JISTaP https://www.jistap.org
Journal of Information Science Theory and Practice

RESEARCH PAPER
J Inf Sci Theory Pract 13(1): 51-69, 2025

eISSN : 2287-4577 pISSN : 2287-9099

https://doi.org/10.1633/JISTaP.2025.13.1.4

Hate Speech Identification and Categorization on Social Media Using Bi-LSTM: An Information Science Perspective

Krishna Kumar Mohbey
Department of Computer Science, Central University of Rajasthan, Ajmer, India
E-mail: kmohbey@curaj.ac.in

Nishtha Kesswani*
Department of Data Science & Analytics, Central University of Rajasthan, Ajmer, India
E-mail: nishtha@curaj.ac.in

Yunevich Nikol
Center of Advanced Studies in Digital Development of JSC "Giprosvyaz," Minsk, Belarus
E-mail: yunevich@giprosvjaz.by

Basant Agarwal
Department of Computer Science and Engineering, Central University of Rajasthan, Ajmer, India
E-mail: basant@curaj.ac.in

Maxim Sterjanov
Department of Computer Science, Belarusian State University of Informatics and Radioelectronics, Minsk, Belarus
E-mail: sterjanov@bsuir.by

Vishnyakova Margarita
Center of Advanced Studies in Digital Development of JSC "Giprosvyaz," Minsk, Belarus
E-mail: vishnyakova@giprosvjaz.by

ABSTRACT

Online social networks empower individuals with limited influence to exert significant control over specific individuals' lives and exploit the anonymity or social disconnect offered by the Internet to engage in harassment. Women are commonly attacked due to the prevalent existence of sexism in our culture. Efforts to detect misogyny have improved, but its subtle and profound nature makes it challenging to diagnose, indicating that statistical methods may not be enough. This research article explores the use of deep learning techniques for the automatic detection of hate speech against women on Twitter. It offers further insights into the practical issues of automating hate speech detection in social media platforms by utilizing the model's capacity to grasp linguistic nuances and context. The results highlight the model's applicability to information science by addressing the expanding need for better retrieval of hazardous content, scalable content moderation, and metadata organization. This work emphasizes content control in the digital ecosystem. The deep learning-based methods discussed improve the retrieval of data connected to hate speech in the context of a digital archive or social media monitoring system, facilitating study in fields including online harassment, policy formation, and social justice campaigning. The findings not only advance the field of natural language processing but also have practical implications for social media platforms, policymakers, and advocacy groups seeking to combat online harassment and foster inclusive digital spaces for women.

Keywords: hate speech detection, social media, deep learning, machine learning, metadata organization, content labelling

Received: September 4, 2024 **Revised:** October 5, 2024
Accepted: November 2, 2024 **Published:** March 30, 2025

***Corresponding Author:** Nishtha Kesswani
https://orcid.org/0000-0002-9044-4160
E-mail: nishtha@curaj.ac.in

1. INTRODUCTION

The rise in popularity of social media platforms has led to a significant increase in the volume of textual information, rendering manual moderation of this content unfeasible (Cao et al., 2020). Social media platforms such as Twitter, Facebook, and Instagram enable users to freely express themselves, which has boosted the proliferation of hate speech and harsh language, thus posing new challenges for information science. Hate speech, especially directed at women, is a major obstacle to creating a secure and welcoming online space. Conventional information retrieval systems usually focus on offering people pertinent content through keyword or semantic search engines. However, disseminating abusive content on digital platforms can pose a significant risk to people, society, governments, and social media platforms (Miškolci et al., 2020). Owing to this, researchers are using powerful computational techniques, such as profound learning, to create automated systems that can identify and reduce hate speech in response to the increasing worry about this issue (Davidson et al., 2017). This study introduces a deep learning (DL) method for automatically detecting hate speech directed against women, specifically on Twitter, aiming to enhance the establishment of a safer and more encouraging online environment. By integrating models that can automatically detect and flag hate speech, this research expands the functionality of retrieval systems. It enhances user safety in digital contexts while also producing higher-quality search results.

1.1. Concept of Hate Speech

Hate speech is significant because it is subjective. According to Fortuna and Nunes (2019), hate speech is rhetoric that targets or belittles groups, instigating violence or hatred based on physical appearance, religion, descent, national or ethnic origin, sexual orientation, gender identity, etc. It may surface in a variety of linguistic approaches, including sharp-witted humor and scathing retorts. Hate speech is potentially detrimental to people and society (Fortuna & Nunes, 2019). Hate speech is prevalent on social media platforms and requires creative methods to tackle this growing problem. Hate speech, which includes abusive words, discriminatory statements, and threats, harms individuals and goes against the values of accessible and respectful communication (Waseem & Hovy, 2016). Targeted hate speech against women is especially worrisome since it encourages gender-based violence and reinforces negative preconceptions and biases.

Detecting hate speech may be challenging due to users' common use of acronyms, slang, and hashtags. Accurate classification of data improves searchability and fosters moral digital curation. For instance, hate speech content needs to be correctly indexed and managed in a public archive that houses millions of tweets to prevent unintentionally spreading damaging viewpoints. The methodology to identify hate speech should enable archivists to identify, separate, and if required, impose access restrictions on particular types of offensive communication. This helps to ensure that sensitive content is used appropriately for research or educational purposes, supporting the ethical management of digital resources. Moreover, the classification methods can be applied to hate speech and other online misconduct, which will help to create complete digital content management systems. Previous research has investigated many methods to detect and address hate speech, including rule-based systems and machine-learning techniques. The changing and developing online language necessitates advanced technologies, leading to the investigation of DL methods.

1.2. Need for the Study

Several research studies have reported the automatic identification of hate speech in benchmark datasets by integrating natural language processing (NLP) with classic machine learning (ML) techniques (Salminen et al., 2020; Watanabe et al., 2018) and DL strategies (Zhang & Luo, 2019). These approaches involve the utilization of metadata, user-based features, and text mining-based features. These features include lexical approaches, grammatical approaches, bag-of-words (BOW), text embedding, sentiment analysis, and others. There is a need to organize metadata surrounding hate speech by creating an effective framework to categorize it and distinguish it from other offensive or neutral language. For academics and politicians alike, it would be essential to organize resources with relevant metadata tags such as "hate speech," "misogyny," or "gender-based harassment" in an online archive or digital library devoted to social media studies. Different models can be incorporated into these systems to automatically classify content, guaranteeing that users can access important information while avoiding undesirable or hazardous content. Such research is expected to improve current hate speech identification methods and guide the creation of better content control tools on social media sites.

Srba et al. (2021) studied hate speech detection from a computer science perspective. They analyzed the contribution of different forms of data, including metadata, textual data, videos, and images to hate speech detection. Both models have researched both techniques. However, for DL models to function well, they need a substantial amount of data that has been labeled. Agarwal and Chowdary (2021) have investigated the use of ensemble learning in the context of hate speech identification. Ensemble learning has also demonstrated robust findings. The possible biases of the datasets and algorithms were not considered in these works, even though various contributions have been devoted to analyzing these topics and have provided good classification scores. In the light of the above discussion, the following research hypotheses can be formulated:

H1. The accuracy of hate speech detection on various social media platforms can be greatly enhanced by ML models trained on a range of annotated datasets, decreasing false positives and false negatives.

H2. Utilizing advanced word embeddings in hate speech detection models will enhance the accuracy of classifying nuanced or context-dependent hate speech compared to traditional text representation techniques.

H3. Automated hate speech detection systems improve the organization and retrieval of digital data by effectively classifying harmful content, thus upholding the integrity of information in extensive information systems and databases.

1.3. Objectives

This study is driven by the necessity for an automated and scalable approach to address hate speech directed at women on Twitter. Utilizing DL shows excellent potential by enabling models to acquire intricate patterns and representations from data. DL has been highly successful in NLP tasks, making it well-suited for tackling the issues presented by the ever-changing and context-specific nature of hate speech on social media. The primary objectives of this research are as follows:

- This research offers useful insights into how automated systems might improve the detection and categorization of hate speech on social media sites like Twitter, especially directed toward women, by employing DL models.
- By creating models that can automatically produce precise and context-aware metadata tags, we can enhance the categorization and retrieval of content relevant to hate speech and contribute to effectively curating large-scale social media datasets.
- Rather than concentrating solely on technical measures, we aim to assess the model's performance in detecting hate speech in real-world scenarios, emphasizing the usefulness of accuracy, precision, recall, and F1-score in improving the functioning of content moderation tools.
- To contribute insights into the linguistic features and contextual elements that distinguish hate speech against women on Twitter, thereby advancing our understanding of online gender-based harassment.

The remainder of this paper is organized as follows: Section 2 reviews relevant literature, highlighting the current research on hate speech detection and DL applications in social media. Section 3 outlines the methodology, detailing the dataset collection and preprocessing steps and the architecture and training of the DL model. Section 4 presents the experimental results and discusses the implications of the findings. Finally, Section 5 concludes the paper, summarizing key contributions and discussing potential areas for future investigation.

