PeerJ Computer Science

# Detecting hate speech in roman Urdu using a convolutional-BiLSTM-based deep hybrid neural network

Muhammad Zohaib[1], Ghulam Farooque[2], Mohammad Alsulami[3,4], Fazeel Abid[2], Ali Alqazzaz[5], Mana Saleh Al Reshan[4,6], Jawad Rasheed[7,8,9] and Asadullah Shaikh[4,6]

[1] Information Systems, University of Management & Technology, Lahore, Lahore, Pakistan
[2] Computer Science and Information Technology, University of Lahore, Lahore, Pakistan
[3] Computer Science, Najran University, Najran, Saudi Arabia
[4] Emerging Technologies Research Lab (ETRL), Najran University, Najran, Saudi Arabia
[5] College of Computing and Information Technology, University of Bisha, Bisha, Saudi Arabia
[6] Information Systems, Najran University, Najran, Saudi Arabia
[7] Software Engineering, Istanbul Nisantasi University, Istanbul, Turkey
[8] Applied Science Research Center, Applied Science Private University, Amman, Jordan
[9] Research Institute, Istanbul Medipol University, Istanbul, Turkey

## ABSTRACT

The detection of hate speech on social media has become a pressing challenge, particularly in multilingual and low-resource language settings such as Roman Urdu, where informal grammar, code-switching, and inconsistent orthography hinder accurate classification. Despite progress in hate speech detection for high-resource languages, limited research exists for Roman Urdu content. This study addresses this gap by proposing a computationally efficient deep learning framework based on a hybrid convolutional neural network and bidirectional long short-term memory (CNN-BiLSTM) architecture. The model leverages FastText pre-trained embeddings to capture subword-level semantics and combines convolutional layers for local feature extraction with BiLSTM for global context modeling. We evaluate our approach on a labeled Roman Urdu dataset and compare it with traditional machine learning models and deep learning baselines. Our proposed CNN-BiLSTM model achieves the highest performance with an accuracy of 80.67% and an F1-score of 81.47%, outperforming competitive baselines. These findings demonstrate the effectiveness and practicality of our lightweight architecture in detecting hate speech in Roman Urdu, offering a scalable solution for multilingual and resource-constrained environments.

**Subjects** Data Mining and Machine Learning, Natural Language and Speech, Text Mining, Neural Networks

**Keywords** Hybrid neural network, Roman Urdu, Deep learning, Short-term memory, Convolutional layers, Text data

Submitted 5 May 2025
Accepted 8 October 2025
Published 3 November 2025

Corresponding author
Mohammad Alsulami,
mmalsulami@nu.edu.sa

Academic editor
Siddhartha Bhattacharyya

Additional Information and Declarations can be found on page 19

DOI **10.7717/peerj-cs.3342**



## INTRODUCTION

The number and popularity of social networking websites on the internet have significantly increased during the past ten years, resulting in an exponential hype in the number of users. These websites offer users the promising freedom to share their thoughts and engage with others from different backgrounds, leading to the formation of relationships and the

**How to cite this article** Zohaib M, Farooque G, Alsulami M, Abid F, Alqazzaz A, Reshan MSA, Rasheed J, Shaikh A. 2025. Detecting hate speech in roman Urdu using a convolutional-BiLSTM-based deep hybrid neural network. **PeerJ Comput. Sci. 11:e3342 DOI 10.7717/peerj-cs.3342**

exchange of ideas (*Rizwan, Shakeel & Karim, 2020*). Conversely, hostile, offensive, derogatory, or obscene language is utilized to disseminate, provoke, incite, or justify hate, violence, and prejudice between individuals based on their ethnicity, religion, gender, association with specific organizations, or opinions on particular events or topics, such as politics. Failure to address this type of content has been shown to cause violent acts and more serious confrontations. This renders it impractical to uphold civil rights, legal frameworks, and expression, all of which are essential for the establishment of a non-discriminatory open to democracy culture (*Khan, Shahzad & Malik, 2021*).

Most social media companies depend on information reporting and manual review by human personnel. However, this approach is limited by the speed of the reviewers, their understanding of emerging slang, jargon, and multilingual content, as well as their experience with such content (*Mahmood et al., 2020*). In addition to these challenges, it is worth noting that by the time the manual review process is carried out—a process that typically takes up to 24 h—the targeted harm may have already occurred. Furthermore, the subjective nature of determining what language is considered offensive and constitutes hate speech raises concerns about the potential for the manual procedure to be misused to silence minority groups and refrain from criticizing government actions, political opponents, and religious convictions. Hence, there is a need for the support the advancement of technologies capable of automatically recognizing inappropriate language and hate speech (*Kovács, Alonso & Saini, 2021*).

Recent incidents in Pakistan, including the execution of a student who was subjected to anti-religious propaganda on the internet, efforts to discredit prominent politicians and social media influencers, and the frequent targeting and abuse of women who express their opinions online, have prompted the government to enact legislation prohibiting hate speech on the internet. In response to attempts to target religious minorities and cause offense to their religious sentiments, the government has introduced the Nation Protection Act (*Alkiviadou, 2019*). These incidents vividly highlight the challenges Pakistan faces in combatting online hate speech and the need for automated methods are required to handle such kind of content. The English language has been the primary focus of hate speech and abusive language, despite Urdu being the country's national language, while English is the official language (*Mullah & Zainon, 2021*). Individuals often use Latin scripts when writing in Urdu and frequently switch between the two languages during a conversation. This phenomenon involves the alternation of Urdu and English within the same language, phrase, sentence, or other linguistic unit (*Noor et al., 2015*). The term "Roman Urdu" refers to the distinctive and informal style of the Urdu language that incorporates characteristics such as colloquial jargon, non-standard grammar, divergent spellings, idiosyncratic abbreviations, and code-switching. As a result of these features, Roman Urdu is significantly more challenging to model than formal languages, which typically follow established grammatical rules and employ standardized terminology (*Shakeel & Karim, 2020*). It's recognized that the content and characteristics of hate speech differ among different socioeconomic groups. Therefore, to enhance research in this area, there is a need for annotated *corpus* and models in multi languages to support the analysis of linguistic materials (*Mandl et al., 2019*).

Organizations such as Facebook and Twitter are believed to be Each year, they spend hundreds of millions of euros to counteract hate speech on their platforms. However, these companies continue to face criticism for not taking sufficient measures to deal with the problem (*Al-Hassan & Al-Dossari, 2022*). One reason for the continued criticism of organizations such as Twitter and Facebook regarding their attempts to oppose hate speech is the fact that traditional methods for detecting and removing inappropriate online content rely on manual analysis (*Khan et al., 2025*). This approach is known to be arduous, time-consuming, and ultimately unsustainable (*Duwairi, Hayajneh & Quwaider, 2021*). Research has been motivated by the vital requirement for automated and scalable hate speech recognition tools, which has led to the development of significant methods centered on machine learning (ML) and natural language processing (NLP) (*Putri et al., 2020*; *Khan et al., 2022a*). Due to the lack of comparative assessments and the use of distinct datasets by each study, it is not possible to evaluate the results of their considerable efforts.

The detection of hate speech in Roman Urdu poses unique challenges due to the language's informal structure, heavy use of slang, inconsistent grammar, and frequent switching between English and Urdu (*Ashiq et al., 2024*; *Khan et al., 2022b*). These characteristics make it difficult for traditional models to effectively capture both the surface-level and contextual cues present in such texts. As a result, there is a pressing need for automated solutions that can interpret and learn from these noisy and unstructured language patterns.

To address this, we adopt a hybrid deep learning strategy that combines the strengths of convolutional neural network and bidirectional long short-term memory (BiLSTM) networks. CNNs are well-suited for extracting localized textual features such as offensive word combinations and repeated patterns, which are often indicative of hate speech. On the other hand, BiLSTM networks are capable of processing sequential dependencies in both directions, enabling the model to understand context and semantic flow across entire sentences. The integration of these two components allows the system to detect nuanced expressions of hate, even in code-mixed and grammatically irregular input. This architectural choice ensures a balance between effectiveness and computational efficiency, which is critical for real-time applications in resource-constrained environments.

This study presents a robust and computationally efficient deep learning framework for detecting hate speech in Roman Urdu, a domain that remains underrepresented in the current literature. The key contributions of our work are as follows.

