# Identification of Hate Speech on Social Media using LSTM

Abhijeet Verma[1], Anshuman Singh[1], Anand Bihari[2,*], Sudhakar Tripathi[1],
Sanjay Agrawal[3], Shivendra Kumar Pandey[1], and Sharad Verma[1]

ARTICLE INFO

*Article history:*
Received: 28 July 2022
Revised: 26 September 2022
Accepted: 15 October 2022

*Keywords*:
Hate speech
Social media
CNN
LSTM
Twitter posts

ABSTRACT

With the advent of micro-blogging services like Twitter, Facebook, and Tumblr, people's communication has become more indirect and reliable. People from different lifestyles and cultures interact with each other. They express their thoughts on many topics every day. This led to interpersonal conflict. As a result, the use of hate is increasing. Malicious language has become a serious problem. Manually searching for such content on these websites can be difficult. In recent years, detecting malicious language in content that is available online has become increasingly important in applications such as controversial event detection and analyzing sentiment. Classifying the text of content that is generated online can be a daunting task because of the complexity of human spoken language and online users' microblogging, which has resulted in many informal and error-prone creations. This article presents a system to classify tweets into two groups (non-hate and hate). Also, it shows the implementation of the hate speech detection problem for text data written in English. Several proposed studies on similar questions use classical machine learning approaches that rely heavily on feature engineering. Moving from one domain to another means redesigning the design of the function. This preliminary study proposes another method based on a deep learning approach that does not require feature engineering and can be applied to various contexts. Using a dataset derived from Twitter posts with more than 27,000 tweets, the proposed method provided better results, with an F1 score of 90% or higher.

## 1. INTRODUCTION

Hate speech spawning and spreading—the main cause of hate crime—is easy in the cyber virtual world that can't be reached by the traditional legal system because there is a problem with anonymity." Any statement that disparages a person or a group on the basis of specified attributes (referred to as hate types or hate classes) such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics" is how hate speech is defined". According to a Pew Research Center research survey of 4,248 US adults conducted in 2017, 41% of online users have experienced online hate and foul language [2]. These types of occurrences occur when unsuitable photographs, offensive words, or messages are shared. Identifying and eliminating such content has been a challenge that social media academics have been working on for years. The automation of this activity has rapidly increased in latest years, in tandem influential and interested in the identification of hate speech online on social media [2]. Although hate speech is one of the aforementioned classifications, we are focusing on filtering out tweets that contain hostile content based on racism from tweets in this study. Because of the vagueness of the work, it is fairly difficult. In natural language, a single word can have numerous meanings in a variety of situations. For instance, the word "fan". Occasionally referred to as an electrical appliance," this term refers to someone who is a celebrity's fan. Similarly, some words share the same situation. Variation in spelling is a different situation in which a user, knowingly or accidentally, alters a character or multiple characters in a word [1]. If a simple term is used, the system does not recognize it as hate speech. The basic approach is employed. In terms of previous work, most of it focuses on manual feature extraction or representation learning, followed by machine learning classifiers. On the other hand, deep learning models have shown significant improvements in task accuracy when it comes to text. As a result, we want to integrate deep learning models with different M.L (machine learning) models to classify tweets. Collaborative learning is very popular these days, and it produces better results than previous methods. In this proposed strategy that

[1]*Department of Information Technology, Rajkiya Engineering College Ambedkarnagar, U. P., India.*
[2]*Department of Computational Intelligence, School of Computer Sciene and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India.*
[3]*Department of Electrical Engineering, Rajkiya Engineering College Ambedkarnagar, U. P., India.*
**Corresponding author:* Anand Bihari, Email: anand.bihari@vit.ac.in, csanandk@gmail.com.

we are using, we used a machine learning solution to overcome the natural language resistance [1]. We're using Kaggle's dataset of more than 27,000 publicly available tweets. The main solution of this paper that we are providing: (a) find training sets with annotations on Twitter and comments that are basic facts, (b) developed several classification models and (c) evaluating the performance of models on each platform separately, and By comparing the results of different approaches, as well as those reported in, our strategy improves accuracy by 5%.

