Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 9, No. 1, February, 2025

Unlocking Potential Score Insights of Sentimental Analysis with a Deep Learning Revolutionizes

Ibrahim R. Alzahrani [1*]

[1] *College of Computer Science and Engineering, University of Hafr Al Batin, Al Jamiah, Hafar Al Batin 39524, Saudi Arabia.*

Abstract

Online hate has emerged as a rapidly growing issue worldwide, often stemming from differences in opinion. It is crucial to use appropriate language and words on social media platforms, as inappropriate communication can negatively impact others. Consequently, detecting hate speech is of significant importance. While manual methods are commonly employed to identify hate and offensive content on social media, they are time-consuming, labor-intensive, and prone to errors. Therefore, AI-based approaches are increasingly being adopted for the effective classification of hate and offensive speech. The proposed model incorporates various text preprocessing techniques, such as removing extraneous elements like URLs, emojis, and blank spaces. Following preprocessing, tokenization is applied to break down the text into smaller components or tokens. The tokenization technique utilized in this study is TF-IDF (Term Frequency–Inverse Document Frequency). After tokenization, the model performs the classification of hate and offensive speech using the proposed BiLSTM-based SM-CJ (Scalable Multi-Channel Joint) framework. The BiLSTM-based SM-CJ model is particularly effective in detecting hate, offensive, and neutral tweets due to its ability to capture both forward and backward contexts within a given text. Detecting hate speech requires a comprehensive understanding of the text and the identification of patterns spanning across multiple words or phrases. To achieve this, the LSTM component of the BiLSTM model is designed to capture long-term dependencies by utilizing information from earlier parts of the text. The proposed SM-CJ framework further aligns the input sequence lengths fetched from the input layer, enabling the model to focus on specific segments of the input sequence that are most relevant for hate speech detection. This approach allows the model to accurately capture derogatory language, and subtle nuances present in hate speech. Finally, the performance of the proposed framework is evaluated using various metrics, including accuracy, recall, F1-score, and precision. The results are compared with state-of-the-art approaches, demonstrating the effectiveness of the proposed model.

Keywords:

Hate Speech;

Offensive Speech;

Classification;

Tokenization;

Bidirectional Long Short-Term Memory (BiLSTM);

Pre-Processing.

Article History:

Received:	24	July	2024
Revised:	13	November	2024
Accepted:	06	December	2024
Published:	01	February	2025

1- Introduction

In recent years, the rapid growth of information flow due to the rise of the internet has been remarkable. As internet usage increases, so does the prevalence of hateful verbal communication, particularly on social media platforms such as Twitter. In India alone, more than 24.45 million people are Twitter users [1, 2], and this surge in activity can partly be attributed to the freedom of speech and the right to express opinions on various topics. While social media and other community forums allow users to share their views on a wide range of subjects, differences in opinion often lead to the use of hateful and offensive language [3]. Hate speech and offensive comments can be complex and harmful, as they frequently target specific individuals or groups. Such toxic discourse can escalate tensions and disputes between communities worldwide. Furthermore, hate speech and offensive remarks on online platforms can have severe consequences on individuals' well-being, potentially leading to mental health issues such as depression and, in extreme

* **CONTACT**: ialzahrani@uhb.edu.sa

DOI: http://dx.doi.org/10.28991/ESJ-2025-09-01-03

cases, even suicide [4]. Given these risks, the automatic detection of hate speech plays a critical role in identifying harmful content in tweets and other social media posts [5]. Social media platforms are increasingly focusing on employing advanced techniques to detect negative sentiments in speech and mitigate content that could adversely impact society.

In the past, various manual techniques were employed to identify hate speech on social media platforms. These methods typically involved human reviewers who examined and assessed content to determine whether it contained hateful or offensive language [6]. Reviewers followed specific guidelines provided by the platform and often conducted contextual analyses to assess whether a tweet qualified as hate speech. This assessment considered factors such as tone, intent, and the potential harm caused by the speech. While these manual techniques were effective to some extent, they had significant limitations. Some of the main drawbacks included the time-consuming nature of the process [7], challenges with scalability, and inefficiency, as reviewing large volumes of content could be resource-intensive and difficult for human reviewers to handle accurately. Additionally, constantly reviewing offensive and hateful content could take an emotional toll on reviewers, leading to psychological burnout [8]. To address these challenges, AI-enabled techniques have been adopted to improve the efficiency [9, 10] and accuracy of hate speech detection [11]. AI algorithms can process massive amounts of data in real-time, enabling the quick analysis of large numbers of tweets. Furthermore, the use of AI significantly reduces the reliance on human resources, making it a cost-effective solution for implementing robust hate speech detection models.

Various studies have employed different approaches for the detection of hate speech, offensive language, and neutral content. For instance, optimized classifiers, ensemble classifiers, and multi-tier meta-learning classifiers [12] have been used to categorize tweets into these three categories. Among these, the multi-tier meta-learning model demonstrated its effectiveness in recognizing hate and offensive comments, though its accuracy was limited to 67%. Similarly, the CNN-based "HateClassify" method [13] was utilized to label social media content as hate, offensive, or neutral. This method leverages a crowdsourcing technique, allowing social media users to vote on speech content they deem hateful or offensive. Additionally, Ayo et al. [14] proposed an approach using an improved cuckoo search neural network (NN) to detect hate speech on Twitter. Their method employed a hybrid approach combining TF-IDF and LSTM for sentence-level feature extraction, followed by classification using the improved cuckoo search NN, which identified tweets as hate, offensive, or neutral.

A three-layered deep learning (DL) technique has also been used for monitoring, detecting, and visualizing hate speech incidents in Twitter messages. This approach combines CNN and RNN to automatically learn abstract feature representations as input data passes through various weighted layers. While the model produced better visualization results, its complexity was relatively high. Similarly, an SVM-based model has been used to classify sentiments in Indonesian tweets [15]. The authors tested the model with two-class and three-class sentiment datasets. However, the SVM model was limited in handling three-class sentiment analysis (SA). Moreover, the study lacked proper pre-processing techniques, leading to higher computational resource requirements. In another study, Siddiqua et al. [16] employed a unified neural network (NN) to detect whether tweets contained hate speech. However, the model faced challenges with underfitting, reducing its efficiency in accurately detecting hate messages.

While existing models have achieved notable results in classifying hate, offensive, and neutral speech, they face several significant challenges. These include low accuracy [12], reliance on binary classification (hate and non-hate), high complexity, insufficient pre-processing techniques [15], underfitting issues [16], and computational inefficiency [17]. These limitations must be addressed to develop a more effective and reliable hate speech detection model. To overcome these challenges, the proposed model employs a BiLSTM-based SM-CJ framework for precise detection of hate, offensive, and neutral tweets. The BiLSTM architecture is particularly effective due to its ability to capture both forward and backward contexts of a given text. Detecting hate speech requires an understanding of the entire text and identifying patterns that span across multiple words or phrases. To achieve this, the LSTM component of the BiLSTM model is designed to capture long-term dependencies by utilizing information from earlier parts of the text. Moreover, the proposed model incorporates multiple attention layers. These layers enable the model to focus on specific parts of the input sequence that are most relevant to hate speech detection. This design allows the model to better capture derogatory language and other nuances present in hate speech.

The primary objectives of the proposed work are as follows:

- To employ advanced pre-processing techniques, including the removal of numbers, punctuation, special characters, and stop words, ensuring clean and structured input data.
- To classify hate, offensive, and neutral tweets using the BiLSTM-based SM-CJ model, improving the accuracy and reliability of hate speech classification.
- To evaluate the model's performance using various metrics, such as accuracy, recall rate, F1-score, and precision, to validate its efficacy.

1-1- Paper Organization

Section II provides a review of existing research, highlighting the identified challenges and limitations. Section III details the proposed techniques, including their workflow and relevant mathematical derivations. Section IV presents the results obtained from simulating the proposed approach. Finally, Section V concludes the study with future recommendations.

2- Literature Review

The following section reviews various studies on the classification of hate speech, offensive content, and neutral language.

Hate speech detection has become increasingly important as hate-filled communication on social media platforms, such as Twitter, is rapidly growing [18]. For this purpose, a study utilized a Random Forest (RF) algorithm to classify social media posts on Twitter into two binary classes: hate and non-hate [19]. Numerical features were extracted using the TF-IDF technique. Although the model demonstrated promising performance, future studies aim to enhance its effectiveness and generalization by addressing linguistic challenges. Similarly, a combination of three machine learning (ML) algorithms—RF, Support Vector Machine (SVM), and an ensemble model—integrated with data augmentation techniques, was applied to detect hate speech in Indonesian tweets [20]. The results showed that the ensemble method achieved an 11% higher evaluation score when using a balanced dataset. Furthermore, Improved Principal Component Analysis (IPCA) and Modified Convolutional Neural Networks (MCNN) [21] have been employed to classify text data from Twitter accurately. While the model provided reasonable accuracy, it lacks the capability to handle large datasets efficiently. Additionally, ML models such as SVM, Naive Bayes, Decision Trees, RF, Logistic Regression, and deep learning (DL) models like CNN, LSTM, and the BERT pre-trained transformer model were used for hate speech detection on Twitter datasets [22]. Among these approaches, the BERT model demonstrated superior performance compared to other ML and DL methods. This highlights that DL-based models generally outperform traditional ML approaches.

Moreover, a hybrid DL model combining BiLSTM and CNN has been employed for hate speech classification in textual data. This model integrates GloVe-based word embedding, dropout, L2 regularization, and global max pooling, achieving impressive results. In another study, d'Sa et al. [7] focused on detecting and removing hate speech from Twitter. The study aimed to classify tweets into three categories: hate, offensive, and neutral. Feature-based (FB) techniques and fine-tuning approaches were utilized. The FB technique used sequences of word embeddings as input, while the fine-tuning approach employed the pre-trained BERT model for hate speech classification. The results indicated that BERT fine-tuning outperformed FB techniques. However, the study noted that some hate speech tweets were misclassified as "neutral" due to the absence of explicit hate words or implicit hate speech. Further research is necessary to address these errors.

Similarly, Saleh et al. [23] used the BERT model to detect hate-related abbreviations, terms, and misspelled words. In addition to BERT, the study employed feature extraction (FE) methods for tweet classification into hate and non-hate categories. For FE, domain-specific word embeddings, Google Word2Vec, and GloVe were used, while a BiLSTM classifier was applied for classification. Although the model achieved notable results, future work aims to extend its capabilities to multi-class hate speech detection.