- We incorporate FastText pre-trained word embeddings, trained on large corpora, to enrich the semantic representation of Roman Urdu text. The model benefits from subword-level information, which is particularly effective for handling informal, morphologically complex, and non-standard language tokens.
- We propose a hybrid architecture that integrates a CNN with a BiLSTM network, further enhanced with dual global pooling layers. This design enables the model to capture both localized lexical patterns and long-range contextual dependencies, striking a balance between accuracy and computational cost.

- Extensive experiments are conducted using multiple classical machine learning baselines. Our results consistently demonstrate that the proposed model outperforms these baselines across all standard evaluation metrics, confirming its effectiveness for real-world applications in low-resource settings.
- In contrast to many existing deep architectures that are computationally intensive, our model is lightweight and resource-efficient, making it well-suited for deployment in scenarios where processing power is constrained.

Moreover, while most hate speech detection research focuses on English or other high-resource languages, our work specifically addresses the linguistic and structural complexities of Roman Urdu, a code-mixed, colloquial form characterized by informal spelling, inconsistent grammar, and frequent switching between Urdu and English. This fills a critical gap in multilingual hate speech detection. The rest of this article is organized as follows: 'Related Work' reviews related work; 'Proposed Methodology' describes the proposed CNN-BiLSTM model; 'Experimentation and Results Analysis' outlines the datasets used; 'Discussion' details the experimental setup and results; and 'Conclusion' concludes the study.

## RELATED WORK

The extensive lexical similarity between disrespectful language and slanderous remarks poses a significant challenge to the detection disrespectful language (*Davidson et al., 2017*). As a result, people have become accustomed to using insulting or vulgar language for leisure, humor, and sarcasm. Moreover, when a tweet is identified as hateful, research on anti-black racism indicates that 86 percent of respondents perceived the post as hostile, which often includes offensive language, making it challenging to distinguish between instances of disrespectful language and other forms of discourse (*Wang et al., 2014*). Previous research on hate speech recognition has established a set of criteria for identifying problematic tweets (*Waseem & Hovy, 2016*; *Ashraf et al., 2022*). Furthermore, research has indicated that the geographic distribution of website visitors does not influence the ability to identify objectionable posts. Another study has utilized statistical assessments to establish a connection between a user's propensity to propagate offensive content, such as sexism and racism, and the corresponding labeled categories. The study found correlations of 0.71 and 0.76 for sexism and racism, respectively (*Waseem, 2016*).

In addition to unsupervised methods, several studies have investigated various approaches for hate speech detection., such as *Nobata et al. (2016)* and *Malmasi & Zampieri (2017)*, Several studies have recommended the the utilization of supervised learning methods for detecting hatred. The first of These research used publicly available data sources. available sources, including financial and media comments from two distinct domains, which were utilized to create a *corpus* of hate speech. Additionally, the study examined various types of embeddically extracted features and syntactic characteristics. In comparison, in *Malmasi & Zampieri (2017)* a supervised classification approach was employed, utilizing word skip-grams and n-grams in their algorithms, resulting in an

accuracy rate of 78 percent. The study employed three subcategories, including Hatred, Offense, and Normal, to classify the experimental dataset.

Recent research has demonstrated the performance of deep learning methods for identifying expressions of hatred. To categorize hate speech, deep learning approaches utilize deep artificial neural networks (DNNs), which utilize multiple stacked layers to learn implicit representations from input data. Various feature encodings, including many used by conventional approaches, may be employed to encode the input. However, these input features are not immediately utilized for classification. Instead, new abstract feature representations are added to the multi-layered structure for learning purposes. Consequently, deep learning systems prioritize the architecture of the network's topology to automatically extract valuable properties from a fundamental input feature space. For instance, *Badjatiya et al. (2017)*, the study utilized long short-term memory (LSTM), FastText, and CNN in their experimentation, and the research demonstrated that CNN outperformed LSTM with FastText Embedding. Additional research studies, such as *Jha & Mamidi (2017)*, have utilized conventional supervised learning techniques, such as support vector machines (SVM), models that convert sequences to sequences and the most advanced FastText classifier, to annotate existing Twitter datasets for instances of sexism. A recently created dataset categorizes tweets into three groups: benevolent, hostile, and other. In another research on hateful speech identification, a group classifier was employed, wherein features were extracted *via* word frequency vectorization and subsequently fed to classifiers based on neural networks. The researchers claimed that their ensemble process is superior to current techniques for classifying short messages (*Pitsilis, Ramampiaro & Langseth, 2018*).

In addition to research on the English language, there has been additional research on identifying hate speech in Asian and European languages. *Biradar, Saumya & Chauhan (2021)* focused on code-mixed Hindi-English (Hinglish) content and used the multilingual bidirectional encoder representations from Transformers (mBERT) model for the detection of sentiment and hate speech. Their findings reinforce the importance of custom architectures for handling multilingual, informal social media content. For instance, *Vigna et al. (2017)*, in the instance of the Italian language, collected a media group containing news articles on politicians, performers, groups, and celebrities was collected. The initial research aimed to distinguish between strong hatred, weak hatred, and no hatred. Compared to the initial round of trials, the categories of "strong hate" and Weak offensive were employed to create a binary categorization problem in the second set. SVM, a common supervised learning methodology, and LSTM, a deep learning method, were employed in experiments. SVM, a conventional method, outperformed deep learning approaches in both instances, with F1-scores of 0.75 and 0.851. The detection of hate speech in the German language has also been investigated (*Eder, Krieg-Holz & Hahn, 2019*). This study measures the relative offensiveness of lexical concepts using a vocabulary of 11,000 entries.

Apart from European languages, research has also been conducted to identify offensive language content on Arabic social networks (*Mubarak, Darwish & Magdy, 2017*). Another study (*Ranasinghe & Zampieri, 2020*) proposed multilingual offensive language

identification with cross-lingual embeddings, which contributed valuable work in low-resource settings. Despite significant progress in detecting hate speech across languages such as Arabic and various European tongues, very limited research has focused on Roman Urdu, a code-mixed language characterized by non-standard orthography, inconsistent grammar, and high lexical variability (*Khan et al., 2021*). This linguistic complexity presents unique challenges for automated detection methods. To address this gap, our study constructs a dedicated Roman Urdu hate speech *corpus* and evaluates the performance of five traditional machine learning algorithms alongside five deep learning models. Unlike many existing approaches that require large-scale computational resources or are tailored for high-resource languages, we propose a lightweight hybrid CNN-BiLSTM architecture specifically designed for the noisy, low-resource environment of Roman Urdu social media content.

## PROPOSED METHODOLOGY

This section presents the proposed CNN-BiLSTM architecture, illustrated in Fig. 1. The architecture begins with an embedding layer that transforms each input text message, treated as a sequence of tokens, into a continuous vector space. This transformation is accomplished by mapping each word to a real-valued vector that captures its semantic representation across multiple dimensions.

For this study, we employed 300-dimensional word embeddings generated using a skip-gram model trained specifically on Roman Urdu text. These embeddings were used to initialize the weights of the embedding layer. To ensure uniform input length for model training, each tweet was standardized to a fixed length of 100 tokens—truncating longer messages and padding shorter ones with zero-valued vectors.