2. RELATED WORK

The consequences for online safety, social cohesiveness, and freedom of expression have made hate speech detection on social media platforms an essential field of research. Hate speech detection spans multiple aspects. Pérez et al. (2023) have explored a novel method of data curation, sampling, and annotation. In their study, they investigated the impact of contextual information in enhancing hate speech detection. DL's recent advances have greatly improved detection accuracy and scalability, unlike earlier methods that depended on manually created features and conventional ML techniques (Waseem et al., 2017). Hate speech on Twitter frequently takes on gendered forms, according to research; specifically, women are more likely to be the targets of insults, threats, and harassment. To effectively combat sexism and misogyny on social media, it is essential to understand these gender-specific trends (Mocanu et al., 2015). A study by Tontodimamma et al. (2021) investigated the expanse of hate speech using information mapping on the Scopus database. The bibliometric metric was based on co-word methods analysis and relied on different ML algorithms. DL approaches have entirely changed the hate speech detection game. With advancements such as transformer-based models, recurrent neural networks (RNNs), and convolutional neural networks (CNNs), models can learn intricate representations and patterns from text input. These methods work particularly well for detecting hate speech against women on Twitter because of their greater effectiveness in collecting contextual subtleties and semantic information (Nobata et al., 2016).

Some have voiced worries about biases in hate speech detection algorithms, even if DL-based systems are effective. Unintentionally reinforcing prejudices and worsening existing inequities are the potential outcomes of models trained on biased datasets. Rahman et al. (2021) proposed an information retrieval-based hate speech detection approach involving pooling and active learning methods. The study emphasized task decomposition and annotator rationale techniques. Hate speech detection systems, especially those that target hate speech based on gender, need to be fair and overcome biases, according to experts (Sap et al., 2019). More recent research has looked into novel ways to make hate speech detection algorithms better and easier to understand. The problems presented by the ever-changing types of hate speech on Twitter can be partially addressed by employing domain adaptation approaches, multimodal learning, and adversarial training. Additionally, there is a rising movement to improve the openness and responsibility of hate speech detection systems using explainable AI approaches (Kheddar et al., 2023). Zhang and Luo (2019) explored the long-tail effect in the Twitter dataset and the consequent impact on classifying hate speech. Their study critiques the common practice of micro-averaging, which can bias evaluations towards non-hate classes, and demonstrates improved classification of hate speech in the long tail of datasets.

Identifying abusive language was the primary emphasis of Wiegand et al. (2018)'s research. The authors constructed an abusive lexicon through the use of a variety of traits and lexical resources. Following that, a built lexicon implemented in an support vector machine (SVM) classification was utilized. This study used datasets accessible to the general public (Waseem & Hovy, 2016). It is important to draw attention to the fact that all of the research stated above was conducted using English.

On the other hand, it is worth noting that there have been a few additional studies undertaken in other languages in recent times, including Italian (Del Vigna et al., 2017), German (Köffer et al., 2018), Russian (Andrusyak et al., 2018), and Indonesian (Alfina et al., 2017). Abozinadah et al. (2015) analyzed several machine-learning algorithms to identify offensive tweets written in Arabic. In addition to using three different categorization techniques, they manually identified and labeled five hundred accounts related to the abusive tweets extracted. The naive Bayes (NB) classifier obtained an F1-score of up to 90%, which was the most efficacious. Haidar et al. (2017) proffered a system for detecting and preventing cyberbullying on social media platforms. The authors manually annotated a large dataset of 35,273 tweets from the Middle East region. Through the use of SVM and NB, the authors were able to get the best results in terms of classification, with SVM reaching an F1-score of up to 0.93. Recently, Alakrot et al. (2018) have described creating an inflammatory dataset of Arabic comments on YouTube. After analyzing 150 videos on YouTube, the writers retrieved 167,549 comments from the platform. Sixteen thousand comments were randomly selected for annotation, and three people were responsible for the annotation process.

Waseem and Hovy (2016) employed the logistic regression (LR) classification method to detect racist and sexist content on social media. After manually annotating a dataset of 16,914 tweets, the authors reported that 3,383 had sexist material, 1,972 contained racist content, and 11,559 contained neither sexist nor racist information. The authors used the Twitter application programming interface to extract tweets with a few terms associated with women to generate the dataset. They were able to attain an F1-score of 0.73. Many researchers (Al-Hassan & Al-Dossari, 2019) use this study as a standard of excellence. Pitsilis et al. (2018) propose that to identify instances of racism and sexism in social media, it would be beneficial to use a neural network solution comprising numerous long-short-term-memory (LSTM) based classifiers. In a large number of tests, the authors were able to get the highest possible F1-score of 0.93.

Another group of researchers, Kshirsagar et al. (2018), focused on identifying racism and sexism, and their methodology is likewise based on neural networks. Nevertheless, the authors of this study also utilized word embedding to collect features and merge them with a classifier based on multi-layer perception. A maximum F1-score of 0.71 was reached. Saha et al. (2018) created a methodology to identify instances of hate speech directed towards women. In order to extract features, the authors utilized several methods, including BOW, term frequency-inverse document frequency, and sentence embeddings, along with various classification algorithms, including LR, Extreme Gradient Boosting (XGBoost), and CatBoost. Using the LR classifier, the best F1-score produced was 0.70. A hybrid model that combines CNN and LSTM was suggested by Zhang and Luo (2019) to identify hate speech. Seven datasets were used to apply the authors' method, of which five are available to the general public.

In previous years, much progress has been made in utilizing DL to automatically detect hate comments against women on Twitter. The fight against prejudice, for equality, and against new kinds of hate speech is far from over. Research into the fight against cyberbullying and other forms of gender-based violence can progress toward more effective and moral solutions if it builds on the work that has already been done in this area.

3. PROPOSED MODEL

In this section, we discuss the process of hate-speech detection along with each phase. Fig. 1 depicts the entire process schematically. Data in its original form is gathered from different social media sites. The information is preprocessed, labeled, and divided into training and test sets. Before feature extraction, tokenization and padding are used to standardize the text sequences. Word vectors, particularly GloVe, represent words in a dense format. Next, the information goes through classifiers, such as traditional ML models and DL models. In the end, the system categorizes the text into either "Hate," "Offensive," or "Neither."

3.1. Data Preparation

We evaluated hate speech detection using data from Twitter. After applying data preprocessing, it was split into training and testing sets of 80% and 20%, respectively, with a validation split of 20%. The text is categorized as hate speech, offensive language, or neither. It is crucial to remember that this dataset contains content that may be interpreted as racist, sexist, homophobic, or simply offensive.

We undertook a four-stage process to prepare the textual data for the sentiment analysis. Firstly, we extracted the sentiment labels from the data and reorganized the data frame. Numeric labels were then mapped to more descriptive categories, such as "hate speech" or "offensive language," for better comprehension. Second, to address the issue of variable-length text data and to ensure the model's efficacy, we employed text tokenization. This process segments each text into words or subwords. The maximum number of vocabulary words to be considered was set at 20,000. However, tokenization still leaves the issue of variable length unattended. This inconsistency can pose problems for ML models that expect inputs of uniform size. Thus, we pad the shorter sequences with zeros to make the sequences have the same dimensionality. Next, we augment the text representations using GloVe, a pre-trained word embedding model. It helps capture the inherent structure and content within the otherwise latent words and forms the embedding matrix. Words with similar semantic meanings have similar embedding vectors. Finally, as stated earlier, the textual content has been classified into categorical labels. These labels classify the content into distinct classes without stating the natural order. Thus, we use one-hot encoding to better represent the categorical labels. The final one-hot encoded vector represents the relationship between the input and the text classes.

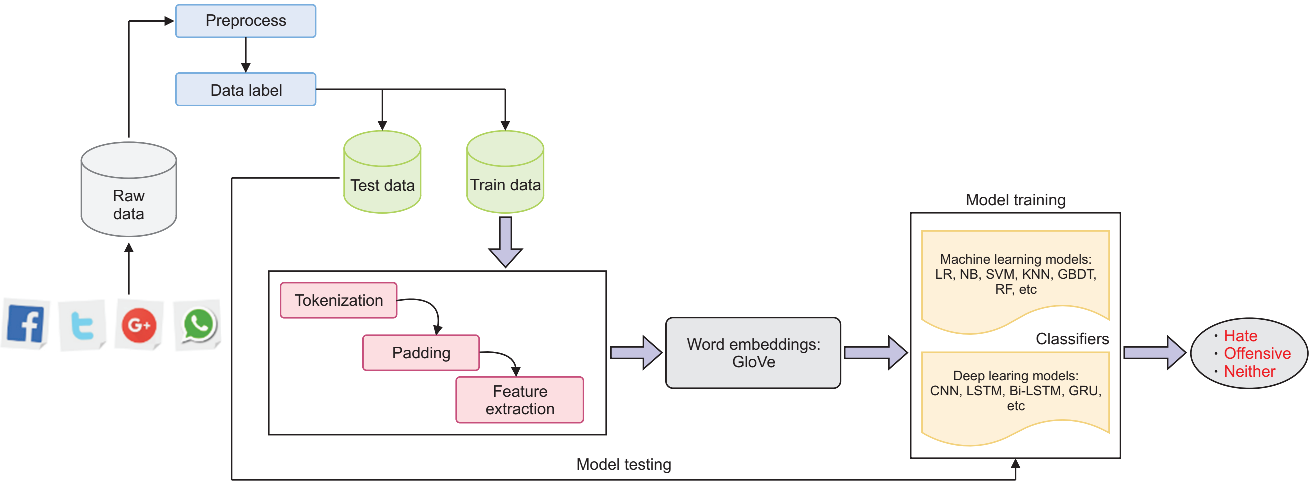


Fig. 1. Schematic framework of hate speech detection using deep learning and machine learning classifiers. LR, logistic regression; NB, naive bayes; SVM, support vector machine; KNN, K-nearest neighbor; GBDT, gradient boosted decision trees; RF, random forest; CNN, convolution neural network; LSTM, long-short-term memory; Bi-LSTM, bidirectional long-short-term memory; GRU, gated recurrent units.

3.2. Classifiers

After applying the data preprocessing, we train different ML and DL classifiers. Here, we discuss each one of them in detail.

