Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Enhancement of Sentiment Analysis of Hate Speech through Ensemble Classifier Model

Renugadevi Somu[1], Subamalai RajaSekar[2]

[1]Assistant Professor, Department of Computer Science and Engineering, Anna University, Chennai, India.
[2]Post Graduate Student, Department of Computer Science and Engineering, Anna University, Chennai, India.
Email: srenuga@annauniv.edu

Hate speech is currently a trending topic on social media and has grown into a big issue. Current practices raise concerns regarding censorship. Our work is largely concerned with human rights, with a focus on developing innovative strategies that respect the freedom of speech while also recognizing and successfully combating discrimination. Using word embedding to record semantic relations found in the text, Glove addresses this problem by making it easier for readers to identify complex contextual patterns. Advanced deep learning models can be very helpful when they combine natural language processing (NLP) methods with the Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BILSTM). These models will be trained using sizable, labeled data sets that contain scenarios with both hate speech and non-hate speech content. Making use of and integrating the strengths of several RNN architectures—each with a special capacity to understand successive dependencies— An ensemble method is demonstrated to improve hate speech classification; neural network implementation is planned, as these methods have become the de facto standard for text categorization problems in the recent past. Our ensemble model offers a thorough approach to comprehending the complex emotions conveyed in text as well as the malicious intent of hate speech. The primary goals of the proposed work are system robustness and a decrease in overall loss. The ensemble model yielded precision, recall, and F1 scores of 0.98, 0.99, and 0.99, respectively. It is discovered that the ensemble model's loss in identifying hateful emotion is 0.19.

Keywords: Hate Speech, Enhanced Classification, Ensemble Model, Sentiment Analysis, Recurrent Neural Networks.

1. Introduction

Social media is a popular medium for online inter- action and public opinion manifestation. Social media can reflect the general public's views on a range of events. The enormous amount of user-generated content more closely mimics the offline world than official news sources

since it is unregulated. This might adversely impact the user's online experience as well as the community overall. Social media platforms make it essential to detect opposing comments posted by individuals once their contents are published. A hateful message is any kind of content that is published and used to criticize members of a particular social group by exhibiting hatred towards individuals. Because their virtual world is unsafe for their users, ecommerce sites and media platforms cannot afford to lose sponsors as well. Due to the excessive risk of associating their brands with harmful and hateful virtual locations, a broader spectrum of enterprises are interested in contributing to studies on systems that can block hate speech. The increasing growth of user-generated material on internet platforms in an era of digital communication has created a pressing need for efficient tools to counter hate speech and understand the sentiments expressed in textual data. Our research aims to remove bias that violates the law while enhancing tactics to protect free speech on social media and other platforms. Since the COVID-19 closure, hate speech pertaining to politics, racism, nationalism, and religion has become increasingly common. Identifying and removing these messages is a laborious task, made more difficult by the seemingly limitless number of damaging tweets. In order to detect and categorize hate speech, deep learning must be used in conjunction with natural language processing (NLP) models. Text categorization tasks and severity detection heavily rely on deep learning techniques.

In response to these urgent issues, this study suggests a novel ensemble model that may be used to build a comprehensive solution that can handle sentiment analysis and hate speech identification at the same time. However, the main issues with the current models are,

i. Manual intervention: The obstacles to manual participation in hate speech detection systems include subjectivity, resulting in inconsistent categorization, scalability issues due to the impracticality of processing the huge volume of internet content, and potential delays in resolving rapidly evolving situations.

ii. Bias exists in algorithms: Biases that exist in the training set of data may continue to influence algorithms. Models developed from biased datasets, which reflect societal imbalances and historical prejudices have a tendency to strengthen existing biases.

iii. Insufficient benchmark datasets: Additional datasets covering additional possible targeted categories are desperately required.

iv. Hate speech elimination: Online businesses and media platforms cannot afford to lose sponsors.

A larger number of businesses are involved in a study that focuses on technology that can eliminate hate speech in order to boost sales for them. The suggested ensemble model achieves enhanced classification while resolving the aforementioned difficulties. Thus the objectives of the Proposed System include:

i. Developing an Ensemble-based hate speech detection system that tackles the constraints currently available in machine learning approaches.

ii. Acquiring well-annotated, top-notch datasets.

iii. Increasing the effectiveness of online content categorization in real-time applications.

iv. Developing techniques to evaluate the model's judgments and provide clear justifications for its classifications.

v. Addressing the issue of Algorithmic Bias and guaranteeing the model's impartiality.

vi. Establishing reliable assessment criteria to gauge how well the models perform in identifying hate speech, such as accuracy, loss, precision, recall, and F1-score.

vii. Classifying the content's emotional tone as beneficial, unfavorable, or possibly dangerous. This offers deeper insights into the feelings behind hate speech.

viii. Developing a system that is flexible to many languages and cultural settings, scalable, and able to deal with a significant amount of text data.

The goal of the proposed work is to provide a fair and reliable sentiment analysis tool that can be used in real-world contexts to comprehend textual sentiment. The rest of this article is structured as follows. Section two describes the literature of this problem domain, Section three describes the methodology adopted for the proposed model, Section four discusses the experimental tests and results and Section five concludes the findings of the work.

2. Literature Review:

Roy et al.[17] focused on machine learning models such as SVM, NB, RF, and LSTM. This may require knowledge, plenty of computing resources. Overall performance is potentially influenced by both the amount and the quality of training data. In another related work Roy et al. [16] proposed a combined method of DCNN and GloVe embedding to identify tweet meanings via convolution operations. However, the complicated structure of algorithms and the need for significant sets of training data could pose restrictions. Another research paper of Makhadmeh et al. [1] investigate linguistic, emotional, unigram and pattern attributes in data from Twitter. The potential drawback of the algorithm could be the complex nature of integrating deep learning models alongside natural language processing techniques, demanding professional expertise regarding both configuration and maintenance.

