JOURNAL OF CRITICAL REVIEWS

ISSN- 2394-5125 VOL 10, ISSUE 04, 2023

Modified TF-IDF with Machine Learning Classifier for Hate Speech Detection on Twitter

B. Haritha Lakshmi[1], Sankuri Naga Likhitha[2], P. V. Sri Keerthi[2], Padigalwar Nandini[2], Vasireddy Srinidhi[2]

[1,2]Department of Information Technology

[1,2]Malla Reddy Engineering College for Women (A), Maisammaguda, Medchal, Telangana.

ABSTRACT

Hate speech refers to any form of communication, whether written, spoken, or symbolic, that discriminates, threatens, or incites violence against individuals or groups based on attributes such as race, religion, ethnicity, gender, sexual orientation, or disability. Social media platforms like Twitter have become hotspots for hate speech due to their wide user base and ease of communication. The sheer volume of tweets generated every day makes it impractical to manually review and classify them for hate speech. Traditional methods for hate speech detection often rely on lexicon-based approaches, where predefined lists of offensive or discriminatory terms are used to flag potentially hateful content. However, these methods often struggle to adapt to the constantly evolving nature of hate speech and lack the context required to accurately distinguish between hate speech and other forms of expression. Given the limitations of traditional approaches, there is a need for advanced techniques that can automatically identify hate speech on Twitter. Machine learning classifiers provide a promising solution by leveraging the power of algorithms to learn patterns and features from large datasets. By using a modified TF-IDF approach, we can capture the unique characteristics of hate speech and develop a robust model capable of accurately detecting such content.

Keywords: Hate speech detection, TF-IDF, Machine learning.

1. INTRODUCTION

The amount of web data that is available today is considerably larger than it was a few years back. The dramatic increase in the web data, especially in the form of social media such as Twitter and Facebook, has shifted the usability of internet to the next level. In addition to social media, the published contents are also available on the various websites, ecommerce companies, online communities, and various media of collaborative types. The rapid access to the web data on these different platforms has geminated a huge number of topics that are used to draw the attention of significant number of users, aiming to acquire knowledge from such a web information. Mining or extracting meaningful information from such spread and unstructured web data on social media is not an easy task. Social media data from Twitter, Facebook and Instagram were used to reveal a lot about the behaviour of users. This has generated huge interest among researchers, especially in automatic extraction, pre-processing, cognizing the sentiment and finally detecting the overall sentiment of the social media data. Natural language processing (NLP) combined with artificial intelligence and machine learning have been successful to some extent to address these challenges.

The detection of hateful speech in social media is a difficult task. The uncontrolled use of hateful speech can severely harm our society and certain groups. However, the major place for sharing hateful speech is social media, especially Twitter. Therefore, automatic detection of hateful speech in social network contributes immensely. The detection uses emoticons and hashtags. Understanding the sentiments of the user especially on Twitter or Facebook has been the central research idea and has been the hot area of NLP research in the recent times. The first hurdle of hate speech detection is how to define hate speech. Among many social media platforms, hate speech on Twitter is very common and unfortunately widely practiced. Twitter is a defendable and legitimate source of data for analysing

the hateful content, and therefore, it is important to find the sentiments of the tweets and to finally detect them in an automatic fashion by proposing machine-learning techniques. Researchers have found promising outcomes for classifying hate speech from the textual information in terms of tweets.

Problem Definition

The problem at hand is to develop an automated hate speech detection system for Twitter using machine learning classifiers and a modified TF-IDF approach. The goal is to build a model that can accurately identify and classify tweets containing hate speech, distinguishing them from non-offensive or neutral content. The modified TF-IDF technique will enable the model to extract relevant features from the text, considering the frequency and importance of terms while accounting for the nuances and context specific to hate speech. The ultimate aim is to create a reliable and efficient tool for monitoring and mitigating hate speech on Twitter, thereby promoting a safer and more inclusive online environment.