### Proposed algorithm

The concatenated convolutional bidirectional long short-term memory network (CCBLSTM) used for hate speech detection is represented mathematically as follows:

1. Let $X = [x_1, x_2, x_3, \ldots, x_n]$ be the input sequence of text data, where $x_i$ represents the $i_{th}$ word in the sequence,
2. Let $E$ be the word embedding matrix, where Every word has been represented by a d-dimensional vector. Then, the embedding of the input sequence $X$ is given by: $Embedding(X) = [e_1, e_2, e_3, \ldots, e_n]$, where $e_i = E(x_i)$.
3. Let $W_i$ be the weight matrix and $b_c$ be the bias vector of the convolutional layer. The output of the convolutional layer is given by: $Convolution(X) = [c_1, c_2, \ldots, c_m]$, where $C_i = relu(W_c \cdot e_i + b_c)$. Here, $relu$ is the rectified linear unit activation function.
4. Let $W_f$ be the weight matrix and $b_f$ be the bias vector of the feedforward layer. The output of the feedforward layer is given by: $Feedforward(Z) = [f_1, f_2, \ldots, f_n]$, where $f_i = relu(W_f \cdot z_i + b_f)$.
5. Let $W_f$ be the weight matrix and $b_s$ be the bias vector of the softmax layer. The output of the softmax layer is given by: $Softmax(F) = [y_1, y_2, \ldots, y_k]$, where $y_i = \frac{exp(W_s \cdot f_i + b_s)}{sum(exp(W_s \cdot f_j + b_s))}$,

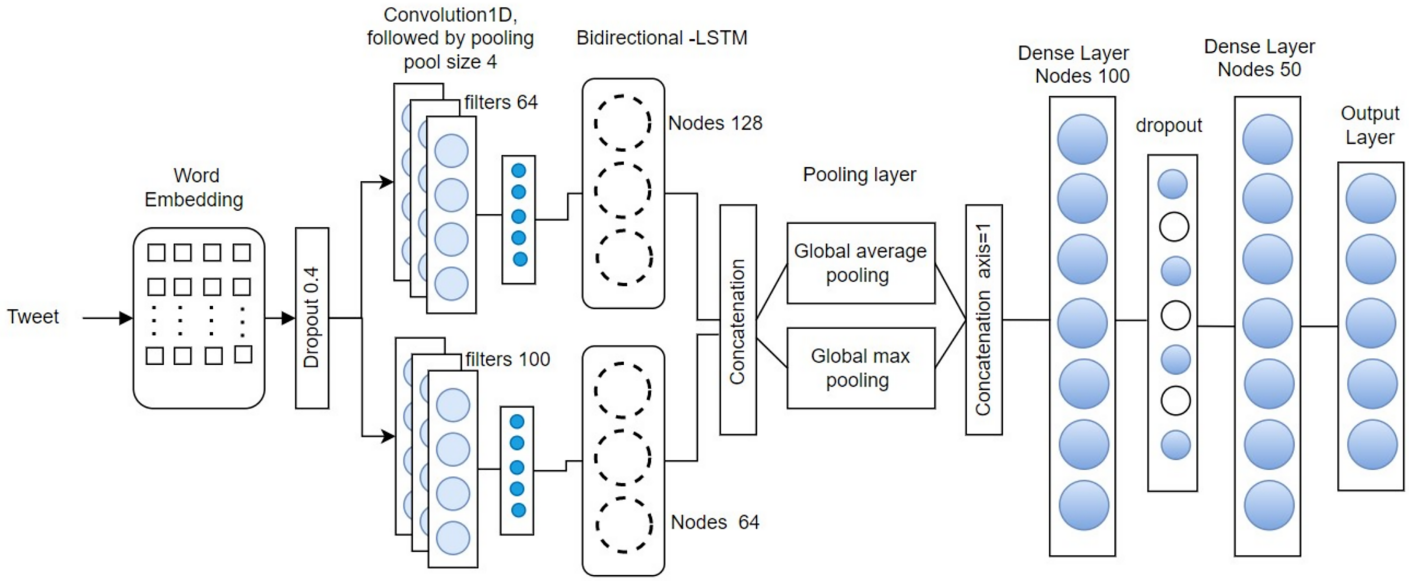


**Figure 1** **CNN-BiLSTM network architecture.** Full-size DOI: 10.7717/peerj-cs.3342/fig-1

for $j = 1$ to $k$. Here, $k$ is the number of output classes and $exp$ is the exponential function.

6. Let $H = [h_1, h_2, \ldots, h_n]$ be the hidden state of the bidirectional LSTM layer. The output of the bidirectional LSTM layer is given by: $BidirectionalLSTM(X) = H$.
7. Let $Z = [z_1, z_2, \ldots, z_n]$ be the concatenated output of the convolutional layer and the bidirectional LSTM layer, where $z_i = [c_i, h_i]$.
8. Let $P$ be the dropout probability. The output of the dropout layer is given by: $Dropout(F) = [f_1', f_2', \ldots, f_n']$, where $f_i' = f \cdot (Bernoulli(1 - P))$. Here, $Bernoulli(1 - P)$ is a random binary vector with probability $P$ of being 0 and 1−P of being 1.
9. $Y = [y_1, y_2, \ldots, y_k]$ be the true labels for the input sequence $X$.
10. The loss function used for training the CCBLSTM model is the cross-entropy loss: $Loss(Y, Softmax(F)) = -sum(y_i \cdot log(y_i'))$, for $i = 1$ to $k$ Here, $y_i'$ is the predicted probability of the $i_{th}$ class.

The CCBLSTM model is trained to utilize a gradient descent optimization algorithm, such as Adam, to minimize the loss function and update the model parameters. The SoftMax function is utilize by the model to predict the probability of the output class for a new input sequence of text data. To prevent overfitting, After the embedding layer, a drop-out layer is applied, with a drop-out rate of 0.4. This has the effect of randomly dropping out a word in a sentence and ensuring that categorization does not rely on specific words.

Next, two 1D convolutional layers with 100, 64 filters each and a kernel size of 4 with padding set to "same" are applied to the drop-out layer's output to make sure the length of the output matches the length of the input data. Activation is accomplished using the

rectified linear unit (ReLU) function. A 100 × 100 representation of the input feature space is generated, and this is further downsampled by a 1D maximum pooling layer with a pool size of four, resulting in an output of shape 25 × 100. There are 25 different dimensions, each of which has an "extractable feature".

Convolutional neural networks (CNNs) are a type of deep neural network that is mostly applied to image-related applications. The block diagram of a convolutional network is depicted in Fig. 2. Among the various types of CNNs, one-dimensional CNNs are specialized for processing signal and time series data. In this research, we introduce a neural network constructed using the one-dimensional CNN architecture for the purpose of detecting hate speech. Within the hidden layers of the network, convolutions are executed, employing kernels as filters to retrieve characteristics from the input data. The convolutional layer typically entails the dot product operation applied to input vectors, and the ReLU (rectified linear unit) stands out as the most commonly employed activation function in this context.

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[n] \cdot h[n-k]. \tag{1}$$

To calculate the number of features in the output, the formula below is employed.

$$n_{out} = \frac{n_{in} + 2p - k}{s} + 1. \tag{2}$$

In the context provided, $n_{in}$ signifies the count of input features, $n_{out}$ represents the number of output features, $k$ stands for the convolution kernel size, $p$ denotes the padding size, and $s$ indicates the stride utilized in the convolution process.

The characteristics derived from the initial convolutional layer are subsequently propagated to LSTM in the BiLSTM layer, where they are treated as individual timesteps, resulting in the generation of 128 hidden units per timestep. The characteristics taken out of the secondary convolutional layer are subsequently channeled into the second BiLSTM layer for further processing.

One type of specialized neural network is the recurrent neural network (RNN) model tailored for the examining of time series data. This model incorporates a feedback loop, enabling it to effectively utilize prior information. However, RNN encounters challenges related to memory limitations and information retention. It has trouble picking up long-term dependencies and is prone to the vanishing gradient issue. To overcome these issues, LSTM architecture was developed. LSTM created expressly to address RNN's limitations in capturing enduring reliance and mitigating the vanishing gradient problem. The LSTM architecture employs memory cells to preserve historical data for extended periods, managing this information by using a gate approach. The LSTM unit has three types of gates that function: the gate for input ($i_t$), the memory gap called forget-gate ($f_t$), and the gate of outflow ($o_t$), illustrated in Fig. 3. To control the state of the memory cells, each gate carries out sigmoid function and point-wise multiplication operations. The present input $x_t$ and the output of the preceding layer's concealed state $h_{t-1}$ are passed into all three gates. Forget gate determines which information to retain or discard, utilizing the