## 2. RELATED WORK

Many scholars have worked on finding hate speech in the English language. These studies used machine learning algorithms and data sources such as Twitter, Facebook, YouTube comment sections, and other online forums [7]. The study's primary aim was to focus on hate speech directed at vulnerable people in society. Most of them were minorities, teenagers, dark colour people, etc. Hate speech is escalating day by day as social media is emerging. Identifying hate speech on social media such as Discord, Youtube, Reddit, Twitter, and Facebook is challenging for researchers [8]. Researchers use various approaches to classifying hate content: SVM, Decision Tree, Naive Bayes, Logistic Regression, LSTM, and CNN [3]-[6]. Researcher Hema Krishnan [12] worked on a hate speech detection system on Twitter using the Naive Bayes Classifier based on sentimental data. Naufal Riza [13] implemented Naive Bayes to classify the Indonesian language's hate speech. Kelvin Kiema and George Okeyo used the Naive Bayes classifier to conduct Twitter's Hate Speech[14]. They produced a better performance by producing an accuracy of 67.47%, a recall of 62%, and a precision of 58%. Along with Naive Bayes, Some researchers used another classifier, such as Neural Network. Sindhu et al. [6] applied the random forest classifier to solve word prediction. And she predicted dangerous words using the same classifier and obtained a 76.42% accuracy. She got an accuracy of 72.4% by using the Naive Bayes algorithm and 72% by using SVM. Researcher Davidson [7] also uses different classifiers like Porter-Steemer and logistic regression, Naive Bayes', random forest, and decision tree. He has developed a linear SVM and logistic regression model to produce a high accuracy level after evaluation by 5-fold cross-validation[7]. Serven Malmasi [15] uses the same dataset that Davidson used. E. Chandra Sekaran, A. Srinivasan, E. Gilbert, and

M. Samory [16] uses a new concept that is a bag of community" to identify abusive content from the online community. They also use different algorithms such as Naive Bayes linear SVC and Logistic regression for the classification. Above, all the algorithms based on naive Bayes performed best. Burnap and William [17] mainly focus on racism and identify hate speech on Twitter. Wasen and Hovy [18] used logistic regression and the n-gram function to classify racism and sexism to classify tweets. Davidson [6] and other researchers found that racism and homosexuality are malicious and offensive. They used many classifiers, such as Naive Bayes, decision trees, random forests, logistic regression, and support vector machines, to classify into different categories. Badjatiya et al. [19] used deep learning models like convolutional neural networks and long short-term memory to detect hate speech content. They used several embeddings such as random, GloVe, and fastText embedding and found the best combination is LSTM[5], Random Embedding, and Gradient Boosted Decision Tree to classify disliked tweets.

In our research, we used different approaches that researchers previously used. We categorized tweets into hate and non-hate. To promote objectivity, we adopted their method of annotating the dataset, which included annotators from various backgrounds. We compared the performances of six classifiers: RF (Random Forest), NB (Naive Bayes), CNN, LR (Logistic regression), SVM, and CNN+LSTM.

## 3. METHODOLOGY

In our research study, we constructed a text classifier machine learning model that may be utilized to detect hate content in a text corpus. We suggested an automated text classification strategy to address the shortcomings of crude hate speech classification. To choose the features of the text corpus and classify them into hate and non-hate categories, we are utilizing feature extraction, which is a word embedding technique. We use many sorts of machine learning models to determine the optimal model for our data set for this goal.

### *3.1. Data set Description*

There are approximately 27,000 rows of tweets in the dataset, which contains three columns, namely: id, label, and tweets. We have extracted the data from Kaggle (https://www.kaggle.com/code/arkhoshghalb/detecting-hate-tweets/data). Hate is classified as a '1', while non-hate is classified as a '0.' In the data collection, there were 27,030 tweets shown in table 1.