Another research study is implemented using multi-label task. In this study Min et al. [10] focused on multi-label training that simultaneously learns sensors on both jobs Already-trained emotion detectors cause biases. It requires more computing resources and training time. This work used ML models in a multitasking manner. SocialHater BERT: [5] represents machine learning models undergo training employing profiles of users and tweet information. The challenge of subjectivity in classifying hate speech, which comes through different opinions based on contextual as well as topical factors that affect classification accuracy. This related task is implemented using ensemble Models and Stacked Approaches. In this study Lee et al. [8] proposed the uses of SVM and GCR-NN models. One of the limitations is that deep learning models may require greater resources from computers because of their level of complexity while training. Ayo et al. [2] proposed fuzzy logic and probabilistic clustering method, thereby improving the representation of features and classification efficacy by automatically recognizing topics. Data inequalities, ambiguity, threshold setting, fragmentation issues, and misclassification errors were possible issues with this type of model. This research study is implemented using Neural Network Architectures and Sentiment Analysis.

The work of Junyi et al. [4] mainly concentrates on sentiment analysis and short-text classification by employing TF-IDF features and Convolutional Neural Networks (CNN). The need to rely on contextual pre-trained embedding of words and the utilization of restricted datasets for Word2Vec's training on short sentences are drawbacks, though. Pitsilis et al. [13] present the techniques by using specific user data. But there are specific limitations that impact validity and generalizability, including the high level of detail of NLP models and the need for several trials to deal with unpredictable phenomena. This research study utilized hybrid models and aspect-based Sentiment Analysis. Nurulhuda et al. [20] explore aspect-based sentiment analysis on Twitter. It is possible that not every component that could have been noticed may be regarded noteworthy and that latent elements may prove challenging to correctly determine are drawbacks.

The following issues have been observed from the above related works,

i. The accuracy of multilingual tweets is relatively low.
ii. The HSD task's interpretability is challenging to achieve.
iii. Problems with threshold setting, imprecision, imbalanced data, fragmentation, subjective nature of hate speech and the unequal distribution of tweets.
iv. The rate of misclassification is elevated in contrast to alternative models.
v. There is not enough data to build a model that can catch all hate speech on the OSN.
vi. The loss percentage is high for tweets.
vii. Ignoring hate speech contained in audio and video files.
viii. There was a lack of classification accuracy with the fuzzy logic linguistic variables.

3. Methodology:

This section covers the design and methodology of the proposed system. Raw tweet stream is taken from the Twitter dataset and preprocessed. This entails tokenization, padding the sequence, removal of punctuation, removal of stop words, removal of frequent words, removal of hash tags, URLs and cleaning tweet data. The proposed method combines the advantages of LSTM, BILSTM, and GRU models in an ensemble manner and uses GloVe embedding for feature ex- traction. By strengthening the system's capacity to recognize hate speech and sentiment in text data, this method seeks to provide more comprehensive and effective techniques for applications such as social media analysis and online content control. For efficient feature extraction, utilize GloVe (Global Vectors for Word Representation) embedding. GloVe embedding are renowned for their ability to accurately represent word associations with semantics and convey an in-depth representation of the text's underlying meaning. By using this step, you can make sure that the model has a solid contextual understanding foundation. We intend to capitalize on the abilities of recurrent neural networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory(BILSTM), and Gated Recurrent Unit (GRU) models, in the proposed method for hate speech identification with sentiment analysis. Optimizing the ability of the algorithm to extract contextual information with complexities and temporal connections from

text data is the goal. The predictions from each individual model (LSTM, BILSTM, and GRU) will be coupled in order to create the ensemble model, potentially with the assistance of a meta-learner or a weighted average. By leveraging the positive aspects of each model separately, this ensemble approach produces a more trustworthy and precise system for sentiment analysis-based hate speech detection. The proposed model for detecting hate speech from twitter tweet data is shown in figure 1.