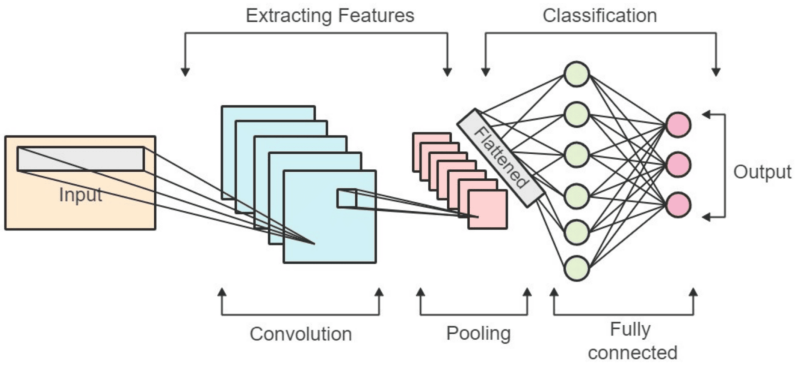


**Figure 2** **Block diagram of the convolutional network.** Full-size DOI: 10.7717/peerj-cs.3342/fig-2

sigmoid function to transition information using the current input $x_t$ and the earlier stage of concealment $h_{t-1}$, forget gate's output ranges from zero to one: a value close to zero signifies that the values will be ignored, while a value close to one indicates that more learning will be retained. The following is the formula for calculating the forget gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}] + b_f) \tag{3}$$

where, $\sigma$ symbolizes the activation of the sigmoid, and $b$ and $W$ represent the bias and weight of each gate unit, corresponding. The present input-data $x_t$ and the preceding hidden-state $h_{t-1}$ are fed into the sigmoid function. Through transformation into a scale ranging from zero to one, The input gate chooses which data should be reused. Here, zero signifies insignificance, while one signifies importance. The following is the expression for the input gate formulation as seen in Eq. (4) *Khan et al. (2022a)*:

$$i_t = \sigma(W_i \cdot [h_{t-1}] + b_i). \tag{4}$$

Subsequently, the present-input $x_t$ and the hidden-state $h_{t-1}$ values are got through the tanh operation. At this stage, the condition of the cell $\hat{C}_t$ is computed, and the cell state is accordingly updated with the revised value. Refer to Eqs. (5) and (6) *Zhu et al. (2020)*.

$$\hat{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t. \tag{6}$$

Here, *tanh* denotes the activation function of hyperbolic tangent. The symbol $\odot$ represents the dot multiplication operator, and $C_t$ signifies the updated recall cell. The output gate selects the next concealed state when the operation comes to an end. The recently updated recall cell $C_t$ and a newly concealed state $h_t$ are subsequently forwarded to subsequent temporal intervals within the series as seen in Eqs. (7) and (8) *Renna (2023)*.

$$o_t = \sigma(W_o \cdot [h_{t-1}, p_t] + b_o) \tag{7}$$

$$h_t = o_t \odot \tanh(C_t). \tag{8}$$

A typical information is handled by LSTM solely from the preceding direction, relying solely on prior data. In contrary, The BiLSTM architecture consists of two LSTM layers: one in the backward direction and the other forward. The diagram representation of

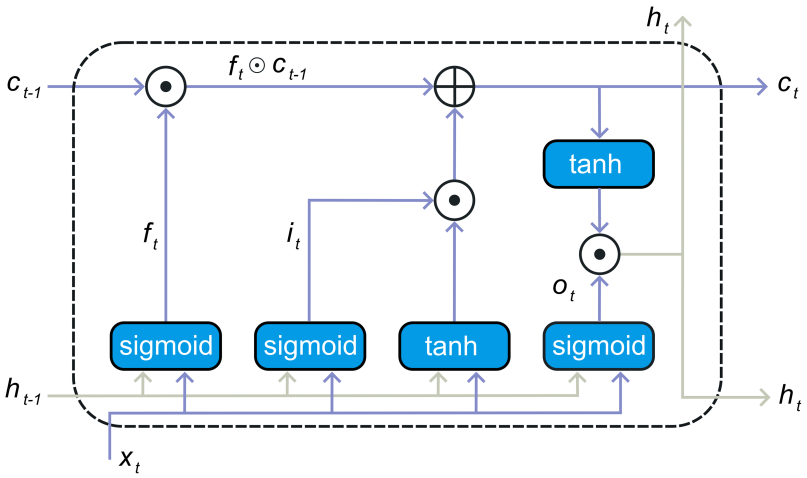


**Figure 3** **An illustrative block diagram of the LSTM network.**
Full-size DOI: 10.7717/peerj-cs.3342/fig-3

BiLSTM is illustrated in Fig. 1. The LSTM in front captures Previous information based on the input sequence, while the backward LSTM gathers upcoming data details, with the outputs from the merging of the two concealed layers. Consequently, the hidden-state $h_t$ of the BiLSTM at the present time $t$ encompasses both the upfront component $\overrightarrow{h_t}$ and the in reverse component $\overleftarrow{h_t}$.

$$h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t}. \tag{9}$$

Here, $\oplus$ represents the component-wise summation operation, utilized to combine the outputs from both the forward and backward elements. BiLSTM offers superior efficiency compared to traditional LSTM and RNN models due to its ability to leverage both preceding and subsequent information in the input sequence.

Reset and update gates are the only two gates in a Gated Recurrent Unit (GRU), whereas there are three in an LSTM (forget, input, and output gates). Consequently, BiLSTM is a complicated structure with more trainable parameters. In theory, this slows down training. After concatenating the outputs of each BiLSTM, A layer known as global max pooling, and a layer of the global average pooling "flatten" the output size generated by selecting the biggest and the mean figure for every timestep dimension to form a $1 \times 100$ vector. This makes an obvious decision to represent a tweet using the qualities with the highest scores. A 0.4 dropout layer is then followed by a 100-node dense layer with full connectivity. Last but not least, this vector is used as input by a SoftMax layer to calculate the likelihood distribution across every potential class (n), in accordance with the databases. To train the model, we employ The cross-entropy loss function for categories and the Adam optimizer. Empirical evidence demonstrates that the first loss function is superior to others, including classification error and sum of the squared error. For classification tasks (*McCaffrey & Colin, 2015*) the benefits of two additional popular extensions of gradient descent with stochasticity to increase the basic stochastic gradient descent's (SGD) effectiveness, utilized optimizer (AdaGrad and RMSProp).

Model parameters. All other parameters are taken from historical data, except batch, epochs, and learning rate, which are derived from experimentations. several the basis for our model's parameters is earlier documented in experimental results, as previously said (more information adhere to). Perhaps Not the finest conditions for the best results,which are always reliant on dataset quality. Later on, we demonstrate, however, experimentally, that the model yields favorable outcomes with minimal data-driven parameter adjustment. Comparison with DNNs of a similar kind. Our network structure resembles that of those mentioned in *Ordóñez & Roggen (2016)*, *Djuric et al. (2015)*, *Tsironi et al. (2017)*. Major variations encompass: (1) We employ a BiLSTM rather than a GRU for similar reasons as stated earlier; (2) Using a drop-out layer to make the learning process more consistent as well as a layer of global max pooling to extract information from the BiLSTM. since convolutional layers are used to get hierarchical image processing features. We don't use such complicated models because we've shown that our CNN +BiLSTM works well for this task and may advantage of both the convolutional qualities and order information from data that BiLSTM gives us, which proves our hypothesis. For the same reason, we only use word embeddings to build our model, even though most people think character-level features are better. Later research shows that this structure is so good that it does a better job than DNN models that are based on character embeddings.

## EXPERIMENTATION AND RESULTS ANALYSIS

In the process of developing and training the CNN-BiLSTM models, Keras, a popular deep learning framework, was utilized in conjunction with Tensorflow as the backend. The Scikit-learn package was used to analyze the data using random forests, naive Bayes, logistic regression, and SVM. Kaggle notebooks equipped with an NVIDIA GPU P100 and 16 GB of GPU internal memory were used for conducting experiments. Stratified five-fold cross-validation was employed to evaluate each model. This approach resulted in five stratified divisions for the *corpus*, each containing a nearly identical distribution of tweet classifications. One random partition was selected from each of the five folds as the testing set, while the remaining dataset served as the training *corpus*. The average recall, accuracy, precision, and F1 over all folds were computed for each model. As the dataset was imbalanced, accuracy was considered unreliable, and therefore F1 was chosen as the primary evaluation metric in this research. The formulae for these metrics are presented in the equations. Equation defines accuracy as the percentage of instances properly categorized relative to the total number of cases (Eq. (10)).

$$Accuracy = \frac{TN + TP}{TP + FN + TN + FP}. \tag{10}$$

The symbols TP, TN, FP, and FN stand for tests that are positive, true negatives, and false negatives, respectively, and the number of correctly detected true negatives. Equation (11) measures the proportion of accurately detected positive instances to the total number of anticipated positive cases. Remember, as expressed in Eq. (12), quantifies the proportion of accurately classified positive cases.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$Recall = \frac{TP}{TP + FN}. \tag{12}$$

The definition of F-measure is that it is meant to reflect the harmonic mean of recall and accuracy. Here is another common use of the phrase, as indicated by Eq. (13).