**Table 1. Dataset Description**

| Class Label | Number of tweets | Maximum Length of Tweet | Number of tokens with stop-words |
|---|---|---|---|
| Hate | 14,741 | 70 words | 6,382 |
| Non-Hate | 12,289 | 70 words | 5,261 |

For this study, we assembled a dataset of publicly released hate speech tweets. This dataset has two types of tweets: non-hate speech and hate speech. The training dataset has 27,030 tweets. Nearly half of the tweets are hate speech, with the rest being non-hate. We're training our model with an online dataset because it has sufficient samples and can be used with whatever model we're working on.

### 3.2. Data Preprocessing

Data preprocessing increases classification results, so it's critical to perform it if we want the best results. As a result, we applied different preprocessing approaches to remove noise and non-informative characteristics from Twitter's tweets in our dataset. Our tweet dataset was cleansed of the following elements:

1. URLs are being eliminated.
2. Special characters are being deleted.
3. Corrections to the spelling.
4. Getting rid of the stop words.

Tokenization and stemming were also done on preprocessed tweets. Tokenization breaks down each tweet into tokens or words, which the porter stemmer then breaks down into root forms, such as furious to insult.

### 3.3 Data Splitting

After doing preprocessing, we partitioned the preprocessed data in an 80-20 ratio. We're dividing the data into two parts so that we can train our model on 80% of it and then test it on the other 20% to see how accurate it is in producing the result. Using the training data, the model that we used to classify tweets is trained to learn classification rules. The test data is also used to test the classification model.

### 3.4 Classifications

We used supervised learning approaches to find hate speech in the English language. We evaluate the performance of all the methods that we are using, i.e., six methods by using the collected dataset: Logistic Regression, Linear SVM [10], RF Classifier, NB Classifier, CNN [4], and CNN+LSTM [6].

### 3.5 Model Description

In this study, we use a variety of ML models, such as CNN, NB, SVM, RF, and LR [21-22]. We measured the accuracy and F1 Score of each model we employed to find one that was the best classifier for our dataset. For classification, we are mainly using CNN+LSTM [6] as our main model.

#### 3.5.1 CNN Model

To categorize hateful words, we suggest using CNN-based models. The most significant constituent is the convolutional layers, which extract features by sliding the filter across the input. Convolution is followed by max pooling to capture the output's most important properties.

To illustrate this idea, consider the example of a comment: "In this pic you are looking ugly and disgusting, you deserve to die". Each of the words, such as 'looking', 'ugly', 'disgusting', 'deserve', and 'die' alone, are not always indicative of hate. But combinations such as "looking ugly," "ugly and disgusting," and "deserve to die" can be more indicative features. And for explicit images, we used image processing to check what's hateful in the image.

Convolutional neural networks (CNNs)[4] have three layer types: Pooling, Convolution and fully connected layers. Pooling and Convolution layers extract information, while a fully connected layer integrates those features into the final result, such as classification [4].

**Convolutional Layer**: This is the Primary layer of the convolutional network. This extracts features by sliding the filter across the input. Its objective is to find a certain collection of features in the pictures supplied as input [4].

**Pooling Layer**: The size of the feature maps is lowered using pooling layers. As a result, it lessens the quantity of network computation and the number of parameters that must be learned.[4]

**Fully connected layer**: The output of the final Pooling Layer serves as the input for the so-called Fully Connected Layer at the end of a CNN. Classification is carried out by fully connected layers using the features extracted by the preceding layers [4].

#### 3.5.2. Long Short Term Memory

Several academics developed various techniques for detecting online hate speech in social community forums, all of which were successful. To detect online hate, we need a significant volume of labelled data that has been divided into hate and non-hate categories. Preprocessing the dataset and using several machine learning models is the typical technique.

LSTM has three gates: input gate, output gate and forgets gate. The task of the input gate is adding information to the state of the cell, and forget gate determines what information must be remembered and what can be forgotten. The output gate chooses what output to produce based on the current internal cell stat [5].