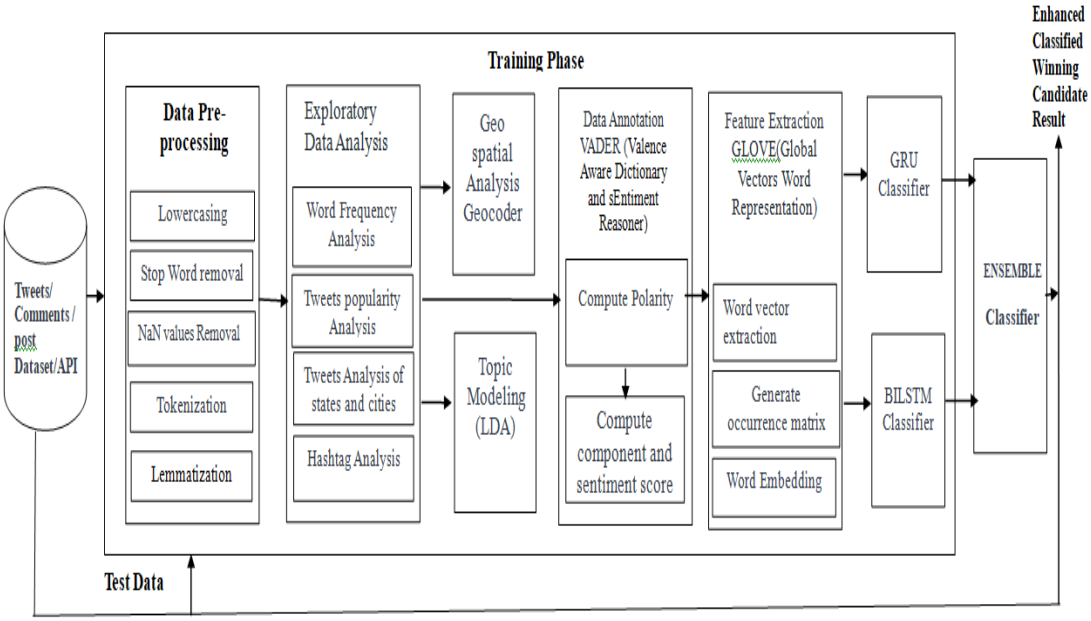

Figure1. Block diagram of Hate Speech Recognition System

3.1) Data Collection

Capturing text sample data is the first level in the detection of hate speech. Such information could originate from test raw data, social media posts, reviews, comments, and various other text sources. Each text sample in the test dataset should be labeled, identifying it either as having hate speech or not. It is capable of holding 25MB of toxic data content. For better context, extra information has been obtained from each tweet in addition to its actual content, such as the number of words, characters, and tags per tweet. The tweets and their IDs have been included in the Twitter test dataset. This article's tweets originate from twitter.com. Each Twitter post is casually composed by the individuals that feature equally hazardous and non-toxic tweets. The dataset, which originated from Twitter, has approximately 17,879 Twitter comments coupled with their respective IDs. The compressed length of the Twitter test dataset, given its size, is approximately 17 GB, while the raw text data comprises approximately 15 GB. The tweets tend to be lengthy, with a typical word count of over 40. Each tweet has roughly 25 words. There are a total of about 80 characters in each tweet. Detecting the presence of a hashtag in a tweet affects in finding the number of hashtags. To offer an unbiased dataset, separate annotator is performed on the Annotated data.

3.2) Data Preprocessing

Text data must be preprocessed in order to be tidied up and made ready to undergo further procedures. To minimize noise and normalize the text, preprocessing is essential. It contains Lower casing, removing characters and numbers from tweets, getting rid of special characters and symbols, eliminating white space, taking off hashtags and URLs, stopping short words, eradicating frequent patterns, removing emoji, avoiding extraneous characters and punctuation, accurately spelling, tokenization, and padding the sequence. Preprocessing is illustrated in figure 2. The major components of data preprocessing are discussed in the following sub-sections.

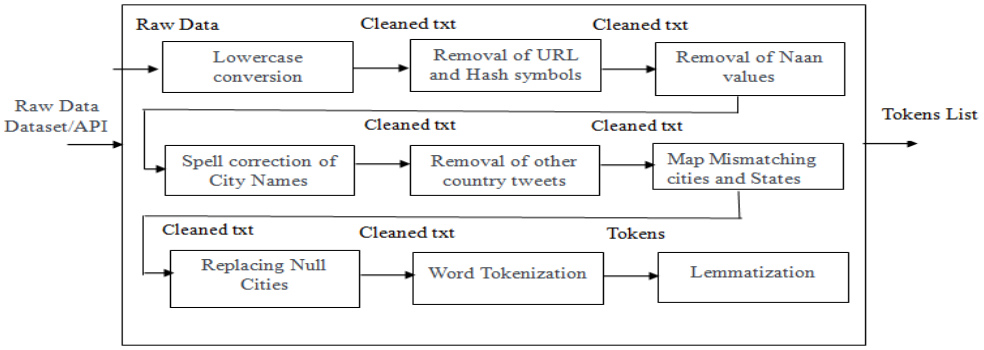

Figure2.Preprocessing Stage

3.2.1) Lowercase Conversion

Tweets contain uppercase letters, lowercase letters, and combinations of both letters. Lowercase conversion converts all these formats into standardized lowercase letter formats.

3.2.2) Stop Words and Frequent Word Removal

Stop words are phrases that are frequently used in language to better communicate specific elements without suggesting things. In addition, the existence of these phrases will have a major impact on the quantity of storage capacity required for data as they are more frequent In addition, since there will be plenty of tokens, which will have potential performance overhead. The majority of frequently utilized phrases can be discovered in tweets. These phrases should be converted to standard words to avoid biases in analysis.

3.2.3) Tokenization

Tokenization breaks an unorganized text into discrete, smaller parts, which are commonly referred to as tokens. The token was one element that acts as the foundation block for an entire tweet. Each tweet has been identified as a distinctive token. For tweet analysis, tokenization is an important preprocessing step. Because the text data is to be processed and input into the deep learning models, such a mathematical conversion of the text data is essential.

3.2.4) Padding the Sequence

Sequence padding is trying to maintain sequences uniform in length, it entails adding placeholder values to them. This enables batch processing, ensures framework compatibility,

and promotes efficient training. Sequences are usually restricted to a particular length, and lesser sequences have been padded with a distinctive token or value, by ensuring that each parameter contains equal dimensions.

3.3) Exploratory Data Analysis

It is an essential starting point in the data analytics process, which also comprises interpreting, summarizing, and expressing the primary characteristics of a dataset. EDA has made it simpler for analysts and data scientists to find patterns in data, detect anomalies, and create ideas for more research. It analyzes the words, characters, and hash tags that can be found within each tweet. Plotting was carried out employing the Matplotlib library to generate the final histogram.

3.4) Data Annotation

The act of categorization, labeling data using relevant details is referred to as data annotation. It may be performed automatically by tools or manually by human annotators. Sentiment analysis also referred to as opinion mining, is a natural language processing (NLP) technique for determining the sentiment of the tweets. Subjectivity analysis is helpful for identifying if a statement is based on a personal opinion, while polarity analysis identifies an opinion's emotional tone as either positive or negative. Many applications, including social media monitoring, brand reputation management, and customer feedback analysis, rely on these attributes. Algorithm-1 depicts the steps involved in Sentiment analysis task where the sentiment analysis function has been applied to each and every tweet of the Dataset.

