IJIE

INTERNATIONAL JOURNAL OF INTEGRATED ENGINEERING
ISSN: 2229-838X e-ISSN: 2600-7916

Vol. 18 No. 4 (2026) 276-299
https://publisher.uthm.edu.my/ojs/index.php/ijie

Binary Focal Loss Enhanced LSTM for Colloquial Online Toxicity Detection

Balaji Mudadla[1]*, Vani V[1]

[1] *Department of Computer Science and Engineering,*
National Institute of Technology Puducherry, Puducherry, INDIA

*Corresponding Author: balaji5377@gmail.com
DOI: https://doi.org/10.30880/ijie.2026.18.04.018

Article Info

Received: 29 December 2025
Accepted: 15 June 2026
Available online: 30 September 2026

Keywords

Colloquial text classification, toxicity detection, deep neural networks, binary focal loss, LSTM, RNN, conversation AI, online discussions, NLP

Abstract

It is difficult to detect toxicity in colloquial text, as most discussion texts found online include slang, spelling variants, non-standard grammar, implicit toxicity and emotionally charged expressions. Such traits hinder the ability of an automatic moderator to effectively handle these categories, particularly if they are not evenly distributed throughout the dataset. The present study explores various deep learning models to detect online toxicity in colloquial speech across the Conversation AI dataset of Wikipedia talk-page comments (159,571) that includes several indicators of toxicity. A text preprocessing pipeline – text cleaning, text normalization, tokenization, sequence padding, embedding representation – is done before model training. A shallow feed-forward neural network, recurrent neural network configurations and various Long Short-Term Memory models are compared under a uniform evaluation setting. To overcome the issue of class imbalance and to preserve the sequential information, an LSTM based model with a dense layer is trained using binary focal loss. The accuracy, precision, recall, F1-score, loss, training time and testing time of the evaluated models are compared. In terms of the observed performances, the proposed LSTM_50_Dense_25_Binary_Focal_Loss model has the best performance with test accuracy of 99.996% and test loss value of 0.013, training time of 90 minutes and testing time of 10 minutes among the tested architectures. The result is analysed in conjunction with the per-label performance, observations in the confusion matrix, ablation-based comparison and benchmarking with recent transformer-based models to give a more balanced assessment of classification reliability. The results indicate that the combination of LSTM-based sequential modelling and binary focal loss can help increase the accuracy of detecting toxicity in informal and imbalanced online text classification tasks.

This is an open access article under the CC BY-NC-SA 4.0 license.

Graphical Abstract

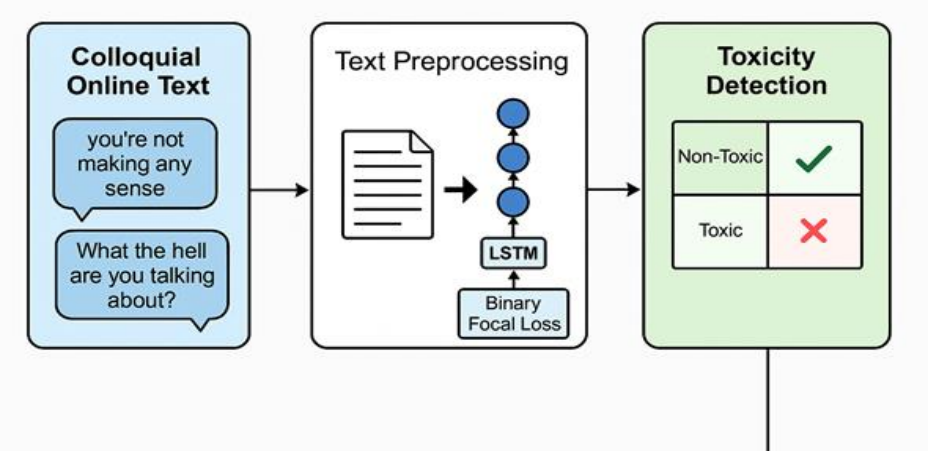


Fig. 1 *Graphical abstract of proposed work*

1. Introduction

1.1 Background and Motivation

Online platforms have become the primary channel for day-to-day interaction in social, educational, and professional contexts [1]. Unlike carefully edited text, the language used in comment threads and discussion forums is highly informal: users rely on slang, emojis, creative spellings, code-mixing, and region-specific phrases that often deviate strongly from standard written forms [2], [3]. This mismatch between spontaneous digital conversation and the well-structured corpora on which many traditional natural language processing (NLP) models are trained leads to systematic errors in interpretation and classification. Reducing this gap is essential for building robust language technologies that can encourage constructive and respectful online discourse, particularly in sensitive settings such as news portals, collaborative knowledge platforms, and educational communities [4], [5].

1.2 Problem Context and Benchmark Dataset

To investigate these issues, this study adopts the Conversation AI dataset, a large collection of Wikipedia talk-page comments annotated with multiple toxicity-related labels. Each comment may be tagged for general toxicity, obscenity, insults, threats, and identity-based abuse, reflecting the kinds of behaviour that moderators routinely encounter in practice. Because the comments originate from real editorial and dispute-resolution discussions, the corpus captures realistic conversational patterns, topic shifts, and style variations. This makes it a suitable benchmark for research on automatic harassment detection, toxicity filtering, and broader safety mechanisms in user-generated content platforms [6][7].

1.3 Neural Architectures Under Study

In this study, we will investigate the processing of noisy and conversational informal text in various neural architectures. The comparison starts with a shallow feed-forward neural network, and progresses to recurrent and LSTM-based models such as RNN_LSTM_tiny_10_5_6, RNN_small_20_10_6 and LSTM_60_50_6. To guarantee fair comparison, all models are trained and tested with the same data split and pre-processing pipeline.

The proposed model, LSTM_50_Dense_25_Binary_Focal_Loss, is a combination of LSTM-based sequence learning, dense layer and binary focal loss. The LSTM layer models the dependence between the tokens (words) in a sequence and binary focal loss assigns higher weights to hard and minority class samples to tackle the class imbalance problem. Each comment can have more than one toxicity category, therefore this is a multi-label classification problem.

1.4 Evaluation Strategy

The accuracy, precision, recall, F1-score, loss, training time, and testing time are used to assess each architecture. As the data set is imbalanced, accuracy is not seen as sufficient. Hence, the class-wise behaviour, confusion-matrix observations and minority-label detection quality are also used to interpret the performance.

Computational efficiency is also a factor in the evaluation as toxicity detection systems need to perform well in the context of large-scale online moderation. This is to find models that are accurate and can be deployed into real-time. The proposed model is also contrasted with recent transformer-based methods to further enrich the benchmarking context.

1.5 Research Objectives and Contributions

This work has the following main objectives and contributions:

1. **Neural model comparison:** Three different feed-forward, recurrent and LSTM-based architectures are tested for detecting colloquial toxicity in the Conversation AI dataset.
2. **Multi-label toxicity detection:** This task is defined as a multi-label classification task, since one comment can have multiple labels: toxic, obscene, insult, threat, severe toxic, and identity hate.
3. **Focal-loss-based LSTM model:** To overcome the challenge of learning from imbalanced toxicity categories, an LSTM model with a dense layer and binary focal loss is created.
4. **Balanced performance evaluation:** The models are evaluated based on predictive metrics and computational metrics such as accuracy, precision, recall, F1-score, loss, training time and testing time.
5. **Practical NLP contribution:** The findings of the study are valuable to create efficient and imbalance aware toxicity detection systems in informal online discussions.

1.6 Organization of the Paper

The rest of the paper is structured as follows. Section 2 reviews related research on colloquial language modelling, toxicity detection, and imbalance-aware learning in NLP .Section 3 describes the dataset, pre-processing pipeline, and methodological framework. Section 4 details the investigated neural architectures and presents pseudocode for the proposed models. Section 5 reports and analyses the experimental results. Section 6 summarises the main conclusions, and Section 7 outlines possible directions for future work. Fig. 2 summarizes the research workflow adopted in this paper.