## 4. PROPOSED APPROACH

This research provides the most significant contribution by offering feature reduction strategies in conjunction with a combination of deep learning models that include two neural network layers, CNN [4] and LSTM [5]. The suggested technique outperforms traditional deep learning models in terms of prediction accuracy. The CNN-LSTM architecture is used to process the data. The model's first layer, the embedding layer, accepts tweets and converts each word into a 100-pixel vector. Because a tweet can only be 90 characters long, this layer will generate a 90*100 matrix. The output matrix will contain the weights obtained from the matrix multiplication, resulting in a vector for each word. The CNN layer uses these embedding vectors to retrieve context information. The CNN layer's output is passed into an LSTM [5] layer, then passed on to a fully linked dense layer; this produces a single output as the final result. This model is trained and tested on batches of size 128[6].

### *4.1 Input and Convolution Layer*

The first step is to use the Keras tokenizer to tokenize the text of the tweets. The embedding layer of Keras converts tokens into word vectors by the use of word2vec word embeddings. The word embedding layer's output, a vector of words, is sent to the convolution layer. The convolutional layers extract a set of semantic or structural properties derived from a set of input arrays. Each word vector is received by CNN[4]; neurons n Filters of varying sizes can be used to generate a variety of features. There are several filters f, each with a different kernel size. (c*e) is the outcome of applying the function c to each word embedding e. Because the kernel size in our work is 5, a filter of size 64 will produce 5-word combinations. Tables 2 and 3 illustrate the input and output shapes of CNN+LSTM with various settings. A detailed description of the defined model is given in Table 2 and Table 3.

**Table 2. Layer Structure of the Defined Model**

| **CNN-LSTM** |
|---|
| Dropout(0.2)<br>Conv(5x5,@64)<br>Dense(3 neuron)<br>Max Pooling (4x4)<br>Activation=’ReLu’<br>LSTM (150 neurons) |

**Table 3: Model Parameter Structure**

| **Layer(type)** | **Output Shape** | **Param #** |
|---|---|---|
| embedding_15(Embedding) | (None,90,100) | 2709100 |
| conv1d_15(Conv1D) | (None, 86, 64) | 32064 |
| max_polling1d_11(MaxPolling1D) | (None, 21, 64) | 0 |
| dropout_14 (Droupot) | (None, 21, 64) | 0 |
| lstm_11 (LSTM) | (None, 150) | 99000 |
| dense_15 (Dense) | (None,3) | 153 |

### *4.2 ActivationFunction, Max-Pooling, and Dropout*

The ReLu activation function affects every CNN neuron's output [4]. It will also assist in lowering the size of the input function to the next level, so overfitting can be avoided. In the first phase of the ReLu activation function, the negative values are converted to zero, which indicates network non-linearity. As there is no negative value in the CNN layer. As a result, the CNN layer's input shape and output shape are the same. After the ReLu function's completion, each neuron's value is sent into a 1-D max-pooling layer [6]. In the convolution procedure known as maximum pooling, the kernel pulls the highest value possible from the convolution area [11]. Since the Twitter dataset values are at least 0.2, the output dropout level is equivalent to the given input.

**ReLu**: The Rectified Linear Unit is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value *x* it returns that value back. So it can be written as $\boldsymbol{f(x)=max(0,x)}$.

### *4.3 LSTM*

LSTM is the next layer, with 100 units. For our data, We must create a sequence structure that looks like a long chain[6]. while keeping track of prior inputs. Because it has three gates: input gate, output gate and forget gate, the LSTM is the ideal option for this task. These gates determine which information is valuable for categorization and what information can be forgotten based on the dropout value. The cell memory block, which is essential for prediction, saves previous input. Each and Every unit in the dense layer is connected to all 100 units in the LSTM layer's output [5, 9], [20].

### *4.4 Dense*

The dense layer, which is the final layer of the proposed model, is highly connected with a single output[6]. The batch size is 128 and the iteration number is 10.

## 5. CLASSIFIER EVALUATION

We must ensure the integrity of our outcomes. As a result, we conducted a quantitative analysis of the frequency of data collection, data release mode, and data types. The obtained data is meticulously annotated before being entered into the constructed model for prediction. So we used an evaluation measure to examine the data and outcomes to see how well the model worked, how skewed the results were, and how generalizable our findings were. We employed the evaluation metric we devised throughout the trial.