3.2.1. Machine Learning Classifiers

Logistic Regression (LR): A fundamental classification algorithm estimates the likelihood that an instance belongs to a specific class. Its high interpretability boosts its extensive use. It is a common choice for binary classification tasks, especially when the relationship between features and the target variable is linear (Sperandei, 2014).

Random Forest (RF): This ensemble learning technique combines several decision trees (DT) to produce correct predictions. It decreases overfitting by averaging predictions from individual trees. It can perform classification and regression tasks and is resistant to outliers and noisy data (Rigatti, 2017).

Support Vector Machine (SVM): SVM is an effective approach for classification and regression. It identifies a hyperplane that best separates various classes while improving their distance. It is suited for high-dimensional spaces and can handle nonlinear interactions with kernel functions (Patle & Chouhan, 2013).

K-Nearest Neighbours (KNN): This classifies occurrences based on the class of their closest neighbors. It is easy to use and requires no training, making it suitable for small datasets. The distance metric selection and the number of neighbors heavily influence its performance (Zhang et al., 2018).

Naive Bayes (NB): A probabilistic algorithm founded on Bayes' theorem. It is especially beneficial for text categorization and other high-dimensional data. Despite its "naive" feature independence assumption, NB is computationally efficient (Berrar, 2019).

Extreme Gradient Boosting (XG Boost): This is a highly efficient version of Gradient Boosting and is well-known for its accelerated performance. Its unique characteristics include a regularization mechanism to check for overfitting and a depth-first strategy for tree pruning to improve generalization capability (Chen & Guestrin, 2016).

3.2.2. Deep Learning Models

Convolution Neural Network (CNN): CNN is a deep neural network that automatically adapts itself to learn the spatial hierarchies of features inherent within the input. CNN consists of convolution layers that use kernels to compute element-wise multiplication on input, aggregate the result to provide a single value, and thus learn the local patterns (Liu et al., 2019).

Bidirectional Long-Short-Term Memory (Bi-LSTM): This is an extension of LSTM that processes input in forward and backward directions using past and future tokens. It has been widely used in sentiment analysis. It uses two separate LSTM layers for processing in different directions. The outputs are concatenated before being used for final predictions (Li et al., 2018).

Long-Short-Term Memory (LSTM): Designed to solve the vanishing gradient issue, LSTM is an extension of RNNs. The input, output, and forget gates are its three gates. They are apt for sequential data processing tasks (Staudemeyer & Morris, 2019).

The reason for selecting a mix of classic ML models and DL models lies in their different strong points, which help tackle different issues in hate speech detection. ML models allow for easier interpretation of results, providing insights into how features contribute to hate speech classification. DL models are suitable for identifying subtle forms of offensive language and hate speech in text by capturing long-term dependencies, sequential patterns, and contextual information that traditional models may struggle to detect. This variety permits a thorough assessment of various approaches, helping the research pinpoint the most effective model or blend of models for tackling issues of hate speech identification.

3.3. Model Architecture

In this section, we discuss detecting Bi-LSTM-based hate speech in detail. Algorithm 1 represents the entire process of hate speech detection.

Algorithm 1: Hate speech detection using Bi-LSTM neural network

```
Input: Text data (T), Preprocessed labels (L), Maximum sequence
length (MAX_LEN), d, Word embedding matrix (W)
Output: Sentence class C={"hate Speech," "Offensive," "Neither"}
1   Apply preprocessing
    (a)     sentence tokenization
    (b)     remove punctuations such as " '!()-[]
            {};:'"\,<>./?@#$%^&*_~" '
    (c)     remove stop words
2.  for each sentence in T
        while sequence length <MAX_LEN do
        (a) Pad the sequence with zeros
        (b) Return padded sequence $S_p$
        end while
    end for
```

```
3.  Lookup_Embeddings($S_p$, W)
    Initialize an empty list embedding to store word embeddings
    for each word in the $S_p$
            (a) Lookup the word's embedding vector from the W
                matrix
            (b) Append the embedding vector to the embedding
                list
            (c) Return embeddings list
    end for
4.  Initialize hyperparameter for model construction
    learning rate=0.001,
    optimizer='adam',
    epochs=100,
    batch_size=64,
    train_set ($T_r$)=80%,
    Validation set ($T_v$)=20%
    test_size ($T_s$)=20%
5.  while each sentence $S \in T_r$ do
    (a)     Generate all word embedding vectors in $S$=[$s_1$, $s_2$, $s_3$,
              . . .., $s_n$]
    (b)     Construct a Bi-LSTM network
    (c)     Train the Bi-LSTM model on $T_r$ along with L
    (d)     Use a SoftMax classifier to categorize the
              Bi-LSTM's output as in $C$
    (e)     Validate the model predictions on $T_v$
    end while
6.  for each statement in $T_s$
            Test the Bi-LSTM model and predict the labels in $C$
    end for
```

3.3.1. Forming Word Embeddings

We represent each word in the vocabulary as a dense vector of size d (embedding dimension), which consists of semantic information about the word. Using GLoVE, a pre-trained word embedding matrix w is formed with size $vocabulary\ size \times d$. This matrix associates each word in the vocabulary with a dense vector (embedding). Words with similar meanings tend to have similar embeddings in this vector space. The model learns these pre-trained embeddings during training and may fine-tune them for the specific sentiment analysis task. Each token in the preprocesses sequence is looked up in the embedding matrix. This retrieves its corresponding embedding vector, transforming the text sequence into a sequence of embedding vectors. Fig. 2 provides a visualization of the embedding formed from 100 samples randomly selected. The principal component analysis (PCA) analysis in Fig. 2A depicts that similar words will be close together, but different words will be further apart in meaning or context. In detecting hate speech, this is crucial as words linked to hate speech could gather together, possibly showing offensive language patterns. PCA preserves the overall layout by representing the relationships between distant points and the dispersion of points in the plot, which mirrors the variability in the original multi-dimensional space. The closeness of these words can aid in teaching models to identify offensive language, as they frequently indicate negative feelings or hostility.

Fig. 2B represents a t-distributed stochastic neighbor embedding (t-SNE) plot that offers important insights into how words related to hate speech are distributed relative to neutral words. Words that are closer together on this t-SNE plot probably have similar meanings or are used in the same way. In the context of hate speech, words such as

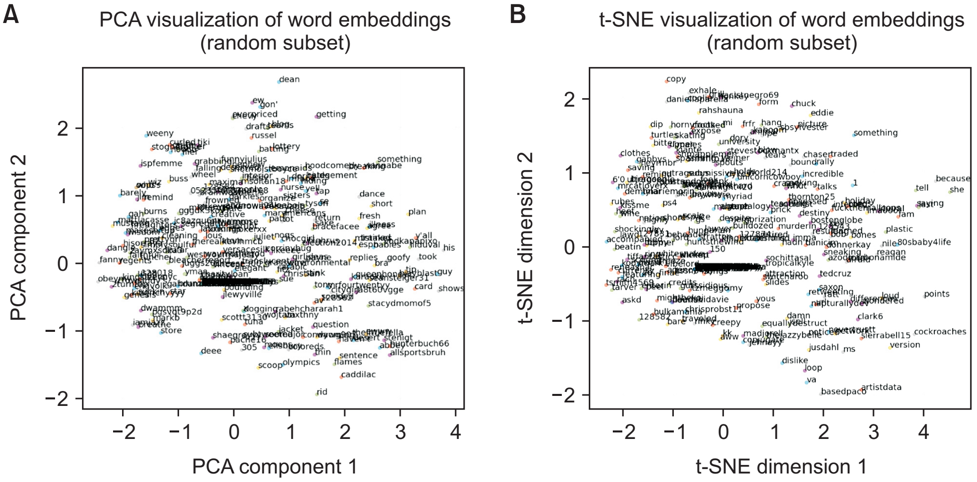


Fig. 2. Visualising embedding: (A) principal component analysis (PCA); (B) t-distributed stochastic neighbor embedding (t-SNE).

racial slurs, offensive terms, and expressions of hate may cluster together. t-SNE emphasizes the preservation of local connections, resulting in compact groupings of related words and facilitating the identification of words with similar meanings or usage contexts.

3.3.2. Bi-LSTM for Hate Speech Detection

In this paper, we propose a Bi-LSTM-based hate speech detection model responsible for capturing the sentiment within the text. Table 1 represents the model summary. It consists of two LSTM layers stacked specially. In forward LSTM, the layer processes the sequence of embedding vectors in a forward direction, capturing the sequential relationships between words from the beginning to the end of the sentence (Equation (1)-(3)). At each time step t in the sequence, the forward LSTM takes the previous hidden state h_{t-1}, the current input embedding x_t from the embedding layer, and a cell state c_{t-1} as input. The forget gate f_t, input gate i_t, and output gate o_t are calculated using sigmoid activation functions.

$$f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f \tag{1}$$

$$i_t = \sigma(W_i \times [h_{(t-1)}, x_t] + b_i) \tag{2}$$

$$o_t = \sigma(W_o \times \left[h_{(t-1)}, x_t\right] + b_o) \tag{3}$$

Where σ is the sigmoid function, W_f, W_i, and W_o are the weight matrices, and b_f, b_i, and b_o are the bias vectors.