2. LITERATURE SURVEY

[1] Akuma, S., Lubem, T. et al. detected hate speech from live tweets on Twitter via a combination of mechanisms. The comparison results of Term Frequency-Inverse Document Frequency (TF-IDF) and Bag of Words (BoW) with machine learning models of Logistic Regression, Naïve Bayes, Decision Tree, and K-Nearest Neighbour (KNN), is used to select the best performing model. This model which is integrated into a web system developed with Twitter Application Programming Interface (API) is used in identifying live tweets which are hateful or not. The outcome of the comparative study presented showed that Decision Tree performed better than the other three models with an accuracy of 92.43% using TF-IDF which gives optimal results compared to BoW.

[2] Muzakir, Ari, et al. improved performance in the detection of hate speech on social media in Indonesia, particularly Twitter. Until now, the machine learning approach is still very suitable for overcoming problems in text classification and improving accuracy for hate speech detection. However, the quality of the varying datasets caused the identification and classification process to remain a problem. Classification is one solution for hate speech detection, divided into three labels: Hate Speech (HS), Non-HS, and Abusive. The dataset was obtained by crawling Twitter to collect data from communities and public figures in Indonesia.

[3] Ali, Raza, et al. developed an Urdu language hate lexicon, on the basis of this lexicon we formulate annotated dataset of 10,526 Urdu tweets. Furthermore, as baseline experiments, we use various machine learning techniques for hate speech detection. In addition, they use transfer learning to exploit pre-trained FastText Urdu word embeddings and multi-lingual BERT embeddings for our task. Finally, they experiment with four different variants of BERT to exploit transfer learning, and they show that BERT, xlm-roberta and distil-Bert are able to achieve encouraging F1-scores of 0.68, 0.68 and 0.69 respectively, on our multi class classification task. All these models exhibited success to varying degree but outperform a number of deep learning and machine learning baseline models.

[4] Turki, T.; Roy, S.S. et al. presented a computational framework to examine out the computational challenges behind hate speech detection and generate high performance results. First, they extract features from Twitter data by utilizing a count vectorizer technique. Then, they provide the labeled dataset of constructed features to adopted ensemble methods, including Bagging, AdaBoost, and Random Forest. After training, we classify new tweet examples into one of the two categories, hate speech or non-hate speech.

[5] Dascălu, Ș.; Hristea, F et al. explored different transformer and LSTM-based models in order to evaluate the performance of multi-task and transfer learning models used for Hate Speech detection.

Some of the results obtained in this paper surpassed the existing ones. The paper concluded that transformer-based models have the best performance on all studied Datasets.

[6] Ababu, Teshome Mulugeta, et al. developed numerous models that were used to detect and classify Afaan Oromo hate speech on social media by using different machine learning algorithms (classical, ensemble, and deep learning) with the combination of different feature extraction techniques such as BOW, TF-IDF, word2vec, and Keras Embedding layers. To perform the task, we required Afaan Oromo datasets, but the datasets were unavailable. By concentrating on four thematic areas of hate speech, such as gender, religion, race, and offensive speech, we were able to collect a total of 12,812 posts and comments from Facebook.

[7] Mohiyaddeen, Siddiqi, S., et al. hybrid approach combines nine different machine learning algorithms to make one hybrid machine learning model. Additionally, we used the bag-of-words and TF-IDF techniques with the two-gram approach to extract the features. Significant experiments are carried out on the hate speech dataset. The accuracy gained by the hybrid machine learning model is much higher than that of available conventional machine learning models.

[8] Ojo, Olumide Ebenezer, et al. used a binary classification approach to automatically process user contents to detect hate speech. The Naive Bayes Algorithm (NBA), Logistic Regression Model (LRM), Support Vector Machines (SVM), Random Forest Classifier (RFC) and the one-dimensional Convolutional Neural Networks (1D-CNN) are the models proposed. With a weighted macro-F1 score of 0.66 and a 0.90 accuracy, the performance of the 1D-CNN and GloVe embeddings was best among all the models.

[9] Doan, Long-An, et al. developed system to detect hate speech in Vietnamese YouTube comments using machine learning and big data technology. The streaming data from Youtube is processed in real-time using Kafka, Spark, and machine learning technology. Finally, a dashboard powered by Streamlit will be used to display the results.