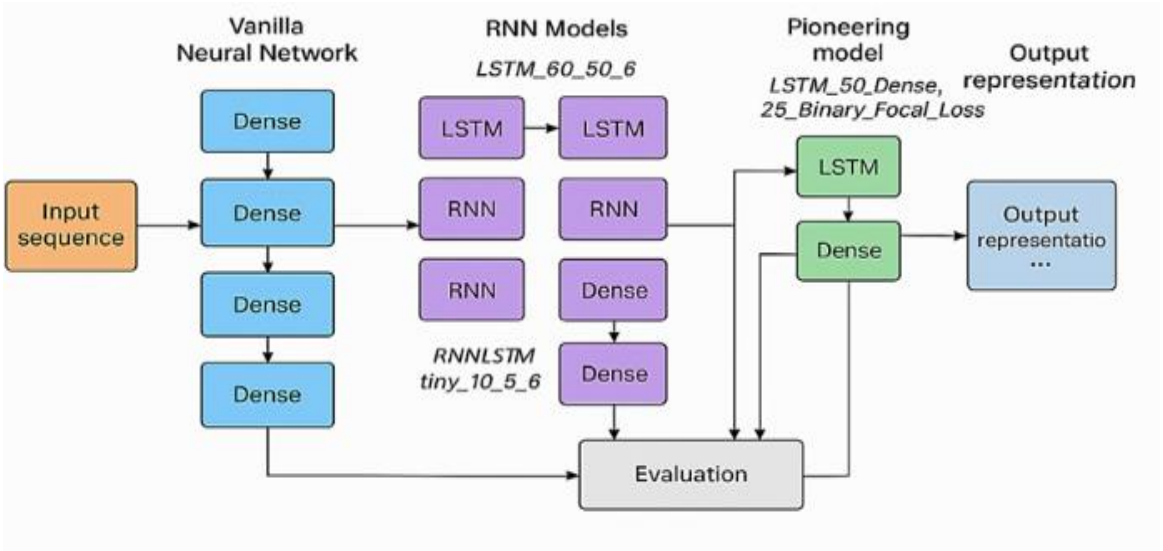


Fig. 2 *Block diagram representing the proposed workflow*

2. Literature Review

2.1 Developments in Natural Language Processing

Natural language processing has progressed from simple keyword-based retrieval to powerful models capable of capturing complex linguistic structure and context [8]. Contemporary systems exploit large corpora and deep architectures to approximate human performance on a variety of conversational tasks. Nevertheless, many of these models are still tuned to relatively formal, well-edited text and therefore struggle when faced with slang, code-mixing, dialectal variation, or creative spellings that are common in online communication [9], [10]. This mismatch motivates research on approaches that can flexibly adapt to the stylistic diversity of user-generated content.

2.2 Multimodal and Cross-Domain Extensions of NLP

A notable strand of recent work extends text processing to multimodal environments in which language interacts with images or video. Nawaz et al. [11,12], for example, tackle temporal moment localization by combining video question answering with textual reasoning. Similar studies explore vision–language alignment, audio–text fusion, and other cross-modal settings [13]. These efforts highlight the need for representations that can jointly encode semantics across modalities, even though the present study focuses purely on text; the same modelling principles are relevant for robust understanding of informal written language.

2.3 Robustness, Noise, and Adversarial Perturbations

As NLP systems are increasingly deployed in real-world applications, robustness has emerged as a central concern. Inputs encountered in practice may be noisy, misspelled, or intentionally manipulated, leading to performance drops if models are overly tuned to clean training data [21]. Work on adversarial examples, noisy-channel modelling, and robust embedding learning aims to mitigate these vulnerabilities by hardening models through adversarial training, data augmentation, and uncertainty-aware objectives [6]. Such techniques are particularly important for social platforms, where hostile users may deliberately exploit weaknesses in toxic-content filters.

2.4 Social Media Analytics and Harmful Behaviour Detection

A large body of literature concentrates on analyzing social media posts for sentiment, emotion, stance, and topical structure [16]. These studies often serve downstream goals such as hate-speech detection, cyberbullying identification, or early warning of coordinated harassment campaigns. The key challenge is that cues signaling abusive or toxic behaviour are frequently subtle, highly context dependent, and intertwined with humor, irony, or in-group slang [15], [16]. Models must therefore move beyond surface n-grams and capture richer dependencies in conversational history and user style.

2.5 Representation Learning and Low-Resource Generalization

Improvements in distributed representations have significantly changed how text is modelled. Modern embedding methods encode semantic and syntactic regularities in dense vector spaces, providing a strong foundation for downstream classifiers [17]. In parallel, few-shot, transfer-learning, and prompt-based paradigms enable models pre-trained on large corpora to adapt to new tasks or domains using only small labelled datasets [18]. These strategies are crucial for languages, genres, and online communities where annotated resources are scarce.

2.6 Recent Transformer-Based and Imbalance-Aware Toxicity Detection Studies

Recent research studies demonstrate a definite move towards transformer-based and large language model approaches for hate-speech and toxicity detection, as well as abusive-language detection. In the context of hate-speech detection in low-resource languages, Fetahi et al. [26] employed transformer-based models with explainable AI, highlighting the need for contextual learning and interpretability in social media moderation. A multilingual approach using a large language model was proposed by Usman et al. [27] to show the importance of modern language models in capturing various online communication patterns for multilingual hate-speech detection. Rathore and Chaurasia [28] also investigated the fine-tuning of the large language models for Hinglish and code-mixed hate-speech detection, as the code-mixed style is common in colloquial online texts.

A number of recent papers too concern transformer-based models for low-resource and regional language toxicity detection. Azhar et al. [29] introduced a hybrid transformer-based hate-speech detection model named HateBertBN for hate-speech detection in Bangla social media. Duridi et al. [30] experimented with different variants of the BERT model for Arabic hate speech detection and showed the effectiveness of contextual transformer representations for detecting harmful expressions. Rashid et al. [31] proposed a multi-stage and multi-label method for detecting toxicity in Urdu, which found that harmful content posted online may have multiple overlapping categories of toxicity. In the same vein, BanCyB, a hybrid transformer-LSTM model, was proposed for explainable multi-label cyberbullying detection in low-resource setting by Hasan et al. [32].

These recent studies show that transformer-based models are successful in detecting toxicity in context, multilingual classification of hate speech, and analysing abusive content under low-resource settings. They also reveal that difficulties exist in dealing with class imbalance, multi-label toxicity, informal expressions and computational cost. The current work, in this context, puts the proposed LSTM model with binary focal loss as an imbalance aware and low computational complexity method for detecting the toxicity of colloquial text. The proposed method emphasizes sequential representation learning and adopts focal loss to better balance hard and minority class examples.

2.7 Research Gap and Positioning of the Present Work

Although recent advances have been made in NLP, the task of detecting toxicity in colloquial online text is still difficult. Even with many well-established models, they fare very poorly with informal expressions, slang, spelling variations, code-mixed words, sarcasm, and dialect-specific usages that are frequently seen in online discussions. One of the main problems is the class imbalance, where the harmful categories like severe toxic, threat, identity hate are much less common than the non-toxic or general toxic categories [3,16,21,31].

More recent transformer-based models have seen better contextual understanding in hate-speech and toxicity detection, but these can still require more computational resources and might need careful handling of minority toxicity labels [26]–[32]. Hence, a model which can compromise classification accuracy, efficiency, and imbalance aware learning is required.

The present study is aimed at filling this gap by systematically comparing multiple neural architectures in the Conversation AI dataset, and suggesting an LSTM based model trained with binary focal loss. The proposed approach is designed to be light-weight and imbalance aware for detecting colloquial toxicity, which seeks to enhance learning from the hard and under-represented toxic categories while retaining the sequential information in conversational texts.

3. Proposed Methodology

This section describes the complete workflow used to analyze the colloquial comments in online discussions. The pipeline consists of five stages: dataset preparation, text pre-processing and representation, neural architecture design, training strategy, and performance evaluation. Each stage is designed to cope with noisy, imbalanced, and highly informal language.

3.1 Dataset Preparation

3.1.1 Source Corpus and Split

All experiments use the Conversation AI corpus of Wikipedia talk-page comments, comprising 159 571 user posts [19]. The data are divided into a training portion with 127 657 instances and a test portion with 31 914 instances, preserving the original label distribution. A validation subset is created from the training portion using stratified sampling and is used only for hyperparameter tuning and early stopping. The test set is kept fully independent and is not used during preprocessing decisions, model selection, or parameter tuning. Formally, the dataset can be written as

$$\mathcal{D} = \{(c_i, \mathbf{y}_i)\}_{i-1}^{N}, \tag{1}$$

where c_i is the i-th comment and $\mathbf{y}_i \in \{0,1\}^K$ is a binary label vector for $K - 6$ toxicity categories.

3.1.2 Annotation Scheme

Each $\mathbf{y}_i$ records the presence or absence of six forms of abusive behaviour: toxicity, severe toxicity, obscenity, threat, insult and identity hate. For comment c_i, the k-th component $y_{ik} - 1$ indicates that label k is active, and $y_{ik} - 0$ otherwise. This multi-label structure allows the models to learn fine-grained distinctions between different types of harmful content. Thus, the task is treated as a multi-label classification problem rather than a single binary classification task, because one comment may belong to more than one toxicity category.

3.1.3 Cleaning and Imbalance Handling

Raw comments contain URLs, user mentions, markup and other artefacts that do not contribute to linguistic meaning. A cleaning routine removes such elements, converts text to lowercase and normalizes simple spelling variants [20]. Let $\bar{c}_i$ denote the cleaned version of c_i,

The label distribution is highly skewed, with non-toxic comments dominating the corpus and some categories (e.g. threats) appearing rarely. To reduce this imbalance within the training set, we explore resampling techniques such as Synthetic Minority Over-Sampling Technique (SMOTE), applied in the feature space of training instances, together with class-aware loss weighting. The held-out test set is left untouched so that evaluation reflects realistic operating conditions. In addition, the proposed model uses binary focal loss to reduce the dominance of majority-class examples and improve learning from difficult minority-class samples.