Hate speech on social media is increasing at an alarming rate, causing significant negative impacts on society. As a result, various artificial intelligence (AI) models have been developed to detect hate speech efficiently. For instance, Ojo et al. [24] implemented a 1D-CNN model for binary classification of hate and non-hate speech. Alongside this, models such as Naive Bayes (NB), Random Forest Classifier (RFC), Support Vector Machine (SVM), Logistic Regression Model (LRM), and 1D-CNN were evaluated. Among them, the 1D-CNN combined with GloVe word embeddings delivered the best results. However, the GloVe embeddings struggled to classify the test dataset effectively due to the limited number of training sentences available. Similarly, Bisht et al. [25] emphasized the classification of hate, offensive, and neutral speech on Twitter in their proposed approach. To improve the classification process, word embedding techniques and a multi-step classifier were employed. The process began with data preprocessing, which included removing stop words, special characters, and punctuation, as well as performing stemming and tokenization. The processed words were then embedded using word embedding techniques. Ultimately, the classification was performed using an LSTM/Bi-LSTM-based classifier, achieving an accuracy of 86% in hate speech detection.

Das et al. [26] used a CNN model with three input layers, an output layer, and 31 units in a hidden layer to classify hate, offensive, and neutral tweets. The dataset consisted of 12,000 tweets, and the model achieved an overall accuracy of 73% by leveraging sentiment features and a 4-gram model. Additionally, machine learning classifiers such as NB and SVM were utilized by another study [27] for hate and non-hate speech detection. The experimental results revealed that SVM outperformed the NB model, which had an accuracy rate of only 50%. Although SVM achieved better performance, other machine learning techniques such as ANN, KNN, RF, and additional models need to be compared to identify the most effective model for hate speech detection. A key limitation of these approaches was their inability to process data in real time.

In another study, CNN was used for classifying hateful, offensive, and neutral tweets [28]. This approach incorporated a factorization technique based on a continuously updated, crowd-sourced dictionary of hate words. The CNN model employed had four hidden layers, three convolutional layers, and one final linear layer. Despite its promise, the model's low accuracy remained a significant limitation, and future improvements could be made by employing alternative techniques. Furthermore, a study by Wani et al. [29] used a decision tree (DT) classifier with bi-grams and a Bag of Words (BOW) vectorizer. To enhance classification accuracy, CNN and LSTM models were later incorporated. The experimental results indicated that the LSTM model with Word2Vec embeddings delivered satisfactory classification outcomes for hate speech detection.

Distinguishing between hate speech and offensive language is a key challenge in eliminating toxic content. To address this, Touahri & Mazroui [30] employed a deep learning (DL) approach for effectively classifying tweets as hate speech, offensive, or neutral. They used a public dataset consisting of various tweets for their study. Initially, a BiLSTM model was pre-trained with GloVe embeddings, followed by transfer learning (TL) using BERT, GPT-2, and DistillBERT for hate speech detection. The study's results indicated that a confusion matrix and cost-benefit model were applied to quantify and assess the model errors. Similarly, BiLSTM and LSTM models [31] were used for binary classification of hate and non-hate speech. The results showed that the BiLSTM model achieved a better recall rate than the LSTM model; however, the LSTM model outperformed BiLSTM in terms of accuracy, precision, and F1 score. Despite the LSTM model's superior accuracy, the BiLSTM model had a better recall rate, which indicates a lower error rate in detecting positive classes. This suggests that the BiLSTM model has a slight edge over the LSTM model. Nonetheless, the implementation of an attention model is recommended for future work, as it has shown significant benefits in natural language processing (NLP) applications.

Machine learning (ML) techniques were also applied for classifying hate, offensive, and neutral tweets from Twitter, using n-gram features with TF-IDF values. Various ML algorithms, such as SVM, Naive Bayes (NB), and Logistic Regression (LR), were tested on different feature sets [32]. The experimental results revealed that LR performed better with an ideal n-gram range of 1 to 3 compared to the other models. The study found that 4.8% of offensive tweets were misclassified as hate speech. This issue could be addressed by incorporating more examples of offensive tweets that do not contain hateful language. A drawback of this model is its failure to account for negative words in a sentence. Furthermore, Zhou et al. [33] used ELMo (Embedding from Language Models), CNN, and BERT models for recognizing hate speech. They suggested that the model's performance could be improved by using a fusion approach, which enhances the model's efficacy in hate speech detection. This approach was applied to the SemEval Task 5 dataset, and further improvements could be made by replacing the basic word embeddings in the CNN with more advanced embedding techniques.

Ensemble methods have also been utilized for hate speech detection. For example, DeL-haTE [34], an ensemble method, combined layers of GRU and CNN models. The CNN layer extracted higher-order features from the word embedding matrix, while the GRU model captured features from the sequence of words. The use of these combined features enabled the automatic detection of hate speech, improving the model's overall performance.

2-1- Gaps Identified

From the review of existing studies, several significant concerns have been identified:

- While the model has shown improved results, there is still a possibility that some "hate speech" tweets may be misclassified as "neither," particularly due to the absence of explicit hate words or the presence of implicit hate speech. Therefore, a more in-depth analysis is necessary in future studies to address these errors [7].
- Despite its overall performance, a major limitation of the model is its relatively low accuracy. The model's performance could be enhanced by incorporating alternative techniques [28]. Similarly, other studies have reported similarly low accuracy levels [25, 26].
- Although the model demonstrates efficacy, the implementation of attention mechanisms is recommended for future work, as they have proven beneficial in natural language processing (NLP) applications [31].
- Some studies have focused exclusively on binary classification, distinguishing between hate speech and non-hate speech [24, 31].

3- Proposed Methodology

Hate speech detection is crucial for protecting individuals and communities from harm and discrimination. It can create a hostile and toxic environment on social media platforms like Twitter. Despite the community guidelines against hate speech implemented by Twitter, some users violate these rules and post hateful content. As a result, hate speech detection becomes essential. However, manual methods for detecting hate speech can be tedious and time-consuming. To address this, AI-based techniques are employed for more efficient detection. Nonetheless, existing AI models often fall short in terms of accuracy and effectiveness. Therefore, the proposed model aims to improve the efficiency and reliability of hate speech detection. Figure 1 illustrates the overall mechanism involved in the process.

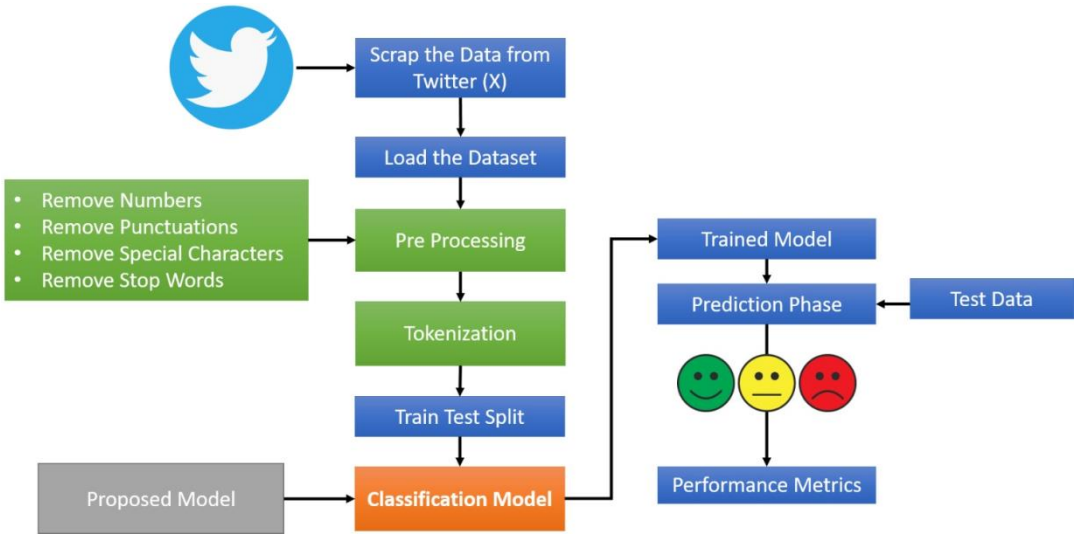

Figure 1. Overall Proposed Model

Figure 1 illustrates the process involved in hate speech classification. The process begins with loading the dataset. After loading the dataset, various pre-processing techniques are applied to the text. One of these techniques, the removal of numbers, focuses on eliminating numerical values, allowing the model to focus more on the textual content rather than numbers. Once the text is pre-processed, tokenization is performed, where the text is broken down into tokens to simplify the analysis and manipulation of the content. The tokenized text is further processed using the TF-IDF technique.

After tokenization, the text is classified using the BiLSTM-based SM-CJ model, which incorporates multiple attention layers to improve the model's effectiveness in classifying hate and offensive speech. The proposed BiLSTM-based SM-CJ model leverages BiLSTM because of its ability to capture both forward and backward contexts within the text. Hate speech detection requires understanding the entire text and recognizing patterns that span multiple words or phrases. To achieve this, the LSTM component of the BiLSTM model is designed to capture long-term dependencies by using information from earlier parts of the text, which enhances the model's learning ability. Additionally, the proposed model incorporates multiple attention layers, which allow the model to focus on specific parts of the input sequence that are most relevant to hate speech detection. Finally, the predicted sentiment is displayed using corresponding emojis, and the effectiveness of the proposed framework is evaluated using various metrics.

3-1- Pre-processing

Text pre-processing refers to the steps taken to clean and transform raw data before it can be used in a model. The proposed framework focuses on pre-processing the text using various techniques, including the removal of numbers, punctuation, special characters, and stop words.

Removal of Numbers: This step involves eliminating numerical values, which shifts the focus toward the textual content rather than numerical data. This helps the classification model capture the linguistic patterns more effectively.

Removal of Punctuation: By removing punctuation marks, the text is simplified and the noise is reduced. This allows the model to concentrate on important words and phrases that contribute to the classification of hate, offensive, and neutral content.

Removal of Stop Words: Stop words, such as "is," "the," "and," and others, are frequently occurring words that do not add significant meaning to the text in terms of classifying hate speech. Removing these stop words reduces the dimensionality of the input and eliminates unnecessary noise, which enhances the classification process. In the proposed model, HTTP entities, URLs, ampersands, user tags, and noisy symbols like "!", ",", and "`" are also removed from the text.

Removal of Special Characters: Special characters such as symbols and hashtags do not provide substantial information for hate speech detection. Removing these characters helps standardize the text and reduces the complexity of the input data.

These are some of the techniques used for pre-processing the text. After the pre-processing steps, the tokenization process takes place, where the text is split into individual tokens or words. Tokenization breaks down the text into its constituent components, which is essential for in-depth analysis. The next subsection will explore the deeper process of tokenization.

3-2- Tokenization

Tokenization is the process of breaking down raw text into smaller units, such as sentences or words, known as tokens. These tokens help in understanding the context, which improves the classification of hate speech. Tokenization also aids in replacing sensitive elements of data with non-sensitive ones. In the proposed model, the TF-IDF technique is used for tokenization.

- **TF (Term Frequency)** calculates the frequency of each token within a document. It is determined by counting how often each token appears in the document. The TF step assigns a numerical value to each token based on its frequency within the document.
- **IDF (Inverse Document Frequency)** calculates the significance of each token across the entire dataset. This step assigns a numerical value to each token depending on its uniqueness in the dataset.