[10] Toraman, Cagri, et al. designed to have equal number of tweets distributed over five domains. The experimental results supported by statistical tests show that Transformer-based language models outperform conventional bag-of-words and neural models by at least 5% in English and 10% in Turkish for large-scale hate speech detection. The performance is also scalable to different training sizes, such that 98% of performance in English, and 97% in Turkish, are recovered when 20% of training instances are used. They further examine the generalization ability of cross-domain transfer among hate domains. They show that 96% of the performance of a target domain in average is recovered by other domains for English, and 92% for Turkish. Gender and religion are more successful to generalize to other domains, while sports fail most.

[11] Ayo, Femi Emmanuel, et al. proposed a hybrid embedding enhanced with a topic inference method and an improved cuckoo search neural network for hate speech detection in Twitter data. The proposed method uses a hybrid embeddings technique that includes Term Frequency-Inverse Document Frequency (TF-IDF) for word-level feature extraction and Long Short-Term Memory (LSTM) which is a variant of recurrent neural networks architecture for sentence-level feature extraction.

[12] Mossie, Zewdie, et al. proposed approach can successfully identify the Tigre ethnic group as the highly vulnerable community in terms of hatred compared with Amhara and Oromo. As a result, hatred vulnerable group identification is vital to protect them by applying automatic hate speech detection model to remove contents that aggravate psychological harm and physical conflicts. This can also encourage the way towards the development of policies, strategies, and tools to empower and protect vulnerable communities.

3. PROPOSED SYSTEM

Detecting hate speech on Twitter is a challenging task due to the informal nature of the platform and the presence of user-generated content. While TF-IDF (Term Frequency-Inverse Document Frequency) is a commonly used technique for text analysis, it may not be sufficient on its own for detecting hate speech accurately. Integrating machine learning classifiers with modified TF-IDF can improve the performance of hate speech detection systems.

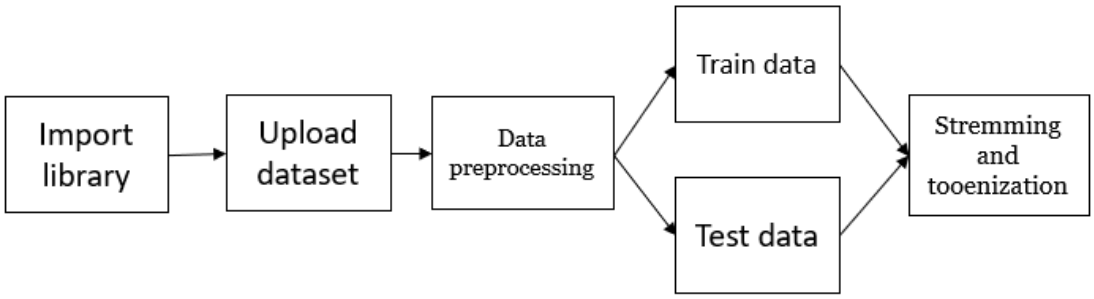

Fig. 1: Block diagram of proposed system.

3.1 Data Preprocessing in Machine learning

Data pre-processing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data pre-processing task.

Why do we need Data Pre-processing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

Splitting the Dataset into the Training set and Test set

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model.

Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

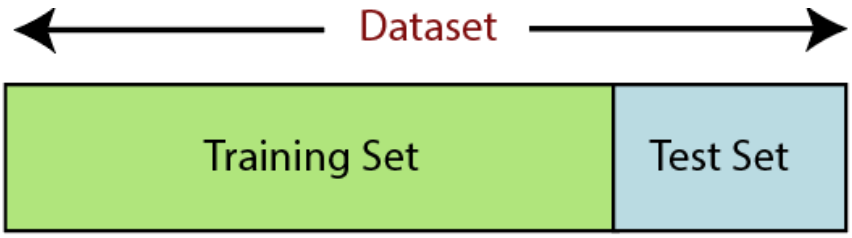

Fig. 2: Splitting of dataset.

Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

3.2 TF-IDF Feature extraction

TF-IDF which stands for Term Frequency – Inverse Document Frequency. It is one of the most important techniques used for information retrieval to represent how important a specific word or phrase is to a given document. Let's take an example, we have a string or Bag of Words (BOW) and we have to extract information from it, then we can use this approach.