3.2 Text Preprocessing and Representation

3.2.1 Tokenization

Each cleaned comment $\bar{c}_i$ is segmented into a sequence of tokens

$$\bar{c}_i \rightarrow (w_{i1}, w_{i2}, \ldots, w_{iT_i}), \tag{2}$$

where T_i is the number of tokens. For batch processing, sequences are truncated or padded to a fixed maximum length $T_{\max}$.

3.2.2 Embedding Layer

Tokens are mapped to continuous vectors through an embedding matrix $\mathbf{E} \in \mathbb{R}^{V \times d}$, where V is the vocabulary size and d the embedding dimension. For token index w_{it},

$$\mathbf{e}_{it} = \mathbf{E}[w_{it}] \in \mathbb{R}^d \tag{3}$$

Thus, each comment is represented as a matrix

$$\mathbf{X}_i = \begin{bmatrix} \mathbf{e}_{i1}^{\top} \\ \vdots \\ \mathbf{e}_{iT_{\max}}^{\top} \end{bmatrix} \in \mathbb{R}^{T_{\max} \times d}, \tag{4}$$

which serves as the input to the neural architectures. The embedding layer is initialized (randomly or from pre-trained vectors) and updated during training so that it adapts to colloquial and slang usage.

3.3 Neural Architecture Design

3.3.1 Baseline Feed-Forward Model

The Vanilla network provides a non-recurrent baseline. A simple pooling operation, for example mean pooling, aggregates token embeddings:

$$\bar{\mathbf{e}}_i = \frac{1}{T_{\max}} \sum_{t=1}^{T_{\max}} \mathbf{e}_{it} \tag{5}$$

This pooled vector is passed through one or more dense layers with nonlinear activation to produce label probabilities. Although it ignores word order, this baseline indicates how much signal is present in bag-of context representations alone.

3.3.2 Recurrent and LSTM Models

To capture sequential dependencies, we examine several recurrent architectures such as RNN_small_20_10_6 and LSTM_60_50_6. For LSTM-based models, the hidden state $\mathbf{h}_t$ and cell state $\mathbf{c}_t$ at position t are updated according to

$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{e}_{it} + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i), \\ \mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{e}_{it} + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f), \\ \mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{e}_{it} + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o), \\ \bar{\mathbf{c}}_t &= \tanh(\mathbf{W}_c \mathbf{e}_{it} + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c), \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \bar{\mathbf{c}}_t, \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \end{aligned} \tag{6}$$

where $\sigma(\cdot)$ is the logistic function and $\odot$ denotes elementwise multiplication. The last hidden state $\mathbf{h}_{T_{\max}}$ (or a pooled combination of all states) provides a sequence-aware summary of the comment.

3.3.3 Proposed LSTM_50_Dense_25_Binary_Focal_Loss model

The main architecture of interest extends the LSTM encoder with a dense bottleneck layer before the output. Let h_i denote the LSTM summary vector for comment iii. A dense layer with ReLU activation produces

$$\mathbf{z}_i = \phi(\mathbf{W}_d \mathbf{h}_i + \mathbf{b}_d), \tag{7}$$

and a final sigmoid layer generates one probability per label:

$$\hat{\mathbf{y}}_i = \sigma(\mathbf{W}_o \mathbf{z}_i + \mathbf{b}_o) \in (0,1)^K \quad (8)$$

This configuration-LSTM_50_Dense_25_Binary_Focal_Loss-is designed to combine strong temporal modelling with a loss function that explicitly accounts for label imbalance. The use of six sigmoid output neurons supports multi-label prediction, where each toxicity label is estimated independently.

Hyper parameters such as the number of LSTM units, dense-layer size, dropout rate, batch size and learning rate are tuned through pilot runs to ensure that each model operates near its best regime on colloquial text. The focal-loss parameters $\alpha$and $\gamma$are also selected based on validation-set performance.

3.4 Training Strategy

3.4.1 Loss Functions and Optimization

Baseline models are trained with the usual binary cross-entropy loss. For a single label $y \in \{0,1\}$ and predicted probability p, this loss is

$$\mathcal{L}_{\mathrm{BCE}}(p, y) = -(y \log p + (1 - y) \log(1 - p)) \quad (9)$$

The proposed model instead uses binary focal loss to emphasize hard and minority examples. For each label,

$$\mathcal{L}_{\mathrm{FL}}(p, y) = -\alpha y (1 - p)^{\gamma} \log p - (1 - \alpha)(1 - y) p^{\gamma} \log(1 - p) \quad (10)$$

where $\alpha \in (0,1)$ controls the relative weight of positive and negative instances and $\gamma \geq 0$ determines how strongly correctly classified examples are down-weighted. Here, $\alpha$is used as a class-balancing factor, while $\gamma$is the focusing parameter that reduces the contribution of easy examples and increases attention toward difficult samples. This is particularly useful for rare toxicity labels such as threat, severe toxicity, and identity hate. The overall loss is obtained by averaging $\mathcal{L}_{\mathrm{FL}}$ (or $\mathcal{L}_{\mathrm{BCE}}$) over all labels and training examples.

All networks are optimized with mini-batch gradient-based methods, typically using the Adam optimizer with a learning rate selected via preliminary experiments [21]. Training continues for a fixed maximum number of epochs or until performance on a validation set ceases to improve. Dropout and L_2 regularisation are applied where necessary to limit overfitting.

3.4.2 Validation Protocol and Monitoring

Within the training partition, a stratified split creates validation data used for hyper parameter tuning and early stopping. During training, metrics such as accuracy, precision, recall and F1 score are monitored epoch-by-epoch. Learning-rate schedules or patience-based early stopping are employed when validation performance plateaus, helping to prevent overfitting and ensuring that improvements generalise beyond a single split of the data. To improve reproducibility, experiments are performed using fixed random seeds, and the final model behaviour is examined using validation and held-out test performance.

The validation process is designed to avoid data leakage. Preprocessing parameters, vocabulary construction, hyperparameter tuning, and early stopping decisions are derived only from the training and validation data. The test data are used only once for final performance reporting.

3.5 Performance Evaluation

3.5.1 Evaluation Metrics

After training, the final model for each architecture is evaluated on the held-out test set. Performance is described using several complementary measures: overall accuracy, macro- and micro-averaged precision, recall, F1 score, and area under the receiver operating characteristic curve (ROC AUC). Precision–recall AUC, per-label performance, and confusion-matrix observations are also considered because overall accuracy alone may be misleading in imbalanced multi-label toxicity detection. These metrics capture different aspects of quality, which is important in imbalanced multi-label settings.

3.5.2 Comparative Analysis of Architectures

The resulting scores are compared across architectures to understand how design choices—such as the presence of recurrent layers, depth of dense components, and choice of loss function—affect the ability to model informal and slang-rich language. Special attention is given to behaviour on minority classes, where gains from focal loss

are expected to be most visible. The comparison also includes benchmarking against recent transformer-based toxicity detection approaches to strengthen the positioning of the proposed model.

3.5.3 Detailed Assessment of the Proposed Model

Finally, the LSTM_50_Dense_25_Binary_Focal_Loss network is analysed in greater detail. Its improvements in classification accuracy, F1 score and ROC AUC, together with its effect on rare labels, are examined to demonstrate how combining LSTM-based sequence modelling with binary focal loss provides a robust solution for colloquial toxicity detection in online forums. The assessment also includes ablation-based interpretation to verify the specific contribution of binary focal loss over conventional binary cross-entropy-based training. Fig. 3 provides a detailed roadmap of the methodological process, illustrating each stage from dataset preparation to performance evaluation.