The final TF-IDF value for each token is obtained by multiplying its TF value by its IDF value. This process results in a numerical value that represents the importance of each token in the context of the dataset. The TF-IDF model is crucial for text classification, as it emphasizes frequency rather than semantic relationships between words.

3-3- Classification – A Scalable Multi Channel Joint Architecture

The text, after tokenization, is fed into the Bi-LSTM model incorporated in the proposed framework, as it offers better performance in classification compared to the conventional LSTM model. Bi-LSTM uses two LSTM layers on the input data, allowing it to capture information from both past and future time steps. This ability enables Bi-LSTM to capture more context and dependencies within the input sequence. In contrast, LSTM is a sequential function that can lack robustness in classifying hate, offensive, and neutral speech. LSTMs are designed to capture information within a specific range, but they may struggle to capture long-range dependencies, especially in Twitter comments. This limitation can restrict the model's ability to comprehend the full context of a given text, potentially leading to inaccurate detection of hate speech. Therefore, the proposed model employs the Bi-LSTM architecture, which replicates the first recurrent layer in the network. The input is provided to this layer in its original form, and Bi-LSTM is trained on both forward and backward information over a given period.

Bi-LSTM processes input data in two directions using the Forward Layer (FL) and the Backward Layer (BL). Each LSTM layer includes a memory cell that stores information over time, as well as gates that control the flow of information. The outputs from both LSTM layers are then concatenated to capture both past and future context. Figure 2 illustrates the process of Bi-LSTM for hate speech detection classification.

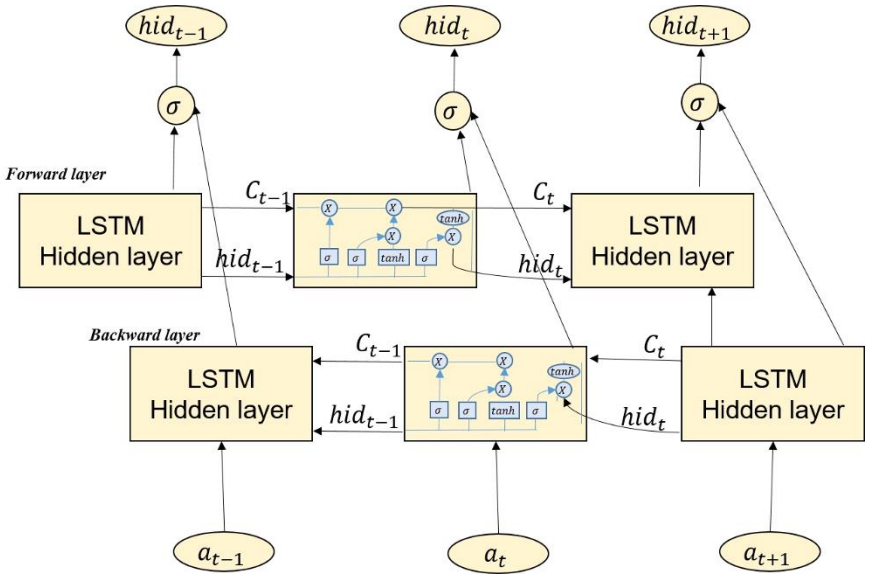

Figure 2. Bi-LSTM architecture

Figure 2 illustrates the process involved in BiLSTM, where the Forward Layer (FL) and Backward Layer (BL) are used to capture information from both preceding and subsequent words in the input sequence. The FL processes the input sequence in a forward direction, capturing the dependencies and context of each word based on the preceding words in the sequence. This enables the model to understand the sequential information and patterns in the input. At the same time, the BL processes the input sequence in reverse, capturing the dependencies and context of each word based on the succeeding words in the sequence. By combining the outputs from both the FL and BL, the BiLSTM model effectively captures a more comprehensive understanding of the input sequence. The relationship between the inputs and outputs is shown in Equation 1:

$$i_t = \sigma(Weg_{ai}\, a_t + Weg_{hidi}\, hid_{t-1} + Weg_{ci}\, c_{t-1} + bi_i) \quad (1)$$

where c is denoted as the cell state, Weg is denoted as the weights of the parameter, b is represented as the biases parameter and eventually, a is represented as the input value.

$$f_t = \sigma(Weg_{af}\, a_t + Weg_{hidf}\, hid_{t-1} + Weg_{cf}\, c_{t-1} + bi_f) \quad (2)$$

$$c_t = f_t\, c_{t-1} + i_t \tanh(Weg_{ac}\, a_t + Weg_{hidc} hid_{t-1} + bi_c) \quad (3)$$

$$o_t = \sigma(Weg_{ao} a_t + Weg_{hido} hid_{t-1} + Weg_{co} c_{t-1} + bi_o \quad (4)$$

$$hid_t = o_t\, tanh(c_t) \quad (5)$$

Likewise, from equations, c persuades new information in the cell state f_t is represented as the forget function, which sieves out the irrelevant information, i_t is represented as the input gate, o_t is denoted as the output gate, where it yield significant information and eventually, hid is represented as the output value. The output obtained is forwarded for obtaining forward hidden layer, backward hidden layer and output value.

$$(\overrightarrow{hid_t}) = (\, Weg_{\overrightarrow{ahid}}\, a_t + Weg_{\overrightarrow{hid}\,\overrightarrow{hid}} \overrightarrow{hid}_{t-1} + bi_{\overrightarrow{hid}}) \quad (6)$$

$$(\overleftarrow{hid}_t = H\, (\, Weg_{\overleftarrow{ahid}}\, a_t + Weg_{\overleftarrow{hid}\,\overleftarrow{hid}} \overleftarrow{hid}_{t+1} + bi_{\overleftarrow{hid}} \quad (7)$$

$$y_t = Weg_{\overleftrightarrow{hidy}}\, \overleftarrow{hid}_t + Weg_{\overleftrightarrow{hidy}}\, \overleftarrow{hid_t} + bi_y \quad (8)$$

In which, Weg denotes the weight matrices, Likewise, $Weg_{\overrightarrow{ahid}}$ is denoted as the forward input hidden weight and $Weg_{\overleftarrow{ahid}}$ is represented as the backward input hidden weight matrices. Similarly, bias vector in the equation is denoted by using b and finally, hidden layer of the model is represented by using H. In BiLSTM model, attention layers called scalable multi-channel joint architecture is used for analyzing the sentiments present in the text that is, identifying if the provided text is hate or offensive or neither. This SM-CJ architecture extracts both original context features and multi-scale high level context features. Figure 3 shows the model for multi attention layers incorporated in the proposed BiLSTM model known as SM-CJ for predicting the sentiments given in the text.

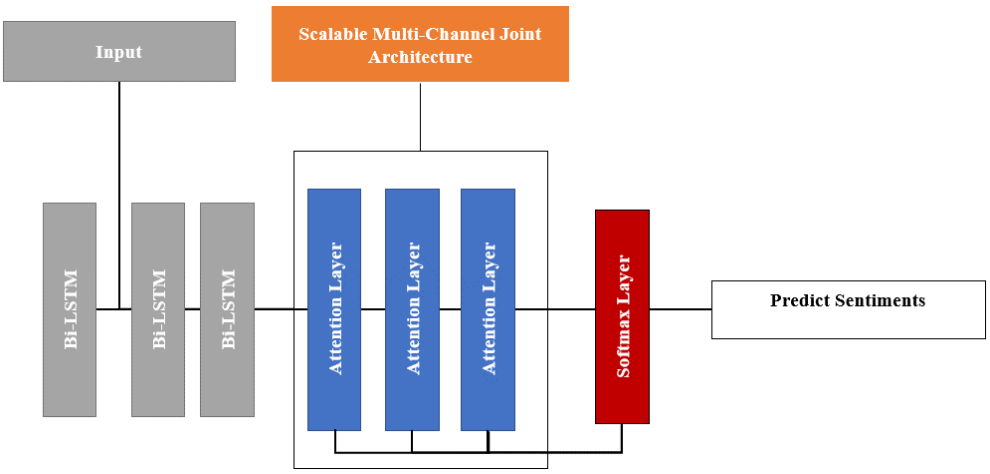

Figure 3. Proposed SM-CJ model

Figure 3 shows the process carried out by the model for predicting the sentiments of the tweets by using different multi attention layers known as SM-CJ. Multiple attention layer provides additional benefits for the proposed model, thereby making the meaning of the sentence more scalable. Initially, the proposed SM-CJ aids in aligning the length of the input sequences fetched from the input layer. This can be accomplished by using the Equation 9,

$$T = w_1 \oplus w_2 \oplus \cdots \oplus w_t \oplus \cdots \oplus w_k,\ t = 1, 2, \ldots, k \quad (9)$$

where $\oplus$ signifies joint mark w_1 is defined as the tth word of the T. Then, the new vector is represented by using Equation 10:

$$A = a_1 \oplus a_2 \oplus \cdots \oplus a_t \oplus \cdots \oplus a_k \quad (10)$$

$$A = \psi\left(conv\,(Weg_{ck}, T, r)\right) \quad (11)$$

In which, Weg_{ck} is defined as Weight of kernel and ψ is defined as hyperbolic unit. The obtained hyperbolic unit is represented as:

$$\psi(a) = \begin{cases} a, & a > 0 \\ \frac{\alpha a}{1-a}, & a \leqslant 0, \end{cases} \quad (12)$$

where, α is defined as the hyperparameter. After this process, Bi-LSTM with hidden neuron size is utilized for capturing the high-level context information that is glued by using forward LSTM layer and backward LSTM layer.

$$f_t = \sigma\left(Weg_f\left[hid_{t-1}, a_t\right] + b_f\right) \quad (13)$$

In Equation 14, f_t is denoted as the forget gate, which determines what information need to be removed through a_t and hid_{t-1}.

$$i_t = \sigma\left(Weg_i\left[hid_{t-1}, a_t\right] + b_i\right), \quad (14)$$

$$\tilde{c}_t = tanh\left(Weg_c\left[hid_{t-1}, a_t\right] + bi_c\right), \quad (15)$$

From the equations, the input gate of LSTM is denoted as i_t . This helps in deciding which information need to be updated via through a_t and hid_{t-1}. Further, the candidate cell $\tilde{c}$ is attained by using the Equation 16,

