Hateful Sentiment Detection in Real-Time Tweets: An LSTM-Based Comparative Approach

Sanjiban Sekhar Roy, *Member, IEEE*, Akash Roy, Pijush Samui, Mostafa Gandomi, and Amir H. Gandomi, *Senior Member, IEEE*

***Abstract*— It is undeniable that social media has improved our lives in many ways, like allowing interactions with others all over the world and network expansion for businesses. However, there are detrimental effects of such accessibility, including the rapid spread of hate through offensive messages typically directed toward gender, religion, race, and disability, which can cause psychological harm. To address this problem of social media, many researchers have recently proposed various algorithms powered by machine learning (ML) and deep learning for the detection of hate speech. This work proposes a hate speech detection model based on long-short term memory (LSTM), using term frequency inverse document frequency (TF-IDF) vectorization, and makes comparisons with support vector machine (SVM), Naïve Bayes (NB), logistic regression (LR), XGBoost (XGB), random forest (RF), K-nearest neighbor (k-NN), artificial neural network (ANN), and bidirectional encoder representations from transformers (BERT) models. To validate and authenticate our proposed work, we obtained and classified a real-time Twitter data stream of a trending topic using Twitter API into two classes: hate speech and nonhate speech. The precision, recall, and $F1$ score achieved by LSTM are 0.98, 0.99, and 0.98, respectively. The accuracy of LSTM for detecting hateful sentiment was found to be 97%, surpassing the accuracy of other models.**

***Index Terms*— Machine learning (ML), social media, text classification, Twitter hate speech detection.**

I. Introduction

DUE to the easy availability of the Internet over the past decade, the web now hosts more than 4.5 billion active users, around 59% of the total global population, particularly attached to social media [1], [2]. Facebook, Twitter, and WeChat are the most popular platforms among the new generation of Internet users. Specifically, Twitter allows all types of discussions, from celebrity gossip, news updates, recipes, and even just thoughts a user has at that time [3], [4]. However, there are no restrictions on the content posted on Twitter, meaning that people have the freedom to express anything they want, whether it is positive or negative. Recently, the number of hate posts on topics, such as politics, racism, nationality, and religion, has largely increased, especially during the COVID-19 lockdown. The manual identification and removal process of such posts is a cumbersome job considering the seemingly infinite number of hateful tweets. Hence, classifying and detecting hateful speech using natural language processing (NLP) and machine learning (ML) models are significant. Yet, it is still very difficult to perform these operations for real-time tweets on a day-to-day basis [5], [6]. Therefore, we extracted a real-time stream from the Twitter platform using Twitter's application programming interface (API) in this work. In the literature, it has been found that the detection of hate speech is mostly achieved through binary classifications, multiclass classifications, or a combination of both. ML algorithms are commonly used in detection methods for hate speech, which are typically only available in the English language. In this work, we first created an English corpus containing hate and nonhate speech from a real-time stream of tweets via API that is available on Twitter. This application program is used to avail a continuous stream of tweets and help us avoid dependence on the already existing corpora of tweets available on the web. Besides, our constructed corpora of tweets add validity and authenticity to the proposed work. To deal with the corpora of tweets, such as preprocessing, normalization, and other preprocessing-related steps, we used Panda's library. Since the corpus cannot address all issues related to hate speech and its detection, we further classified the tweets using our proposed ML and deep learning models. To identify the words that are responsible for hate speech detection, we incorporated the term frequency inverse document frequency (TF-IDF) technique. This technique adds weight to the word or term to determine its significance in the whole corpus. The value of TF-IDF is proportionate to the number of times a particular word appears in a document. Considering that 83% of document searches on the Internet are conducted using TF-IDF, it seemed to be the best technique to identify hateful and nonhateful tweets in this work.

The word cloud technique was employed to further prepare a visual representation of the tweets. Moreover, we developed a comprehensive strategy to perform the challenging task of finding sentiments based on the considered hashtags using TextBlob, which is a Python-based NLP-supported framework. TextBlob returns the polarity of a sentence along with subjectivity, where a polarity of -1 refers to negative sentiment and $+1$ to positive sentiment.

Manuscript received 25 August 2022; revised 2 December 2022; accepted 13 March 2023. Date of publication 10 May 2023; date of current version 2 August 2024. *(Corresponding author: Amir H. Gandomi.)*

Sanjiban Sekhar Roy and Akash Roy are with the School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, India (e-mail: sanjibansroy@ieee.org; akash.roy2020@vitstudent.ac.in).

Pijush Samui is with the National Institute of Technology Patna, Patna 800005, India (e-mail: pijush@nitp.ac.in).

Mostafa Gandomi is with the College of Engineering, University of Tehran, Tehran 1416634793, Iran (e-mail: mostafa.gandomi@gmail.com).

Amir H. Gandomi is with the Faculty of Engineering and Information Systems, University of Technology Sydney, Ultimo, NSW 2007, Australia, and also with the University Research and Innovation Center (EKIK), Óbuda University, 1034 Budapest, Hungary (e-mail: gandomi@uts.edu.au).

This article has supplementary downloadable material available at https://doi.org/10.1109/TCSS.2023.3260217, provided by the authors.

Digital Object Identifier 10.1109/TCSS.2023.3260217

To analyze any hashtag, online content, or anybody's Twitter account, several analytical tools are available but are not easy to develop [7], [8]. Since meaningful information can be found from real-time tweets, a live extraction of tweets was one of the motivations of this work. Also, incorporating state-of-the-art Python packages, such as TextBlob library, and word cloud representation of significant terms have been successfully included in this research. Moreover, utilizing different deep learning models, such as long-short term memory (LSTM) and other advanced ML, for hate speech detection was another motivational factor for this study.

Overall, the following steps were performed.

1) To validate and authenticate our proposed method, we extracted real-time tweets using Tweepy API. Then, we cleaned and preprocessed tweets to create a data frame for subsequent analysis.
2) We analyzed the sentiment of the tweets using the TextBlob library of Python in order to find the subjectivity and polarity scores. Based on the calculated polarity scores, the tweets were identified as hate speech or the other of two as nonhate speech.
3) The TF-IDF of the Twitter data was calculated, and n-gram feature extraction was performed to determine the importance of a particular word in the dataset. To mark the words with the greatest contribution to the hateful sentiments, word cloud, also called tag cloud, was used to prepare a visual frame.
4) The proposed LSTM model achieved an accuracy of 97% for detecting hate speech and, thus, appears to be a good choice for evaluating textual data that is sequential by nature. Besides, the comparative analysis to Naïve Bayes (NB), logistic regression (LR), support vector machine (SVM), XGBoost (XGB), random forest (RF), K-nearest neighbor (k-NN), and artificial neural network (ANN) and bidirectional encoder representations from transformers (BERT) models further confirms the effectiveness of our method.