Algorithm - 1 : Ensemble Model Framework Algorithm

```
Input:  Preprocessed Tweets
Output:  Sentiment score and Polarity
1: procedure  SENTIMENT ANALYSIS(tweets)
2:        subjectivity  ⇐ 0
3:        polarity ⇐ 0
4:       for all tweet in tweets do
5:          b ⇐ TextBlob(tweet)
6:          sscore ⇐ b.sentiment.subjectivity
7:          pscore ⇐ b.sentiment.polarity
8:          subjectivity.append(sscore)
9:          polarity.append(pscore)
10:      end for
11:      return subjectivity, polarity
12: end procedure
```

3.5) Feature Extraction

Words have been transformed into dense vectors utilizing an embedding methodology called GloVe [12] that preserves semantic information. It aims to capture the semantic connections between words through the use of both word co-occurrence data and matrix factorization. The regularity of the words that occur together in the dataset can be seen in the embedding that GloVe generates by factoring the word co-occurrence matrix, indicating the total number of times a word (i, j) appears in the context(j), where the context is defined by the window size that exists before and after the $word_i$.

An objective function is used to optimize these embedding; the product of word vectors corresponds with the actual cooccurrence probabilities. GloVe embedding enhances the accuracy of text categorization and sentiment analysis is frequently used for tasks related to natural language processing. The algorithm – 2 depicts the GloVe word embedding steps.

Algorithm - 2 : GloVe Word Embedding

Input: Sentiment Score

Output: Word Vectors

1: procedure WORD VECTORS(score)
2: $w \Leftarrow 0$
3: $b \Leftarrow 0$
4: for all Sentiment Score in tweets do
5: for Each word pair (i, j) and X_{ij} do
6: $P_{ij} = X_{ij} + C$
7: calculate F(x) and δ
8: end for
9: end for
10: end procedure

3.6) Recurrent Neural Networks

The potential of Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and Gated Recurrent Unit (GRU) to deal with the complex nature of data in sequence makes them remarkable. Originally created as remedies to the diminishing gradient issue that afflicted conventional RNNs, LSTMs and GRUs have grown into effective tools for capturing complicated interactions in sequences. It controls data flows through input, output and forget gates. LSTMs have the ability to carry out this amazing feat. On the flip side, GRUs had an additional simple technique, employing update and reset gates to gain control over the traversal of incoming data. BiLSTMs is another model that takes advantage of bidirectional processing and effectively traverses input sequences in both forward and reverse directions at the exact

same time. This dual perspective integrates information from the past and future domains to provide a greater understanding of context.

3.7) Hyper Parameters

3.7.1) Global max pooling and Average pooling

For the purpose of minimizing the dimensionality of the features before feeding them to layers that are completely linked for classification global max pooling and global average pooling are commonly employed. They contribute to retaining important geographic data despite reducing the network's computational complexity. The highest possible value for each of the input's map features is determined via global max pooling. The average value for every map feature in the input is determined by employing global average pooling.

3.7.2) Activation Function

The tanh activation function of equation 1 is often employed to resolve the diminishing gradient issue and capture complex nonlinear interactions in normalized form. It has an output range of [-1, 1].

$$\mathbf{tanh}(\mathbf{x}) = \frac{\mathbf{e^x - e^{-x}}}{\mathbf{e^x + e^{-x}}} \qquad (1)$$

$$\mathbf{sigmoid}(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{1 + e^{-x}}} \qquad (2)$$

The outcome of binary classification models primarily employs the sigmoid activation function (equation 2), and its outcome range is [0, 1]. For tasks involving binary classification, Sigmoid is essential due to its ability to generate understandable probabilities and diminish the gradient problem.

3.7.3) Dropout

Prior to the classification layer and following the embedding layer, dropout layers are added at a 30 percentage rate to mitigate the over-fitting hostility that arose during training as a consequence of the imbalance in the class situation. A lesser figure can be experimented with until the rate of dropout is determined.

3.7.4) Optimizer

The advantageous features of AdaGrad and RMSProp optimization techniques are brought together in the Adam optimizer. Adam determines the initial and subsequent moments of the gradient to preserve rates of adaptive learning for each parameter. As a result of this, it is able to automatically alter the learning rate for each parameter, thereby rendering it excellent for jobs containing sparsely distributed information. The method possesses rapid convergence as well as excellent generalization efficiency, which makes it cost effective as well as efficient.

3.7.5) Dense Layer

Multiple labels can be assigned to an instance in multilabel classification. Each unit decides whether or not the corresponding label fits the input instance. A dense layer has been employed with the number of labels as units and the sigmoid as the activation function for performing multi-label classification. As a result, every neuron in the dense layer will be assigned with a binary value.

3.8) Classification models

3.8.1) Classification Model 1: LSTM

Sequences of inputs go through processing continuously within Long Short-Term Memory (LSTM) networks, involving distinct actions at every step. Initially, the forget gate f selects what information to disregard from the cell state c that came preceding it. The fresh data that needs to be incorporated into the state of the cell through the candidate cell ct is subsequently identified by the input gate I. Constantly, modifying the cell state CT integrates fresh data while selectively discarding previous data according to gate choices. The modified cell state then affects the output gate o, which regulates how much data flows to the output and ultimately generates a fresh hidden state h. This approach allows LSTM networks to tackle difficulties like vanishing and overflowing gradients [19].The L_t, n_t, F_t, u_t,O_t are defined as in equation 3 to equation 7.

$$\mathbf{L_t} = \boldsymbol{\sigma}(\mathbf{V_I}[\mathbf{b_{t-1}}, \mathbf{i_t}] + \mathbf{S_I}) \quad (3)$$

Where, V_I is the weight and S_I is the bias. It was followed by the updating the previous data u_{t-1} to new data u_t defined in (Eq. (6)).

$$\mathbf{n_t} = \mathbf{tanh}(\mathbf{V_c}[\mathbf{b_{t-1}}, \mathbf{i_t}] + \mathbf{S_c}) \quad (4)$$

Where, V_c is the weight and S_c is the bias.

$$\mathbf{F_t} = \boldsymbol{\sigma}(\mathbf{V_f}[\mathbf{u_{t-1}}, \mathbf{i_t}] + \mathbf{S_f}) \quad (5)$$

$$\mathbf{u_t} = \mathbf{F_t} * \mathbf{u_{t-1}} + \mathbf{I_t} * \mathbf{c} + \mathbf{S_f} \quad (6)$$

where V_f is the weight,u_{t-1} is the previous timestamp output, it is the new input message, and S_f is the bias.

$$\mathbf{O_t} = \boldsymbol{\sigma}(\mathbf{V_o}[\mathbf{B_{t-1}}, \mathbf{i_t}] + \mathbf{S_o}) \quad (7)$$

$$\mathbf{x_t} = \mathbf{O_t} * \mathbf{tanh}\,(\mathbf{c_t}) \quad (8)$$

where x_t of equation 8 is the new information sent to the next LSTM cell.