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t, \quad (16)$$

Here, c_{t-1} denotes the old cell information and information which needs to forget through f_t is defined as c_{t-1}. $\tilde{c}_t$ decides which information needs to updated via input gate and eventually, new cell information is obtained by using c_t

$$o_t = \sigma\left(Weg_o\left[hid_{t-1}, a_t\right] + bi_o\right), \quad (17)$$

$$hid_t = o_t \times tanh\left(c_t\right), \quad (18)$$

Output gate of LSTM is defined by using o_t. Then, the output of the LSTM cell h_t can be contracted by cell information and output gate. Eventually, the BiLSTM output B is denoted as,

$$B = bi_1 \oplus bi_2 \oplus \cdots \oplus bi_t \oplus \cdots \oplus bi_k, \quad (19)$$

$$bi_t = [hid_t^f, hid_t^{bi}], \quad (20)$$

Here, hid_t^{bi} is denoted as output of tth cell in backward LSTM and hid_t^f is represented as the output of tth cell in forward LSTM. Additionally, attention mechanism is the proposed model for assigning different weights to output features of BiLSTM and the output is preserved as single channel coding features. The equations are depicted as follows,

$$K = tanh\left(Weg_k B^T\right) \quad (21)$$

$$A = softmax\left(Weg_a K\right) \quad (22)$$

$$M = AB, \quad (23)$$

In which, B and M is denoted as the output of the AL (Attention Layer) AL, B^T is denoted as transpose matrix and Weg_a and Weg_k is represented as trainable parameter. The above-mentioned mathematical expression has focused on standard structure of a single channel. However, multi-channel is built. This helps in mining wide variety of sentiment features and the output of n channels are fused using,

$$M^{multi} = M^{r=1} + M^{r=2} + \cdots + M^{r=n} \quad (24)$$

$$\widehat{M} = M^{ori} + M^{multi} \quad (25)$$

Here, M^{ori} examines in denoting the output of the channel and eventually the output of the model is described by using Equation 25. After constructing a multi-channel model, sentiment decoder is utilized for decoding the sentiment features which are generated by using multi-channel encoder. In this model, attention mechanism is used for filtering the features and assigning the weights, which is depicted in the equations,

$$\tilde{C}_i = conv_{1\times1}\left(Weg_i^C, \widehat{M}\right) \quad (26)$$

$$Q_i = \tilde{C}_i\, Weg_i^Q \quad (27)$$

$$K_i = \tilde{C}_i\, Weg_i^K \quad (28)$$

$$V_i = \tilde{C}_i \, Weg_i^V \tag{29}$$

$$\widetilde{M}_i = softmax\left(\frac{Q_i K_i^T}{\sqrt{d_i}}\right) V_i \tag{30}$$

In which, input of sentiment decoder is represented by using variable $\widehat{M}$ and output is represented as $\widetilde{M}_i$. In the sentiment decoder, concatenation layer and global attention spaces are used for fusing the output of global attention spaces.

$$M\tilde{} = [\widetilde{M}_1, \widetilde{M}_2, \ldots, \widetilde{M}_i, \ldots, \widetilde{M}_n] \tag{31}$$

However, in order to tweak the attention distribution of the AM (Attention Module) a technique called regularization is used. Regularization helps in preventing the idleness of weighted sentiment features. This can be accomplished by using Equation 32,

$$H = softmax\ (\widetilde{M}^T\ \widetilde{M}) \tag{32}$$

$$\tilde{r} = \sum_{i=1}^{d_H} \sum_{j=1}^{d_H} \frac{(hid_{ij} - e_{ij})^2}{d_H} \tag{33}$$

$\widetilde{M}$ is the output of the global AM, H is defined as the matrix of d_H X d_H, hid_{ij} is represented as the value of j^{th} column and i^{th} row of H. then e_{ij} is defined as the value of i^{th} row and j^{th} column of the matrix. Then, Kmodule is utilized for filtering the appropriate information and for choosing the right features. Moreover, k helps reducing the parameters of the model and assist in preventing the overfitting of the model. Thus, k is represented as,

$$\mathrm{k} = \mathrm{o}\ (\mathrm{INT}\ (\mathrm{leg_m}) + \mathrm{INT}\ (\tfrac{\mathrm{leg_i}}{10})), \tag{34}$$

Here, length of the input is denoted using I_i, $\mathrm{leg_m}$ is denoted as the average distance of the text and $INT(.)$ is denoted as the rounding function,

$$o(X) = \begin{cases} \mathrm{x}, \mathrm{f}\ (\mathrm{x}) \leqslant \delta, \\ \gamma\ , \mathrm{f}\ (\mathrm{x}) > \delta, \end{cases} \tag{35}$$

Number of features are defined by $f(.)$ and the threshold is denoted as δ. After Sentiment decoder, sentiment classifier is used for judging the sentiments. In order to proceed this, softmax layer activation function is utilized.

$$P = softmax\ (tanh\left(\widetilde{M}^{decoder}\ W_1 + B_1\right)W_2 + B_2), \tag{36}$$

where, W_1 and W_2 is denoted as weights and B_1 and B_2 represents bias parameters and $\widetilde{M}^{decoder}$ represents the output of K- module. Finally, cross entropy function is used as the rudimentary loss function and for training and evaluating the class weights for upholding the balance of the training process. Thus, the adaptive weighted loss function is defined using, Equation 37,

$$\mathrm{L} = \beta \times\ \mathrm{L^{ce}} + (1 - \beta) \times \mathrm{L^{wce}} \tag{37}$$

$$= -\beta \times \tfrac{1}{\mathrm{n}} \textstyle\sum_{\mathrm{i}=1}^{n} \mathrm{y_i} \log \mathrm{y_i^p} - (1 - \beta) \times \tfrac{1}{\mathrm{n}} \textstyle\sum_{\mathrm{i}=1}^{n} \mathrm{w_i y_i} \log \mathrm{y_i^p} \tag{38}$$

$$= -\tfrac{1}{\mathrm{n}} \textstyle\sum_{i=1}^{n} (\beta + (1 - \beta) \times \mathrm{w_i}) \times \mathrm{y_i} \log \mathrm{y_i^p} \tag{39}$$

where $\mathrm{L^{wce}}$ is defined as the weighted cross entropy loss and $\mathrm{L^{ce}}$ is denoted as the cross-entropy loss. β is defined as the harmonic factor and n is denoted as number of classes:

$$\mathrm{w_i} = \mathrm{w_i^t} \times \mathrm{w_i^e} \tag{40}$$

$$\mathrm{w_i^t} = \min\ (\tilde{\mathrm{w}}_\mathrm{i}^\mathrm{t}, \wp) \tag{41}$$

$$\mathrm{w_i^t} = \frac{\sum_{i=1}^{n} \mathrm{N_i^t}}{\mathrm{n} \times \mathrm{N_i^t}} \tag{42}$$

$$\mathrm{w_i^e} = \frac{\hat{\mathrm{w}}_\mathrm{i}^e}{\min\ (\hat{\mathrm{w}}_1^\mathrm{e}, \ldots, \hat{\mathrm{w}}_\mathrm{n}^\mathrm{e})} \tag{43}$$

$$\hat{\mathrm{w}}_\mathrm{i}^e = \exp(\ 1\ - \tfrac{\mathrm{T_i^e}}{\mathrm{N_i^e}}) \tag{44}$$

Here, the terms $\mathrm{w_i^t}$ is denoted as the training class weights and $\mathrm{w_e^t}$ represented as the training class. $\wp$ is denoted as the threshold value, T_i^e is denoted as the correct prediction samples and $\mathrm{N_i^e}$ is denoted as the ith samples. Thus, proposed model aids in correctly predicting the sentiments of the given text with intensely concealed sentiments. This implementation of the proposed work helps in preventing the overfitting of the model. Hence, Figure 4 depicts the overall illustration of then proposed model.

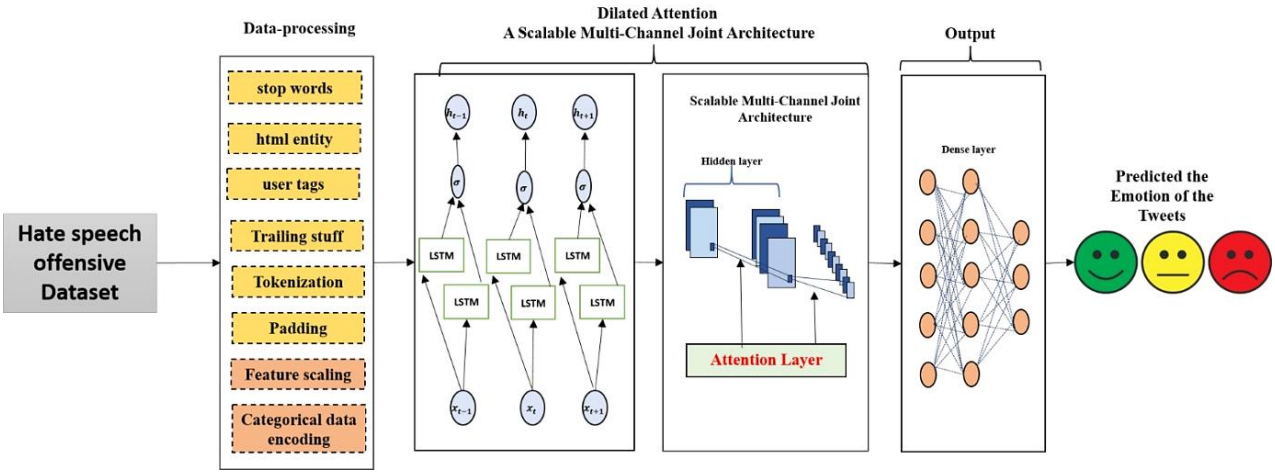

Figure 4. Illustration Diagram

Figure 4 illustrates the overall process involved in the proposed classification of hate speech, offensive language, and neutral content. The process begins with loading the dataset, followed by pre-processing using various techniques. The tweets are then classified based on the severity of the language used, categorizing them as hateful, offensive, or neither. Finally, the model displays an emoji corresponding to the sentiment of each tweet.

4- Results and Discussion

The results obtained from implementing the proposed framework are presented in this section. This includes a description of the dataset, the metrics used to evaluate the effectiveness of the proposed mechanism, performance analysis, and a comparison of the proposed work with existing models.

4-1- Dataset Description

The annotated dataset consists of various tweets classified as hate, offensive, or neutral. Some sample tweets are shown in Table 1.

Table 1. Sample Tweets in dataset

Sl. No.	Class	Tweets
1	Offensive	"@ComedyPosts: Harlem shake is just an excuse to go full retard for 30 seconds."
2	Offensive	"@CoryBandz: having one loyal female is wayyyyy better than having hoes , idc 💯"
3	Offensive	"@GirlThatsVonte: Yall bitches wit no edges be doing the most talking 😩😴✋"😂😂😂
4	Hate	"@MarkRoundtreeJr: LMFAOOOO I HATE BLACK PEOPLE https://t.co/RNvD2nLCDR" This is why there's black people and niggers
5	Neither	"@MotherJones: 10 birds your grandkids may never see, thanks to climate change http://t.co/XqmXHkAsWt http://t.co/RbITeGRnhm" #Climate
6	Neither	"@Theo17100: http://t.co/BYj1HOyhmG" this scally lad would get it"

Table 1 displays various hate, offensive, and neutral words used in tweets. Similarly, the dataset contains a range of hate, offensive, and neutral tweets. Ultimately, the proposed model effectively classifies these tweets based on their content.*

4-2- Performance Metrics

Performance metrics are primarily used to assess the effectiveness and efficiency of the proposed technique. Various metrics are employed to evaluate the model's performance.