Table 1. Bidirectional long-short-term memory model summary

Layer (type)	Output shape	Param#
embedding (Embedding)	(None, None, 100)	2,000,000
dropout (Dropout)	(None, None, 100)	0
bidirectional (Bidirectional)	(None, None, 64)	34,048
dropout_1 (Dropout)	(None, None, 64)	0
bidirectional_1 (Bidirectional)	(None, 32)	10,368
dropout_2 (Dropout)	(None, 32)	0
dense (Dense)	(None, 16)	528
dropout_3 (Dropout)	(None, 16)	0
dense_1 (Dense)	(None, 16)	272
dropout_4 (Dropout)	(None, 16)	0
dense_2 (Dense)	(None, 3)	51
Total params: 2,045,267 (7.80 MB) Trainable params: 45,267 (176.82 KB) Non-trainable params: 2,000,000 (7.63 MB)		

c_t is the candidate cell state computed using *tanh* activation function as Equation (4) and updated as Equation (5). Hidden states are updated as Equation (6).

$$c_t = \tanh(W_c \times [h_{t-1}, x_t] + b_c) \tag{4}$$

$$c_t = f_t \times c_{t-1} + i_t \times c_t \tag{5}$$

$$h_t = o_t \times \tanh(c_t) \tag{6}$$

In *backward LSTM*, the layer processes the same sequence of embeddings in a backward direction, understanding the relationships from the end toward the beginning. By combining the outputs of both LSTMs, the model can learn long-term dependencies within the text. This is crucial for sentiment analysis, as the context of surrounding words can influence sentiment. The equations have a similar structure with different weight and bias matrices specific to the backward direction W'_f, W'_i, etc.

3.3.3. Dense Layer

The combined output from the Bi-LSTM is fed into a dense layer with a weight matrix W_d, a bias vector b_d, and a softmax activation function. The softmax function converts the output into a probability distribution over all possible sentiment labels (e.g., "hate speech," "offensive language," "neutral," and "positive"). Each element in the output vector represents the probability of the corresponding sentiment label for the input text sample. The output y is calculated as $y=softmax(W_d \times [h_f, h_b] + b_d)$, where *h_f* and h_b represent the final hidden states from the forward and backward LSTMs, respectively.

3.3.4. Categorical Crossentropy

Categorical cross-entropy (Ho & Wookey, 2020) is a loss function often employed in multi-class classification situations that combines the projected probability distribution from the Bi-LSTM model with the true probability distribution representing the sentiment labels. Loss is defined as

$$loss = -\sum(y_{true} \times \log\left(1 - y_{pred}\right))$$

where y_{true} represents the true label and y_{pred} defines the probability of the same label from the Bi-LSTM model. During training, the Bi-LSTM model seeks to reduce the total category cross-entropy loss across all training samples by modifying the model's weights and biases using an optimization technique.

4. EXPERIMENTATION AND RESULTS

4.1. Dataset Gathering and Collection Process

The dataset used in this study is provided by Davidson et al. (2017). The data compiled by *Hatebase.org* was searched for tweets with *hate speech* lexicon. These tweets were randomly sampled to form a 25k tweet dataset that CrowdFlower workers manually coded. The tweets were in three categories: hate speech, offensive but not hate speech, or neither offensive nor hate speech. The categorization did not solely rely on the words but also considered their context. Three or more people labeled each tweet, and the final decision was made based on a majority decision. Some tweets were not labeled because of absentia of a majority decision. The process resulted in 24,802 tweets, of which 5% were labeled as hate speech, with 76% at 2/3 measure and 53% meeting a 3/3 measure of offensive language. The remainder were non-offensive. Fig. 3 depicts the statistics of the dataset.

4.2. Experimental Setup

With a 64-bit Windows operating system installed, the experiment was conducted on an Intel(R) Core (TM) i7-6700 CPU @ 3.40GHz 3.41 GHz processor. It has a 16 GB memory capacity. In Python 3.7, the Jupyter Notebook environment was used to program the code. Different libraries were used, such as sci-kit-learn for model analysis and computing performance, pandas for data manipulation and visualization, and numpy for computing operations. We have implemented TensorFlow 2.10, a DL framework. The Natural Language Toolkit Python package with punkt tokenizer was utilized to remove stop words.

4.3. Hyperparameter Tuning

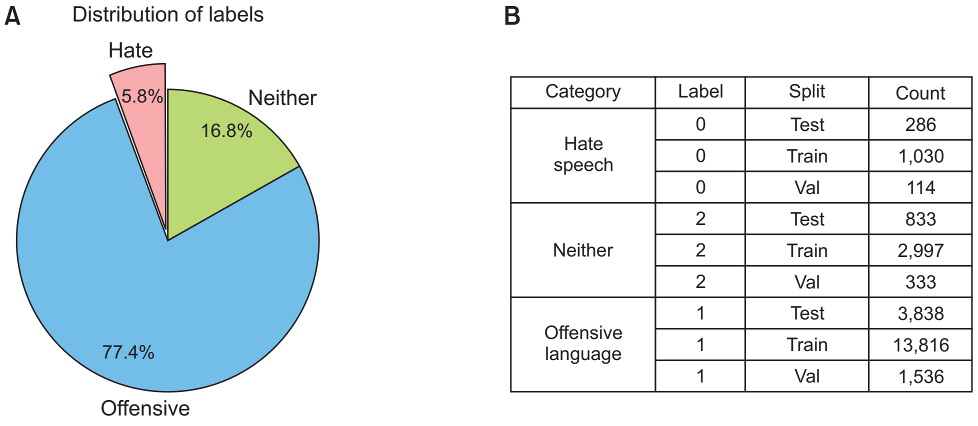
For experimentation purposes, we used batch size 64 and categorical cross-entropy for loss, and each model was optimized by Adam optimizer. We have used early stopping with patience 5 on validation accuracy to prevent overfitting. The embedding dimensions were set to 100. We selected a maximum of 20,000 features and 512 as text length to be considered on each tweet.

4.4. Evaluation Technique

The model's performance was evaluated based on precision, recall, and F1-score, as discussed in Equation (7)-(9). Besides these attributes, ML models have also been evaluated on micro and macro averages for precision, recall, and F1-score. The micro-average metrics are calculated by adding true-positive $True_{pos}$, $False_{Neg}$, and $False_{pos}$ occurrences of hate speech tweets in the predictions, independent of the instances of the classes. The macro average is the precision, recall, and F1-score average for different classes. As already stated, the dataset is imbalanced, and there are more non-hate speech tweets than hate speech tweets; thus, micro-averaging cannot depict the results of the minority class. We have also constructed a confusion matrix that records the number of *True Positives* ($True_{pos}$), *True Negatives* ($True_{Neg}$), *False Positives* ($False_{pos}$), and *False Negatives* ($False_{Neg}$) and in turn, represents the actual and predicted positives and negatives. Besides this, the qualitative analysis of performance has also been evaluated on accuracy measures as depicted in Equation (10).

a. Precision (P): The ratio of tweets classified as hate speech among the total retrieved tweets as represented by Equation (7).

$$Precision = \frac{True_{pos}}{True_{pos}+False_{pos}} \tag{7}$$



Category	Label	Split	Count
Hate speech	0	Test	286
	0	Train	1,030
	0	Val	114
Neither	2	Test	833
	2	Train	2,997
	2	Val	333
Offensive language	1	Test	3,838
	1	Train	13,816
	1	Val	1,536

Fig. 3. Dataset statistics: (A) Original dataset; (B) dataset after split.

b. Recall (R): This is the ratio of tweets defined truly as hate speech to the total number of hate speech tweets in ground truth (Equation (8)).

$$Recall = \frac{True_{pos}}{True_{pos}+False_{Neg}} \tag{8}$$

c. F1-Score (F1-score): F1-score is the harmonic mean of P and R depicted by Equation (9).

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

d. The confusion matrix comprises the following:

1. *True Positives* (*True$_{pos}$*): For a data point to be considered positive, its actual class and the prediction must be true.

2. *True Negatives* (*True$_{Neg}$*): For data points, a true negative occurs when the actual class and the forecast are incorrect.

3. *False Positives* (*False$_{pos}$*): A false positive is called a false positive when a data point's genuine prediction is based on an incorrect class.

4. *False Negatives* (*False$_{Neg}$*): An example of a false negative would be a data point whose real class contradicts the prediction.

Table 2 represents the various cells in the confusion matrix. The predicted positives and negatives are the predicted values by the models against actual values. A high *True$_{pos}$* indicates the high performance of the model, while a high *False$_{Neg}$* suggests that the model missed many instances of hate speech. Similarly, a high *False$_{pos}$* means incorrect flagging of non-hate speech content, and high *True$_{Neg}$* values indicate that the model is correctly identifying non-hate speech.

e. Accuracy: This is the ratio of the total number of true predictions (positives and negatives) to total predictions (true and false positives and negatives), as represented in Equation (10).

Table 2. Confusion matrix model

	Predicted positive	Predicted negative
Actually positive	*True Positive* (*True$_{pos}$*)	*False Negative* (*False$_{Neg}$*)
Actually negative	*False Positive* (*False$_{pos}$*)	*True Negative* (*True$_{Neg}$*)

f.

$$Accuracy = \frac{True_{pos}+True_{Neg}}{True_{pos}+True_{Neg}+False_{pos}+False_{Neg}} \tag{10}$$

4.5. Result Analysis

XG Boost emerged as the best performer among the ML variants. It scored an outstanding precision (class 0) of 99.19%, suggesting a high level of accuracy in accurately detecting non-hate speech occurrences, and a recall (class 0) of 97.22%, effectively catching the majority of genuine non-hate speech instances. For hate speech (class 1), the model attained a precision of 18.64%, indicating that it correctly predicts hate speech 18.64% of the time. The recall (class 1) for hate speech was 28.18%, implying that it correctly detected 28.18% of actual hate speech cases. DT demonstrated great accuracy but needs improvement in class 1 recall (30.69%). It struggled to catch every incident of hate speech. LR fared well for non-hate speech (class 0), with 98.63% precision and 96.92% recall. However, its performance in hate speech (class 1) was unsatisfactory, with low precision (17.92%) and recall (25.45%). RF, like LR, excelled at class 0 precision and recall. For hate speech (class 1), RF had a 23.66% precision and a 30.21% recall. SVM had the maximum precision for non-hate speech (class 0) at 99.38%. However, its recall for hate speech (class 1) was extremely low at 18.29%, implying that it missed many actual hate speech incidents. XG Boost exhibited a balanced approach with high accuracy (99.19%) for non-hate speech (class 0). Hate speech had a moderate recall rate (28.18%) in class 1. The F1-score reflects this balance, making XG Boost an attractive option. Table 3 represents the comparative results of different ML models.