3.8.2) Classification Model 2: BILSTM

A BiLSTM analyzes an input sequence from the left to the right throughout its forward pass. The forward LSTM cell utilizes the input value y_i along with the previously computed forward hidden state $\vec{\mathbf{o}}_{\mathbf{i-1}}$ to determine the current forward hidden state $\overrightarrow{\mathbf{o}_\mathbf{i}}$ at each time step t. The equations are given below [7]

$$\vec{\mathbf{o}}_\mathbf{i} = \mathbf{LSTM}\,(\mathbf{y_i}, \vec{\mathbf{o}}_{\mathbf{i-1}}) \quad (9)$$

where, y_i represents the input at step t in time. $\vec{\mathbf{o}}_{\mathbf{i-1}}$ is the state that was hidden during the prior time step. $\vec{\mathbf{o}}_\mathbf{i}$ indicates the state which it is hidden at time step t. A BiLSTM analyzes an input sequence from the right to the left throughout every reverse pass. The inverse LSTM cell utilizes the input yi and the previous backward hidden state $\overleftarrow{\mathbf{o}}_{\mathbf{i-1}}$to determine the next backward hidden state $\overleftarrow{\mathbf{o}}_\mathbf{i}$ at each subsequent step t.

$$\overleftarrow{\mathbf{o}}_\mathbf{i} = \mathbf{LSTM}\,(\mathbf{y_i}, \overleftarrow{\mathbf{o}}_{\mathbf{i+1}}) \quad (10)$$

where y_i represents the input at step t in time. $\overleftarrow{\mathbf{o}}_{\mathbf{i+1}}$ is the state that was hidden during the next time step. $\overleftarrow{\mathbf{o}}_{\mathbf{i}}$ indicates the state which is hidden at time step t.

The BiLSTM is able to gather information in both past and potential future contexts due to these two passes, which is helpful for tasks such as sentiment analysis and sequence tagging where it is essential to fully understand the context of words in both directions as well. For the purpose of getting the final representation, both forward and backward hidden states typically get concatenated at each time step.

$$\mathbf{o_t} = [\vec{\mathbf{o}}_{\mathbf{i}}, \overleftarrow{\mathbf{o}}_{\mathbf{i}}] \quad (11)$$

where o_t represents the concatenation of past and future pass results $\vec{\mathbf{o}}_{\mathbf{i}}$ indicates the forward pass and $\overleftarrow{\mathbf{o}}_{\mathbf{i}}$indicates the backward pass.

3.8.3) Classification Model 3: GRU

Gated Recurrent Unit (GRU) is another form of Recurrent Neural Network (RNN) architecture employed for sequential data processing. It is also substantially less expensive compared to the LSTM (long short-term memory) design because it contains fewer parameters and operations. The GRU model operates through the use of the gates r_t and z_t , the current input x, the previous hidden state h_{t-1}, and additional information to modify its hidden state h in every step. The equations 12 to 15 indicate the computation of the GRU parameters [11].

$$\mathbf{r_t} = \boldsymbol{\sigma}(\mathbf{W_r}.[\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_r}) \quad (12)$$

Where, r_t indicates the reset gate, h_{t-1} indicates the previous hidden state, x_t represents the input value, W_r indicates the weights, b_r represents the bias, σ represents sigmoid function.

$$\mathbf{z_t} = \boldsymbol{\sigma}(\mathbf{W_z}.[\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_z}) \quad (13)$$

Where, z_t represents the update gate.

$$\tilde{\mathbf{h}}_{\mathbf{t}} = \mathbf{tanh}\,(\mathbf{W_h}.[\mathbf{r_t} * \mathbf{h_{t-1}}, \mathbf{x_t} + \mathbf{b_h} \quad (14)$$

Where, $\tilde{\mathbf{h}}_{\mathbf{t}}$ represents the candidate gate.

$$\tilde{\mathbf{h}}_{\mathbf{t}} = (\mathbf{1} - \mathbf{z_t}) * \mathbf{h_{t-1}} + \mathbf{z_t} * \tilde{\mathbf{h}}_{\mathbf{t}} \quad (15)$$

Where, z_t represents the updated hidden state at each time t.

3.8.4) Enhanced Classification Model: Ensemble Model

The benefits of each design are put together in an ensemble model utilizing LSTM, BILSTM, and GRU models to boost performance as a whole in consecutive data issues. For the purpose of generating a final outcome, the ensemble model combines the results from every individual model. By utilizing the mutual beneficial features of each architecture, this ensemble approach enhances efficiency and consistency in applications like machine translation, language modeling, and sequence prediction. Algorithm - 3 depicts the Ensemble framework.

Algorithm - 3 : Ensemble Model Framework Algorithm

1: Input: X ={x_1, x_2, ..., x_T}

2: Output: Enhanced predicted output

```
procedure ENHANCED CLASSIFICATION(X)
    i ⇐ 0
    for each model in LSTM, BILSTM, GRU do
    Train models with X
    Do prediction for each and every model
    Add the prediction to the list l
    end for
    Enhanced predicted output l
    return l
end procedure
```

The result of the ensemble approach has been determined to categorize any particular case through evaluating the cumulative ensemble and average weighted ensemble of the separate models as mentioned in [15].Weight has been assigned to every single model that makes up the ensemble model through the weighted averaging procedure according to how effectively it held its own in the test data. The weights assigned to the models range from zero (0) to one (1), and the total weight value is one. The average prediction and weighted average prediction formulas are highlighted in equations 16 and 17 respectively.

$$\mathbf{AvgPrediction} = \frac{\mathbf{1}}{\mathbf{M}} \sum_{\mathbf{i=1}}^{\mathbf{M}} \mathbf{P_i} \qquad (16)$$

$$\mathbf{WeightedAvgPrediction} = \sum_{\mathbf{i=1}}^{\mathbf{M}} \mathbf{w_i P_i} \qquad (17)$$

Where P_i indicates i^{th} model prediction and M is the number of models and w_i denotes i^{th} model weight.