4-2-1- Accuracy

Accuracy is considered a measure of the total correct classifications. It is calculated using Equation 45.

$$Acc = \frac{TRN+TRP}{TRN+FLN+TRP+FLP} \quad (45)$$

where, TRN signifies True negative, TRP is True positive, FLN is False negative, and FLP is False positive.

* Dataset Link: https://huggingface.co/datasets/hate_speech_offensive.

4-2-2- Precision

Precision is calculated by measuring the number of correct classifications. It is determined by identifying improper classifications. Equation 46 presents the formula used to calculate precision.

$$precision = \frac{TRP}{FLP+TRP} \tag{46}$$

4-2-3- F-measure

The F-measure, also known as the F1 score, is the weighted harmonic mean of recall and precision. The F1 score is calculated using Equation 47.

$$F1 - score = 2 \times \frac{Rc \times Pc}{Rc+Pc} \tag{47}$$

Here, P is denoted as precision and R is denoted as recall.

4-2-4- Recall

Recall refers to a measure that calculates the total number of correctly identified positive instances across all positive groups. It is evaluated using the formula in Equation 48.

$$R\_c = \frac{TRP}{FLN+TRP} \tag{48}$$

4-3- EDA

This section presents the Exploratory Data Analysis (EDA) conducted using the proposed method. EDA is a technique used to examine and summarize datasets in order to gain insights and understand the underlying patterns and relationships within the data. One of the primary functions of EDA is to identify outliers and anomalies that may skew results or suggest data quality issues. Detecting these irregularities early in the analysis allows for proper handling, ensuring more reliable outcomes. Additionally, EDA can highlight the most relevant features for the analysis, helping to streamline the dataset by removing unnecessary or redundant variables. This focus on significant features can improve model performance by reducing complexity and enhancing interpretability.

Figures 5 to 7 display the key words identified by the model using the proposed technique. Figure 5 shows words recognized as hate speech in tweets, such as "nigga," "shit," "fag," "bitch," "trash," "racist," "bitches," "queer," and others. Similarly, Figure 6 depicts words identified as offensive, including "hoe," "want," "lol," and more. Figure 7 shows neutral words detected in the tweets, indicating that these words are not inherently hateful or offensive. However, neutral words can take on hateful meanings depending on the surrounding context. Furthermore, these neutral words can help in understanding the overall context of a statement. Hate speech often involves subtle or coded language, where neutral terms may be used to indirectly convey hateful sentiments. By recognizing these terms, models can better assess the intent behind the language. Some of the neutral words identified include "kike," "birds," "monkey," "colored," "yellow," "game," and "trash."

Figure 5. Hateful words

Figure 6. Offensive words

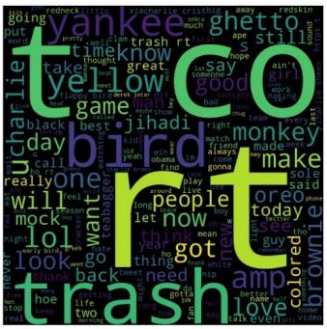

Figure 7. Neutral Words

Figure 8 displays the top 10 words identified in the tweets. By highlighting the most frequently used words, this plot helps identify trending topics, sentiments, or themes prevalent in the tweets. This can be particularly useful during events or discussions that generate significant social media activity. The words include "a," "RT," "bitch," "the," "I," "to," "you," "and," "that," and "my." Among these, the most frequently used word is "a," with a frequency of over 8,000 occurrences.

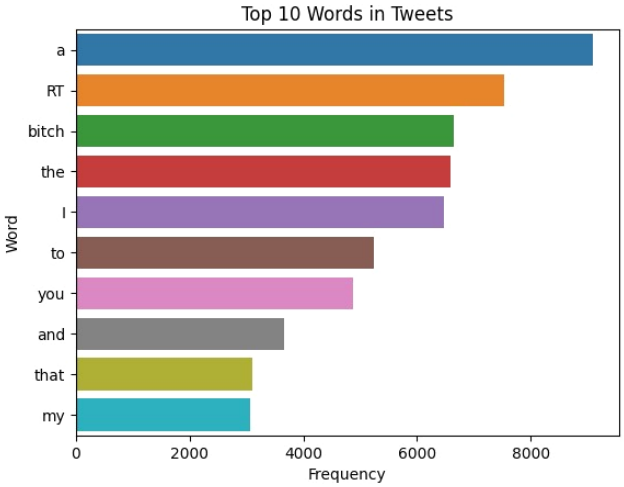

Figure 8. Top 10 Words in Tweets

Figure 9 shows the distribution of classes identified by the proposed model. This plot is essential for visualizing the balance or imbalance of classes in the hate speech detection dataset, where the value 1 represents "hate speech," 2 represents "offensive speech," and 0 represents "neither." From the figure, it is evident that the model has identified more instances of hate speech than offensive or neutral speech. Similarly, Figure 10 illustrates the distribution of counts for each class, with the three distinct classes (0, 1, and 2) represented on the X-axis and the count on the Y-axis. The data points for each class are stacked vertically, indicating that the distribution of counts within each class is consistent and discrete.

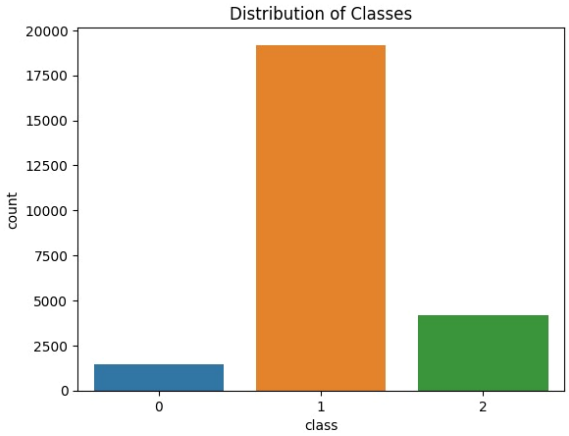

Figure 9. Distribution of classes

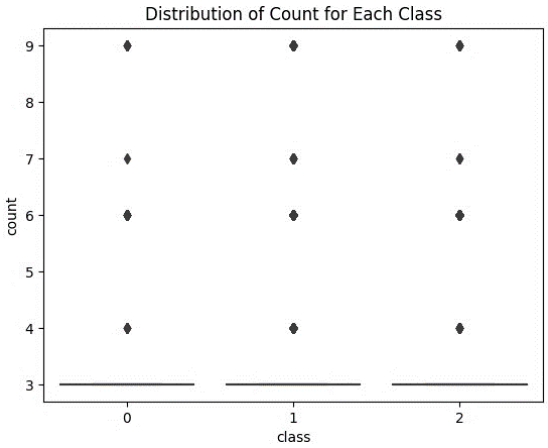

Figure 10. Distribution of count of each class

Figure 11 shows the distribution of hate speech, offensive language, and neutral tweets, with the distribution count for hate speech ranging from 0 to 20,000. For offensive language, the count ranges from 0 to 12,500, and for neutral tweets, it ranges from 0 to 15,000. From these observations, it is evident that the count for hate speech is higher than that for the other categories.

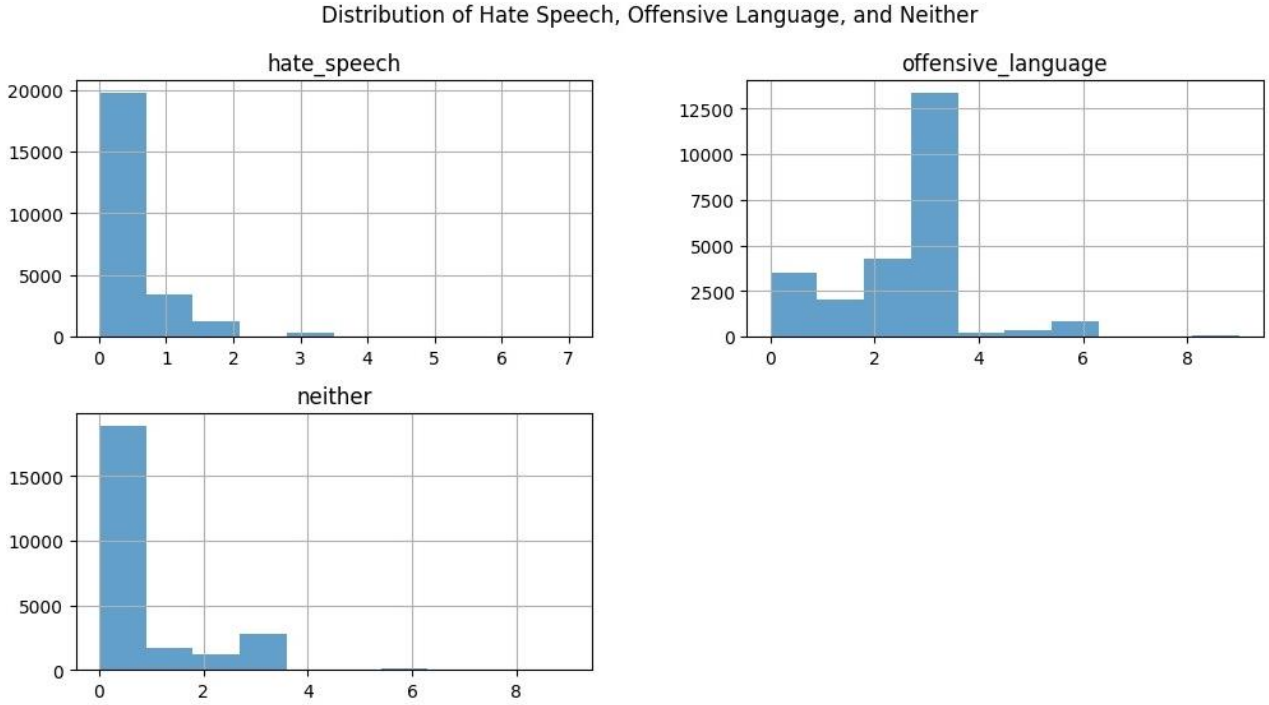

Figure 11. Distribution of hate, offensive and neither speech

Figure 12 depicts the distribution of tweet lengths, providing insights into how tweet lengths vary across the dataset. This analysis is especially useful for understanding user engagement and content style on platforms like Twitter. Figure 13 represents sentiment analysis of tweets, which helps identify trends, patterns, and outliers in the sentiment expressed through tweets. In Figure 12, the Y-axis represents the word count of the tweets, while the X-axis shows the length of the tweets. Similarly, in Figure 13, the X-axis represents sentiment polarity, and the Y-axis denotes the word count of the tweets.