The DL models were evaluated across training, validation, and test data sets, and their effectiveness was determined. Table 4 shows how well three models—CNN, LSTM, and Bi-LSTM—performed on various data partitions (train, validation, and test) using four metrics: accuracy, precision, recall, and F1-score. During training, the CNN model attained a notable accuracy of 96.48%. However, its performance declined on the validation set with an accuracy of about 85.35% on test sets, with precision, recall, and F1-score also indicating a similar decrease. The LSTM model showed good performance on the training set, achieving 94.92% accuracy, and continued to exhibit strong performance on the validation (89.18%) and test (89.15%) sets. LSTM achieved high precision, recall, and F1-score, with a slight decrease between training and testing. Consistently, the Bi-LSTM model surpassed both CNN and LSTM models, achieving the highest training

Table 3. Performance comparison of different ML models

Model	Class	Accuracy	Precision	Recall	F1-score
NB	0.0		0.956289461	0.42090637	0.584533175
	1.0		0.065217391	0.677419355	0.11898017
	Overall	0.435343958			
	Macro average		0.510753426	0.549162863	0.351756672
	Weighted avgerage		0.906136322	0.435343958	0.55832997
DT	0.0		0.959508938	0.952330056	0.955906019
	1.0		0.289808917	0.326164875	0.306913997
	Overall	0.917086948			
	Macro average		0.624658928	0.639247465	0.631410008
	Weighted avgerage		0.921815514	0.917086948	0.919378124
KNN	0.0		0.958018472	0.975630611	0.966744334
	1.0		0.409326425	0.283154122	0.334745763
	Overall	0.936655235			
	Macro average		0.683672448	0.629392367	0.650745048
	Weighted avgerage		0.927135865	0.936655235	0.931172899
LR	0.0		0.952715259	0.98631894	0.969225922
	1.0		0.438596491	0.17921147	0.254452926
	Overall	0.940891668			
	Macro average		0.695655875	0.582765205	0.611839424
	Weighted avgerage		0.923778576	0.940891668	0.928995608
RF	0.0		0.955615753	0.980333476	0.96781682
	1.0		0.417721519	0.23655914	0.302059497
	Overall	0.938470849			
	Macro average		0.686668636	0.608446308	0.634938158
	Weighted avgerage		0.925340891	0.938470849	0.930345306
SVM	0.0		0.949356749	0.99380077	0.971070496
	1.0		0.516666667	0.111111111	0.182890855
	Overall	0.944119427			
	Macro average		0.733011708	0.55245594	0.576980676
	Weighted avgerage		0.925003202	0.944119427	0.926708559
XG Boost	0.0		0.953359359	0.99187687	0.972236773
	1.0		0.577777778	0.186379928	0.281842818
	Overall	0.946540246			
	Macro average		0.765568568	0.589128399	0.627039796
	Weighted avgerage		0.932220109	0.946540246	0.93337861

NB, naive Bayes; DT, decision tree; KNN, K-nearest neighbours; LR, logistic regression; RF, random forest; SVM, support vector machine; XG Boost, Extreme Gradient Boosting.

Table 4. Performance comparison of deep learning models on train, validation, and test set, respectively

Model	Split	Accuracy	Precision	Recall	F1-score
CNN	Train	0.9648	0.9672	0.9639	0.9655
	Validation	0.8535	0.8549	0.8513	0.8513
	Test	0.85031	0.84908	0.85164	0.85034
LSTM	Train	0.9492	0.9522	0.9457	0.9489
	Validation	0.8918	0.8946	0.8903	0.8924
	Test	0.89146	0.888243	0.894047	0.891078
Bi-LSTM	Train	0.9651	0.9674	0.9632	0.9653
	Validation	0.8901	0.8927	0.8886	0.8906
	Test	0.89449	0.892456	0.896078	0.894235

CNN, convolutional neural network; LSTM, long-short-term memory; Bi-LSTM, bidirectional long-short-term memory.

accuracy of 96.51%, impressive validation accuracy of 89.01%, and test accuracy of 89.45%. The Bi-LSTM model had the highest precision, recall, and F1-score values, showing balanced performance and an increase in recall (89.61%) on the test set, indicating its superior ability to detect more true positives than other models.

From the results, we can deduce that CNN achieved competing results. Its high accuracy means that it correctly distinguishes a large proportion of hate speech and non-hate speech incidents. However, we find a trade-off between precision and recall. While the precision for non-hate speech is high, the recall for hate speech is lower. This shows that while CNN is good at identifying non-hate speech, it may overlook some cases of hate speech. LSTM regularly outperforms across all criteria. Its balanced precision and recall for both classes show that it balances erroneous positives and false negatives. However, LSTM's accuracy is slightly lower than that of CNN. Table 4 represents the performance of different DL models.

The performance of different ML-based classifiers can be visualized using the confusion matrix. From Fig. 4, we can deduce that the DT model correctly predicted TP instances (true positives) and TN instances (true negatives). It made some FP errors (false positives) and a few FN errors (false negatives). Overall, the DT model shows a balanced performance. The KNN model achieved high TP and TN counts and had minimal FP and FN errors. Thus, it exhibits robust classification. The LR model performed well regarding TP and TN but had a moderate number of FP and FN errors. The LR model is reliable but not perfect. The NB model had a high TN count but relatively low TP, and it made several FP errors. Thus, it struggles with sensitivity. The RF model excelled in both TP and TN and had minimal FP and FN errors. Thus, it turns out to be a potential second-best performer. The SVM model achieved high TN but relatively low TP and can be tuned for better sensitivity. The XG Boost model had the highest TP count and made few FP and FN errors. Thus, the XG Boost model is the most accurate among the models.

Fig. 5 depicts the confusion matrices for DL models. The LSTM accurately predicted 95 instances of label 0 but misclassified 112 instances of label 1 as label 0, and 16 cases of label 2 as label 0. It correctly identified label 1 173 times while incorrectly predicting label 0 101 times. It exhibited a higher accuracy for label 2, with 698 correct predictions. Bi-LSTM accurately recognized label 0 in 163 instances but misidentified it as label 1 108 times and label 2 2 times. Label 1 had 115 valid identifications, while label 2 had 693 appropriate classifications. CNN outperformed LSTM and Bi-LSTM, with 216 correct predictions. However, its accuracy for label 0 (53 correct identifications) was lower than LSTM (95) and Bi-LSTM (163). For label 2, CNN made 541 right predictions. Thus, LSTM excelled at predicting label 2, whereas Bi-LSTM performed very well for both label 2 and label 1. CNN outperformed the other algorithms in identifying label 1 but struggled with label 0. Differences in performance could be related to architectural variances and training information.

Fig. 6 shows the variation in training and loss across epochs for CNN, LSTM, and Bi-LSTM models. CNN's training loss lowers steadily as the number of epochs grows. This shows that the model is effectively learning from the training data. CNN's training accuracy gradually improves, indicating that it accurately predicts more instances as training advances. The training loss of LSTMs diminishes with epochs; however, it is more pronounced