The rest of this article is structured as follows. Section II describes the related works, Section III introduces the proposed model, Section IV discusses the experimental results, Section V presents the discussion, and Section VI concludes the findings and suggests directions for future works.

II. Related Works

Many researchers have implemented different techniques based on dictionary, bag-of-word, n-gram, TF-IDF, and deep learning, for hate speech detection. For instance, Mohapatra et al. [9] investigated hate speech in a Facebook dataset containing 27 422 posts from different popular pages in mixed Odia–English language and classified the posts into three categories: "hate," "offensive," and "nonhate" speech using word-to-vector, TF-IDF, and n-gram techniques. The researchers applied SVM, NB, and RF ML models to check the features of "hate," "nonhate," and "offensive" language and found that their proposed RF model achieved the best accuracy of 75%. In another work, Bhora et al. [10] employed word-n-gram, lexicon, character-n-gram, and negation for feature extraction to evaluate an English–Hindi mixed social media text for hate speech detection and SVM as a predictive model, achieving an accuracy of 71.7%. Another work [11] focused on both topics and hate speech targets at a coarser level and discussed how that coarser level knowledge transfer happens across a variety of topics. This article also implemented a multitask approach of ML models. Sharma et al. [12] proposed a model that concentrates on examining hate speech in Hindi–English code-switched language and achieved 15% better performance compared to other existing models. In [13], the zero-shot style-transfer method is introduced, which uses an unsupervised technique for hate to nonhate conversion. García-Díaz et al. [14] analyzed which features are significant for selecting hate speech in Spanish, and a combination of linguistic features and transformers influenced their model. In [15], a deep learning-inspired multitask model that uses shared-features and task-specific features was developed, which achieved an encouraging $F1$ value in hate-speech detection. Moreover, Mossie and Wang [16] present a model to classify hate speech and nonhate speech in the Amharic language using Apache, an open-source tool. Table I displays some recent works on hate speech detection in social networks, including the models used, feature extraction strategy, and accuracy.

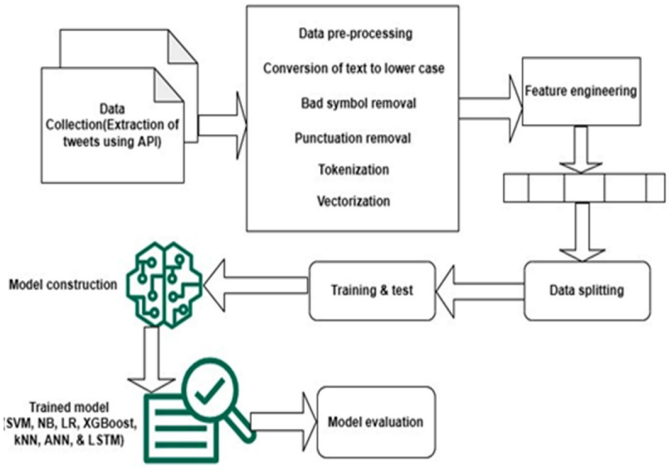


Fig. 1. Architecture of the hate speech detection method.

III. Proposed Model

The proposed model for detecting hate speech from real-time tweet data is shown in Fig. 1. As seen, data from a raw tweets stream are extracted using API and preprocessed, which involves removing punctuation, vectorization, and cleaning. Subsequently, using the TF-IDF technique [25], features are extracted from the collected tweets, and then, the n-gram method is used. After completing the feature extraction, the model is trained by the proposed ML LSTM algorithm. LSTM is an advanced structure of recurrent neural networks (RNNs) that is primarily helpful for sequence learning, such as NLP-related problems. Although RNNs are not that successful in the vanishing gradient problem for analyzing tweets in terms of

TABLE I
PREVIOUS WORKS ON HATE SPEECH DETECTION

Ref.	Study	Feature extraction	
		Model	Accuracy
Pereira-Kohatsu et al. [17]	Detecting hate speech from Twitter	Feature Extraction: Unigrams, POS tags, Tokenization, TF-IDF; ML Model: LSTM + MLP	82.8%
Agarwal and Chowdary [18]	Hate speech detection with a case study on COVID-19	Feature Extraction: Bag of words; ML Model: RNN and LSTM	88.99%
Sadiq et al. [19]	Aggression detection on Twitter with deep neural networks	Feature Extraction: Unigram, Bi-gram, TF-IDF; ML Model: CNN, LSTM, MLP, BiLSTM	91.3% (Maximum Accuracy)
Araque and Iglesias [20]	Hate speech detection online	Feature Extraction: SIMON, TF-IDF, n-gram;ML Model: LSTM, SVM	90.63%
Alammary [21]	Arabic question classification using TF-IDF	Feature Extraction: TF-IDF ML Model: NB, LR, SVM, KNN, Decision Tree	NB-78.72% LR-77.95% SVM-77.78% KNN-76.65% Decision Tree-59.71%
Bujar Raufi and Ildi Xhaferri [22]	Automatic hate Speech detection in mobile application	ML Model: ANN	94-95%
Roy et al. [23]	Hate speech detection using deep convolutional neural network	Feature Extraction: n-gram ML Model: SVM, LSTM	SVM-80% LSTM-97%
Malmasi and Zampieri [24]	Hate speech detection in social network	Feature Extraction: Word skip-gram, surface n-gram	78%

their sentiment, LSTM has shown to attain better accuracy than convolution networks and, thus, has found popularity among researchers. The hate speech detection problem is similar to a sequence learning problem, for which LSTM networks are a state-of-the-art method and can learn the sequence of the tweets [26]. The prime reason for using LSTM is that it can learn long-term dependencies unlike RNNs [27]. In this work, the results obtained by LSTM are compared with those of the SVM, NB, LR, RF, k-NN ANN, and BERT models.

A. Corpus Building of Tweets

We performed the following steps to successfully build a corpus of tweets consisting of real data streaming from Twitter. First, APIs were considered for the extraction of real-time Tweets.

Two different kinds of API are used for extracting tweets from Twitter: one for searching and the other for streaming tweets. In this work, REST API and queries were used to gather tweets from users. Based on the search of typical hashtags or terms, searching API returns the tweets that match the placed query, whereas streaming API helps to avail a continuous stream of real-time tweets from Twitter. Therefore, REST API tweepy was used for the search in this work. Twitter provides a consumer key and consumer secret key for the purpose of user authentication. On the other hand, API offers an access token and access token secret key for accessing the data. By using Oauth Handler and API authentication, the user gets authenticated, and then after using the user_time line function, any tweet can be fetched by providing the screen_name, count, lang, and tweet_mode.