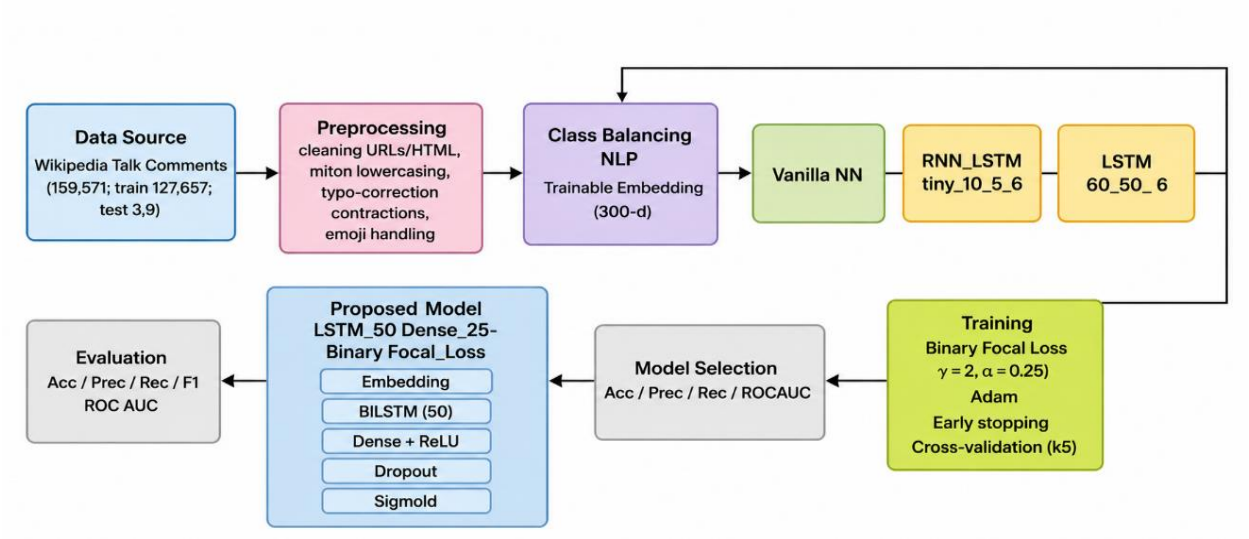


Fig. 3 *Workflow of proposed work*

4. Architecture Developed

This section covers the new neural network architectures developed during the course of this work. Every architecture mentioned here is specifically designed to solve a problem in the area of Natural Language Processing/Understanding such as analyzing online text. The target is architectures that are both effective at the task and which challenge existing NLP methodologies. The architectures discussed in this paper are developed through extensive experimentation. They use both old and new techniques, combining what works with innovative techniques. These models have undergone a comprehensive evaluation process to determine their effectiveness in classifying the nuances of digital communication [22].

This section examines these architectures to gain insight into the rationale for their design, their layer configurations, and the various features that distinguish them. According to developers, the code is no longer just a technical achievement. It suggests advanced neural networks could lead to more capable and nuanced standardized digital communications.

4.1 Case -1. Keras 10, 6 Neural Network Vanilla

This research paper proposes a new neural network model for binary classification. The proposed new model focuses on classifying toxic and non-toxic words. The model utilizes a Vanilla Neural Network implemented in Keras. The input layer has the same dimension as that of the training data and a hidden layer of 10 units with a ReLU activation [23]. The top layer has 6 units and uses a sigmoid activation. During the training process, we make use of the optimizer stochastic gradient descent and the loss function binary cross entropy with metric accuracy. The model will train for 50 epochs, a batch size of 200, and a validation split of 20%.

The primary objective of this model is to efficiently detect toxic language, a key factor in promoting healthier and more constructive online discussions. The research investigates the impact of different neural network architectures on identifying toxic language, focusing on the effectiveness of the Vanilla Neural Network in discerning toxic from non-toxic words within online conversations. Insights from this research, particularly regarding the Vanilla Neural Network's performance, provide significant contributions to the field of online content moderation. This is visually represented in Fig. 4, which illustrates the Vanilla Neural Network Architecture.

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

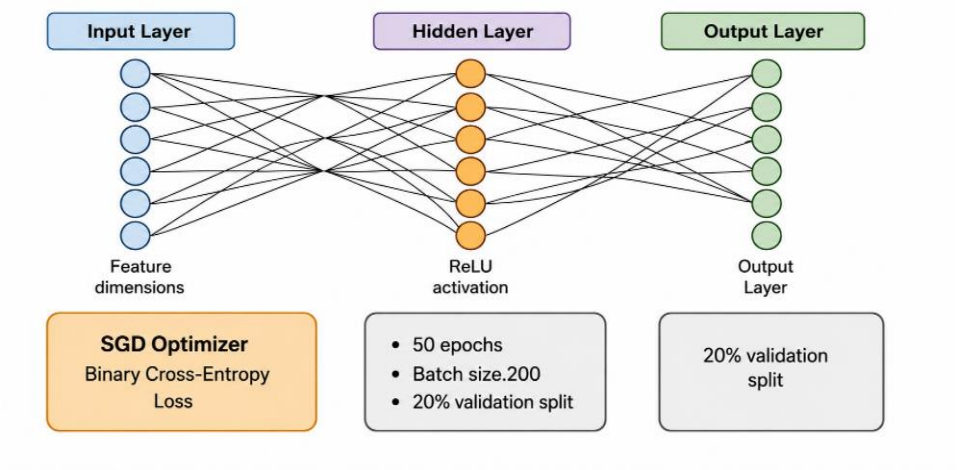


Fig. 4 *Vanilla neural network architecture*

Pseudocode Modelling-I

Let:

X_t be the input features with dimensions (m, n), where m is the number of samples and n is the number of features.

y_{train} be the binary labels for toxicity (0 for non-toxic, 1 for toxic) [6].

$W^{[1]}$ and $b^{[1]}$ represent the weights and biases of the first hidden layer.

$W^{[2]}$ and $b^{[2]}$ represent the weights and biases of the output layer [6].

$A^{[1]}$ is the activation of the first hidden layer (with ReLU activation).

$A^{[2]}$ is the activation of the output layer (with sigmoid activation).

The algorithm can be mathematically represented as follows:

Step 1: - Forward Propagation:

Calculate the activation of the first hidden layer:

$$Z^{[1]} = X_t \cdot W^{[1]} + b^{[1]} \tag{11}$$
$$A^{[1]} = \text{ReLU}\left(Z^{[1]}\right)$$

Calculate the activation of the output layer:

$$Z^{[2]} = A^{[1]} \cdot W^{[2]} + b^{[2]} \tag{12}$$
$$A^{[2]} = \text{Sigmoid}\left(Z^{[2]}\right)$$

Step 2 :- Cost Function:

Compute the binary cross-entropy loss for the entire training dataset:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_{\text{train}}^{(i)}\ \log\left(A^{[2]}\right)^{(i)} + \left(1 - y_{\text{train}}^{(i)}\right)\log\left(1 - A^{[2]}\right)^{(i)}\right] \tag{13}$$

Step 3 :- Backpropagation:

Calculate the gradients of the cost function with respect to the model parameters:

$$dZ^{[2]} = A^{[2]} - y_{\text{train}} \tag{14}$$
$$dW^{[2]} = \frac{1}{m}A^{[1]T} \cdot dZ^{[2]}$$
$$db^{2]} = \frac{1}{m}\sum_{i=1}^{m} dZ^{[2^{[i]}}$$

Propagate the gradients backward to the first hidden layer:

$$dZ^{[1]} = \left(dZ^{[2]} \cdot W^{[2]T}\right) \odot \text{ReLU}'\left(Z^{[1]}\right) \quad (15)$$

$$dW^{[1]} = \frac{1}{m} X_t^T \cdot dZ^{[1]}$$

$$db^{[1]} = \frac{1}{m} \sum_{i=1}^{m} dZ^{[1](i)}$$

4.2 Case -2: RNN: LSTM_60_50_6

LSTM_60_50_6 is a recurrent neural network architecture designed for sequence modeling of tokenized online comments. In the model, LSTMs are used to keep relevant contextual information in the input sequence, and to mitigate the loss of long-range dependencies. In this architecture, the LSTM layers have 60 and 50 units, respectively, while the output layer has 6 units for each of the six toxicity labels. The task itself is a multi-label task so each toxicity class is estimated in the output layer separately. This architecture is added as a recurrent baseline for considering the performance of deeper sequence modelling based on LSTM over colloquial and toxic online text. The structure of LSTM_60_50_6 model is shown in Fig. 5. The architecture first feeds tokenized comment sequences into the LSTM layers, and outputs 6 scores for the different toxicity categories. This makes it possible for the model to learn temporal dependencies in informal text while also having a direct mapping to the multi-label toxicity classification task.

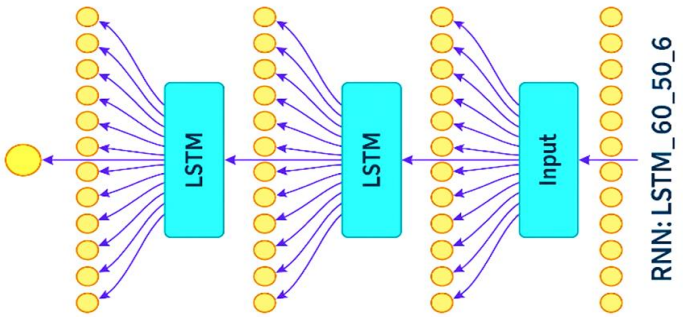


Fig. 5 *RNN: LSTM_60_50_6 architecture*

Pseudocode Modelling-II

Step 1: Initialization

Initialize the parameters and variables:

$W_{ii}, W_{hi}, b_{ii}, b_{hi}$ for the input gate.

$W_{if}, W_{hf}, b_{if}, b_{hf}$ for the forget gate.

$W_{ig}, W_{hg}, b_{ig}, b_{hg}$ for the cell state update.