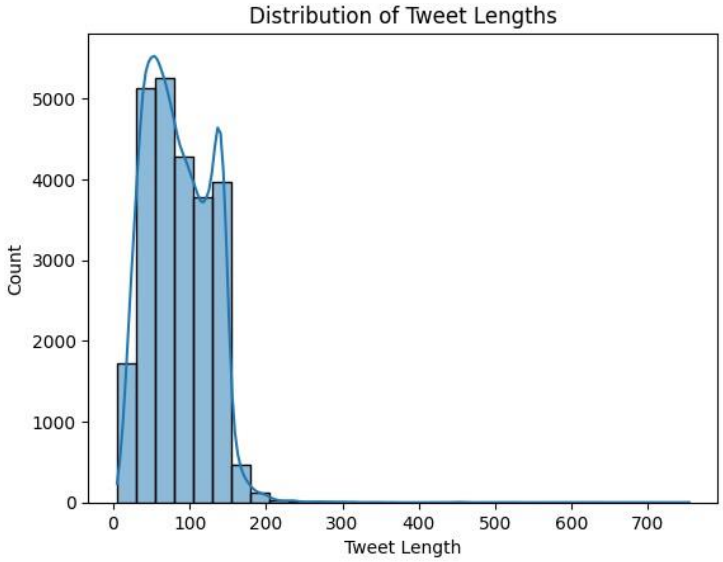

Figure 12. Distribution of tweet lengths

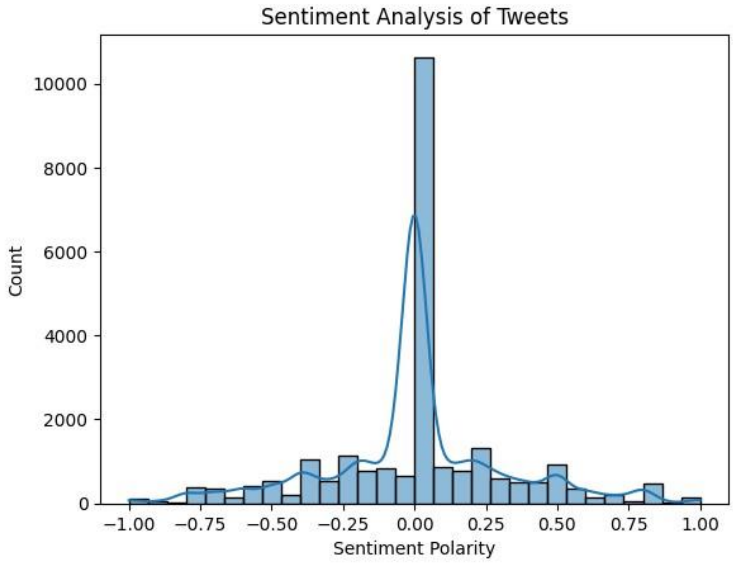

Figure 13. Sentiment analysis of tweets

Similarly, the correlation matrix is depicted in the Figure 14. Correlation matrix is a table which displays the correlation co-efficient between numerous variables. It provides insights into variable relationships, aids in feature selection, and helps identify potential multicollinearity issues. Thus, correlation matrix helps with comprehending the relationships between variables in the dataset. From the Figure 14, it can be identified that, model has perfect correlation as the correlated value obtained is 1. Experimental outcome identified by using proposed model aids in understanding the efficacy of the framework for classification of hate speech, offensive and neither. Thus, the performance of the proposed BiLSTM with SM-CJ is further analyzed using different metrics which will be explored in the subsequent section.

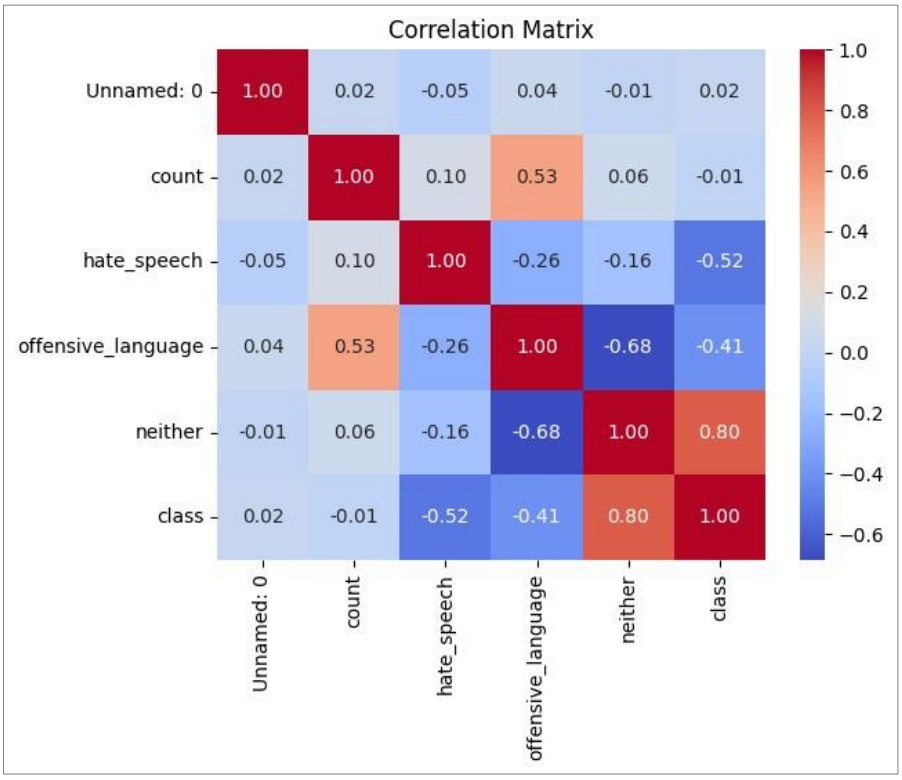

Figure 14. Correlation Matrix

4-4- Performance Analysis

Performance analysis aids in examining the efficacy of the proposed models using different techniques like confusion matrix, correlation matrix, value of accuracy, rate of recall, value of F1 score and value of precision. Thus, Figure 15 shows the confusion matrix of the proposed mechanism.

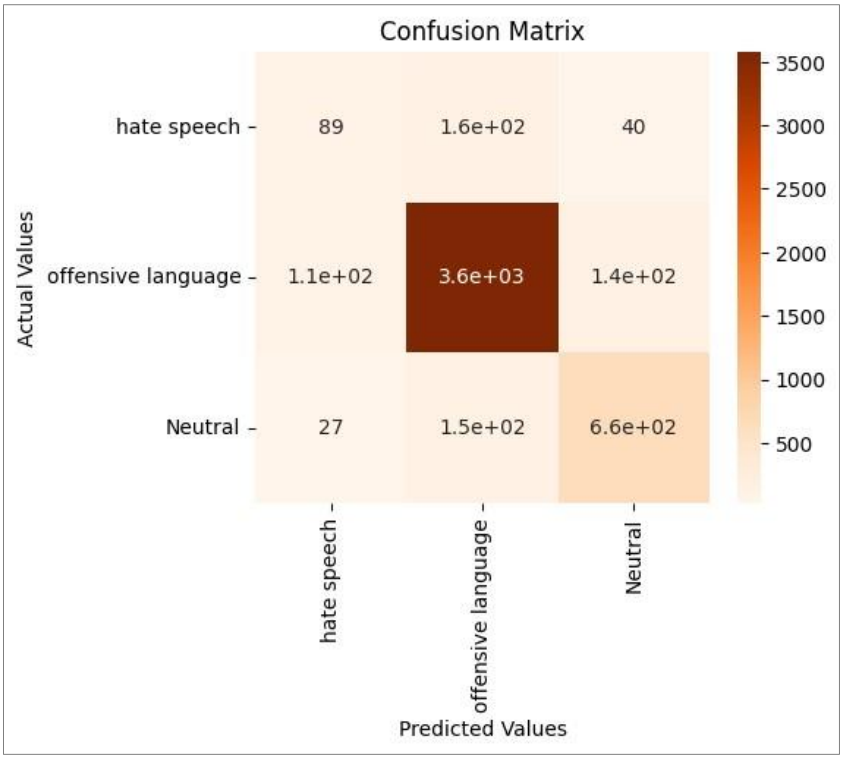

Figure 15. Confusion matrix

The confusion matrix is used to assess the performance of a classification model by providing a comprehensive summary of the model's predictions and the true labels of the data. This matrix helps identify errors made by the model and aids in understanding misclassifications. Figure 15 illustrates the confusion matrix of the proposed framework. From Figure 15, it can be seen that 89, 3578, and 656 instances were correctly classified. The visual representation of the confusion matrix is shown in Figure 15 for the classification of hate speech, offensive language, and neutral tweets.

Similarly, Figures 16 and 17 display the model's accuracy and loss, respectively, attained by the proposed framework. Model loss is a metric that quantifies how well a machine learning model performs on a given dataset. It measures the error between the model's predictions and the true labels. Model accuracy refers to the percentage of correctly predicted instances by the proposed model. In Figures 16 and 17, the blue line represents training accuracy and training loss, while the orange line represents validation accuracy and validation loss. Figure 18 presents the classification report, showing the values obtained by the proposed mechanism for metrics such as accuracy, F1 score, recall, and precision. The proposed model achieves a better accuracy value. Finally, Figure 19 illustrates the sentiment of the text along with the corresponding emoji.