B. Feature Extraction Model

For cleaning and feature extraction, we implemented the TF-IDF vectorizer, which applies the term referred to as t in document d. This TF-IDF vectorizer technique is used to find

the significance of the terms in the corpus [28]

$$tf(t,d) = \frac{C_{t,d}}{\sum_k C_{t,d}} \tag{1}$$

where $C_{t,d}$ is the total number of times t appears in document d; and $\sum_k C_{t,d}$ represents the total number of terms that are available in document d. The significance of t in document d depends on how many times t occurs in document d.

The calculation of IDF is as follows:

$$\text{idf}(t, D) = \log\left(\log \frac{D}{d_t}\right) + 1 \tag{2}$$

where D represents the total documents available in the corpus; and d_t denotes the total documents that consist of the term t. The greater the value of t, the more significantly it identifies document d in the corpus of tweets. Usually, TF-IDF is computed by multiplying (1) and (2)

$$\text{TF} - \text{IDF}(t, d, D) = tf(t, d) \times \text{idf}(t, D). \tag{3}$$

The algorithm for finding TF-IDF is given in the following [16].

Algorithm 1 Calculation of TF-IDF Score

```
Input: an array of sentences.
Output: TF-IDF score
Begin
    Import TF-IDF Vectorizer Module from sklearn
    Create an array containing all tweets
    Perform fit_transform on the array and store the
    output in a matrix

end
```

C. Word Cloud Visualization for Prominent Hateful Words

The representation of text data in terms of graphical means plays a vital role in understanding the overall significance of words contributing to hateful sentiments. In this work, the word cloud technique displays significant words related to hateful comments [29]. The word cloud, also known as tag cloud, usually consists of tags that are single words and shows the significance of those words using the increased font size. This form of visualization helps to understand the dominant terms or words in the tweets contributing heavily to hateful sentiments [30].

D. N-Gram

In the literature, it can be seen that text-to-feature vector conversion can be done in many ways. The most effective feature engineering methods for hate speech detection are character n-gram and the word or bag of word method. Moreover, n-gram is a contiguous sequence of n-items for a given sample of text or speech. Herein, the features extracted from the real-time tweets using Twitter API with the values of n_gram = 1 or 2 are passed in the function named generate_n-grams(), here, n refers to the number of words used in the probability sequence. An n-gram of one word is called a unigram (1-g), whereas a bigram (2-g) is the n-gram of two words. The choice of n determines the feature extraction process.

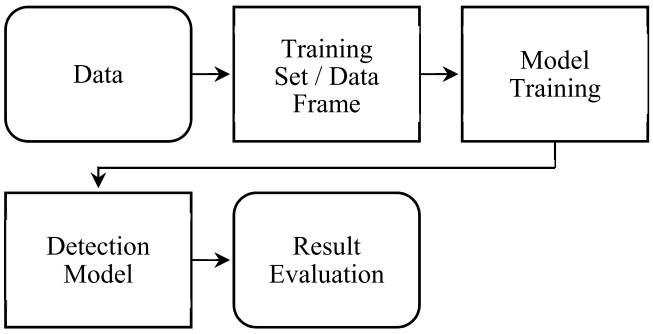


Fig. 2. Hate speech detection model using training and test data.

E. Resolution

The proposed ML model, LSTM, was applied to the training dataset collected from the real-time tweets using tweepy API [31]. The results of LSTM were then compared with other learning algorithms, such as SVM [32], NB [33], XGB [34], RF [35], k-NN [36], LR [37], ANN [31], and BERT [38], [39]. We used a supervised learning approach for detecting hate speech from the collected Twitter dataset. The models were applied to the training and test data, and the outcomes are shown in Fig. 2.

1) *Support Vector Machine:* SVM, originally developed by Cortes and Vapnik [32], was applied in this work to differentiate between hate speech and nonhate speech in tweets. The training vector D of SVM can be calculated using the following equation, where the n dimension of $x \in R^n$ signifies every instance of a tweet that is classified as either hateful or not hateful:

$$D = \left\{\left(x^1, y^1\right), \ldots\ldots, \left(x^n, y^n\right), \text{ where } \in R^n \right.$$
$$\left. y \in \{-1, +1\}.\right.$$

In SVM, the differentiator is a hyperplane, which is given as

$$f(x) = w \times x + k = 0 \tag{4}$$

where w is the weight to differentiate the hyperplane; and bias is represented as k$\in R$. Furthermore, the following equations are used to show whether the hyperplane should belong to the hate speech or nonhate speech class [20]:

$$w.x_i + k \geq 1, \quad (\text{for } y_i = 1) \tag{5}$$
$$w.x_i + k \leq -1, \quad (\text{for } y_i = -1). \tag{6}$$

After considering the slack variable ξ_i and adjusting the misclassification rate, the above equations can be combined as

$$y_i(w.x_i + k) \geq 1 - \xi_i. \tag{7}$$

2) *Bayes Classifier:* In the Bayes classifier [33], we assume that a pair (X, Y) takes a value in the range of $R^d X\{1, 2, 3, \ldots, n\}$, where X is considered as tuples and Y

is considered as a class label. The condition distribution of X provided by Y yields the value r as $(X|Y = r) \sim P_r$, where $r = 1 - n$. The operator $\sim$ signifies the distribution, and P_r refers to the probability of distribution. The equation for the Bayes classifier is given as

$$C^{\text{Bayes}}(x) = r \in 1 \text{ to nargmax } P(X = x). \tag{8}$$

The classifier C is a function $R^d \rightarrow \{1, 2, 3, \ldots, n\}$ that classifies X to the corresponding class. In this work, the class label is either hate speech or nonhate speech.

3) *XGBoost:* XGB is a distributed gradient boosting library that performs efficiently for text classification [34]. It applies the Newton–Raphson method, which uses the Taylor approximation of second order in the loss function. The log loss function for the evaluation of classification is given as

$$\frac{1}{N}\sum_{q=1}^{N} y_i \log \log\left(p(y_i)\right) + (1 - y_i) \cdot \log(1 - p(y_i)). \tag{9}$$

The working principle follows an ensemble learning inspired by decision trees that applies an optimized gradient boosted framework. The method tends to perform best for small to medium structured data but not unstructured data.

4) *Random Forest:* In RF, t number of trees are created using a discrimination function. The probability that class x belongs to class c (c = 1–2) is written as [35]

$$P\left(v_j(x)\right) = \frac{P(m, v_j(x)}{\sum_{l=1}^{n} P(m, v_j(x)}. \tag{10}$$

The discriminant function is given as

$$g_c(x) = \frac{1}{t}\sum_{j=1}^{t} P(c, v_j(x). \tag{11}$$

RF tends to grow as the n number of classification trees increases; subsequently, the input vector passing through each such tree in the forest also grows. The classification results are generated by each tree's "vote" for a particular test instance toward a specific class; then, the forest chooses the class with the most votes for that particular instance.