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{13}$$

To ensure reproducibility, we report the key hyperparameters used in training our CNN-BiLSTM model. The maximum input sequence length was set to 150 tokens. The model was trained using the Adam optimizer with an initial learning rate of 0.02 and a learning rate decay factor of 0.2 applied after 5 epochs. A dropout rate of 0.4 was used to prevent overfitting. The model was trained for a maximum of 50 epochs with early stopping based on validation loss, using a patience value of 10. The batch size was set to 258, and L1 and L2 regularization were applied to the kernel weights.

## Dataset description

The Roman Urdu language lacks sufficient linguistic resources, particularly for identifying hate speech. In the available literature, only a few corpora exist for Roman Urdu, which is a language with limited resources. There has been little effort made in this field for Roman Urdu, which poses challenges due to its informal nature, with frequent misspellings, elongated letters, and variations in spelling. Additionally, several terms are used interchangeably in both English and Urdu, further complicating the dataset acquisition process. As is well known, the scarcity of datasets is a significant challenge in this area of research. Many previous studies have utilized privately collected datasets to address various issues. To construct the largest dataset for offensive language, the author annotated tweets and made the dataset publicly available. The dataset was created by seeking for tweets that contained commonly appearing phrases, which were then evaluated manually for hate speech or specific entity identifiers.

Roman Urdu Hate-Speech and Offensive Language Detection (RUHSOLD) database (*Rizwan, Shakeel & Karim, 2020*) undergoes meticulous annotation by three distinct annotators. To handle conflicts, a resolution is achieved through a majority vote among the annotators. If consensus cannot be reached or if there is inadequate information for labeling, the tweets in question are excluded and substituted with randomly selected tweets from the dataset. Two subtasks are defined as the standard for annotation. The first subtask involves binary labels for Hateful content and Normal content, representing offensive and inoffensive language respectively. This subtask is referred to as "coarse classification." The second subtask involves characterizing hateful content with five specific descriptions, which are deemed most suitable for the demographics of Roman Urdu speakers based on relevant research. Table 1 presents the Twitter tags together with their respective counts.

To highlight the linguistic challenges posed by Roman Urdu, Table 2 provides examples of hate speech instances in Roman Urdu along with their English translations. Roman

**Table 1** Identifiable counts of tweets, tagged with their respective labels.

| Label | Count |
|---|---|
| Sexism | 839 |
| Offensive | 2,402 |
| Religious hate | 782 |
| Profane | 640 |
| Normal | 5,349 |
| Total | 10,012 |

**Table 2** Examples of hate speech in Roman Urdu with English translations.

| Roman Urdu text | English translation | Label |
|---|---|---|
| Tum kitni ghatiya aur beghairat aurat ho | You are such a vile and shameless woman | Offensive |
| Masjid ko ura do sab kafir hain | Blow up the mosque, all are infidels | Religious hate |

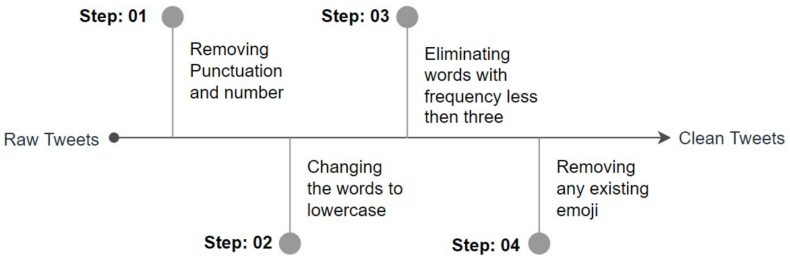


**Figure 4** Steps pre-processing. Full-size DOI: 10.7717/peerj-cs.3342/fig-4

Urdu exhibits characteristics such as inconsistent spelling, lack of grammatical standardization, and frequent use of colloquialisms, making automated processing particularly difficult. These examples also reflect the presence of offensive and religiously charged content, which further complicates classification.

## Pre-processing

Pre-processing is a fundamental procedure that plays a crucial part in ensuring the accuracy of the classification results. Our pre-processing approach begins with standardizing the information included within each tweet. This involves several steps, including the removal of punctuation and numbers, conversion of words to lowercase, and reduction of word variations and accents. Sparse features that are not essential for learning are removed by eliminating tokens that occur in a document fewer than three times. We observed that this led to an improvement in classification accuracy. Additionally, we removed any existing emojis and encoded the class names using the Label Encoder. All of these tweets were cleaned up using pre-processing techniques, leaving only the natural language text for use in subsequent phases. Figure 4 below illustrates the steps involved in our pre-processing strategy.

## RESULTS

This section showcases the outcomes derived from the experiments conducted within this study, elucidating the efficacy of our proposed model. Concurrently, we analyze the implications of these findings within the broader scope of our research objectives. Our results unequivocally demonstrate the superiority of the CNN-BiLSTM model over the baseline model when evaluated against the five alternative models developed and tested. It is essential to note that multiple baseline models were established to facilitate comprehensive comparative analysis.

In the endeavor to identify instances of offensive conversation in Roman Urdu, we have engineered and trained five distinct machine learning models grounded in the encoded representations of the comments. Each of these models possesses its unique hyper-parameter space, encompassing the specific parameters tailored to each learning model. Following the allocation of a validation set from the training data in each of the five folds, rigorous manual scrutiny of the hyper-parameters for each model ensues. For instance, adjustments are made to parameters such as *C* and kernel in support vector machines, as well as the number of estimators, tree depth, and criteria for random forests. The outcomes derived from these five diverse machine learning models are illustrated in Fig. 5.

To establish additional baseline models for deep learning, we implemented three models: CNN, BiLSTM, and bidirectional GRU (BiGRU). We then modified our CNN to construct the proposed model by removing the BiLSTM and replacing it with a BiGRU network. This made it possible for us to compare the two architectures and see how the changes affected a core CNN and GRU structure. Furthermore, the second model enabled us to determine whether GRU can extract relevant order information from short communications like tweets. These baseline models were applied to the RUSHOLD datasets, and the results were compared to those obtained by the proposed model, as seen in Fig. 6.

In this study, a comprehensive approach was taken, encompassing the training of 10 distinct learning models rooted in the encoded representations of tweets to identify hate speech in Roman Urdu. Each learning model possessed its distinctive hyper-parameter space, comprising unique parameters tailored to its specifications. To identify the optimal hyper-parameters for each model, a validation set was meticulously preserved from the training data in each of the five folds. These hyper-parameters underwent careful manual scrutiny and adjustment. For instance, parameters such as penalty, *C*, and solver for logistic regression, the number of estimators, tree depth, and criterion for random forests, as well as penalty, *C*, kernel, and gamma for support vector machines, Furthermore adjustments were made to: The parameters for deep models were adjusted, including batch size, learning rate decay, weight initialization, dropout rate, regularization, and early stopping. Hyper-parameter optimization was executed with the objective of maximizing the test set's F1-score. To comprehensively evaluate the models across all folds, average accuracy,recall, precision, and F1-score was calculated for each model. Given the *corpus* is imbalanced structure, where accuracy might present a misleading picture, F1-score was