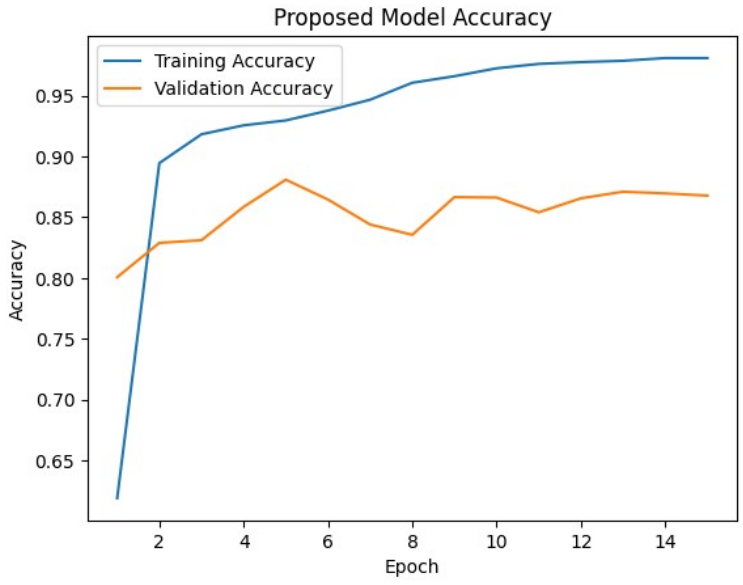

Figure 16. Model accuracy

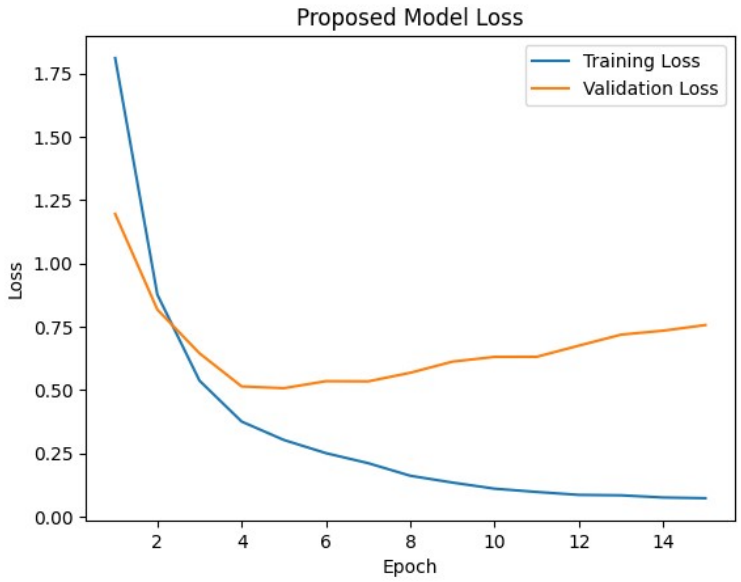

Figure 17. Model loss

Classification report for proposed model

	precision	recall	f1-score	support
hate_speech_count	0.39	0.31	0.34	290
offensive_language_count	0.92	0.93	0.93	3832
neither_count	0.78	0.79	0.79	835
accuracy			0.87	4957
macro avg	0.70	0.68	0.68	4957
weighted avg	0.87	0.87	0.87	4957

Figure 18. Classification Report

```
Enter the number of Test data and hit (Enter): 3
Test data 1: True label (before preprocessing) = [[0.24067163 0.97621596 0.04418003]
 [0.17152911 0.9843046  0.0556826 ]
 [0.15701172 0.98729974 0.08358875]
 ...
 [0.19643252 0.9832761  0.04523829]
 [0.3054062  0.9594943  0.06534348]
 [0.18622366 0.9851459  0.05043177]], Predicted label = 😡


Test data 2: True label (before preprocessing) = [[0.24067163 0.97621596 0.04418003]
 [0.17152911 0.9843046  0.0556826 ]
 [0.15701172 0.98729974 0.08358875]
 ...
 [0.19643252 0.9832761  0.04523829]
 [0.3054062  0.9594943  0.06534348]
 [0.18622366 0.9851459  0.05043177]], Predicted label = 😡


Test data 3: True label (before preprocessing) = [[0.24067163 0.97621596 0.04418003]
 [0.17152911 0.9843046  0.0556826 ]
 [0.15701172 0.98729974 0.08358875]
 ...
 [0.19643252 0.9832761  0.04523829]
 [0.3054062  0.9594943  0.06534348]
 [0.18622366 0.9851459  0.05043177]], Predicted label = 😐
```

Figure 19. Prediction of the proposed model

Although the model has provided efficient outcomes for classifying hate, offensive, and neutral content, it is important to compare the proposed model with different existing models to assess its effectiveness. This comparison will be detailed in the following section.

4-5- Comparative Analysis

A comparative analysis is performed to evaluate the proposed framework against existing works. Table 2 presents the comparative analysis of state-of-the-art approaches with the proposed model. From Table 2, it is evident that the accuracy of the proposed model outperforms other prevailing models, such as CNN-GRU with an accuracy rate of 62.7%, BERT [Attn] with an accuracy rate of 69.0%, and ERT-HateXplain with an accuracy of 69.8%, among others. A visual representation of the data from Table 2 is shown in Figure 20.

Table 2. Comparative Analysis [35]

Model	Accuracy
CNN-GRU [LIME]	0.627
BiRNN [LIME]	0.595
BiiRNN-Attn [Attn	0.621
BiRNN-Attn [LIME]	0.621
BiiRNN-HateXplain [Attn]	0.629
BiRNN-HateXplain [LIME]	0.629
BERT [Attn]	0.690
BERT [LIME]	0.690
BERT-HateXplain [Attn]	0.698
ERT-HateXplain [LIME]	0.698
Proposed Model	***0.87***

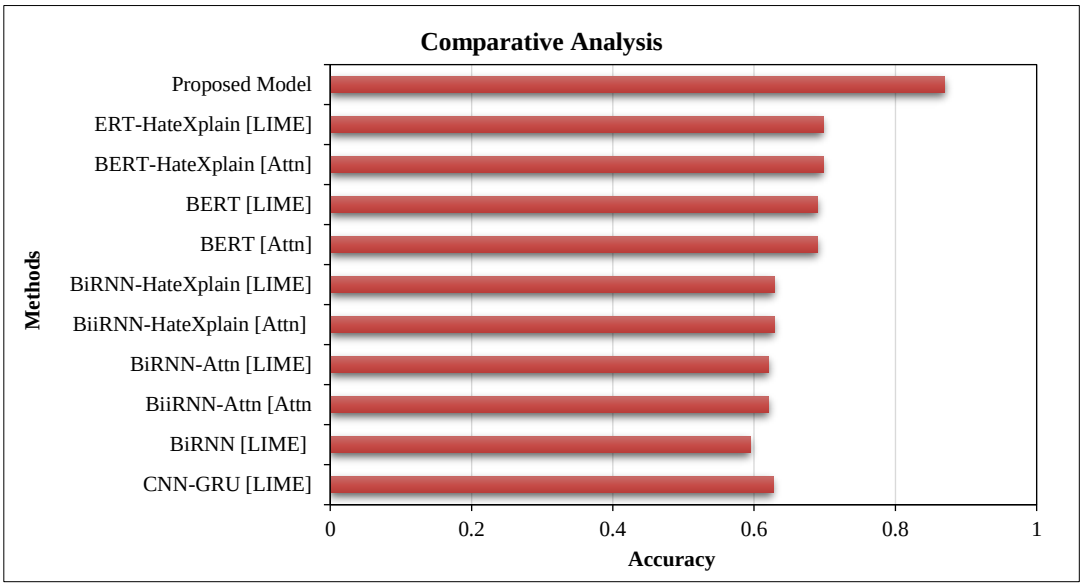

Figure 20. Pictorial representation

Similarly, Table 3 indicates the comparative analysis of the existing works with the proposed model, in which the proposed model has delivered better accuracy, precision, recall and F1 score than the prevailing models as the existing RF model has attained accuracy rate of 0.75%, F1 score of 0.74%, Precision value of 0.73% and Recall rate of 0.75%. Likewise, the prevailing AdaBoost has obtained accuracy value of 0.78%, value of precision 0.75%, recall rate of 0.78% and value of F1 score of 0.73. Figure 21 displays the graphical demonstration of the Table 3.

Table 3. Comparative Analysis [36]

Model	Accuracy	Precision	Recall	F1-Score
LR	0.75	0.72	0.75	0.72
NB	0.73	0.71	0.73	0.68
RF	0.75	0.73	0.75	0.74
SVM	0.79	0.77	0.79	0.77
KNN	0.57	0.61	0.57	0.47
DT	0.73	0.71	0.73	0.71
Adaboost	0.78	0.75	0.78	0.73
MLP	0.70	0.58	0.70	0.63
Proposed Model	**0.87**	**0.70**	**0.68**	**0.68**

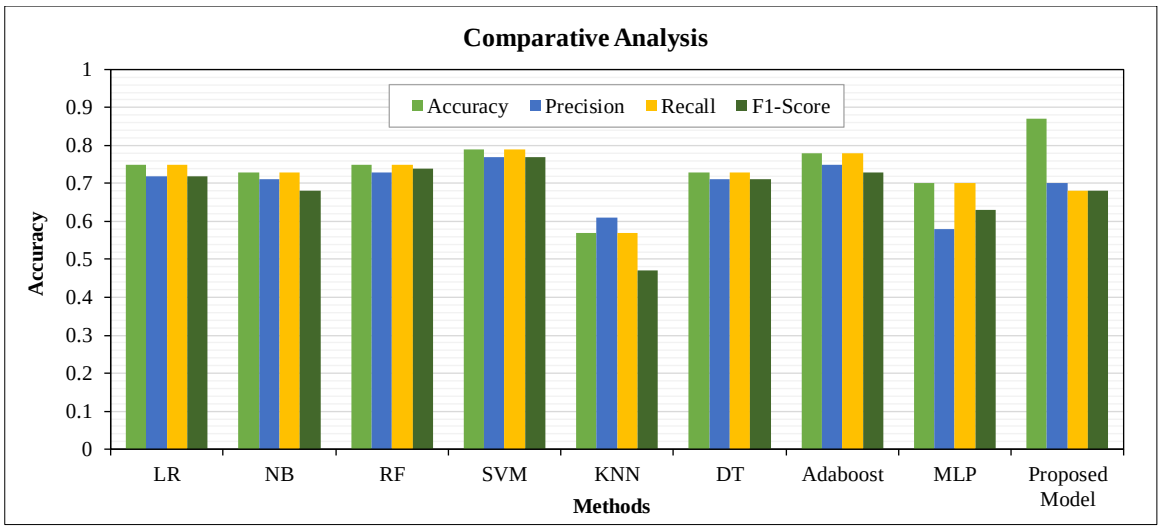

Figure 21. Graphical illustration

Based on the experimental results, it can be concluded that the proposed approach outperforms existing methods in terms of accuracy, precision, recall rate, and F1 score. This improvement is primarily attributed to the implementation of the SM-CJ architecture within the BiLSTM model, as well as various pre-processing techniques, such as the removal of stop words, punctuation, special characters, and numbers from the input text. Additionally, the inclusion of multiple attention layers enhances the model's performance by allowing it to more accurately classify hate, offensive, and neutral content within the dataset.

5- Conclusion

Hate speech detection has become increasingly important in recent years due to the rapid spread of hate on social media platforms. Detecting hate speech on Twitter, for example, is crucial and needs to be done quickly. However, manual detection of hate speech is time-consuming, labor-intensive, and prone to errors. Therefore, AI-based methods are used in the proposed framework for the detection of hate and offensive speech. The proposed model incorporates various pre-processing techniques to achieve better results. Moreover, the model employs a tokenization process, breaking unstructured data into smaller chunks of information, making it more manageable and easier to analyze. Following this, the BiLSTM-based SM-CJ model is applied for effective classification of hate and offensive speech in tweets.

BiLSTM is particularly useful for capturing both forward and backward context in a given text, which is essential for accurately detecting hate speech. Since hate speech detection requires understanding the entire text and identifying patterns that may span multiple words or phrases, the LSTM component of the BiLSTM model is designed to capture long-term dependencies, utilizing information from earlier parts of the text. Additionally, the proposed model includes multiple attention layers, allowing it to focus on the most relevant parts of the input sequence for hate speech detection. This enables the model to better capture derogatory language and other nuances characteristic of hate speech.