5) *Logistic Regression:* The LR model is written as [37]

$$\log\frac{p(x)}{1 - p(x)} = \beta_0 + x\beta \tag{12}$$

where the linear function $p(x)$ is unbounded; and the value of p is between 0 and 1.

6) *K-Nearest Neighbor*: Using the k-NN search [40], [41], [42], the closest k neighbors from the real-time queries, i.e., tweets, are chosen for training documents. The KNN algorithm believes that similar points exist in nearby proximity, meaning similar points are nearby. Using this strategy, the algorithm predicts any unknown tweets regardless of whether they are considered hate speech or nonhate speech. For k observations near a test observation x_0, the conditional probability that x_0 belongs to class j is determined by

$$P_r(Y = j \mid X = x_0) = \frac{1}{k}\sum_{i \in N_0} I(y_i = j) \tag{13}$$

where k is the number of neighbors to be considered by k-neighbors queries, which was 5 in this work.

7) Artificial Neural Network: The output function of ANN is expressed as [31]

$$f(x) = b + W^{\mathrm{T}}X \tag{14}$$

where $W = [w_1\ w_2\ w_3, \ldots, w_n]$, $X = [x_1\ x_2\ x_3, \ldots, x_n]$, b is the bias, and x is the input variable, i.e., the input to the neurons.

Back-propagation progressively calculates the error of each layer l; then, the $\delta^{(l)}$ term at a particular layer l can be referred by in the following equation:

$$\delta^{(l)} = \left(\left(W^{(l)}\right)^{\mathrm{T}}\delta^{(l+1)}\right) f'\left(z^{(l)}\right). \tag{15}$$

The gradient can be calculated by

$$\nabla_{W^{(l)}} J(W, b; x, y) = \delta^{(l+1)}\left(a^{(l)}\right)^{\mathrm{T}} \tag{16}$$

$$\nabla_{b^{(l)}} J(W, b; x, y) = \delta^{(l+1)} \tag{17}$$

where W and b are the parameters; and x and y represent the input tweets. The back-propagation algorithm optimizes the results, or reduces the error, through propagation and weight updates.

8) BERT: We also considered the BERT language model for the detection of hate speech as it has shown high performance in many NLP-related problems [38]. Table II and Figs. 8 and 9 present the results of BERT in-terms of $F1$ score, accuracy, precision, and recall. While BERT [38] was developed by Google Artificial Intelligence (AI) and heavily used to solve NLP-related problems, it is now used to generate a new representation of words based on words presented in the existing tweet corpus by considering both previous and following words in the tweet. This model can check the context of a word and its relation to other words in the sequence of text via an attention mechanism. Using three embedding layers, BERT creates the final model. Three distinct types of embeddings, which can be classified as token, segment, and position, are the input to the encoder layer of the proposed BERT. In terms of a formula, it can be represented as follows:

$$\begin{aligned} E = {} & \text{TokenEmbeddings} + \text{SegmentEmbeddings} \\ & + \text{PositionEmbeddings}. \end{aligned} \tag{18}$$

9) LSTM Model: Finally, we considered LSTM as a prime learning algorithm to efficiently differentiate hate speech from nonhate speech [19]. By incorporating three gates, including input–output, and forget gates, LSTM performs better in comparison with the other classifiers. The equations of LSTM are [26], [27]

$$e_t = \sigma_h(W_e x_t + P_e l_{t-1} + a_e) \tag{19}$$

$$j_t = \sigma_h\left(W_j x_t + P_j l_{t-1} + a_j\right) \tag{20}$$

$$p_t = \sigma_h\left(W_p x_t + P_p l_{t-1} + a_p\right) \tag{21}$$

$$\tilde{d}_t = \sigma_h(W_d x_t + P_d l_{t-1} + a_d) \tag{22}$$

$$d_t = e_t o e_{t-1} + j_t o \tilde{d}_t \tag{23}$$

$$L_t = p_t o \sigma_l(d_t) \tag{24}$$

TABLE II
IMAGE OF THE DATA FRAME AFTER SENTIMENT ANALYSIS

S. no	Tweets	Subjectivity	Polarity	Analysis
1	India has administered 1 million vaccine doses.	0.000000	0.000000	1
2	The world is so close to ending polio	0.200000	0.200000	1
3	Our funding is only part of the solutions.	0.757778	0.280000	1
4	Everyone deserves equitable access to healthcare.	0.333333	0.333333	1
5	We have done this before in 2017 the worked w…	0.250000	0.250000	1
6	I have been watching some of my favourite movies.	1.000000	0.500000	1
7	Getting an early start on my weekend reading.	0.341166	0.266667	1

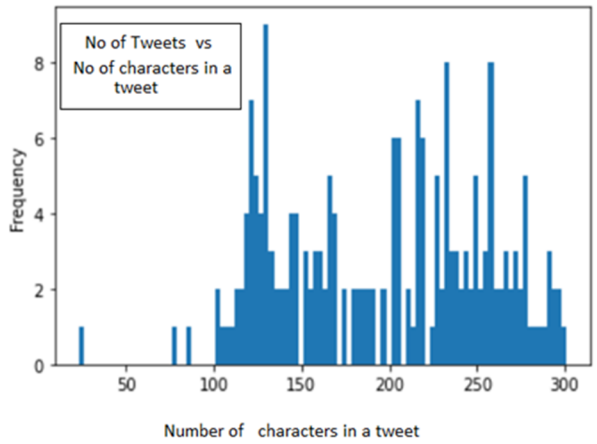


Fig. 3. X-axis represents a number of characters in a tweet, and Y-axis represents the number of tweets.

where t refers to time, $\vec{x}_t$ is the real number in m dimensional space, e_t is the activation vector refers to forget gate, j_t is the activation vector refers input gate, p_t refers to the vector of activation of the output gate, L_t = activation vector of hidden state, $\widetilde{d}_t$ refers to the activation vector of the input cell, d_t refers to the vector of cell state, W is the weight matrix, a refers to the bias vector, and operator o is known as the Hadamard product; and initially, $d_0 = 0$ and $l_0 = 0$.

The algorithms can be found in Appendix A of the Supplementary Materials, including Algorithm 2: SVM, Algorithm 3: NB, Algorithm 4: LR, Algorithm 5: XGB, Algorithm 6: RF, Algorithm 7: k-NN, Algorithm 8: BERT, and Algorithm 9: LSTM.