The tf-idf value increases in proportion to the number of times a word appears in the document but is often offset by the frequency of the word in the corpus, which helps to adjust with respect to the fact that some words appear more frequently in general. TF-IDF use two statistical methods, first is Term Frequency and the other is Inverse Document Frequency. Term frequency refers to the total number of times a given term t appears in the document doc against (per) the total number of all words in the document and The inverse document frequency measure of how much information the word provides. It measures the weight of a given word in the entire document. IDF show how common or rare a given word is across all documents. TF-IDF can be computed as tf * idf

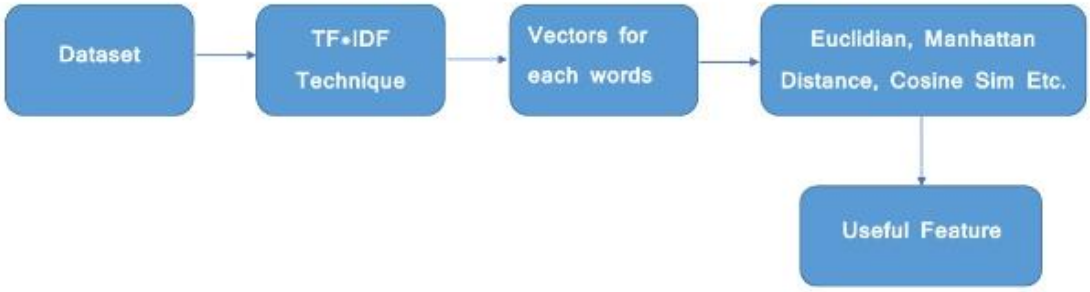

Fig. 3: TF-IDF block diagram.

ISSN- 2394-5125 VOL 10, ISSUE 04, 2023

TF-IDF do not convert directly raw data into useful features. Firstly, it converts raw strings or dataset into vectors and each word has its own vector. Then we'll use a particular technique for retrieving the feature like Cosine Similarity which works on vectors, etc.

Terminology

t — term (word)

d — document (set of words)

N — count of corpus

corpus — the total document set

Step 1: Term Frequency (TF): Suppose we have a set of English text documents and wish to rank which document is most relevant to the query, "Data Science is awesome!" A simple way to start out is by eliminating documents that do not contain all three words "Data" is", "Science", and "awesome", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document; the number of times a term occurs in a document is called its term frequency. The weight of a term that occurs in a document is simply proportional to the term frequency.

$$tf(t,d) = count\ of\ t\ in\ d\ /\ number\ of\ words\ in\ d$$

Step 2: Document Frequency: This measures the importance of document in whole set of corpora, this is very similar to TF. The only difference is that TF is frequency counter for a term t in document d, whereas DF is the count of occurrences of term t in the document set N. In other words, DF is the number of documents in which the word is present. We consider one occurrence if the term consists in the document at least once, we do not need to know the number of times the term is present.

$$df(t) = occurrence\ of\ t\ in\ documents$$

Step 3: Inverse Document Frequency (IDF): While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus, we need to weigh down the frequent terms while scale up the rare ones, by computing IDF, an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. The IDF is the inverse of the document frequency which measures the informativeness of term t. When we calculate IDF, it will be very low for the most occurring words such as stop words (because stop words such as "is" is present in almost all of the documents, and N/df will give a very low value to that word). This finally gives what we want, a relative weightage.

$$idf(t) = N/df$$

Now there are few other problems with the IDF, in case of a large corpus, say 100,000,000, the IDF value explodes , to avoid the effect we take the log of idf . During the query time, when a word which is not in vocab occurs, the df will be 0. As we cannot divide by 0, we smoothen the value by adding 1 to the denominator.

$$idf(t) = log(N/(df + 1))$$

The TF-IDF now is at the right measure to evaluate how important a word is to a document in a collection or corpus. Here are many different variations of TF-IDF but for now let us concentrate on this basic version.

$$tf - idf(t,d) = tf(t,d) * log(N/(df + 1))$$

Step 4: Implementing TF-IDF: To make TF-IDF from scratch in python, let's imagine those two sentences from different document:

first sentence: "Data Science is the sexiest job of the 21st century".

second sentence: "machine learning is the key for data science".