The performance of the model was evaluated using various metrics, including accuracy, precision, recall rate, and F1 score. The proposed framework achieved an accuracy of 87%, a precision of 70%, an F1 score of 68%, and a recall rate of 68%. In future work, different deep learning approaches can be explored to further improve the model's performance. Additionally, experiments can be conducted on diverse datasets to assess the model's adaptability across different platforms and languages, ensuring its broader applicability and robustness.

6- Declarations

6-1- Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6-2- Funding

The author extends his appreciation to the Deanship of Scientific Research, University of Hafr Al Batin for funding this work through the research group project No. 0024-1443-S.

6-3- Institutional Review Board Statement

Not applicable.

6-4- Informed Consent Statement

Not applicable.

6-5- Conflicts of Interest

The author declares that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the author.

7- References

[1] Chishty, B. A., & Hashmi, P. (2024). Workplace Cyberbullying in the Healthcare Organization. Workplace Cyberbullying and Behavior in Health Professions. IGI Global, Pennsylvania, United States. doi:10.4018/9798369311394.ch008.

[2] Kumari, K., & Jamatia, A. (2023). An Approach of Hate Speech Identification on Twitter Corpus. Evolution in Computational Intelligence, FICTA 2022, Smart Innovation, Systems and Technologies, 326, Springer, Singapore. doi:10.1007/978-981-19-7513-4_11.

[3] Putra, C. D., & Wang, H.-C. (2024). Semi-meta-supervised hate speech detection. Knowledge-Based Systems, 287, 111386. doi:10.1016/j.knosys.2024.111386.

[4] García-Díaz, J. A., Jiménez-Zafra, S. M., García-Cumbreras, M. A., & Valencia-García, R. (2022). Evaluating feature combination strategies for hate-speech detection in Spanish using linguistic features and transformers. Complex & Intelligent Systems, 9(3), 2893–2914. doi:10.1007/s40747-022-00693-x.

[5] Salminen, J., Hopf, M., Chowdhury, S. A., Jung, S., Almerekhi, H., & Jansen, B. J. (2020). Developing an online hate classifier for multiple social media platforms. Human-Centric Computing and Information Sciences, 10(1), 1-34. doi:10.1186/s13673-019-0205-6.

[6] Rusert, J., Shafiq, Z., & Srinivasan, P. (2022). On the Robustness of Offensive Language Classifiers. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 1, 7424–7438. doi:10.18653/v1/2022.acl-long.513.

[7] d'Sa, A. G., Illina, I., & Fohr, D. (2020). Classification of hate speech using deep neural networks. Revue d'Information Scientifique & Technique, 25(01), 1-12.

[8] Spence, R., Bifulco, A., Bradbury, P., Martellozzo, E., & DeMarco, J. (2023). The psychological impacts of content moderation on content moderators: A qualitative study. Cyberpsychology: Journal of Psychosocial Research on Cyberspace, 17(4), 8. doi:10.5817/cp2023-4-8.

[9] Liu, Y., Alzahrani, I. R., Jaleel, R. A., & Sulaie, S. A. (2023). An efficient smart data mining framework based cloud internet of things for developing artificial intelligence of marketing information analysis. Information Processing & Management, 60(1), 103121. doi:10.1016/j.ipm.2022.103121.

[10] Lakhan, A., Mastoi, Q., Dootio, M. A., Alqahtani, F., Alzahrani, I. R., Baothman, F., Shah, S. Y., Shah, S. A., Anjum, N., Abbasi, Q. H., & Khokhar, M. S. (2021). Hybrid Workload Enabled and Secure Healthcare Monitoring Sensing Framework in Distributed Fog-Cloud Network. Electronics, 10(16), 1974. doi:10.3390/electronics10161974.

[11] Mehta, H., & Passi, K. (2022). Social Media Hate Speech Detection Using Explainable Artificial Intelligence (XAI). Algorithms, 15(8), 291. doi:10.3390/a15080291.

[12] Oriola, O., & Kotze, E. (2020). Evaluating Machine Learning Techniques for Detecting Offensive and Hate Speech in South African Tweets. IEEE Access, 8, 21496–21509. doi:10.1109/access.2020.2968173.

[13] Khan, M. U. S., Abbas, A., Rehman, A., & Nawaz, R. (2021). HateClassify: A Service Framework for Hate Speech Identification on Social Media. IEEE Internet Computing, 25(1), 40–49. doi:10.1109/mic.2020.3037034.

[14] Ayo, F. E., Folorunso, O., Ibharalu, F. T., & Osinuga, I. A. (2020). Hate speech detection in Twitter using hybrid embeddings and improved cuckoo search-based neural networks. International Journal of Intelligent Computing and Cybernetics, 13(4), 485–525. doi:10.1108/ijicc-06-2020-0061.

[15] Murfi, H., Siagian, F. L., & Satria, Y. (2019). Topic features for machine learning-based sentiment analysis in Indonesian tweets. International Journal of Intelligent Computing and Cybernetics, 12(1), 70–81. doi:10.1108/ijicc-04-2018-0057.

[16] Siddiqua, U. A., Chy, A. N., & Aono, M. (2019). KDEHatEval at SemEval-2019 Task 5: A Neural Network Model for Detecting Hate Speech in Twitter. Proceedings of the 13th International Workshop on Semantic Evaluation, 365–370. doi:10.18653/v1/s19-2064.

[17] Winter, K., & Kern, R. (2019). Know-Center at SemEval-2019 Task 5: Multilingual Hate Speech Detection on Twitter using CNNs. Proceedings of the 13th International Workshop on Semantic Evaluation, 431–435. doi:10.18653/v1/s19-2076.

[18] Mossie, Z., & Wang, J.-H. (2020). Vulnerable community identification using hate speech detection on social media. Information Processing & Management, 57(3), 102087. doi:10.1016/j.ipm.2019.102087.

[19] Bade, G., Kolesnikova, O., Sidorov, G., & Oropeza, J. (2024). Social Media Hate and Offensive Speech Detection Using Machine Learning Method. Proceedings of the Fourth Workshop on Speech, Vision, and Language Technologies for Dravidian Languages, 22 March, 2024, St. Julian's, Malta.

[20] Widiarta Nandana Githa, I. P., Syananda, A., Faustine, R., Edbert, I. S., & Suhartono, D. (2024). Hate Speech Classification in Indonesian Tweets Using TF-IDF and Data Augmentation. 2024 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), 61–65. doi:10.1109/gecost60902.2024.10474781.

[21] Manikandan, R., Hariharasitaraman, S., Ramkumar, S., & Gobinath, R. (2024). Leveraging Deep Learning in Hate Speech Analysis on Social Platform. Deep Learning for Smart Healthcare, 38–53, Auerbach Publications, Boca Raton, United States. doi:10.1201/9781003469605-3.

[22] Shawkat, N., Saquer, J., & Shatnawi, H. (2024). Evaluation of Different Machine Learning and Deep Learning Techniques for Hate Speech Detection. Proceedings of the 2024 ACM Southeast Conference on ZZZ, 253–258. doi:10.1145/3603287.3651218.

[23] Saleh, H., Alhothali, A., & Moria, K. (2023). Detection of Hate Speech using BERT and Hate Speech Word Embedding with Deep Model. Applied Artificial Intelligence, 37(1). doi:10.1080/08839514.2023.2166719.

[24] Ojo, O. E., Hoang, T. T., Gelbukh, A., Calvo, H., Sidorov, G., & Adebanji, O. O. (2022). Automatic Hate Speech Detection Using CNN Model and Word Embedding. Computación y Sistemas, 26(2), 1007-1013. doi:10.13053/cys-26-2-4107.

[25] Bisht, A., Singh, A., Bhadauria, H. S., Virmani, J., & Kriti. (2020). Detection of Hate Speech and Offensive Language in Twitter Data Using LSTM Model. Recent Trends in Image and Signal Processing in Computer Vision. Advances in Intelligent Systems and Computing, 1124. Springer, Singapore. doi:10.1007/978-981-15-2740-1_17.

[26] Das, D., Mondal, S., & Ray, A. (2021). Classifying Hate Speeches Shared in Twitter. Advances in Speech and Music Technology. Advances in Intelligent Systems and Computing, 1320. Springer, Singapore. doi:10.1007/978-981-33-6881-1_31.

[27] Asogwa, D. C., Chukwuneke, C. I., Ngene, C. C., & Anigbogu, G. N. (2022). Hate speech classification using SVM and naive BAYES. arXiv preprint, arXiv:2204.07057. doi:10.48550/arXiv.2204.07057.

[28] Kupi, M., Bodnar, M., Schmidt, N., & Posada, C. E. (2021). dictNN: A Dictionary-Enhanced CNN Approach for Classifying Hate Speech on Twitter. arXiv preprint, arXiv:2103.08780. doi:10.48550/arXiv.2103.08780.

[29] Wani, A. H., Molvi, N. S., & Ashraf, S. I. (2020). Detection of Hate and Offensive Speech in Text. Intelligent Human Computer Interaction. IHCI 2019. Lecture Notes in Computer Science, 11886, Springer, Cham, Switzerland. doi:10.1007/978-3-030-44689-5_8.

[30] Touahri, I., & Mazroui, A. (2022). Offensive Language and Hate Speech Detection Based on Transfer Learning. Advanced Intelligent Systems for Sustainable Development (AI2SD'2020). AI2SD 2020, Advances in Intelligent Systems and Computing, 1418, Springer, Cham, Switzerland. doi:10.1007/978-3-030-90639-9_24.

[31] Paul, C., & Bora, P. (2021). Detecting Hate Speech using Deep Learning Techniques. International Journal of Advanced Computer Science and Applications, 12(2), 78. doi:10.14569/ijacsa.2021.0120278.

[32] Viswapriya, S. E., Gour, A., & Chand, B. G. (2021). Detecting Hate Speech and Offensive Language on Twitter using Machine Learning. International Journal of Computer Science and Mobile Computing, 10(4), 22–27. doi:10.47760/ijcsmc.2021.v10i04.004.

[33] Zhou, Y., Yang, Y., Liu, H., Liu, X., & Savage, N. (2020). Deep Learning Based Fusion Approach for Hate Speech Detection. IEEE Access, 8, 128923–128929. doi:10.1109/access.2020.3009244.

[34] Melton, J., Bagavathi, A., & Krishnan, S. (2020). DeL-haTE: A Deep Learning Tunable Ensemble for Hate Speech Detection. 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), 1015–1022. doi:10.1109/icmla51294.2020.00165.

[35] Mathew, B., Saha, P., Yimam, S. M., Biemann, C., Goyal, P., & Mukherjee, A. (2021). HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. Proceedings of the AAAI Conference on Artificial Intelligence, 35(17), 14867–14875. doi:10.1609/aaai.v35i17.17745.

[36] Abro, S., Shaikh, S., Hussain, Z., Ali, Z., 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), 61. doi:10.14569/ijacsa.2020.0110861.