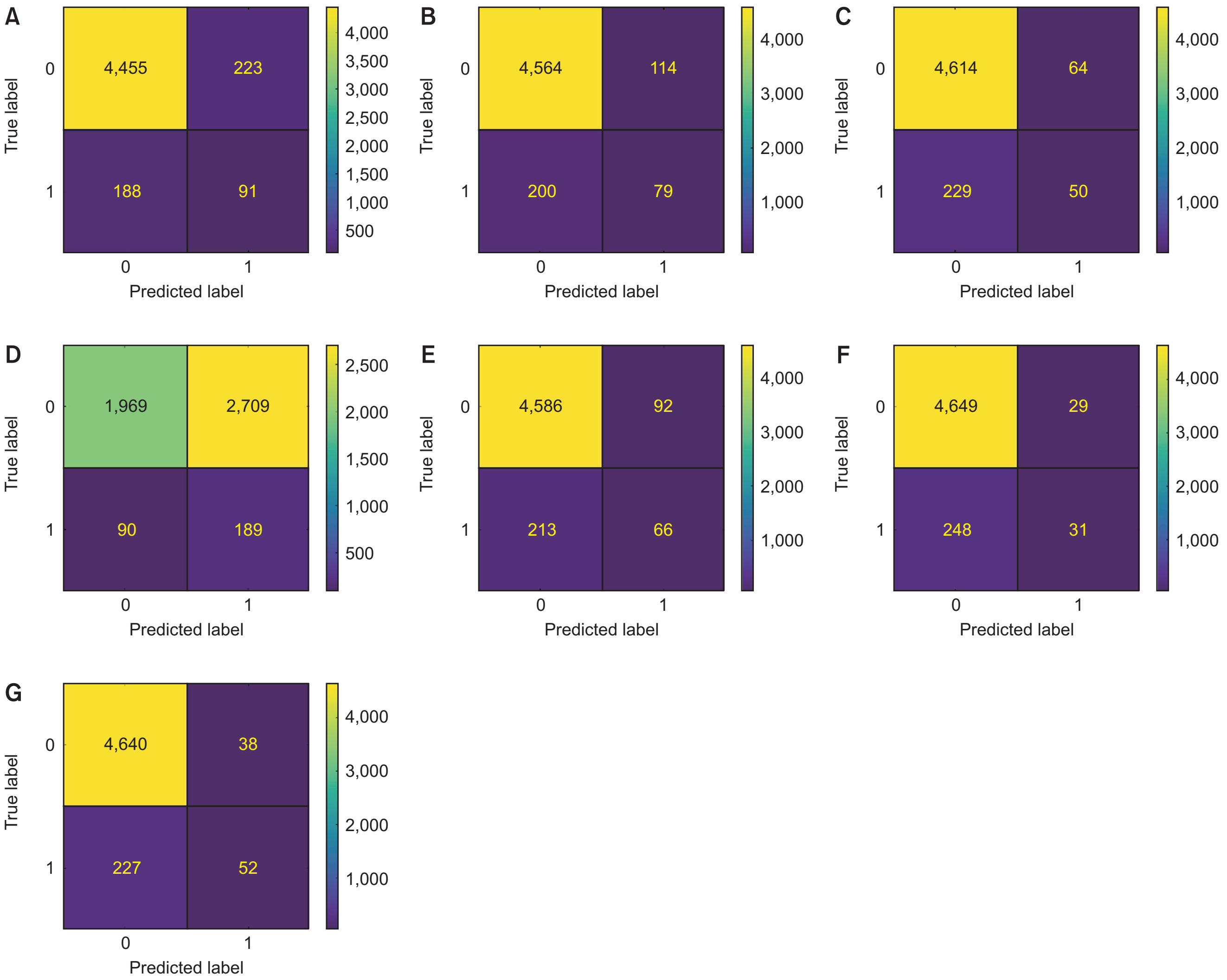


Fig. 4. Confusion matrices for machine learning models: (A) Decision tree; (B) K-nearest neighbours; (C) logistic regression; (D) naive Bayes; (E) random forest; (F) support vector machine; (G) Extreme Gradient Boosting.

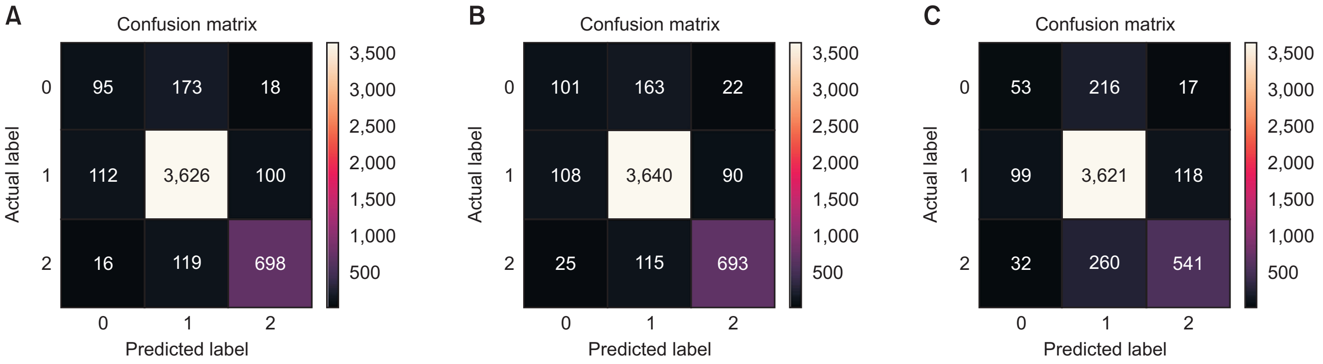


Fig. 5. Confusion matrix of deep learning models: (A) Long-short-term-memory; (B) bidirectional long-short-term memory; (C) convolutional neural network.

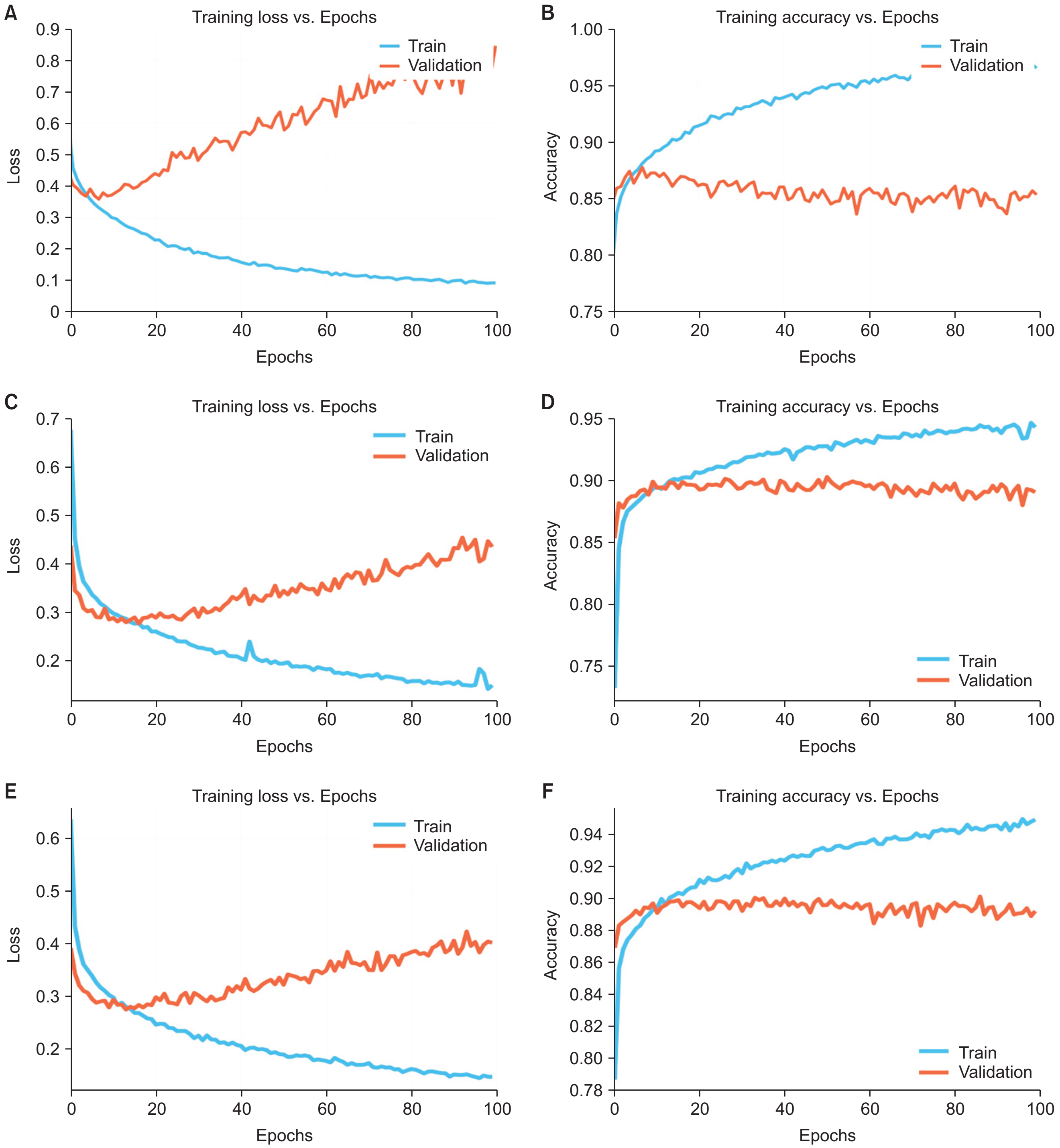


Fig. 6. Training and loss variation with epochs: (A), (B) convolutional neural network; (C), (D) long-short-term-memory; (E), (F) bidirectional long-short-term memory.

than that of CNNs. This indicates that the LSTM is learning efficiently. LSTM training accuracy improves similarly to CNN, albeit with a slightly steeper gradient. Bi-LSTM's training loss is rapidly decreasing, indicating good learning. It converges faster than either CNN or LSTM. Bi-LSTM's training accuracy rapidly increases, outperforming the other models. It reaches high accuracy. During training, the Bi-LSTM model outperforms other models in terms of accuracy and loss reduction. LSTM follows closely, with CNN trailing significantly.

4.6. Discussion

This experiment examined the efficacy of various DL architectures for detecting hate speech in text data. We evaluated the performance of a CNN, LSTM, and Bi-LSTM model. To isolate the impact of the architecture, all models were trained using the same hyperparameters. The Bi-LSTM model outperformed the CNN and LSTM algorithms when assessing hate speech. This shows that Bi-LSTM's capacity to identify long-term dependencies in both directions of text sequences may be critical for accurately detecting hate speech.

CNN underperformed compared to Bi-LSTM in hate speech detection. CNNs handle text data with convolutional filters that have a fixed window size. This window captures local word associations but may struggle with hate speech that relies on context dispersed throughout the text. Sarcasm or implied hate speech, for example, may necessitate understanding the overall sentiment of the sentence rather than just a few adjacent words. Standard CNNs analyze text in a single direction (often left to right). Hate speech frequently uses subtle wordings and word placement to communicate its vile meaning. Bi-LSTMs, which can evaluate text in both directions, can better capture these sentence-level order-dependent correlations.