The idea of positive and negative affects all of the numbers we assess for accuracy, precision, recall, and F-scores. Positive speech is defined as hate speech, whereas negative speech is defined as speech that is not hateful. Figure 8 shows the definitions of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN)[7].

*1) Recall*: Percentage of actual consent is expected to be positive.

$$Recall=\frac{TP}{(TP+FN)} \tag{1}$$

*2) Precision*: It tests how many outcomes actually are positive outcomes for all well-predicted outcomes.

$$Precison = \frac{TP}{(TP+FP)} \tag{2}$$

*3) Accuracy*: It is calculated by dividing the total number of relevant predictions by all the predictions [7].

$$Accuracy = \frac{(\mathrm{TP + TN})}{(\mathrm{TP + TN + FP + FN})} \tag{3}$$

*4) F1 Score*: It is the precision and recall harmonic mean. It accounts for both false positives and false negatives[7].

$$F1\ Score = \frac{2*(\mathrm{Precision}*\mathrm{Recall})}{(\mathrm{Recall}+\ \mathrm{Precision})} \tag{4}$$

## 6. RESULTS

This section describes the functionality of the six classifier models we have used in this research. These models used different feature sets to regard the evaluation matrix, i.e., precision accuracy, F1 score and recall.

In our research, we performed multiple classifiers and selected the top three classifiers among all classifiers. The selected classifiers are Random Forest, CNN, and CNN+LSTM[6]. Table 6 shows the outcomes of the experiments we conducted using various methodologies.

**Table 4. Hyperparameters used in the proposed approach**

| Parameter Description | Values |
|---|---|
| Maximum length of tweet | 90 |
| Size of filters | 5 |
| Number of filters | 64 |
| Pooling size | 4 |
| Activation function | Relu |
| Number of convolution layers | 3 |
| Learning rate | 0.01 |
| Batch size | 128 |
| Loss function | binary_crossentropy |
| Optimizer | Adam |
| Epoch | 10 |

The approaches we are using as our main models produce good outcomes. "CNN+ Long Short Term Memory" is the best of all approaches on our given dataset. The performance of various machine learning methods, including Random Forest, Naive Bayes, Logistic Regression and Random Forests was compared to that of the proposed CNN+LSTM[6] model. The same data set and split training and testing techniques were applied. In Table 4, the hyper-parameters of our main model, CNN+LSTM [6], are detailed. The proposed model used a total of 64 filters to find the data stream, and the ReLu activation function has been used with four pooling sizes.

The proposed model has runs with total 10 epoch to get the optimized result. After 10 epoch, it is found that the precision, recall and f-measures score is above of the 0.9. The Recall, Precision and f1-score values for the CNN+LSTM model are shown in Table 5 and the confusion metrics of the proposed model is shown in Figure 1.

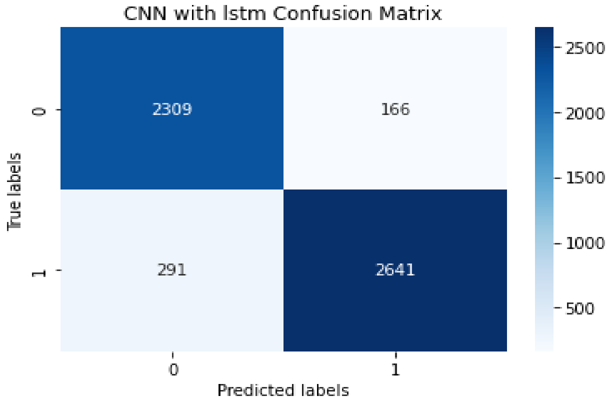

**Fig. 1. Confusion metrics of the proposed model.**

**Table 5. Accuracy values for CNN+LSTM Classifier**

| | Precision | Recall | F1-Score |
|---|---|---|---|
| Hate | 0.92 | 0.92 | 0.92 |
| Non-Hate | 0.90 | 0.91 | 0.91 |

However, the CNN+LSTM confusion matrix in figure1 shows that the model successfully predicted 2248 out of 2475 (non-hate class) and 2694 out of 2887 (hate class) predictions.