4. Experimental Test and Results

The dataset utilized for the proposed work has been stratified into training and testing sets with an 80:20 ratio for each prediction task using the stratified sampling approach. Using the testing dataset, the performance has been assessed for both single and ensemble classifiers. The outcomes of the proposed approach for analyzing sentiment using deep learning and natural language processing are presented in this section. Final results for accuracy, precision, recall, F1 score, number of correct predictions (CP), number of incorrect predictions (WP), loss, and classification report are shown for each and every model individually [6].

4.1) Analysis of Dataset

Utilizing the Twitter dataset, this strategy's capability to determine hate speech has been shown [3] through the use of numerous NLP processing steps and deep learning techniques. The datasets encompass seventeen thousand tweets that require continual evaluation at both

sentence and word levels so that social media networks are able to eliminate inappropriate posts in particular circumstances. For this purpose, 80 percent of the Twitter data that had been collected was used as the training model. An ensemble-optimized deep learning method and the Keras library function were applied. The remaining 20 percentage of Twitter data served as the testing model, whereas the hate speech detection technique was developed using the TensorFlow backend.

4.2) Exploratory Data Analysis Results

Figure 3 illustrates the occurrence of every tweet on the y-axis and the number of words on the x-axis. The result and the histogram were created using the Matplotlib library's plot function.

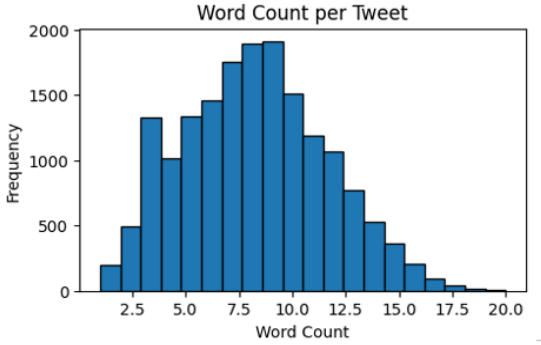

Figure 3. Exploratory Data Analysis

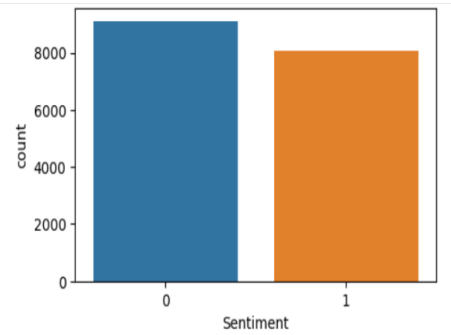

Figure 4. Sentiment Analysis

4.3) Analysis of Sentiments Results

Based on the sentiment analysis of the examined tweet, Figure 4 illustrates the polarity of tweets through a bar chart, in accordance with the sentimental assessment of the examined tweets. As indicated, the sentiment analyzer's TextBlob module was employed to figure out subjectivity and polarity, and the matplotlib module's plot method was subsequently employed to show the outcome. Each tweet's subjectivity, polarity, and ultimate analytical score are shown in Table 1. More specifically, a tweet is assigned with a value of 0 if the polarity score is less than 0, suggesting that the tweet is a "hated" post. A tweet is categorized a "non-hated" if its inverse polarity score is more than or equal to 0. The value 1 represents the outcome of the sentiment Analysis Module.

Table 1. Analysis of Sentiments

S.No.	Tweets	Subjectivity	Polarity	Score
1	life requires willpower find new challenges	0.227273	0.068182	1
2	limited rain shine set edition got today	0.142857	-0.071429	0
3	blacktina ill thick never women again	0.737500	-0.400000	0
4	last exams happy tomorrow hey yay guy	0.533333	0.400000	1
5	weekend healthy filled sun- beams have everyone enjoy the day	0.700000	0.450000	1

4.4) Feature Extraction Results

Word clouds and GloVe word embedding were the two different methods of feature extraction employed in the present study. Figure 5 is a visual representation of a word cloud, also known as a tag cloud, that highlights the key keyword or terms that influence the sentiments. Another

method used to determine the semantic relationship among each text in the multi-tweets was the word embedding. For specific tasks, pre-trained GloVe vectors have proved the benefits without having the model be manually retrained.

Figure 5. Feature Extraction - Visualization of a tweet word cloud

4.5) Comparative Analysis of Model Evaluation Results

For the purpose of training the learning algorithms to categorize data that was obtained from an Twitter stream, all of the deep learning models, under this study, have been taken into account. Additionally, the accuracy, precision, recall, and F1 score were also determined. Loss function [9] is given in Equation 18.

$$\mathbf{Loss} = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} (\mathbf{y_i} - \hat{\mathbf{y}}_\mathbf{i})^{\mathbf{2}} \qquad (18)$$

where n is the number of samples, yi is the true value and ˆyi is the predicted value Considering losses of 0.33, the LSTM and present models exhibit significant forecasting errors, revealing limitations to recognizing relevant similarities. With a loss of 0.26, the BiLSTM model performs better than expected, indicating higher precision. On the flip side, the GRU model beats other recurrent designs and exhibits better results with a loss of 0.21, demonstrating its efficacy in capturing temporal dependencies. The ensemble model outperforms individual models and illustrates the potential benefits of model consolidation by combining predictions from numerous models to achieve a competitive loss of 0.19. With everything considered, the ensemble and GRU models perform extremely well and have been feasible choices to mitigate error rates in prediction. Figure 6 shows the graphical representation of this loss function.