This research enhances metadata tagging systems and information retrieval frameworks by ensuring that the Bi-LSTM model can detect subtle linguistic patterns, including slang and implicit bias. This work shows how ML models can be included in more comprehensive content organization systems, especially in the information science domain, going beyond recall and precision metrics. The model's capacity to differentiate between neutral content, hate speech, and inflammatory language has immediate applications for social media platforms and digital archives looking to improve their metadata systems. These findings highlight the difficulties in detecting hate speech in the real world, including language diversity, irony, and the dynamic nature of online debate. This model provides a scalable approach to these problems by detecting hate speech and classifying it to facilitate content retrieval, analysis, and moral moderation.

Fig. 7 compares the accuracy of hate speech detection on ML and DL models. This distribution highlights the trade-offs between simplicity and performance in ML models. While ML approaches are interpretable and computationally efficient, they struggle with noisy social media data because of their simple feature representations and inability to handle context. DL models use neural networks to learn relevant features from raw text data, making them capable of capturing complicated patterns and contexts. They excel at comprehending context, processing loud text, and detecting subtle linguistic signs. Looking at it from the viewpoint of information science, this automation does not just make content moderation more efficient, but boosts information retrieval systems. Organizing content by how harmful it is, these models help improve search results and suggestions, lowering the likelihood of users coming across dangerous material. Furthermore, the capability to identify and categorize hate speech guarantees that data stays compliant with ethical

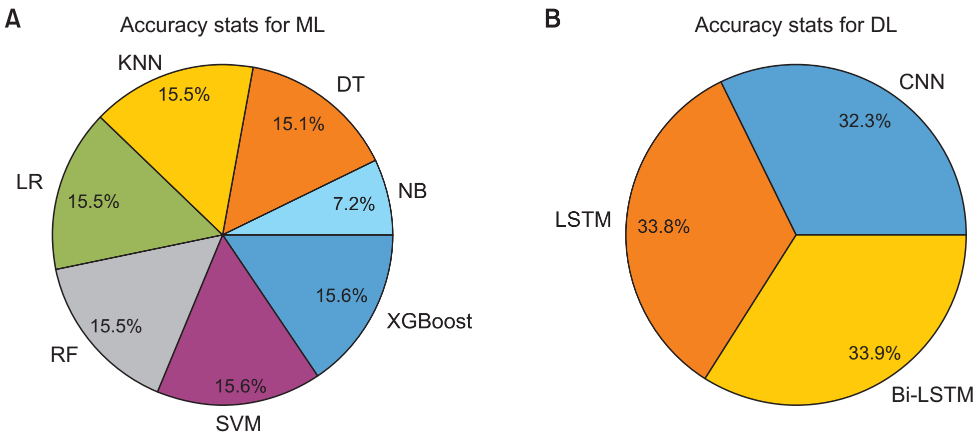


Fig. 7. Accuracy comparison for machine learning (ML) and DL methods. KNN, K-nearest neighbours; DT, decision tree; NB, naive Bayes; XG Boost, Extreme Gradient Boosting; SVM, support vector machine; RF, random forest; LR, logistic regression; CNN, convolutional neural network; Bi-LSTM, bidirectional long-short-term memory; LSTM, long-short-term memory.

and legal norms, contributing to upholding the integrity and reliability of the data in these systems. Therefore, hate speech detection based on ML is in line with the objectives of information science, promoting the creation of secure and structured digital spaces while ensuring data integrity and fairness.

5. FUTURE STUDIES

Several challenges in hate speech detection plague information systems, as listed below:

5.1. Cultural Variability

Each culture's definition of hate speech is influenced by its distinctive linguistic nuances, social norms, and historical contexts. This diversity requires the creation of flexible detection systems that can identify and understand culturally unique phrases and insults. For example, a term considered offensive in one culture could be deemed appropriate in another, resulting in potential misidentifications. Moreover, with the development of online communication, different slang and references constantly appear, often differing greatly depending on the community. Information science needs to focus on context-aware algorithms that consider cultural factors to improve accuracy and relevance in hate speech detection. In the end, focusing on cultural differences will enhance the efficiency of information retrieval systems and help create fairer digital spaces.

5.2. Evolving Nature of Online Language

The changing online language greatly affects the detection of hate speech, especially in information retrieval systems. Detecting offensive language is made difficult by the emergence of new slang, acronyms, and rapidly evolving terminologies on social media platforms such as Twitter and Facebook. Models must constantly adjust to new expressions and cultural contexts to keep up with this dynamic evolution, as once harmless language may now be linked to hate speech. Moreover, the complexity is increased by multilingualism and code-switching, which involve mixing languages, making it challenging for conventional keyword-based systems. To tackle this issue, information retrieval systems must include real-time updates and models aware of context. Utilizing methods like transfer learning and cross-linguistic training aids in maintaining the effectiveness of these systems. Adjusting to the constantly changing nature of online language helps categorize harmful content more accurately and enhance user experiences.

5.3. Transfer Learning Across Languages

In information science, transfer learning is crucial for surpassing language obstacles and improving model generalization. Transfer learning enables efficient identification of hate speech in various linguistic environments by utilizing pre-trained models from well-resourced languages like English and adjusting them with data from limited resources. This is especially beneficial because hate speech can manifest uniquely depending on cultural and regional influences in different languages. It helps adjust detection systems for poorly represented languages, ensuring models are more inclusive and can be used worldwide. Furthermore, exposing hate speech classifiers to different linguistic structures and expressions can enhance their robustness through cross-linguistic transfer. This aligns with information science objectives, aimed at creating more accessible, equitable, and effective systems for retrieving information and moderating content. Future research can concentrate on creating multilingual datasets and use of transfer learning methods to improve the scalability and efficiency of hate speech detection in various languages.

5.4. Lack of Annotated Data

It is often difficult to find high-quality, annotated hate speech datasets in many languages, particularly for languages that are not well-represented. Insufficient training data may limit the model's generalization capacity across different languages. A model trained on English data might not work effectively in languages with fewer resources, where the labeling of offensive material may be scarce or unreliable. Hence, it is a potential future direction to work on expanding the annotated data.

5.5. Ethical Concerns

Furthermore, addressing the ethical implications of automated hate speech identification, such as privacy problems and the possibility of censorship, is critical. Incorporating user feedback loops can increase the system's accuracy over time, transforming it into a dynamic tool for developing methods against hate speech.

6. CONCLUSION

In social media, the proliferation of hate speech has become a pressing issue, necessitating the development of sophisticated algorithms capable of accurately detecting and mitigating such content. The comparative analysis of

ML models for tweet hate speech detection revealed that DL models, particularly the Bi-LSTM network, offer superior performance over traditional ML approaches. The Bi-LSTM model's proficiency stems from its unique architecture, which processes data sequences in both forward and backward directions, thereby capturing contextual information from all temporal dependencies within the text. This bidirectional processing is particularly advantageous for understanding the nuanced semantics and complex structures of natural language, which are characteristic of social media communication.

Several metrics have been considered when assessing the performance of various models. The accuracy of the Bi-LSTM model in classifying tweets as hate speech or non-hate speech has consistently outperformed other models. This high accuracy rate indicates the model's ability to discern subtle linguistic cues and patterns that distinguish hate speech from benign expressions. By enabling digital libraries, social media platforms, and other information systems to better categorize hate speech, the Bi-LSTM model created in this study contributes to the ethical management and general accessibility of digital content. This research emphasizes the valuable effects of these technologies on content management and retrieval. As the landscape of online discourse continues to change, it provides a flexible and scalable strategy that is in keeping with the larger objectives of information science.

Moreover, the training and validation loss graphs for the Bi-LSTM model demonstrate a rapid convergence to a lower loss value, suggesting that the model is learning effectively and generalizing well to unseen data. This is further corroborated by the stability of the learning curve, which exhibits minimal fluctuations in accuracy and loss across epochs, indicating a robust model less prone to overfitting. The efficiency of the Bi-LSTM model is also noteworthy. Despite the inherent complexity of processing natural language data, the model's architecture enables it to handle large volumes of text while maintaining high computational efficiency. This is crucial for real-time applications where timely detection of hate speech is essential. By leveraging the Bi-LSTM model, social media platforms can filter out hate speech more effectively, creating a safer and more inclusive online environment. This enhances the user experience and aligns with the ethical and legal standards governing digital communication. As the digital landscape continues to evolve, the Bi-LSTM model's adaptability and scalability will be instrumental in addressing the challenges of moderating content on social media platforms.

Future research in hate speech detection with DL could greatly benefit from advances in transformer-based models such as bidirectional encoder representations from transformers, which have demonstrated exceptional success in detecting context and nuances in text. These models might be fine-tuned to detect hate speech, improve accuracy, and lower false positives. Another area of focus could be creating multimodal models that consider not only textual content but also images, videos, and user metadata. This holistic approach may provide a more complete comprehension of the content and its intent. Furthermore, creating larger and more diverse datasets encompassing numerous modes of communication, platforms, and languages would aid in developing more robust models. Collaboration with social media networks to access real-time data could improve the models' usefulness.

CONFLICTS OF INTEREST

The authors of this manuscript state that they have no conflict of interest.

ACKNOWLEDGMENTS

This work was supported by the Department of Science and Technology India, International Cooperation Division (Grant No: DST/INT/BLR/P-38/2023(G)) and the Belarusian Republican Foundation for Fundamental Research (Agreement No. F23INDG-011).