IV. EXPERIMENTAL RESULTS

To evaluate the detection accuracy of hate speech by the deep learning-inspired LSTM and other ML models, several metrics related to performance evaluation were calculated, such as precision, recall, accuracy, and $F1$-Score. To perform the analysis, features were initially extracted from the dataset of tweets in order to preprocess the data. Fig. 3 displays the length of the tweets in terms of characters on the x-axis and the frequency of each tweet on the y-axis. The plot function of the matplotlib library was used to prepare the histogram and determine the output. Fig. 4 presents the subjectivity and tweets' polarity using TextBlob of Python. Fig. 5 displays the tweets and their corresponding sentiment scores. A visualization of word cloud, also called tag cloud, is illustrated in Fig. 6, which reveals the most significant word or terms that contribute to the sentiments.

A. Dataset Description

To build the dataset for this study, we used tweepy API to fetch data from a real-time stream of Twitter, specifically 150–200 tweets. By searching "Bill Gates" as the subject, we obtained 200 raw tweets. Then, using the data frame functionality of pandas in Python, we created the data frame for further preprocessing and data cleaning.

B. Preprocessing

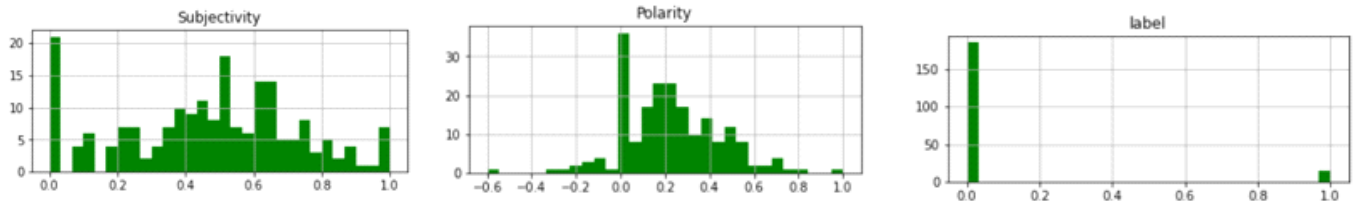
During the data processing and cleaning, some special characters and emojis were deleted from every tweet.

Fig. 4(a) and (b) presents the line charts of the subjectivity and polarity of the extracted tweets, respectively, based on the frequency of the considered tweet. As mentioned, the subjectivity and polarity were calculated using the module TextBlob and then graphed using the plot function of the module matplotlib. Fig. 4(c) displays the labels of the extracted tweets in terms of their frequency. In addition, comprehensive sentiment analysis was performed via the TextBlob module in Python. Fig. 5 shows each tweet's assigned subjectivity, polarity, and total analysis score. Specifically, a tweet with a polarity score less than 0, which indicates a "hated" post, is assigned a value of 0. A tweet with a polarity score greater than or equal to 0, which refers to a "nonhated" post, is assigned a value of 1.

C. Feature-Extraction Results

This work applied three methods for feature extraction: word cloud, n-gram, and TF-IDF. For word cloud, we cleaned the data and obtained a visualization frame using the "WordCloud" module.

Afterward, we performed the TF-IDF extraction with the help of the text feature extraction module of scikit-learn provided by Python. To create the target array corpus, we achieved different scores for the vectors. Table S.1 in Appendix B contains the obtained Id and scores after applying TF-IDF.



(a) (b) (c)

Fig. 4. (a) Subjectivity, (b) polarity, and (c) labels of tweets, which are represented on x-axis, in terms of frequency (y-axis).

Fig. 5. Word cloud visualization from the collected dataset.

Finally, we performed both unigram and bigram feature extraction on real-time tweets considering the length of the tweets, as shown in Figs. 6 and 7, respectively.

D. Model Evaluation Results

According to the literature, the evaluation metrics for the numerous ML and deep learning techniques are typically the same for all works, which include accuracy, precision, recall, and $F1$ score. For example, Sadiq et al. [19] obtained 92% accuracy applying the TF-IDF-based unigram and bigram approach. Alammary [21] achieved 78% accuracy in hate speech detection from their developed approach. In this work, we employed all considered models to train the learning algorithms to evaluate the extracted real-time data from a continuous Twitter stream as hate speech or nonhate speech. By feeding the training tweets into the models, sentiment analysis was conducted on the extracted real-time tweets based on accuracy area under curve (AUC) score, and $F1$ score, as shown in Figs. 8 and 9, and Table II, respectively. In addition, the following equations were used to calculate for accuracy, precision, recall, and $F1$ score:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \tag{25}$$

$$\text{Precision} = \frac{\text{TP}}{\text{FP} + \text{FP}} \tag{26}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{27}$$

$$F1 \text{ score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{28}$$

For each model, the complete classification reports of all classifiers on test data are provided in Table II.

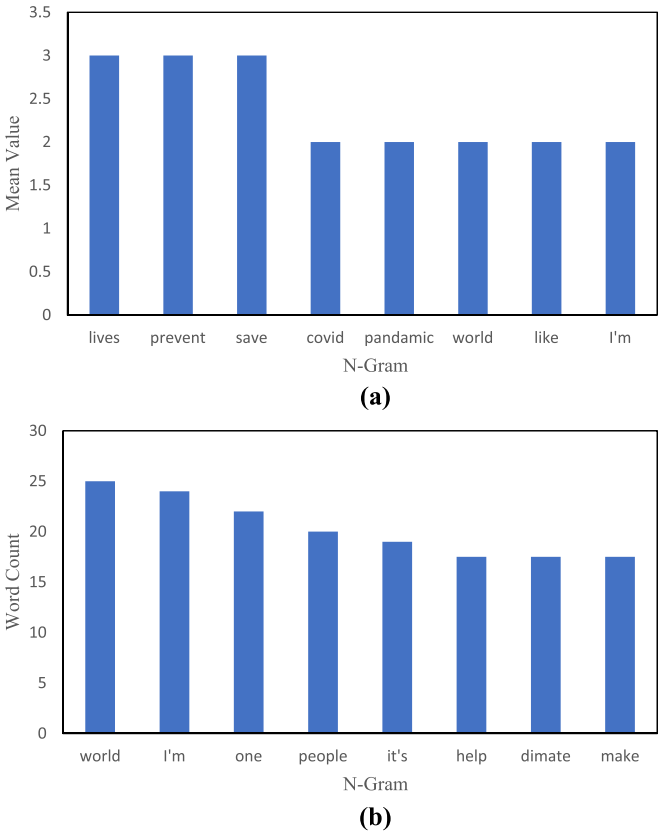


Fig. 6. Unigram feature extraction bar diagram. (a) N-Gram versus mean value. (b) N-Gram versus word count.