Figure 7 represents the graphical representation of this Precision for Hate Speech Detection. Precision [14] is computed as in Equation 19.

$$\mathbf{Precision} = \frac{\mathbf{True\ Positive}}{\mathbf{True\ Positive + False\ Positive}} \qquad (19)$$

The precision values derived from the different models, ranging from 0.96 to 0.99, indicate continuously excellent results. Considering precision scores of 0.96, all the LSTM and BiLSTM models demonstrate an elevated level of capability to correctly identify events that are positive. Given a precision score of 0.97, the GRU model does slightly better than them, demonstrating higher precision in positive instance identification. The Ensemble model achieves the highest precision (0.99), illustrating the extent to which integrating predictions from numerous models can enhance precision. All of the models exhibit good precision overall, with the Ensemble model obtaining the highest precision and emphasizing the beneficial effects of model consolidation for achieving higher anticipated accuracy.

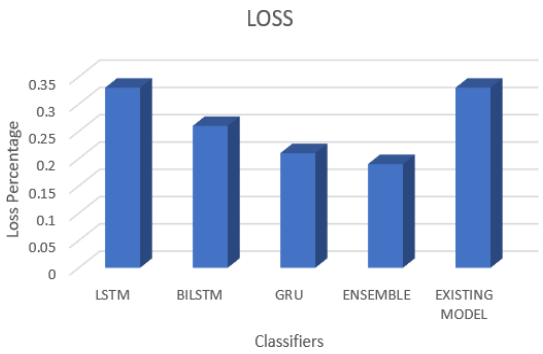

Figure 6. Performance metrics - Loss

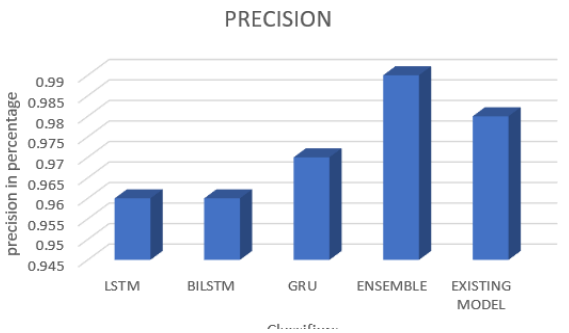

Figure 7. Performance metrics - Precision

Figure 8 is the graphical representation of the Recall values for Hate Speech Detection. Recall value [14] is computed as in Equation 20.

$$\textbf{Recall} = \frac{\textbf{True Positive}}{\textbf{True Positive+False Negative}} \qquad (20)$$

Evaluating the recall values for the various models indicates uniformly excellent outcomes, with overall scores fluctuating between 0.96 and 0.99. Both the LSTM and BILSTM models achieve recall values of 0.96 and 0.97, respectively, illustrating that they are reliable in correctly recognizing events that are positive. With a score of 0.98, the GRU model exhibits better recall, suggesting greater accuracy for recognizing positive cases. Interestingly, the Ensemble and the present models both acquire the highest recall scores of 0.99, showing their extraordinary capacity to properly identify instances that are positive. All models exhibit high recall overall, but the Ensemble and present models do exceptionally well within the area of recall, emphasizing how well they are able to detect beneficial instances and indicating their future potential for applications with excellent performance.

Figure 9 represents the graphical representation of this F1score for Hate Speech Detection. F1 score [18] is estimated as follows:

$$\mathbf{F_1} = \frac{\mathbf{2*precision*recall}}{\mathbf{precision+recall}} \qquad (21)$$

It can be made apparent when examining the F1 scores of the various models that they all perform effectively, having scores ranging from 0.96 to 0.99. High precision and recall are shown with F1 scores of 0.97, especially through the GRU and BiLSTM models. Despite an F1 score of 0.96, indicating that it is slightly lower but still satisfactory, the LSTM model follows shortly after. With an F1 score of 0.99, the ensemble model impressively outperforms individual models, highlighting the importance of bringing together numerous models. The present model, with an F1 score of 0.98, similarly works effectively. Most of the models performed well overall; however, the ensemble model stood out as the most efficient, highlighting the positive effects of model aggregation and achieving higher predicted reliability.

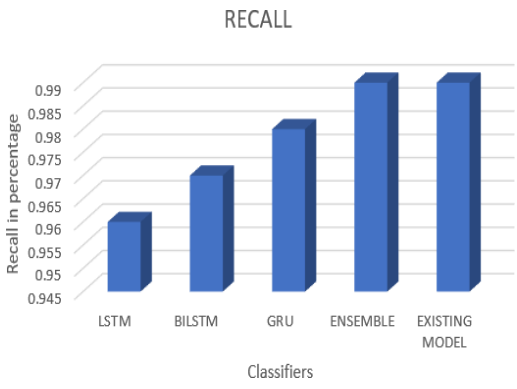

Figure 8. Performance metrics - Recall

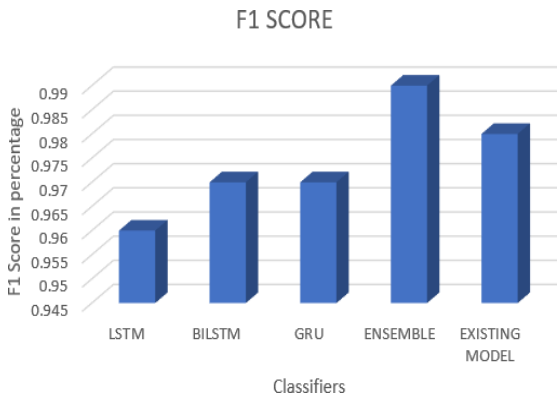

Figure 9. Performance metrics – F1 Score

Table.2 illustrates the various performance evaluation metrics assessed for the proposed Hate Speech Detection model.