REFERENCES

Abozinadah, E. A., Mbaziira, A. V., & Jones, J. H., Jr. (2015). Detection of abusive accounts with Arabic Tweets. *International Journal of Knowledge Engineering*, 1(2), 113-119. https://www.ijke.org/show-36-54-1.html

Agarwal, S., & Chowdary, C. R. (2021). Combating hate speech using an adaptive ensemble learning model with a case study on COVID-19. *Expert Systems with Applications*, 185, 115632. https://doi.org/10.1016/j.eswa.2021.115632

Alakrot, A., Murray, L., & Nikolov, N. S. (2018). Dataset construction for the detection of anti-social behaviour in online communication in Arabic. *Procedia Computer Science*, 142, 174-181. https://doi.org/10.1016/j.procs.2018.10.473

Alfina, I., Mulia, R., Fanany, M. I., & Ekanata, Y. (2017, October 28-29). Hate speech detection in the Indonesian language: A dataset and preliminary study. *Proceedings of the 2017 International Conference on Advanced Computer Science* *and Information Systems* (pp. 233-238). IEEE.

Al-Hassan, A., & Al-Dossari, H. (2019, February 23-24). Detection of hate speech in social networks: A survey on multilingual corpus. In D. Nagamalai, & D. C. Wyld (Eds.), *Proceedings of the 6th International Conference on Computer Science and Information Technology* (pp. 83-100). CS & IT-CSCP.

Andrusyak, B., Rimel, M., & Kern, R. (2018, December 7-9). Detection of abusive speech for mixed sociolects of Russian and Ukrainian languages. In A. Horák, P. Rychlý, & A. Rambousek (Eds.), *Proceedings of the 12th Workshop on Recent Advances in Slavonic Natural Language Processing (RASLAN 2018)* (pp. 77-84). Tribun EU.

Berrar, D. (2019). Bayes' theorem and naive bayes classifier. In S. Ranganathan, M. Gribskov, K. Nakai, & C. Schönbach (Eds.), *Encyclopedia of bioinformatics and computational biology* (Vol. 1, pp. 403-412). Elsevier.

Cao, R., Lee, R. K. W., & Hoang, T. A. (2020, July 6-10). DeepHate: Hate speech detection via multi-faceted text representations. *Proceedings of the 12th ACM Conference on Web Science* (pp. 11-20). ACM.

Chen, T., & Guestrin, C. (2016, August 13-17). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794). ACM.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated hate speech detection and the problem of offensive language. *Proceedings of the International AAAI Conference on Web and Social Media,* 11(1), 512-515. https://doi.org/10.1609/icwsm.v11i1.14955

Del Vigna, F., Cimino, A., Dell'Orletta, F., Petrocchi, M., & Tesconi, M. (2017, January 17-20). Hate me, hate me not: Hate speech detection on Facebook. In A. Armando, R. Baldoni, & R. Focardi (Eds.), *Proceedings of the 1st Italian Conference on Cybersecurity (ITASEC17)* (pp. 86-95). CEUR.

Fortuna, P., & Nunes, S. (2019). A survey on automatic detection of hate speech in text. *ACM Computing Surveys,* 51(4), 85. https://doi.org/10.1145/3232676

Haidar, B., Chamoun, M., & Serhrouchni, A. (2017). A multilingual system for cyberbullying detection: Arabic content detection using machine learning. *Advances in Science, Technology and Engineering Systems Journal,* 2(6), 275-284. https://doi.org/10.25046/aj020634

Ho, Y., & Wookey, S. (2020). The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. *IEEE Access,* 8, 4806-4813. https://doi.org/10.1109/ACCESS.2019.2962617

Kheddar, H., Himeur, Y., Al-Maadeed, S., Amira, A., & Bensaali, F. (2023). Deep transfer learning for automatic speech recognition: Towards better generalization. *Knowledge-Based Systems, 277*, 110851. https://doi.org/10.1016/j.knosys.2023.110851

Köffer, S., Riehle, D. M., Höhenberger, S., & Becker, J. (2018). *Discussing the value of automatic hate speech detection in online debates*. Paper presented at the Multikonferenz Wirtschaftsinformatik 2018 (MKWI 2018): Data Driven X-Turning Data in Value, Lüneburg, Germany.

Kshirsagar, R., Cukuvac, T., McKeown, K., & McGregor, S. (2018, October 31). Predictive embeddings for hate speech detection on Twitter. In D. Fišer, R. Huang, V. Prabhakaran, R. Voigt, Z. Waseem, & J. Wernimont (Eds.), *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)* (pp. 26-32). Association for Computational Linguistics.

Li, C., Zhan, G., & Li, Z. (2018, October 19-21). News text classification based on improved Bi-LSTM-CNN. *Proceedings of 9th International Conference on Information Technology in Medicine and Education* (pp. 890-893). IEEE.

Liu, L., Chen, J., Fieguth, P., Zhao, G., Chellappa, R., & Pietikäinen, M. (2019). From BoW to CNN: Two decades of texture representation for texture classification. *International Journal of Computer Vision*, 127(1), 74-109. https://doi.org/10.1007/s11263-018-1125-z

Miškolci, J., Kováčová, L., & Rigová, E. (2020). Countering hate speech on Facebook: The case of the Roma minority in Slovakia. *Social Science Computer Review,* 38(2), 128-146. https://doi.org/10.1177/0894439318791786

Mocanu, D., Rossi, L., Zhang, Q., Karsai, M., & Quattrociocchi, W. (2015). Collective attention in the age of (mis)information. *Computers in Human Behavior,* 51(Pt B), 1198-1204. https://doi.org/10.1016/j.chb.2015.01.024

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016, April 11-15). Abusive language detection in online user content. *Proceedings of the 25th International Conference on World Wide Web* (pp. 145-153). ACM.

Patle, A., & Chouhan, D. S. (2013, January 23-25). SVM kernel functions for classification. *Proceedings of the 2013 International Conference on Advances in Technology and Engineering* (pp. 1-9). IEEE.

Pérez, J. M., Luque, F. M., Zayat, D., Kondratzky, M., Moro, A., Serrati, P. S., Zajac, J., Miguel, P., Debandi, N., Gravano, A., & Cotik, V. (2023). Assessing the impact of contextual information in hate speech detection. *IEEE Access,* 11, 30575-30590. https://doi.org/10.1109/ACCESS.2023.3258973

Pitsilis, G. K., Ramampiaro, H., & Langseth, H. (2018). Effective hate-speech detection in Twitter data using recurrent neural networks. *Applied Intelligence,* 48(12), 4730-4742. https://doi.org/10.1007/s10489-018-1242-y

Rahman, M. M., Balakrishnan, D., Murthy, D., Kutlu, M.,

& Lease, M. (2021). An information retrieval approach to building datasets for hate speech detection. *arXiv*, 2106.09775. https://doi.org/10.48550/arXiv.2106.09775

Rigatti, S. J. (2017). Random forest. *Journal of Insurance Medicine*, 47(1), 31-39. https://doi.org/10.17849/insm-47-01-31-39.1

Saha, P., Mathew, B., Goyal, P., & Mukherjee, A. (2018). Hateminers: Detecting hate speech against women. *arXiv*, 1812.06700. https://doi.org/10.48550/arXiv.1812.06700

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. https://doi.org/10.1186/s13673-019-0205-6

Sap, M., Card, D., Gabriel, S., Choi, Y., & Smith, N. A. (2019, July 28-August 2). The risk of racial bias in hate speech detection. In A. Korhonen, D. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 1668-1678). Association for Computational Linguistics.

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, 24(1), 12-18. https://doi.org/10.11613/BM.2014.003

Srba, I., Lenzini, G., Pikuliak, M., & Pecar, S. (2021). Addressing hate speech with data science: An overview from computer science perspective. In S. Wachs, B. Koch-Priewe, & A. Zick (Eds.), *Hate speech - Multidisziplinäre analysen und handlungsoptionen* (pp. 317-336). Springer.

Staudemeyer, R. C., & Morris, E. R. (2019). Understanding LSTM -- A tutorial into long short-term memory recurrent neural networks. *arXiv*, 1909.09586. https://doi.org/10.48550/arXiv.1909.09586

Tontodimamma, A., Nissi, E., Sarra, A., & Fontanella, L. (2021). Thirty years of research into hate speech: Topics of interest and their evolution. *Scientometrics*, 126(1), 157-179. https://doi.org/10.1007/s11192-020-03737-6

Waseem, Z., Davidson, T., Warmsley, D., & Weber, I. (2017). Understanding abuse: A typology of abusive language detection subtasks. *arXiv*, 1705.09899. https://doi.org/10.48550/arXiv.1705.09899

Waseem, Z., & Hovy, D. (2016, June 12-17). Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In J. Andreas, E. Choi, & A. Lazaridou (Eds.), *Proceedings of the NAACL Student Research Workshop* (pp. 88-93). Association for Computational Linguistics.

Watanabe, H., Bouazizi, M., & Ohtsuki, T. (2018). Hate speech on Twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. *IEEE Access*, 6, 13825-13835. https://doi.org/10.1109/ACCESS.2018.2806394

Wiegand, M., Ruppenhofer, J., Schmidt, A., & Greenberg, C. (2018, June 1-6). Inducing a lexicon of abusive words – A feature-based approach. In M. Walker, H. Ji, & A. Stent (Eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 1046-1056). Association for Computational Linguistics.

Zhang, S., Li, X., Zong, M., Zhu, X., & Wang, R. (2018). Efficient KNN classification with different numbers of nearest neighbors. *IEEE Transactions on Neural Networks and Learning Systems*, 29(5), 1774-1785. https://doi.org/10.1109/TNNLS.2017.2673241

Zhang, Z., & Luo, L. (2019). Hate speech detection: A solved problem? The challenging case of long tail on Twitter. *Semantic Web*, 10(5), 925-945. https://semantic-web-journal.net/content/hate-speech-detection-solved-problem-challenging-case-long-tail-twitter-1