Next, we have compared the result of the proposed method with other machine learning algorithms. The proposed comparison is available in Table 6. Table 6 compares the accuracy, precision, F1 score, and recall of many different machine learning models to our CNN+LSTM[6] model in great detail. The graphical representation of the comparison is given in Figure 2.

**Table 6. The results of several classification algorithms and approaches for hate speech detection tasks are shown**

| Model | Accuracy | Precision | F1-Score | Recall |
|---|---|---|---|---|
| SVM | 0.61 | 0.60 | 0.71 | 0.87 |
| **CNN+ LSTM** | **0.92** | **0.94** | **0.92** | **0.90** |
| Logistic Regression | 0.62 | 0.61 | 0.70 | 0.83 |
| Naive Bayes | 0.57 | 0.56 | 0.71 | 0.96 |
| Random Forest | 0.77 | 0.81 | 0.78 | 0.76 |
| CNN | 0.91 | 0.92 | 0.92 | 0.91 |

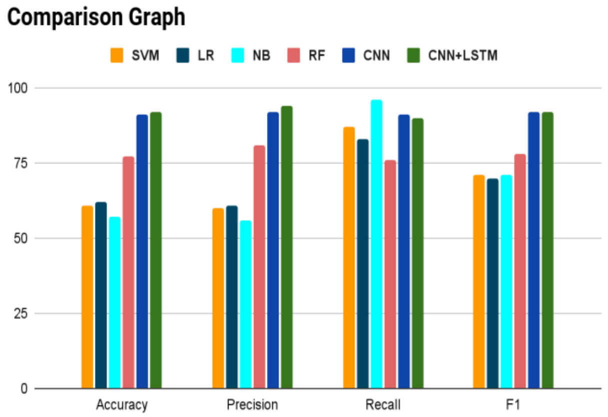

**Fig. 2. Comparative result of all models.**

Table 6 and figure 2 show that the proposed model's accuracy, precision, recall and f1-score are high compared to other models. Based on these two comparisons, we can conclude that the proposed model gives better results than the other models mentioned in Table 6. The above statements say that the proposed model can be used for Hate speech identification.

## 7. CONCLUSION

Hate speech has become a major concern, and a computerized hate speech detection system is one of the solutions to this major problem. Machine learning classifiers have been proposed as a method for detecting hate speech and offensive language among different online platforms that are prone to face hate. The main objective was to develop an automated approach for eliminating social hatred on social media and online communities. The defined model CNN+LSTM outperformed as compared to other existing approaches. When we tested the model using test data, we obtained 92.1% accuracy. More examples of offensive language that do not contain hostile phrases can be found here to address this issue. The model's lack of consideration for the negative word in a phrase is one of its drawbacks. This region can be improved by incorporating linguistic components. Furthermore, the performance of Naive Bayes was the worst. There are two significant limitations to our research. First, our model is lacking in predicting accuracy in real-time. Finally, it divides hate speech into two groups and cannot identify the message's intensity. Consequently, the suggested machine learning model is also useful for predicting how strong hateful messages will be. Hate speech identification is a difficult task for academics. The text in the social media post is followed by photographs, which are subsequently followed by text. Even in text, code-mixed languages are used. To make the algorithm more robust, future work on hate speech identification could include multilingual examples in several languages as well as multi-modal types of social media posts.

## APPENDIX

List of abbreviation used in the article:

| Abbreviation | Full Name |
|---|---|
| ML | Machine Learning |
| CNN | Convolutional Neural Network |
| LSTM | Long Short Term Memory |
| RF | Random Forests |
| NB | Naive Bayes |
| LR | Logistic regression |
| SVM | Support Vector Machine |
| TP | True Positive |
| TN | True Negative |
| FP | False Positive |
| FN | False Negative. |

## REFERENCES

[1] M. Sajjad, F. Zulifqar, M. U. G. Khan and M. Azeem, "Hate Speech Detection using Fusion Approach," 2019 International Conference on Applied and Engineering Mathematics (ICAEM), 2019, pp. 251-255, doi: 10.1109/ICAEM.2019.8853762.