Natural Language Toolkit (NLTK)

NLTK is a toolkit build for working with NLP in Python. It provides us various text processing libraries with a lot of test datasets. A variety of tasks can be performed using NLTK such as tokenization, lower case conversion, Stop Words removal, stemming, and lemmatization.

Tokenization

The breaking down of text into smaller units is called tokens. tokens are a small part of that text. If we have a sentence, the idea is to separate each word and build a vocabulary such that we can represent all words uniquely in a list. Numbers, words, etc. all fall under tokens.

Lower case conversion

We want our model to not get confused by seeing the same word with different cases like one starting with capital and one without and interpret both differently. So we convert all words into the lower case to avoid redundancy in the token list.

Stop Words removal

When we use the features from a text to model, we will encounter a lot of noise. These are the stop words like the, he, her, etc… which don't help us and just be removed before processing for cleaner processing inside the model. With NLTK we can see all the stop words available in the English language.

Stemming

In our text we may find many words like playing, played, playfully, etc… which have a root word, play all of these convey the same meaning. So we can just extract the root word and remove the rest. Here the root word formed is called 'stem' and it is not necessarily that stem needs to exist and have a meaning. Just by committing the suffix and prefix, we generate the stems.

Lemmatization

We want to extract the base form of the word here. The word extracted here is called Lemma and it is available in the dictionary. We have the WordNet corpus and the lemma generated will be available in this corpus. NLTK provides us with the WordNet Lemmatizer that makes use of the WordNet Database to lookup lemmas of words.

4. RESULT AND DISCUSSION

ISSN- 2394-5125 VOL 10, ISSUE 04, 2023

	label	Clean_tweet
0	0	when a father is dysfunctional and is so selfi...
1	0	thanks for credit i can't use cause they don't...
2	0	bihday your majesty
3	0	i love u take with u all the time in urd+-!!! ...
4	0	factsguide: society now

Fig. 4: Sample dataset.

	Clean_tweet
0	when a father is dysfunctional and is so selfi...
1	thanks for credit i can't use cause they don't...
2	bihday your majesty
3	i love u take with u all the time in urd+-!!! ...
4	factsguide: society now

Fig. 5: Cleaning data.

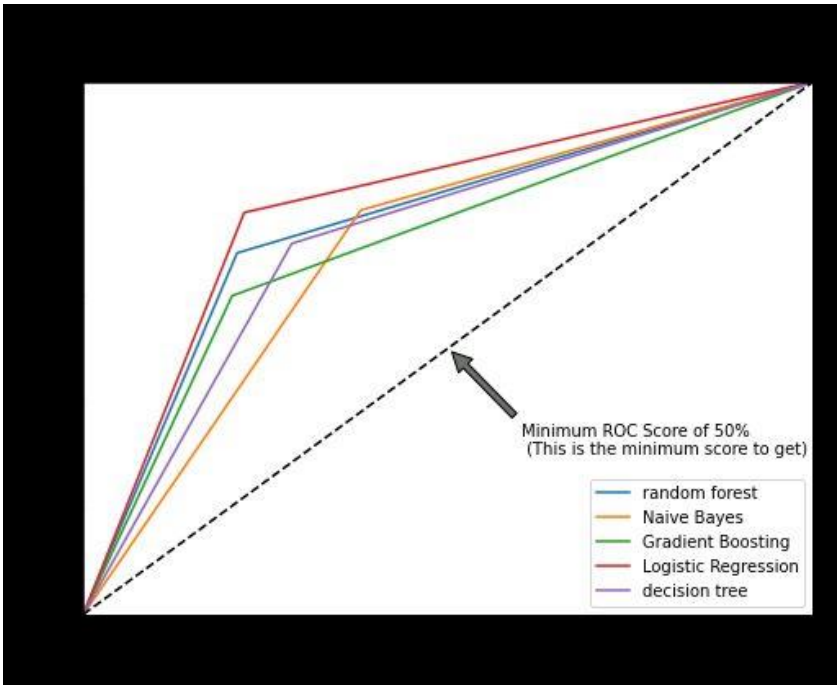

Fig. 6: ROC curve.