The training accuracy for ten Epochs of LSTM was also very satisfactory, as indicated in Fig. 10.

V. Discussion

This study reveals the importance of the detection of hate speech on social media, particularly on the Twitter platform, which has been investigated in several prior works. For instance, Mahapatra et al. [9] developed ML models for English–Odia mixed comments from many Facebook pages and processed them by filtering and cleaning out hateful speech. They used NB, SVM, and RF binary models with feature extractions via word2vec, n-gram, unigram, bigram, trigram TF-IDF, and TF-IDF+ combined n-grams. Among these, SVM achieved the highest average accuracy of 72.54%. For combined n-grams, NB achieved 74.77%, and word2vec features RF obtained 75.39% accuracy. Sadiq et al. [19] proposed CNN-LSTM and CNN-BiLSTM for aggression detection and obtained an overall accuracy of 92% but failed

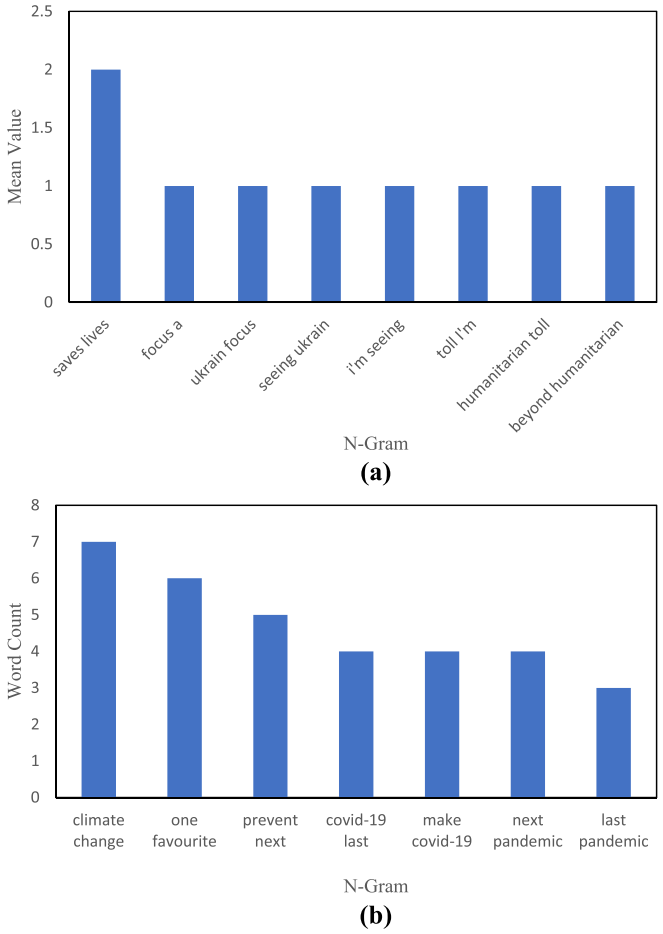


Fig. 7. Bigram feature extraction bar diagram. (a) N-Gram verses mean value. (b) N-Gram verses word count.

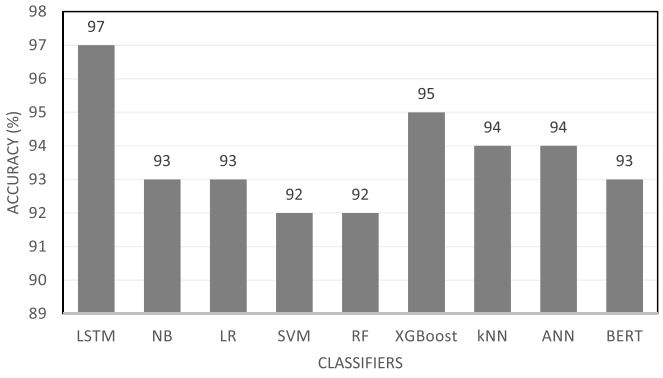


Fig. 8. Accuracy of all methods.

to perform a detailed analysis of the aggression detection of various features. In addition, Araque and Iglesias [20] reported the Affective Space and SenticNet-based ensemble method with the highest $F1$ scores of 97.77% and 98.21% achieved by LR and SVM, respectively. Oriola and Kotzé [42] introduced SVM, LR, RF, and gradient boosting algorithms for hate speech discovery based on South African Twitter data. The optimized SVM n-gram feature vector has obtained the highest TP rate of 0.894 [42]. Herein, we propose an LSTM-based method for hate speech detection, which achieved 97% accuracy. Therefore, the developed LSTM method performed

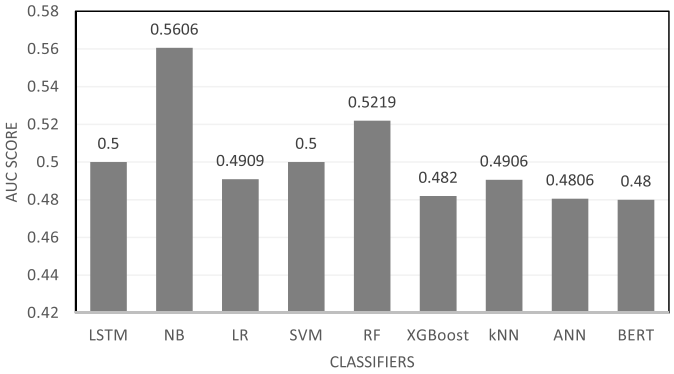


Fig. 9. AUC scores obtained by all methods.

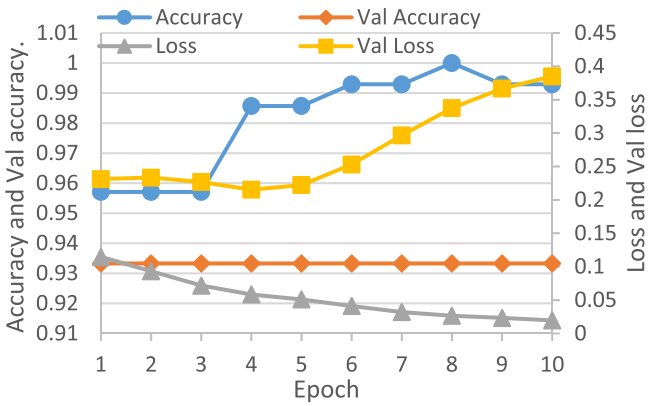


Fig. 10. Training accuracy for ten epochs of LSTM.

superior to other contemporary ML approaches, including NB, SVM, LR, XGB, RF, k-NN, ANN, and BERT that obtained accuracies of 92%, 93%, 93%, 95%, 92%, 94%, 94%, and 93%, respectively. Initially, we prepared the dataset by extracting real-time tweets using Twitter API called REST API, which proved to be more diverse compared to datasets produced in related works. Our approach employs TF-IDF vectorization for the feature representations, which is superior to the count vectorizer method. The resulting visual frame of hateful tweets was generated using the word cloud method and reveals that the proposed method could effectively distinguish between hateful and nonhateful tweets.

