavailable.

References

[1] K. Ahmad, "The 'Return' and 'Volatility' of Sentiments: An Attempt to Quantify the Behaviur of the Markets?," in Affective Computing and Sentiment Analysis: Emotion, Metaphor and Terminology, vol. 45, Springer Science+Business Media, 2011, pp. 89-99.

[2] B. Liu, Sentiment Analysis: Mining Opinions, Sentiments and Emotions, New York: Cambridge University Press, 2015.

[3] P. Mika, Social Networks and the Semantic Web, New York: Springer Science+Business Media, 2007.

[4] A. Kumar and T. M. Sebastian, "Sentiment Analysis: A Perspective on its Past, Present and Future," I.J. Intelligent Systems and Applications, vol. 10, pp. 1-14, 2012.

[5] B. Pang and L. Lee, Opinion Mining and Sentiment Analysis, Now Publishers Inc.: Hanover, 2008.

[6] E. Cambria, D. Das, S. Bandyopadhyay and A. Feraco, "Affective Computing and Sentiment Analysis," in A Practical Guide to Sentiment Analysis, Springer International Publishing, Cham, 2017, pp. 1-10.

[7] I. Twitter, "Investor Relations," 2021. [Online]. Available: https://s22.q4cdn.com/826641620/files/doc_financials/2021/q4/Final-Q4'21-Selected-Metrics-and-Financials.pdf. [Accessed 27 3 2022].

[8] I. Twitter, "Developer Platform - Counting characters," [Online]. Available: https://developer.twitter.com/en/docs/counting-characters. [Accessed 27 3 2022].

[9] N. Wolfson, Hate Speech, Sex Speech, Free Speech, Westport: Praeger Publishers, 1997.

[10] M. Williams, "Hatred Behind the Screens: A Report on the Rise of Online Hate Speech," Mishcon de Reya, London, 2019.

[11] L. Anderson and M. Barnes, "Hate Speech," Metaphysics Research Lab, Stanford University, 2022.

[12] J. Weinstein, Hate speech, pornography, and the radical attack on free speech doctrine, Boulder: Westview Press, 1999.

[13] C. A. Calderón, G. de la Vega and D. B. Herrero, "Topic Modeling and Characterization of Hate Speech against Immigrants on Twitter around the Emergence of a Far-Right Party in Spain," Social Sciences, vol. 9, no. 11, 2020.

[14] H. M. M. Caldera, N. Meedin and I. Perera, "Time Series Based Trend Analysis for Hate Speech in Twitter During COVID 19 Pandemic," in 20th International Conference on Advances in ICT for Emerging Regions (ICTer 2020), Colombo, 2020.

[15] S. Kumar and R. R. Pranesh, TweetBLM: A Hate Speech Dataset and Analysis of Black Lives Matter-related Microblogs on Twitter, arXiv, 2021.

[16] T. Zia, M. S. Akhram, M. S. Nawaz, B. Shahzad, A. M. Abdullatif, R. U. Mustafa and M. I. Lali, "Identification of hatred speeches on Twitter," International Journal of Advances in Electronics and Computer Science, vol. 4, no. 1, pp. 46-51, 2017.

[17] I. Kwok and Y. Wang, "Locate the Hate: Detecting Tweets against Blacks," in Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, Bellevue, 2013.

[18] J. Garland, K. Ghazi-Zahedi, L. Hébert-Dufresne and M. Galesic, "Countering hate on social media: Large-scale classification of hate and counter speech," in Proceedings of the Fourth Workshop on Online Abuse and Harms, 2020.

[19] F. H. A. Shibly, "A Measurement Study on Racist Hate Speech in Twitter using Tweet Binder," Journal of Information Systems & Information Technology (JISIT), vol. 4, no. 1, pp. 1-9, 2019.

[20] R. Jain, D. Goel, P. Sahu, A. Kumar and J. P. Singh, "Profiling Hate Speech Spreaders on Twitter," in CEUR Workshop Proceedings - CLEF 2021 – Conference and Labs of the Evaluation Forum, Bucharest, 2021.

[21] Z. Zhang, D. Robinson and J. Tepper, "Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network," in ESWC 2018: The, Heraklion, 2018.

[22] E. F. Unsvag and B. Gamback, "The Effects of User Features on Twitter Hate Speech Detection," in Proceedings of the Second Workshop on Abusive Language Online (ALW2), Brussels, 2018.

[23] C. M. V. de Andrade and M. A. Gonçalves, "Profiling Hate Speech Spreaders on Twitter: Exploiting Textual Analysis of Tweets and Combinations of Multiple Textual Representations," in CLEF 2021 – Conference and Labs of the Evaluation Forum, Bucharest, 2021.

[24] T. Davidson, D. Warmsley, M. Macy and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language," in Proceedings of the 11th International AAAI Conference on Web and Social Media, Montreal, 2017.

[25] Z. Waseem and D. Hovy, "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter," in Proceedings of the NAACL Student Research Workshop, San Diego, 2016.

[26] A.-M. Founta, C. Djouvas, D. Chatzakou, I. Leontiadis, J. Blackburn, G. Stringhini, A. Vakali, M. Sirivianos and N. Kourtellis, "Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior," in 11th International Conference on Web and Social Media, ICWSM 2018, Palo Alto, 2018.

[27] S. Zannettou, M. ElSherief, E. Belding, S. Nilizadeh and G. Stringhini, "Measuring and Characterizing Hate Speech on News Websites," in WebSci '20: 12th ACM Conference on Web Science, Southampton, 2020.