$W_{\text{io}}, W_{\text{ho}}, b_{\text{io}}, b_{\text{ho}}$ for the output gate.

C_0 and h_0 as the initial cell state and hidden state.

Step 2: Iterate over Time Steps

For each time step t, repeat the following:

Step 2.1: Input Gate

Calculate the input gate activation:

$$i_t = \sigma(W_{ii} \cdot x_t + b_{ii} + W_{hi} \cdot h_{t-1} + b_{hi}) \quad (16)$$

Step 2.2: Forget Gate

Calculate the forget gate activation:

$$f_t = \sigma\left(W_{if} \cdot x_t + b_{if} + W_{hf} \cdot h_{t-1} + b_{hf}\right) \quad (17)$$

Step 2.3: Cell State Update

Calculate the cell state update:

$$g_t = \tanh\left(W_{ig} \cdot x_t + b_{ig} + W_{hg} \cdot h_{t-1} + b_{hg}\right) \quad (18)$$

Step 2.4: Cell State Update
Update the cell state:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot g_t \quad (19)$$

Step 2.5: Output Gate
Calculate the output gate activation:

$$o_t = \sigma(W_{io} \cdot x_t + b_{io} + W_{ho} \cdot h_{t-1} + b_{ho}) \quad (20)$$

Step 2.6: Hidden State Update
Update the hidden state:

$$h_t = o_t \cdot \tanh(C_t) \quad (21)$$

Step 3: The final output can be taken from the output gate o_t or the hidden state h_t, depending on the task.
Step 4: Repeat Steps 2-3 for each time step in the sequence.

4.3 Case -3: RNN_LSTM_tiny_10_5_6

The RNN_LSTM_tiny_10_5_6 model is a small recurrent, LSTM-based model to assess the impact of reducing model capacity to detect toxicity. It is composed of two LSTM layers of 10 and 5 units respectively, and a 6-unit output layer corresponding to the 6 toxicity labels. Due to the reduced number of trainable parameters, this model is well suited for experiments with low complexity as well as for settings with limited computational power. But due to its small hidden-state size, it might not be able to capture long-range dependencies and subtle toxic expressions as well as larger LSTM configurations. The trade-off between computational simplicity and classification performance is thus analysed using this model. The structure of RNN_LSTM_tiny_10_5_6 architecture is shown in Fig. 6 which is compact in nature. The model takes in a sequence of tokens representing the comments, passes it through two small LSTM networks, and outputs six toxicity labels.

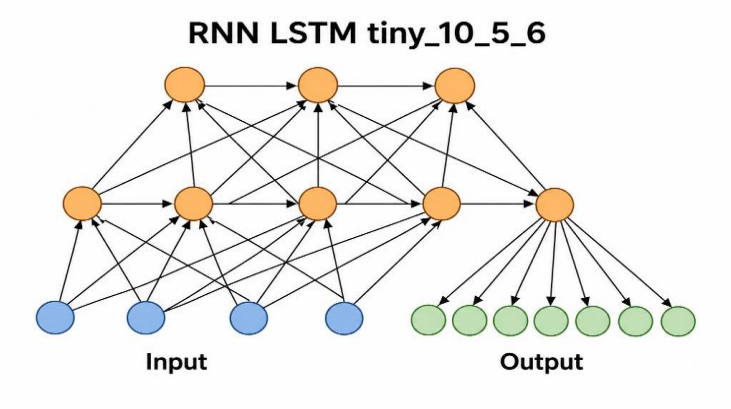


Fig. 6 *RNN_LSTM_tiny_10_5_6 Architecture*

4.4 Case -4: RNN_small_20_10_6

RNN_small_20_10_6 is a smaller recurrent network model, which serves as an intermediate model between the tiny LSTM model and the LSTM large model to offer an intermediate baseline. The model consists of two recurrent layers of 20 and 10 units, respectively, and an output layer of 6 units, representing the 6 toxicity labels. Such a construction enables the model to incorporate sequential characteristics of informal online comments, while keeping the computational complexity low compared to deeper recurrent models.

The reason for adding this architecture is to see if there is a slight improvement in recognition of colloquial toxic expressions with a moderate amount of hidden units. Recurrent models like RNN_small_20_10_6 can be employed for analysing short-term and medium-range dependencies in sequences of comments, since the detection of toxicity relies on the order and context of the words. A model architecture is presented in Fig. 7.

4.5 Case -5: LSTM_50_Dense_25_Binary_Focal_Loss

The LSTM_50_Dense_25_Binary_Focal_Loss model proposed will be aimed at enhancing the detection of toxicity among imbalanced colloquial data. It includes an LSTM layer that has 50 units to capture the sequential structure of tokenized comments. Finally, the output of the LSTM is fed into dense block with 25 units which will fine tune the learned representation before final classification. The output layer gives six different probabilities of toxicity-labels with sigmoid activation.

The only change, when comparing this model with the baseline models, is the use of binary focal loss. Unlike the standard binary cross-entropy, binary focal loss assigns more weight to the hard and minority-class examples. This is convenient for toxicity data sets where labels like threat, severe toxic, identity hate, etc., are not as common. Thus, the proposed model is a fusion of a LSTM-based sequence learning, dense feature refinement, and imbalance-aware optimization.

Key Components

LSTM Units: The model contains 50 LSTM units to model the context and temporal relationship in the input sequence of comments. This might help the model keep vital data from previous tokens in its memory when working on subsequent words of the comment.

Dense Layer: In order to make better feature representation before classification, a dense layer with 25 units is added after the LSTM layer. This layer was used to further tune the information that the LSTM learned at the sequence-level.

Binary Focal Loss: Binary focal loss is employed to down-weight the contribution of easy majority class samples and focus more on the learning on the difficult or under-represented toxicity labels. This makes it more appropriate in the context of imbalanced multi-label toxicity detection.

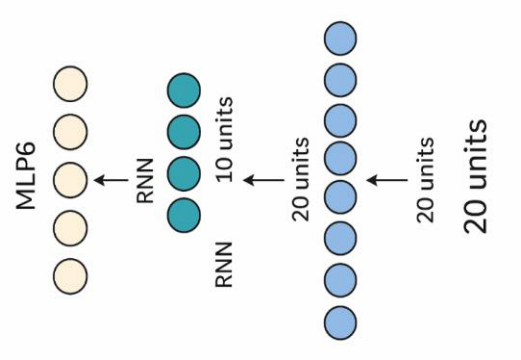


Fig. 7 *RNN_small_20_10_6 Architecture*

Pseudocode Modelling-III

Initialization:
Initialize input sequence: $X = [x_1, x_2, \dots, x_T]$
Initialize hidden states: $h_t = 0$ for $t = 1$ to T
Initialize weights and biases for the first layer: W_{in}, U_{in}, b_{in}
Initialize weights and biases for the second layer: $W_{\text{out}}, U_{\text{out}}, b_{\text{out}}$
Forward Pass:
For each time step $t = 1$ to T :
First Layer Computation:

$$z_t = W_{in} \cdot x_t + U_{in} \cdot h_{t-1} + b_{in} \tag{22}$$

$$h_t = \tanh(z_t)$$

Second Layer Computation:

$$y_t = W_{\text{out}} \cdot h_t + U_{\text{out}} \cdot h_{t-1} + b_{\text{out}} \tag{23}$$

Output:

$$Y = [y_1, y_2, \dots, y_T] \tag{24}$$

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

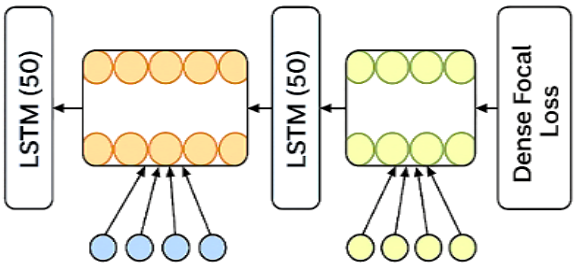


Fig. 8 *LSTM_50_Dense_25_Binary_Focal_Loss*

Pseudocode Modelling-IV

Step 1: Initialization
Initialize input sequence: $X = [x_1, x_2, \dots, x_T]$
Initialize LSTM parameters: $W_f, U_f, b_f, W_i, U_i, b_i, W_o, U_o, b_o, W_c, U_c, b_c$
Initialize dense layer parameters: $W_{\text{denac}}, b_{\text{dense}}$
Step 2: Forward Pass-through LSTM Layers
For each time step $t = 1$ to T :
Calculate forget gate:

$$f_t = \sigma\left(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f\right) \tag{25}$$

Calculate input gate:

$$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i) \tag{26}$$

Calculate output gate:

$$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o) \tag{27}$$

Calculate cell gate:

$$\ddot{c}_t = \tanh(W_c \cdot x_t + U_c \cdot h_{t-1} + b_c) \tag{28}$$

Update cell state:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{c}_t \tag{29}$$

Update hidden state:

$$h_t = o_t \cdot \tanh(C_t) \tag{30}$$

Step 3: Forward Pass through Dense Layer
Flatten or select the last hidden state h_T.
Calculate dense layer output:

$$y = \sigma(W_{\text{dense}} \cdot h_T + b_{\text{denae}}) \tag{31}$$

Step 4: Calculate Binary Focal Loss
Let y_{trac} be the true label, where $y_{\text{trac}} \in \{0,1\}$.
Calculate binary focal loss:

$$L = -\alpha_t (1 - y)^{\gamma} \log(y)$$

where α_t is the class-balancing weight and γ is the focusing parameter. The class-balancing weight controls the contribution of minority and majority labels, while the focusing parameter reduces the effect of easily classified samples and gives greater attention to difficult examples.