VI. Conclusion and Future Work

In this work, we propose an LSTM-based approach to detect hateful speech from a real-time tweet stream on Twitter using specific hashtags. The TFID mechanism applies vector representation that attaches weights to the meaningful terms associated with hateful or nonhateful tweets. The word cloud technique depicts the words responsible for hateful sentiments on a visual frame. The efficiency of our proposed approach to detect hate speech was compared to other reported methods, including NB, SVM, LR, XGB, RF, k-NN, ANN, and BERT. Results show that LSTM obtained the highest accuracy of 97%. In future studies, we aim to extend our approach for languages other than English, such as Bengali, Oria, Marathi, Tamil, and Telugu. Also, we plan to develop a hybrid model

composed of advanced ML methods, evolutionary computation, and swarm intelligence to enhance the accuracy of hate speech detection for multilingual tweets.

REFERENCES

[1] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter," in *Proc. NAACL Student Res. Workshop*, 2016, pp. 88–93.

[2] U. Ahmed and J. C.-W. Lin, "Deep explainable hate speech active learning on social-media data," *IEEE Trans. Computat. Social Syst.*, early access, Apr. 20, 2022, doi: 10.1109/TCSS.2022.3165136.

[3] D. S. Naz and M. O. Shafiq, "Hate speeches on Twitter and Facebook in South Asia: A case study of malala yousufzai," *Pakistan J. Int. Affairs*, vol. 4, no. 1, pp. 328–351, Mar. 2021.

[4] S. Ghosh, A. Ekbal, P. Bhattacharyya, T. Saha, A. Kumar, and S. Srivastava, "SEHC: A benchmark setup to identify online hate speech in English," *IEEE Trans. Computat. Social Syst.*, early access, Mar. 21, 2022, doi: 10.1109/TCSS.2022.3157474.

[5] S. Modha, P. Majumder, T. Mandl, and C. Mandalia, "Detecting and visualizing hate speech in social media: A cyber watchdog for surveillance," *Expert Syst. Appl.*, vol. 161, Dec. 2020, Art. no. 113725.

[6] H. Liu, P. Burnap, W. Alorainy, and M. L. Williams, "A fuzzy approach to text classification with two-stage training for ambiguous instances," *IEEE Trans. Computat. Social Syst.*, vol. 6, no. 2, pp. 227–240, Apr. 2019.

[7] K. Maity, S. Saha, and P. Bhattacharyya, "Emoji, sentiment and emotion aided cyberbullying detection in Hinglish," *IEEE Trans. Computat. Social Syst.*, early access, Jul. 7, 2022, doi: 10.1109/TCSS.2022.3183046.

[8] S. Khan et al., "HCovBi-caps: Hate speech detection using convolutional and Bi-directional gated recurrent unit with Capsule network," *IEEE Access*, vol. 10, pp. 7881–7894, 2022.

[9] S. K. Mohapatra, S. Prasad, D. K. Bebarta, T. K. Das, K. Srinivasan, and Y.-C. Hu, "Automatic hate speech detection in English-Odia code mixed social media data using machine learning techniques," *Appl. Sci.*, vol. 11, no. 18, p. 8575, Sep. 2021.

[10] A. Bohra, D. Vijay, V. Singh, S. S. Akhtar, and M. Shrivastava, "A dataset of Hindi-English code-mixed social media text for hate speech detection," in *Proc. 2nd Workshop Comput. Modeling People's Opinions, Personality, Emotions Social Media*, 2018, pp. 36–41.

[11] P. Chiril, E. W. Pamungkas, F. Benamara, V. Moriceau, and V. Patti, "Emotionally informed hate speech detection: A multi-target perspective," *Cognit. Comput.*, vol. 14, no. 1, pp. 322–352, Jan. 2022.

[12] A. Sharma, A. Kabra, and M. Jain, "Ceasing hate with MOH: Hate speech detection in Hindi–English code-switched language," *Inf. Process. Manag.*, vol. 59, no. 1, 2022, Art. no. 102760.

[13] Z. Ahmad, V. S. Sujeeth, and A. Ekbal, "Zero-shot hate to non-hate text conversion using lexical constraints," *IEEE Trans. Computat. Social Syst.*, early access, May 24, 2022, doi: 10.1109/TCSS.2022.3175259.

[14] J. A. García-Díaz, S. M. Jiménez-Zafra, M. A. García-Cumbreras, and R. Valencia-García, "Evaluating feature combination strategies for hate-speech detection in Spanish using linguistic features and transformers," *Complex Intell. Syst.*, pp. 1–22.

[15] P. Kapil and A. Ekbal, "A deep neural network based multi-task learning approach to hate speech detection," *Knowl.-Based Syst.*, vol. 210, Dec. 2020, Art. no. 106458.

[16] Z. Mossie and J.-H. Wang, "Social network hate speech detection for Amharic language," in *Computer Science & Information Technology*. Dubai, UAE, 2018, pp. 41–55.

[17] J. C. Pereira-Kohatsu, L. Quijano-Sánchez, F. Liberatore, and M. Camacho-Collados, "Detecting and monitoring hate speech in Twitter," *Sensors*, vol. 19, no. 21, p. 4654, 2019.

[18] S. Agarwal and C. R. Chowdary, "Combating hate speech using an adaptive ensemble learning model with a case study on COVID-19," *Expert Syst. Appl.*, vol. 185, Dec. 2021, Art. no. 115632.

[19] S. Sadiq, A. Mehmood, S. Ullah, M. Ahmad, G. S. Choi, and B.-W. On, "Aggression detection through deep neural model on Twitter," *Future Gener. Comput. Syst.*, vol. 114, pp. 120–129, Jan. 2021.

[20] O. Araque and C. A. Iglesias, "An ensemble method for radicalization and hate speech detection online empowered by sentic computing," *Cogn. Comput.*, vol. 14, no. 1, pp. 48–61, 2022.

[21] A. S. Alammary, "Arabic questions classification using modified TF-IDF," *IEEE Access*, vol. 9, pp. 95109–95122, 2021.

[22] B. Raufi and I. Xhaferri, "Application of machine learning techniques for hate speech detection in mobile applications," in *Proc. Int. Conf. Inf. Technol. (InfoTech)*, Sep. 2018, pp. 1–4.