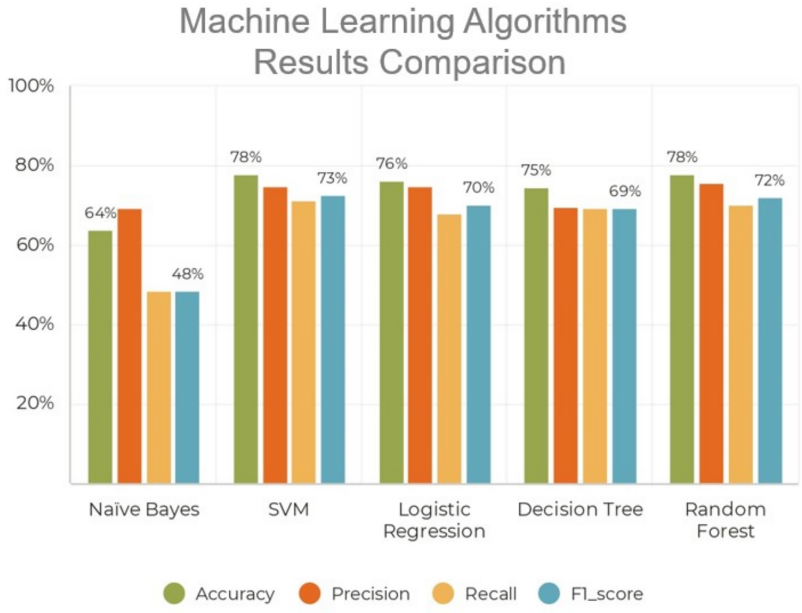


**Figure 5** **Performance analysis of traditional machine learning algorithms.**
Full-size DOI: 10.7717/peerj-cs.3342/fig-5

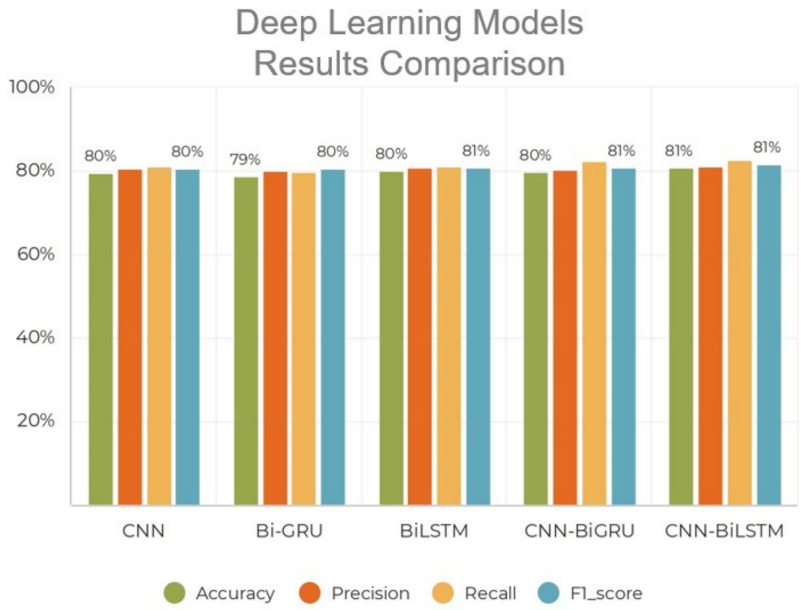


**Figure 6** **Performance comparison of advanced deep learning algorithms with proposed scheme.**
Full-size DOI: 10.7717/peerj-cs.3342/fig-6

employed as the primary evaluation metric, as highlighted in prior research (*Khan, Shahzad & Malik, 2021*). The accuracies and loss curves of the proposed CNN-BiLSTM model are visually shown in Figs. 7 and 8.

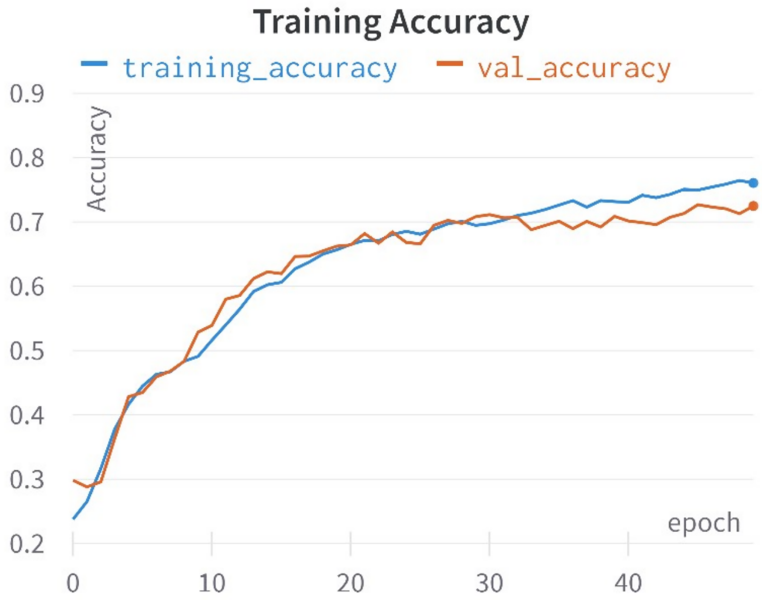


**Figure 7** **Training and validation accuracy curves.** Full-size DOI: 10.7717/peerj-cs.3342/fig-7

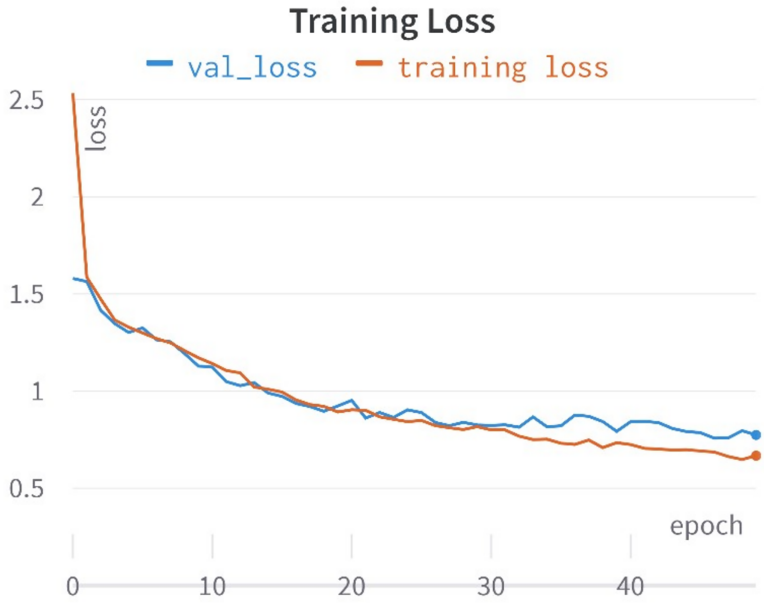


**Figure 8** **Training and validation lose curves.** Full-size DOI: 10.7717/peerj-cs.3342/fig-8

As shown in Table 3, deep learning models significantly outperform traditional machine learning approaches in detecting hate speech in Roman Urdu. The classical models, such as naïve Bayes, logistic regression, and decision tree, demonstrate notably lower F1-scores, primarily due to their limited ability to capture complex linguistic patterns, informal syntax, and code-switching behaviors common in Roman Urdu text. In contrast, deep learning architectures like CNN, BiLSTM, and BiGRU are capable of automatically

**Table 3** Comparison of exploited and proposed models for identification of hate speech in Roman Urdu.

| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| Naïve Bayes | 62.06 | 43.33 | 28.80 | 26.30 |
| SVM | 75.84 | 72.14 | 61.23 | 65.35 |
| Logistic regression | 73.78 | 73.21 | 51.28 | 56.96 |
| Decision tree | 72.79 | 62.08 | 60.79 | 61.35 |
| Random forest | 75.02 | 73.14 | 57.16 | 62.35 |
| CNN | 79.56 | 80.47 | 80.92 | 80.45 |
| BiGRU | 78.67 | 80.03 | 89.57 | 80.47 |
| BiLSTM | 79.88 | 80.88 | 80.94 | 80.78 |
| CNN-BiGRU | 79.83 | 80.38 | 81.21 | 80.71 |
| BERT-CNN-gram (*Rizwan, Shakeel & Karim, 2020*) | 80.0 | 75.0 | 74.0 | 75.0 |
| CNN-BiLSTM | 80.67 | 81.03 | 82.57 | 81.47 |