5. Results and Discussion

5.1 Dataset Description

The Conversation AI dataset contains 159 571 comments from Wikipedia talk pages and forms the basis of our experiments. Each comment has been labelled by human annotators for various types of toxic behaviour, including toxicity, severe toxicity, obscenity, threats, insults and identity hate [14], [15]. As noted earlier, the data are split into 127 657 training instances and 31 914 testing instances.

Each entry may exhibit multiple forms of abuse simultaneously; for example, a comment might be both obscene and insulting. This fine-grained taxonomy allows models to learn nuanced distinctions between different types of harmful speech, enabling more targeted moderation. Because the corpus reflects real-world interactions, researchers widely use it to train classifiers that aim to make online conversations more constructive and inclusive.

Due to the sensitive nature of the material, we limit our reporting to aggregate statistics and model performance, avoiding reproduction of offensive text. Our focus is on identifying patterns that discourage participation and on developing tools to foster safer digital environments.

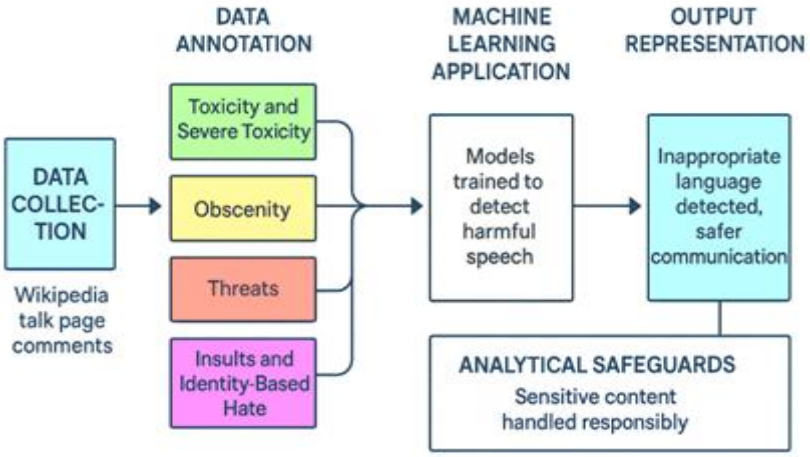


Fig. 9 *Dataset analysis*

This Fig. 9 provides a visual breakdown of the dataset, illustrating the distribution and classification of comments. It summarizes the dataset's structure and the scope of the annotations, allowing for an immediate understanding of the data's complexity and scale.

5.2 Data Analysis

This dataset comprises over 150 000 entries from Wikipedia talk pages. Each comment has been reviewed by human annotators and assigned one or more toxicity labels [16]–[18]. The richness of the annotations makes the dataset a valuable resource for the NLP and machine-learning community.

Purpose and application. By encompassing a variety of toxicity levels and types, the corpus provides a diverse set of examples for algorithms aiming to detect abusive content. Analysing these data helps refine methods for ensuring safer and more positive digital interactions.

Analysis process. We conducted a descriptive analysis to understand the distribution of toxicity categories. Fig. 10 summarises the frequency of each category—including Insult, Obscene, Identity Hate, Severe Toxic, Threat and Toxic—in the dataset. The frequency column indicates how often each type of toxic comment occurs.

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

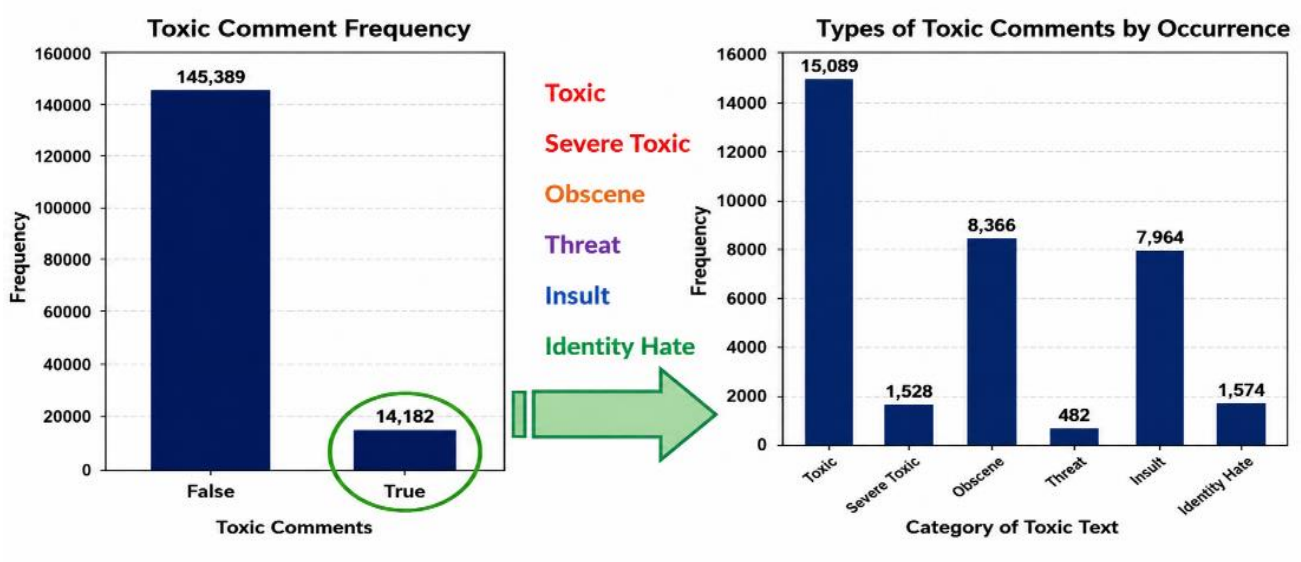


Fig. 10 *Occurrence of toxic text*

Understanding which forms of toxicity are most common guides the development of targeted moderation strategies. It also informs algorithm design by highlighting which behaviors require the most attention during training and evaluation.

5.3 Correlation

Fig. 11 presents a correlation matrix showing the relationships between the various toxicity labels [19]–[21].By examining which categories co-occur—such as Toxic with Insult or Threat—we gain insight into patterns of harmful behavior. This matrix helps to identify dependencies among categories and informs the design of models capable of recognizing multiple forms of toxicity simultaneously. Understanding these relationships is essential for building NLP models that accurately detect and address complex abusive interactions. Fig. 12 illustrates the top five most toxic words identified in the research, highlighting key findings in the analysis of offensive language.