[2] Das, A. K., Al Asif, A., Paul, A., & Hossain, M. N. (2021). Bangla hate speech detection on social media using attention-based recurrent neural network. Journal of Intelligent Systems, 30(1), 578-591.

[3] Nugroho, K., Noersasongko, E., Fanani, A. Z., & Basuki, R. S. (2019, July). Improving random forest method to detect hatespeech and offensive word. In 2019 International Conference on Information and Communications Technology (ICOIACT) (pp. 514-518). IEEE.

[4] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., ... & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. Journal of big Data, 8(1), 1-

74.

[5] Abro, S., Shaikh, S., Khand, Z. H., Zafar, A., Khan, S., & Mujtaba, G. (2020). Automatic hate speech detection using machine learning: A comparative study. International Journal of Advanced Computer Science and Applications, 11(8).

[6] Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017, May). Automated hate speech detection and the problem of offensive language. In Proceedings of the international AAAI conference on web and social media (Vol. 11, No. 1, pp. 512-515).

[7] Sazany, E., & Budi, I. (2018, September). Deep learning-based implementation of hate speech identification on texts in indonesian: Preliminary study. In 2018 International Conference on Applied Information Technology and Innovation (ICAITI) (pp. 114-117). IEEE.

[8] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[9] Zhang, Z., Robinson, D., & Tepper, J. (2018, June). Detecting hate speech on twitter using a convolution-gru based deep neural network. In European semantic web conference (pp. 745-760). Springer, Cham..

[10] Joachims, T. (1998, April). Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning (pp. 137-142). Springer, Berlin, Heidelberg..

[11] Severyn, A., & Moschitti, A. (2015, August). Twitter sentiment analysis with deep convolutional neural networks. In Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval (pp. 959-962)..

[12] Krishnan, H., Elayidom, M. S., & Santhanakrishnan, T. (2017). Emotion detection of tweets using naive bayes classifier. Emotion, 4(11), 457-62..

[13] Fatahillah, N. R., Suryati, P., & Haryawan, C. (2017, November). Implementation of Naive Bayes classifier algorithm on social media (Twitter) to the teaching of Indonesian hate speech. In 2017 International Conference on Sustainable Information Engineering and Technology (SIET) (pp. 128-131). IEEE.

[14] Kiilu, K. K., Okeyo, G., Rimiru, R., & Ogada, K. (2018). Using Naïve Bayes algorithm in detection of hate tweets. International Journal of Scientific and Research Publications, 8(3), 99-107..

[15] Malmasi, S., & Zampieri, M. (2017). Detecting hate speech in social media. arXiv preprint arXiv:1712.06427.

[16] Chandrasekharan, E., Samory, M., Srinivasan, A., & Gilbert, E. (2017, May). The bag of communities: Identifying abusive behavior online with preexisting internet data. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 3175-3187).

[17] Burnap, P., & Williams, M. L. (2015). Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. Policy & internet, 7(2), 223-242.

[18] Waseem, Z., & Hovy, D. (2016, June). Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In Proceedings of the NAACL student research workshop (pp. 88-93).

[19] Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017, April). Deep learning for hate speech detection in tweets. In Proceedings of the 26th international conference on World Wide Web companion (pp. 759-760).

[20] Uoc, N. Q., Duong, N. T., Son, L. A., & Thanh, B. D. A (2022) Novel Automatic Detecting System for Cucumber Disease Based on the Convolution Neural Network Algorithm. GMSARN International Journal 16 (2022) 295-301

[21] KS, A. K., Sarita, K., Kumar, S., Saket, R. K., & Swami, A.(2022) Machine Learning-based Approach for Prevention of COVID-19 using Steam Vaporizer GMSARN International Journal 16 (2022) 399-404.

[22] Gupta, C. L., Bihari, A., & Tripathi, S. (2021). Protein Classification Using Machine Learning and Statistical Techniques. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 14(5), 1616-1632.