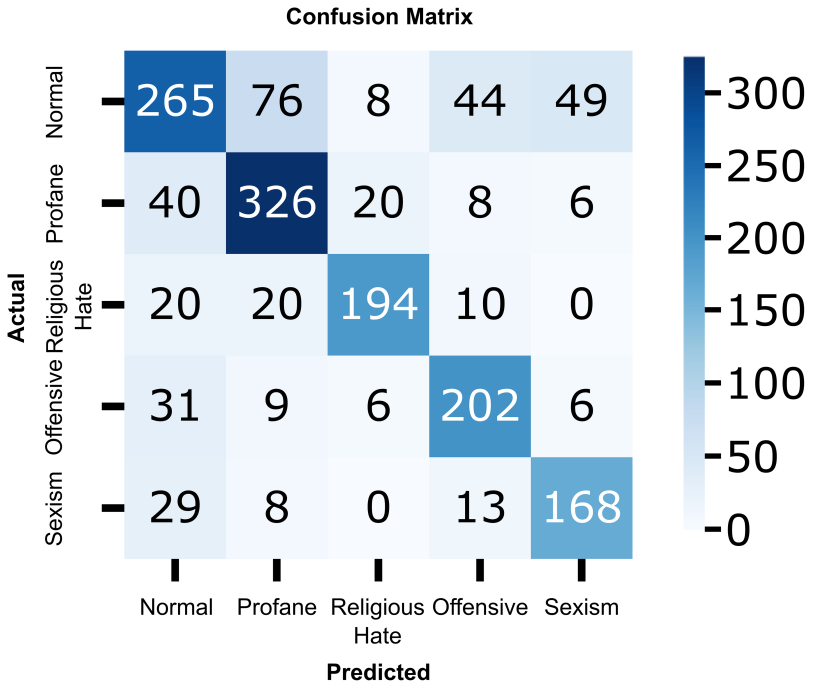


**Figure 9** Confusion matrix of the proposed CNN-BiLSTM model.
Full-size DOI: 10.7717/peerj-cs.3342/fig-9

learning hierarchical and sequential representations from raw text. The combination of convolutional layers (for capturing local patterns) and BiLSTM layers (for modeling long-range dependencies) in our proposed CNN-BiLSTM model further enhances performance, achieving the highest F1-score of 81.47%. This demonstrates the model's robustness in handling noisy, informal language and its superiority in generalizing across complex data distributions. Additionally, the results highlight the importance of leveraging

pre-trained embeddings and end-to-end learning in low-resource and code-mixed language scenarios like Roman Urdu.

To provide a more detailed analysis of the model's classification performance, we include the confusion matrix of the proposed CNN-BiLSTM model as shown in Fig. 9. The matrix offers insights into how well the model distinguishes between hate and non-hate speech in Roman Urdu. It highlights the distribution of true positives, true negatives, false positives, and false negatives, allowing us to identify potential areas of misclassification. As observed, the model achieves strong performance in correctly identifying both classes, with a relatively low rate of false positives and false negatives, confirming its effectiveness in handling the nuanced and informal nature of Roman Urdu text.

**Computational efficiency:** To ensure real-world feasibility, we prioritized a lightweight architecture in the model design. All experiments were executed on a single NVIDIA P100 GPU with 16 GB VRAM, using Kaggle notebooks. The proposed CNN-BiLSTM model required approximately 3 min and 17 s for training and 0.425 s for testing. These results underscore the model's efficiency and potential for real-time deployment in environments with limited computational resources.

## DISCUSSION

Our comprehensive evaluation demonstrates that the proposed CNN-BiLSTM architecture exhibits superior performance compared to conventional machine learning algorithms and competing deep learning frameworks. The hybrid model leverages the complementary strengths of CNN and BiLSTM networks, enabling simultaneous extraction of fine-grained semantic patterns and comprehensive modeling of extended contextual dependencies inherent in Roman Urdu text. This dual-stage feature extraction methodology proves particularly efficacious for processing low-resource, code-mixed languages that exhibit substantial linguistic variability and spelling irregularities.

A distinguishing characteristic of the CNN-BiLSTM framework is its demonstrated resilience to noisy, informal Roman Urdu inputs that commonly manifest inconsistent spelling conventions, frequent code-switching phenomena, and colloquial linguistic expressions. The convolutional layers systematically identify salient local linguistic features, while the BiLSTM component effectively integrates these features within broader contextual frameworks, thereby facilitating robust generalization across diverse linguistic phenomena. Furthermore, the initialization with pre-trained Roman Urdu word embeddings substantially enriches semantic representations, effectively mitigating the adverse effects of data sparsity while enhancing overall classification performance. Empirical results across multiple hate speech detection experiments consistently demonstrate that the hybrid CNN-BiLSTM approach achieves superior F1-scores relative to baseline models, including standalone CNN and BiLSTM implementations.

Despite these strengths, certain limitations must be acknowledged. The training process for the CNN-BiLSTM model requires substantial computational resources, which may constrain its practical deployment in resource-limited environments. Additionally, while

the proposed approach demonstrates effective generalization on the RUHSOLD dataset, its transferability and adaptability to other low-resource linguistic contexts require systematic.

## CONCLUSION

The proliferation of hate speech on social media platforms has raised serious societal concerns, necessitating the development of automated and scalable detection techniques. In this study, we presented a deep learning-based approach, specifically a hybrid CNN-BiLSTM architecture, for the identification of hate speech in Roman Urdu, a linguistically complex and underrepresented language. The proposed model effectively combines convolutional layers to extract local textual patterns and BiLSTM layers to capture long-range dependencies, resulting in improved classification performance. We evaluated our method on a Roman Urdu dataset and demonstrated that it consistently outperforms several traditional machine learning classifiers and baseline deep learning models in terms of accuracy, precision, recall, and F1-score. Our results highlight the model's capability to handle non-standard spellings (*e.g.*, "Khoobsurat," "Khobsorat," "kubsoret") and informal expressions typical in Roman Urdu content. Additionally, we investigated the impact of word embedding choices on model performance and discussed user-centered factors such as posting frequency and social engagement. This work establishes a foundation for future research in hate speech detection in low-resource, code-mixed languages. In the future, we aim to extend this study by incorporating multimodal features and evaluating the model across additional datasets to further assess its robustness and generalizability.

## ADDITIONAL INFORMATION AND DECLARATIONS

### Funding

This work was supported by the Deanship of Graduate Studies and Scientific Research at University of Bisha through the Fast-Track Research Support Program. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

### Grant Disclosures

The following grant information was disclosed by the authors:
Deanship of Graduate Studies and Scientific Research at University of Bisha through the Fast-Track Research Support Program.

### Competing Interests

The authors declare that they have no competing interests.

### Author Contributions

- Muhammad Zohaib conceived and designed the experiments, analyzed the data, performed the computation work, authored or reviewed drafts of the article, and approved the final draft.

- Ghulam Farooque conceived and designed the experiments, analyzed the data, performed the computation work, authored or reviewed drafts of the article, and approved the final draft.
- Mohammad Alsulami conceived and designed the experiments, analyzed the data, authored or reviewed drafts of the article, and approved the final draft.
- Fazeel Abid performed the experiments, prepared figures and/or tables, and approved the final draft.
- Ali Alqazzaz conceived and designed the experiments, performed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, and approved the final draft.
- Mana Saleh Al Reshan performed the experiments, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Jawad Rasheed performed the experiments, authored or reviewed drafts of the article, and approved the final draft.
- Asadullah Shaikh conceived and designed the experiments, analyzed the data, authored or reviewed drafts of the article, and approved the final draft.

## Data Availability

The following information was supplied regarding data availability:

The dataset and code are available at Zenodo:

- Zohaib, M. (2025). Roman-Urdu-Hate-Speech-dataset [Data set]. Zenodo. https://doi.org/10.5281/zenodo.17102699.

- Muhammad, Z. (2025). Roman-Urdu-Hate-Speech-dataset [Data set]. Zenodo. https://doi.org/10.5281/zenodo.17102709.

# REFERENCES

**Al-Hassan A, Al-Dossari H. 2022.** Detection of hate speech in Arabic tweets using deep learning. *Multimedia Systems* **28(6)**:1963–1974 DOI 10.1007/s00530-020-00742-w.

**Alkiviadou N. 2019.** Hate speech on social media networks: towards a regulatory framework? *Information & Communications Technology Law* **28(1)**:19–35 DOI 10.1080/13600834.2018.1494417.

**Ashiq W, Kanwal S, Rafique A, Waqas M, Khurshaid T, Montero EC, Alonso AB, Ashraf I. 2024.** Roman Urdu hate speech detection using hybrid machine learning models and hyperparameter optimization. *Scientific Reports* **14**:28590 DOI 10.1038/s41598-024-79106-7.