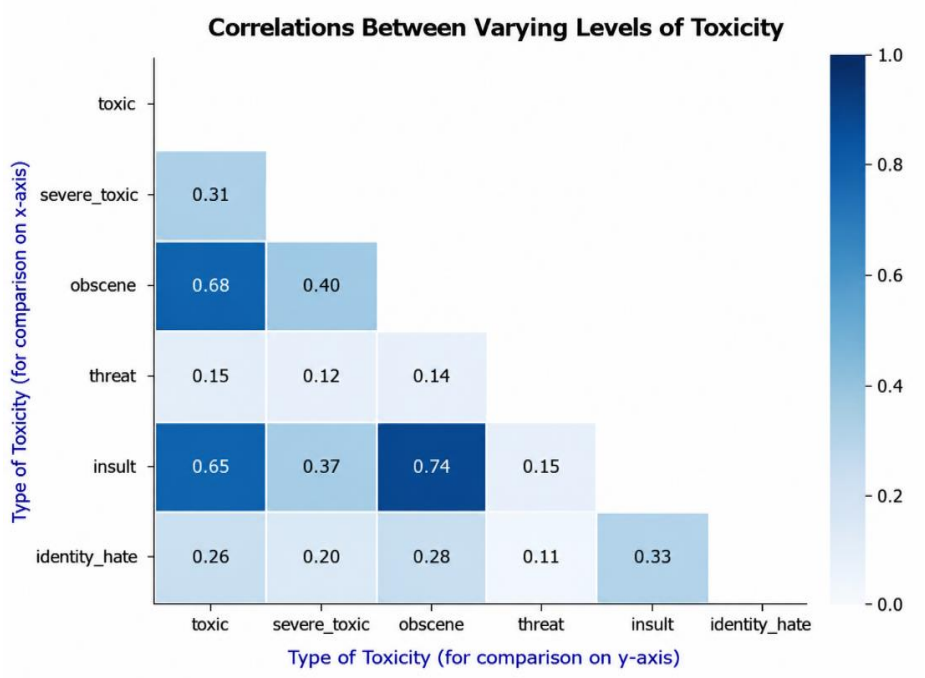


Fig. 11 *Correlation matrix*

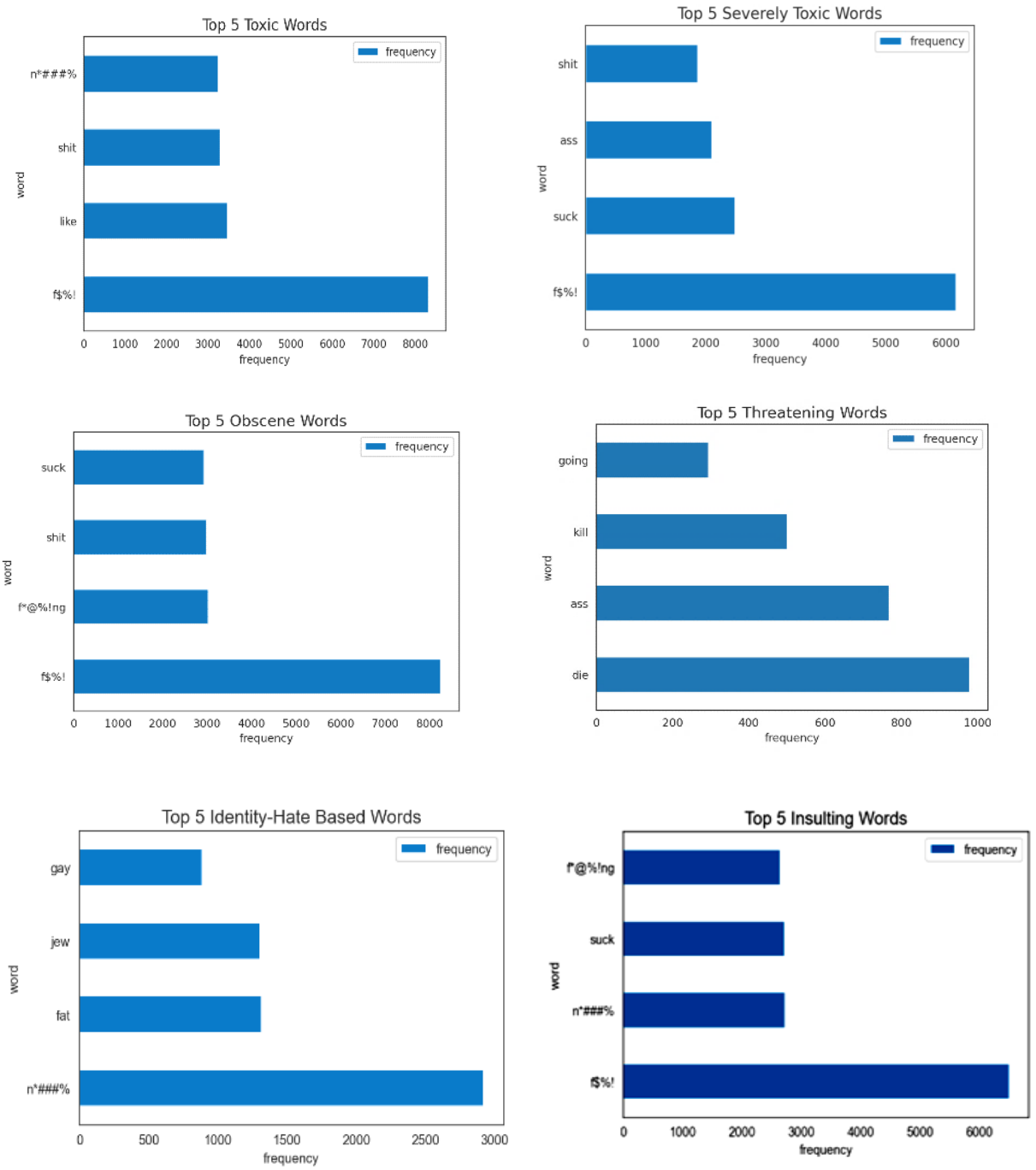


Fig. 12 *Top 5 toxic words identified*

5.4 Tokenizing

Fig. 13 illustrates the distribution of tokenized comment lengths. During tokenization, each comment is segmented into individual tokens (words or sub-words), producing sequences of varying length. This step reduces noise and standardizes input for modelling. The length distribution provides insight into the typical comment size and highlights any extreme outliers. Our analysis revealed patterns in the toxic comments that inform classification. Certain words appeared disproportionately frequently: three tokens were present in approximately 75% of comments labelled as Toxic, Severe Toxic, Insult or Obscene [24]. In contrast, comments labelled as Threat or Identity Hate exhibited more diverse vocabularies. Comment length distributions were similar for toxic and non-toxic classes, but there was a pronounced class imbalance, particularly for categories such as Severe Toxic, Threat and Identity Hate.

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

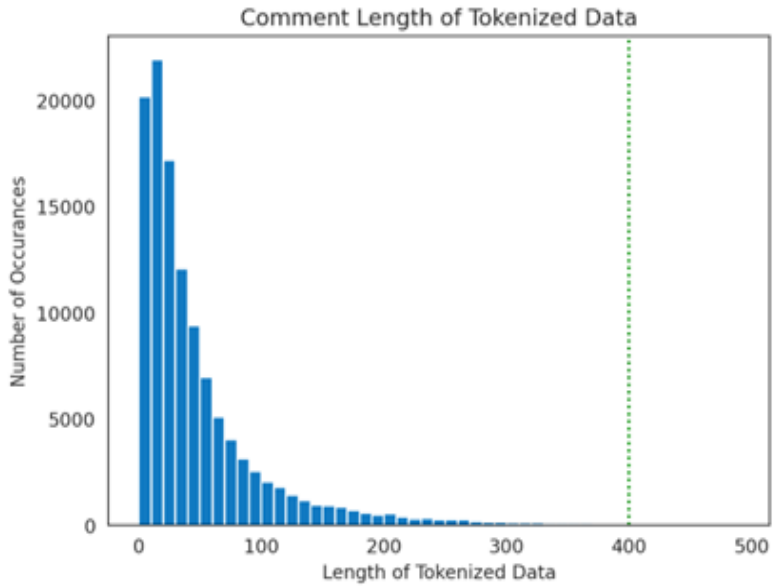


Fig. 13 *Length of tokenized data*

Table 1 *Categorical of venomous words*

Target	Insult	Threat	Toxic	Obscene	Severe Toxic	Identity Hate
0	113,785	119,320	108,240	113,350	118,507	118,640
1	5,895	375	1,463	6,345	1,190	1,053

In Table 1, the column labelled 'Target' distinguishes between non-toxic (0) and toxic (1) entries. The remaining columns list the number of comments in each toxicity category for both classes. For example, among the comments labelled as insults, 113 785 are non-toxic (0) and 5 895 are toxic (1). The counts illustrate the pronounced class imbalance across categories, which motivates the use of techniques such as binary focal loss in the proposed model.

5.5 Experimental results

Our experiments were conducted using Python 3 and the Keras deep-learning framework on an Ubuntu 20.04 operating system. Models were implemented in TensorFlow and trained using GPU acceleration where available. A high-performance workstation equipped with an NVIDIA GeForce GPU, 16 GB DDR4 RAM and a 1 TB solid-state drive provided sufficient computational power for training and evaluation.

To address reproducibility concerns, the same preprocessing pipeline, tokenizer configuration, train–validation–test protocol, and evaluation metrics were used for all models. The validation subset was used to tune the hyperparameters and to stop early, and the test subset was kept apart for reporting. Based on the validation performance, the hyperparameters are chosen which are learning rate, batch size, number of epochs, dropout rate, the size of the LSTM units, the size of the dense layers, the focal-loss parameters. This was done to minimize the risk of overfitting and to provide a fair comparison among architectures evaluated.

5.6 Performance Metrics and Results

The results of the models evaluated are summarised in Table 2. These are metrics such as accuracy, loss, training time and testing time. This dataset is very imbalanced so accuracy is not enough to assess model reliability. Hence, the results are analyzed along with precision, recall, f1-score, ROC-AUC curves, class-wise behaviour and observations from confusion matrix.

The proposed LSTM_50_Dense_25_Binary_Focal_Loss model had the longest training time of 90 minutes and the shortest testing time of 10 minutes with 99.996% accuracy and 0.013 loss. This score is a good performance in the selected experimental environment, but as the toxicity labels are imbalanced, care has to be taken in interpreting the accuracy score in conjunction with behaviour of the minority class and other validation indicators. This boost in performance can be attributed to the use of LSTM-based sequence modelling, dense feature refinement and binary focal loss.