Table 2. Performance Evaluation Metrics

Performance Metrics	Percentage
Accuracy	96
Loss	19
F1 Score	99
Precision	99
Recall	99

Figure 10 demonstrates the comparison of performance metrics applied on various methods for Hate Speech Detection.

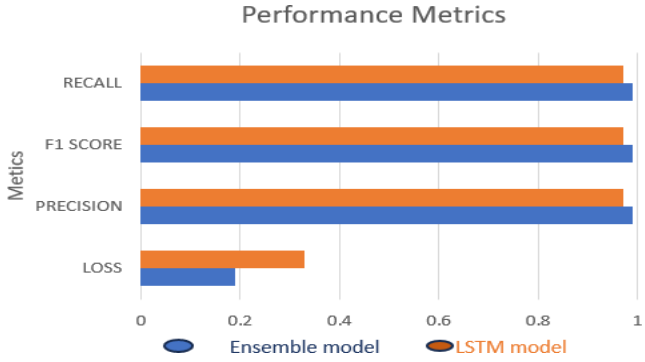

Figure 10. Comparison of Performance Metrics for Hate Speech Detection methods

By comparing the ensemble model's performance indicators to those used by the other models, it is obvious that the ensemble model works better than others in many ways. The ensemble model is capable of mitigating anticipated errors more effectively than the established model, as demonstrated by its significantly smaller loss of 0.19 relative to the latter's loss of 0.33. In addition, the ensemble model surpasses the existing system with scores of 0.98 with regard to

precision, F1 score, and recall, every one of which is higher at 0.99. Additionally, even though both models exhibit excellent accuracy, the ensemble model scoring 96 Percentage and the existing model scoring 97 Percentage, the ensemble model exceeds the other models by a substantial amount. In conclusion, the Ensemble model outperformed all other models, showing the capacity to accomplish high recall, precision, and overall predictive accuracy while also minimizing predictive errors.

5. Conclusion

Overall, in a nutshell, the current endeavor utilized ensemble modeling techniques to tackle the critical tasks of sentiment analysis and hate speech identification. We have demonstrated the effectiveness of classifier models, including ensemble approaches, in accurately acknowledging sentiments and identifying hate speech in text data via comprehensive assessment and evaluation. The ensemble model has especially grown into an effective tool, displaying enhanced recall, precision, and overall predictive accuracy. Ensemble methods have been shown to be essential for improving the reliability and robustness of sentiment analysis and hate speech detection systems through the use of the combined benefits of several models. These outcomes emphasize the significance of ensemble modeling in addressing difficult problems associated with natural language processing. This research could potentially be enhanced further in a number of ways in order to enhance hate speech and sentiment analysis detection systems. To get started, additional investigations into complex deep learning architectures like BERT or GPT models could boost their understanding of text context and semantics, leading to predictions that are more likely to be correct. Combining multimodal data sources such as audio, video, and images might boost the overall comprehension of internet content by providing analysis in a fuller context. Strategies from explainable AI could offer details regarding model choices, improving transparency and trust among users.

Acknowledgement:

I thank my co-author Ms.Suba Malai R for her immense contribution to this work.

Funding Statement:

The authors did not receive financing for the development of this research.

Data Availability:

Data sharing is not applicable to this article.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. Zafer Al-Makhadmeh and Amr Tolba. Automatic hate speech detection using killer natural language processing optimizing ensemble deep learning approach. Computing, 102:501–522, 2020.

2. Femi Emmanuel Ayo, Olusegun Folorunso, Fri- day Thomas Ibharalu, Idowu Ademola Osinuga, and Adebayo Abayomi-Alli. A probabilistic clustering model for hate speech classification in twitter. Expert Systems with Applications, 173:114762, 2021.
3. Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. In Proceedings of the 26th inter- national conference on World Wide Web companion, pages 759–760, 2017.
4. Junyi Chen, Shankai Yan, and Ka-Chun Wong. Ver- bal aggression detection on twitter comments: convolutional neural network for short-text sentiment analysis. Neural Computing and Applications, 32:10809–10818, 2020.
5. Gloria del Valle-Cano, Lara Quijano-Sa´nchez, Federico Liberatore, and Jesu´s Go´mez. Socialhaterbert: A dichotomous approach for automatically detecting hate speech on twitter through textual analysis and user pro- files. Expert Systems with Applications, 216:119446, 2023.
6. Roge´rio P Esp´ındola and Nelson FF Ebecken. On ex- tending f-measure and g-mean metrics to multi-class problems. WIT Transactions on Information and Communication Technologies, 35:25–34, 2005.
7. RedvanGhasemlounia, Amin Gharehbaghi, Farshad Ahmadi, and Hamid Saadatnejadgharahassanlou. Developing a novel framework for forecasting groundwater level fluctuations using bi-directional long short- term memory (biLSTM) deep neural network. Computers and Electronics in Agriculture, 191:106568, 2021.
8. Ernesto Lee, Furqan Rustam, Patrick Bernard Washington, Fatima El Barakaz, Wajdi Aljedaani, and Imran Ashraf. Racism detection by analyzing differential opinions through sentiment analysis of tweets using stacked ensemble gcr-nn model. IEEE Access, 10:9717–9728, 2022.
9. Tomas Mikolov, AnoopDeoras, Stefan Kombrink, Lukas Burget, and Jan Cernocky`. Empirical evaluation and combination of advanced language modeling techniques. In Interspeech, number s 1, pages 605–608, 2011.
10. Changrong Min, Hongfei Lin, Ximing Li, He Zhao, Junyu Lu, Liang Yang, and Bo Xu. Finding hate speech with auxiliary emotion detection from self- training multi-label learning perspective. Information Fusion, 96:214–223, 2023.
11. M Pavithra, K Saruladha, and K Sathyabama. GRU based deep learning model for prognosis prediction of disease progression. pages 840–844, 2019.
12. Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representa- tion. pages 1532–1543, 2014.