[23] P. K. Roy, A. K. Tripathy, T. K. Das, and X.-Z. Gao, "A framework for hate speech detection using deep convolutional neural network," *IEEE Access*, vol. 8, pp. 204951–204962, 2020.

[24] S. Malmasi and M. Zampieri, "Detecting hate speech in social media," 2017, *arXiv:1712.06427*.

[25] L. Silva, M. Mondal, D. Correa, F. Benevenuto, and I. Weber, "Analyzing the targets of hate in online social media," in *Proc. Int. AAAI Conf. Web Social Media*, 2016, pp. 687–690.

[26] C. Olah. *Understanding LSTM Networks*. Accessed: Jun. 26, 2018. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[27] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. D, Nonlinear Phenomena*, vol. 404, Mar. 2020, Art. no. 132306.

[28] J. Ramos, "Using TF-IDF to determine word relevance in document queries," in *Proc. 1st Instructional Conf. Mach. Learn.*, 2003, vol. 242, no. 1, pp. 29–48.

[29] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl, "Word cloud explorer: Text analytics based on word clouds," in *Proc. 47th Hawaii Int. Conf. Syst. Sci.*, Jan. 2014, pp. 1833–1842.

[30] W.-C. Chang and M.-S. Chung, "Automatic applying Bloom's taxonomy to classify and analysis the cognition level of English question items," in *Proc. Joint Conf. Pervasive Comput. (JCPC)*, Dec. 2009, pp. 727–734.

[31] A. Bisht, A. Singh, H. S. Bhadauria, and J. Virmani, "Detection of hate speech and offensive language in Twitter data using LSTM model," in *Recent Trends in Image and Signal Processing in Computer Vision*. Singapore: Springer, 2020, pp. 243–264.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[33] A. Bhoi and R. Chandra Balabantaray, "Hate tweet extraction from social media text using autoencoder wrapped multinomial Naive Bayes classifier," in *Data Engineering and Intelligent Computing*. Singapore: Springer, 2021, pp. 619–628.

[34] R. Mitchell and E. Frank, "Accelerating the XGBoost algorithm using GPU computing," *PeerJ Comput. Sci.*, vol. 3, p. e127, Jan. 2017.

[35] T. K. Ho, "Random decision forests," in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, 1995, vol. 1, pp. 278–282.

[36] A. Briliani, B. Irawan, and C. Setianingsih, "Hate speech detection in Indonesian language on Instagram comment section using K-nearest neighbor classification method," in *Proc. IEEE Int. Conf. Internet Things Intell. Syst. (IoTaIS)*, Nov. 2019, pp. 98–104.

[37] R. E. Wright, "Logistic regression," in *Reading and Understanding Multivariate Statistics*, L. G. Grimm and P. R. Yarnold, Eds. USA: American Psychological Association, 1995, pp. 217–244.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[39] M. Singh, A. K. Jakhar, and S. Pandey, "Sentiment analysis on the impact of coronavirus in social life using the BERT model," *Social Netw. Anal. Mining*, vol. 11, no. 1, pp. 1–11, Dec. 2021.

[40] E. Fix and J. L. Hodges Jr., "Discriminatory analysis. Nonparametric discrimination: Consistency properties," *Int. Stat. Rev.*, vol. 57, no. 3, pp. 238–247, 1989.

[41] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[42] O. Oriola and E. Kotze, "Evaluating machine learning techniques for detecting offensive and hate speech in South African tweets," *IEEE Access*, vol. 8, pp. 21496–21509, 2020.

Sanjiban Sekhar Roy (Member, IEEE) received the Ph.D. degree from the Vellore Institute of Technology, Vellore, India, in 2016.

He was an Associate Researcher with Ton Duc Thang University, Ho Chi Minh City, Vietnam, from 2019 to 2020. He is currently a Professor with the School of Computer Science and Engineering, Vellore Institute of Technology. He has edited handful of special issues for journals, published numerous articles in SCI high impact journals, and many books with reputed international publishers. His current research interests include deep learning and advanced machine learning.

Dr. Roy was a recipient of the "Diploma of Excellence" Award for academic research from the Ministry of National Education, Romania.

Akash Roy is currently pursuing the B.Tech. degree in computer science and engineering with the Vellore Institute of Technology, Vellore, India.

His research interests include web development and machine learning.

Pijush Samui received the Ph.D. degree in geotechnical earthquake engineering from the Indian Institute of Science, Bengaluru, India, in 2008.

He is currently a Professor with the Department of Civil Engineering, NIT Patna, Patna, India.

Dr. Samui is an Editorial Member to many highly reputed journals, has edited many books with reputed publishers and published many articles to high SCI impact journals.

Mostafa Gandomi received the Ph.D. degree in engineering from the University of Tehran, Tehran, Iran, in 2022.

He has authored or coauthored several articles in leading international journals and conferences and has received more than 700 citations for his research contributions. His research interests include computational intelligence, optimization, and machine learning, with a particular focus on their applications in engineering problems.

Amir H. Gandomi (Senior Member, IEEE) was an Assistant Professor with the Stevens Institute of Technology, Hoboken, NJ, USA, and a Distinguished Research Fellow with the BEACON Center, Michigan State University, East Lansing, MI, USA. He is currently a Professor in data science with the Faculty of Engineering and Information Technology, University of Technology Sydney (UTS), Ultimo, NSW, Australia. He is also a Distinguished Professor with Óbuda University, Budapest, Hungary. He has authored or coauthored over 350 journal articles and 14 books, which collectively have been cited 40 000+ times (H-index = 89). His research interests include global optimization and (big) data analytics using machine learning and evolutionary computations in particular.

Prof. Gandomi received multiple prestigious awards for his research excellence and impact, such as the 2022 Walter L. Huber Prize, the Highest-Level Mid-Career Research Award in all areas of civil engineering and the Discovery Early Career Researcher Award (DECRA) from Australian Research Council (ARC). He has been named as one of the most influential scientific minds and received the Highly Cited Researcher Award (top 1% publications and 0.1% researchers) from the Web of Science for six consecutive years, from 2017 to 2022. In the recent most impactful researcher list, done by Stanford University, Stanford, CA, USA, and released by Elsevier, he is ranked as the top 1000 researchers (top 0.01%) and top 50 researchers in Artificial Intelligence (AI) and image processing subfield in 2021! He also ranked 17th in Genetic Programming (GP) bibliography among more than 15 000 researchers. He has served as an Associate Editor, an Editor, and a Guest editor in several prestigious journals, such as an Associate Editor of *IEEE Network Magazine* and IEEE INTERNET OF THINGS JOURNAL(IoTJ). He is active in delivering keynotes and invited talks.