**Ashraf N, Khan L, Butt S, Chang H-T, Sidorov G, Gelbukh A. 2022.** Multi-label emotion classification of Urdu tweets. *PeerJ Computer Science* **8(3)**:e896 DOI 10.7717/peerj-cs.896.

**Badjatiya P, Gupta S, Gupta M, Varma V. 2017.** Deep learning for hate speech detection in tweets. In: *Proceedings of the 26th International Conference on World Wide Web Companion, WWW '17 Companion*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 759–760.

**Biradar S, Saumya S, Chauhan A. 2021.** Hate or non-hate: translation based hate speech identification in code-mixed Hinglish data set. In: *2021 IEEE International Conference on Big Data (Big Data)*. Piscataway: IEEE, 2470–2475.

**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 DOI 10.1609/icwsm.v11i1.14955.

**Djuric N, Zhou J, Morris R, Grbovic M, Radosavljevic V, Bhamidipati N. 2015.** Hate speech detection with comment embeddings. In: *Proceedings of the 24th International Conference on World Wide Web, WWW '15 Companion*. New York, NY, USA: Association for Computing Machinery, 29–30.

**Duwairi R, Hayajneh A, Quwaider M. 2021.** A deep learning framework for automatic detection of hate speech embedded in Arabic tweets. *Arabian Journal for Science and Engineering* **46(4)**:4001–4014 DOI 10.1007/s13369-021-05383-3.

**Eder E, Krieg-Holz U, Hahn U. 2019.** At the lower end of language exploring the vulgar and obscene side of German. In: *Proceedings of the Third Workshop on Abusive Language Online*. Stroudsburg: Association for Computational Linguistics, 119–128.

**Jha A, Mamidi R. 2017.** When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data. In: *Proceedings of the Second Workshop on NLP and Computational Social Science*. Stroudsburg: Association for Computational Linguistics, 7–16.

**Khan L, Amjad A, Afaq KM, Chang H-T. 2022a.** Deep sentiment analysis using CNN-LSTM architecture of English and Roman Urdu text shared in social media. *Applied Sciences* **12(5)**:2694 DOI 10.3390/app12052694.

**Khan L, Amjad A, Ashraf N, Chang H-T. 2022b.** Multi-class sentiment analysis of Urdu text using multilingual BERT. *Scientific Reports* **12(1)**:5436 DOI 10.1038/s41598-022-09381-9.

**Khan L, Amjad A, Ashraf N, Chang H-T, Gelbukh A. 2021.** Urdu sentiment analysis with deep learning methods. *IEEE Access* **9**:97803–97812 DOI 10.1109/access.2021.3093078.

**Khan L, Qazi A, Chang H-T, Alhajlah M, Mahmood A. 2025.** Empowering Urdu sentiment analysis: an attention-based stacked CNN-Bi-LSTM DNN with multilingual BERT. *Complex & Intelligent Systems* **11**:10 DOI 10.1007/s40747-024-01631-9.

**Khan MM, Shahzad K, Malik MK. 2021.** Hate speech detection in roman Urdu. *ACM Transactions on Asian and Low-Resource Language Information Processing* **20(1)**:1–19 DOI 10.1145/3414524.

**Kovács G, Alonso P, Saini R. 2021.** Challenges of hate speech detection in social media. *SN Computer Science* **2(2)**:102233 DOI 10.1007/s42979-021-00457-3.

**Mahmood Z, Safder I, Nawab RMA, Bukhari F, Nawaz R, Alfakeeh AS, Aljohani NR, Hassan S-U. 2020.** Deep sentiments in Roman Urdu text using recurrent convolutional neural network model. *Information Processing & Management* **57(4)**:102233 DOI 10.1016/j.ipm.2020.102233.

**Malmasi S, Zampieri M. 2017.** Detecting hate speech in social media. In: *Proceedings of the International Conference Recent Advances in Natural Language Processing*. Varna, Bulgaria: INCOMA Ltd, 467–472.

**Mandl T, Modha S, Majumder P, Patel D, Dave M, Mandlia C, Patel A. 2019.** Overview of the HASOC track at FIRE 2019: hate speech and offensive content identification in indo-European languages. In: *Proceedings of the 11th Annual Meeting of the Forum for Information Retrieval Evaluation, FIRE '19*. New York, NY, USA: Association for Computing Machinery, 14–17.

**McCaffrey JD, Colin M. 2015.** Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training. *Available at https://jamesmccaffreyblog.com/2013/11/05/why-you-should-use-cross-entropy-error-instead-of-classification-error-or-mean-squared-error-for-neural-network-classifier-training/*.

**Mubarak H, Darwish K, Magdy W. 2017.** Abusive language detection on Arabic social media. In: *Proceedings of the First Workshop on Abusive Language Online*. Stroudsburg: Association for Computational Linguistics, 52–56.

**Mullah NS, Zainon WMNW. 2021.** Advances in machine learning algorithms for hate speech detection in social media: a review. *IEEE Access* **9**:88364–88376 DOI 10.1109/access.2021.3089515.

**Nobata C, Tetreault J, Thomas A, Mehdad Y, Chang Y. 2016.** Abusive language detection in online user content. In: *Proceedings of the 25th International Conference on World Wide Web, WWW '16*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 145–153.

**Noor M, Anwar B, Muhabat F, Kazemian B. 2015.** Code-switching in Urdu books of Punjab text book board, Lahore, Pakistan. *Communication and Linguistics Studies* **1**:13–20.

**Ordóñez FJ, Roggen D. 2016.** Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors* **16(1)**:115 DOI 10.3390/s16010115.

**Pitsilis GK, Ramampiaro H, Langseth H. 2018.** Effective hate-speech detection in twitter data using recurrent neural networks. *Applied Intelligence* **48(12)**:4730–4742 DOI 10.1007/s10489-018-1242-y.

**Putri TTA, Sriadhi S, Sari RD, Rahmadani R, Hutahaean HD. 2020.** A comparison of classification algorithms for hate speech detection. *IOP Conference Series: Materials Science and Engineering* **830(3)**:032006 DOI 10.1088/1757-899x/830/3/032006.

**Ranasinghe T, Zampieri M. 2020.** Multilingual offensive language identification with cross-lingual embeddings. ArXiv DOI 10.48550/arXiv.2010.05324.

**Renna F. 2023.** *New insights in machine learning and deep neural networks*. Basel, Switzerland: MDPI-Multidisciplinary Digital Publishing Institute.

**Rizwan H, Shakeel MH, Karim A. 2020.** Hate-speech and offensive language detection in roman urdu. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Stroudsburg: Association for Computational Linguistics, 2512–2522.

**Shakeel MH, Karim A. 2020.** Adapting deep learning for sentiment classification of code-switched informal short text. In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing, SAC '20*. New York, NY, USA: Association for Computing Machinery, 903–906.

**Tsironi E, Barros P, Weber C, Wermter S. 2017.** An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. *Neurocomputing* **268(7)**:76–86 DOI 10.1016/j.neucom.2016.12.088.

**Vigna FD, Cimino A, Dell'Orletta F, Petrocchi M, Tesconi M. 2017.** Hate me, hate me not: hate speech detection on facebook. In: *Proceedings of the First Italian Conference on Cybersecurity (ITASEC17)*. Stroudsburg: Association for Computational Linguistics, 86–95.

**Wang W, Chen L, Thirunarayan K, Sheth AP. 2014.** Cursing in English on Twitter. In: *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, CSCW '14*. New York, NY, USA: Association for Computing Machinery, 415–425.

**Waseem Z. 2016.** Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter. In: *Proceedings of the First Workshop on NLP and Computational Social Science*. Stroudsburg: Association for Computational Linguistics, 138–142.

**Waseem Z, Hovy D. 2016.** Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In: *Proceedings of the NAACL Student Research Workshop*. Stroudsburg: Association for Computational Linguistics, 88–93.

**Zhu S, Sun H, Duan Y, Dai X, Saha S. 2020.** Travel mode recognition from GPS data based on LSTM. *Computing and Informatics* **39(1–2)**:298–317 DOI 10.31577/cai_2020_1-2_298.