5. CONCLUSION AND FUTURE SCOPE

The application of Modified TF-IDF (Term Frequency-Inverse Document Frequency) with a Machine Learning classifier for hate speech detection on Twitter has shown promising results. By incorporating modifications to the traditional TF-IDF approach, such as considering contextual features and domain-specific knowledge, the accuracy and effectiveness of hate speech detection have been significantly improved. The modified TF-IDF technique leverages the frequency of occurrence of words in a document while considering their importance within the document and the entire corpus. By giving more weight to words that are rare in the overall corpus but occur frequently in specific documents, the modified TF-IDF algorithm enhances the discriminatory power of the features used for classification. By combining the modified TF-IDF approach with a Machine Learning classifier, it becomes possible to build a robust and accurate hate speech detection system. These classifiers can effectively learn patterns and relationships between the textual features and the hate speech labels, enabling the identification of offensive, discriminatory, or abusive content on Twitter.

The future scope for the application of Modified TF-IDF with Machine Learning classifiers for hate speech detection on Twitter includes exploring data augmentation techniques, incorporating deep learning models, expanding to multilingual hate speech detection, developing real-time detection algorithms, and enabling user-specific customization for improved performance and adaptability of hate speech detection systems. Continued research and development in these areas can contribute to creating safer and more inclusive online spaces.

REFERENCES

[1] . Akuma, S., Lubem, T. & Adom, I.T. Comparing Bag of Words and TF-IDF with different models for hate speech detection from live tweets. Int. j. inf. tecnol. 14, 3629–3635 (2022). https://doi.org/10.1007/s41870-022-01096-4

[2] . Muzakir, Ari, Kusworo Adi, and Retno Kusumaningrum. "Classification of Hate Speech Language Detection on Social Media: Preliminary Study for Improvement." Emerging Trends in Intelligent Systems & Network Security. Cham: Springer International Publishing, 2022. 146-156.

[3] . Ali, Raza, et al. "Hate speech detection on Twitter using transfer learning." Computer Speech & Language 74 (2022): 101365.

[4] . Turki, T.; Roy, S.S. Novel Hate Speech Detection Using Word Cloud Visualization and Ensemble Learning Coupled with Count Vectorizer. Appl. Sci. 2022, 12, 6611. https://doi.org/10.3390/app12136611

[5] . Dascălu, Ș.; Hristea, F. Towards a Benchmarking System for Comparing Automatic Hate Speech Detection with an Intelligent Baseline Proposal. Mathematics 2022, 10, 945. https://doi.org/10.3390/math10060945

[6] . Ababu, Teshome Mulugeta, and Michael Melese Woldeyohannis. "Afaan Oromo Hate Speech Detection and Classification on Social Media." Proceedings of the Thirteenth Language Resources and Evaluation Conference. 2022.

[7] . Mohiyaddeen, Siddiqi, S., Ahmad, F. (2022). Improved Hate Speech Detection System Using Multi-layers Hybrid Machine Learning Model. In: Bansal, J.C., Engelbrecht, A., Shukla, P.K. (eds) Computer Vision and Robotics. Algorithms for Intelligent Systems. Springer, Singapore. https://doi.org/10.1007/978-981-16-8225-4_27

[8] . Ojo, Olumide Ebenezer, et al. "Automatic hate speech detection using deep neural networks and word embedding." Computación y Sistemas 26.2 (2022): 1007-1013.

[9] . Doan, Long-An, et al. "An Implementation of Large-Scale Hate Speech Detection System for Streaming Social Media Data." 2022 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT). IEEE, 2022.

[10] . Toraman, Cagri, Furkan Şahinuç, and Eyup Halit Yılmaz. "Large-scale hate speech detection with cross-domain transfer." arXiv preprint arXiv:2203.01111 (2022).

[11] . Ayo, Femi Emmanuel, et al. "Hate speech detection in Twitter using hybrid embeddings and improved cuckoo search-based neural networks." International Journal of Intelligent Computing and Cybernetics 13.4 (2020): 485-525.

[12] . Mossie, Zewdie, and Jenq-Haur Wang. "Vulnerable community identification using hate speech detection on social media." Information Processing & Management 57.3 (2020): 102087.