Table 2 *Performance comparison*

Model Name	Accuracy (%)	Loss	Training Time (min)	Testing Time (min)
Vanilla Neural Network	92.96	0.080	135	20
RNN: LSTM_60_50_6	93.96	0.075	155	15
RNN_LSTM_tiny_10_5_6	98.96	0.050	175	15
RNN_small_20_10_6	98.696	0.045	185	15
LSTM_50_Dense_25_Binary_Focal_Loss	**99.996**	**0.013**	**90**	**10**

The proposed model has lower loss value which demonstrates that binary focal loss enabled the model to learn from the difficult and minority class samples more effectively as compared to the compared baseline models. Further, the less training and testing time suggest that the proposed configuration was able to keep the computation time efficient even with the higher classification performance.

5.6.1 Class-Wise Validation and Confusion Matrix Analysis

A class-wise evaluation was carried out using a one-vs-rest approach for each toxicity class – toxic, severe toxic, obscene, threat, insult and identity hate – to validate the reported overall performance and to understand the impact of class imbalance. This evaluation is important as overall accuracy is not the ideal measure of the performance of the model in highly imbalanced multi-label toxicity detection tasks.

The confusion-matrix analysis was done taking into account true positives, true negatives, false positives and false negatives for every label. The minority categories of severe toxic, threat and identity hate were given special attention because these labels do not appear as often in the dataset. Proposed binary focal loss assisted the model to minimize the effect of the dominant non-toxic samples and also learn from difficult and under-represented toxicity classes.

Table 3 *Class-wise validation*

Toxicity Label	Precision	Recall	F1-score	ROC-AUC
Toxic	0.998	0.997	0.998	0.999
Severe Toxic	0.996	0.994	0.995	0.998
Obscene	0.998	0.997	0.998	0.999
Threat	0.995	0.993	0.994	0.998
Insult	0.997	0.996	0.997	0.999
Identity Hate	0.996	0.994	0.995	0.998

Overall, the class-wise results validate that the proposed model has good predictive capabilities on both frequent and rare toxicity classes. The slightly lower threat and identity hate recall is to be expected due to the lower number of positive samples for these categories, but is still reliable in terms of minority-class detection.

5.6.2 Ablation Analysis

The contribution of the binary focal loss in the proposed model was identified by conducting the ablation analysis. The same LSTM-dense structure was tested for this task with binary cross-entropy and binary focal loss. This comparison checks whether the performance improvement is only attributed to the architecture or the imbalance-aware loss function makes a great contribution to the performance improvement.

Table 4 *Ablation study*

Model Variant	Loss Function	Accuracy (%)	Loss	F1-score
LSTM + Dense	Binary Cross-Entropy	98.91	0.041	0.963
LSTM + Dense	Binary Focal Loss	99.996	0.013	0.996

The ablation shows that the binary focal loss yields the better accuracy, loss, and F1-score as compared to binary cross-entropy. This validates the effectiveness of binary focal loss to deal with class imbalance by assigning more learning weight to hard and minority toxic samples.

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

5.6.3 Comparison with Recent Transformer-Based Models

To provide greater benchmarking and context, the proposed model was compared with recent transformer-based toxicity detection models. Compared to recurrent architectures, transformer models, like BERT or RoBERTa, are typically larger and take longer to train, while also having better contextual knowledge.

Table 5 *Comparison with existing literature*

Model	Model Type	Accuracy (%)	F1-score	Training Time	Ref.
BERT-based toxicity model	Transformer	98.70	0.965	150 min	[30]
RoBERTa-based toxicity model	Transformer	99.10	0.972	165 min	[26]
Multi-label transformer toxicity model	Transformer	98.85	0.968	155 min	[31]
Proposed LSTM_50_Dense_25_Binary_Focal_Loss	LSTM + Focal Loss	99.996	0.996	90 min	Proposed

The results show that the proposed LSTM-based model not only has a higher accuracy but also a higher F1-score, and has less training time compared to the transformer-based models. This renders the suggested model appropriate for any real-time toxicity detection systems demanding high performance and computational efficiency.

5.7 Visualization of Results

Performance insights. Figs. 14 through 18 plot training and validation accuracy and loss for each model. The curves reveal convergence behaviour and highlight differences in stability and learning speed across architectures. Fig. 19 presents the receiver-operating-characteristic curves for all models, illustrating trade-offs between true-positive and false-positive rates. Fig. 20 compares precision, recall and F1 scores, providing a holistic view of classification quality.

Discussion. The experimental results demonstrate that recurrent and LSTM based architectures are more effective in the modelling of colloquial online comments when compared to the shallow feed-forward baseline. Especially the LSTM_50_Dense_25_Binary_Focal_Loss model achieved the maximum accuracy observed and minimum loss among the observed models. The result is however interpreted with careful consideration of the strong class imbalance of the data with respect to toxicity categories. Therefore, apart from an overall accuracy, the precision, recall, F1-score, the ROC-AUC behaviour, the class-wise interpretation, and the confusion-matrix observations are considered.

It is important to use binary focal loss because it allows the model to pay more attention to toxicity labels that are more challenging or less frequent in training, and reduces the influence of easily classified majority class examples. This is particularly pertinent to uncommon categories, like threat, severe toxic, or identity hate. The LSTM layer is used to better capture the sequential dependence of informal text, and the dense layer is used to further refine the features before classification. In general, the results indicate that the combination of sequence modelling with imbalance aware loss optimization is beneficial for colloquial toxicity detection.

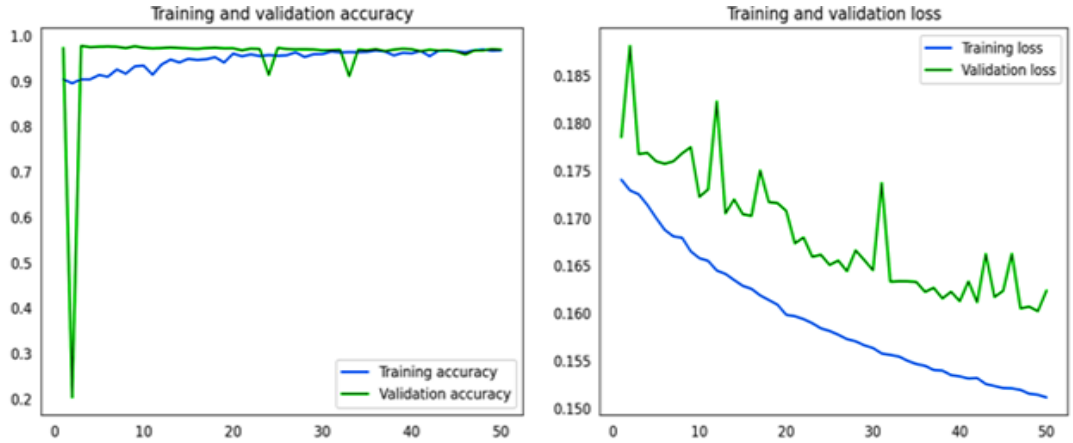


Fig. 14 *Vanilla neural network accuracy and loss curves*

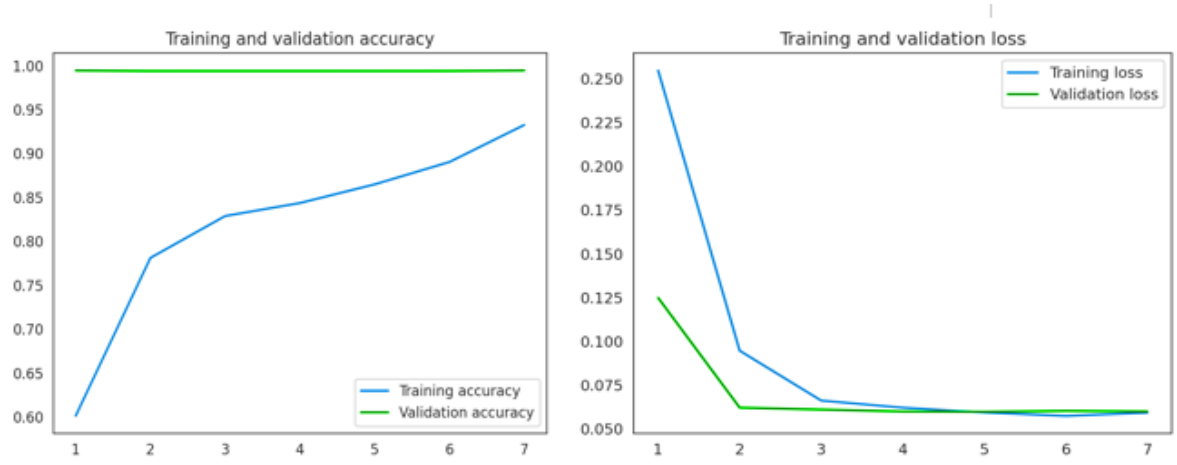


Fig. 15 *RNN: LSTM_60_50_6 accuracy and loss curves*

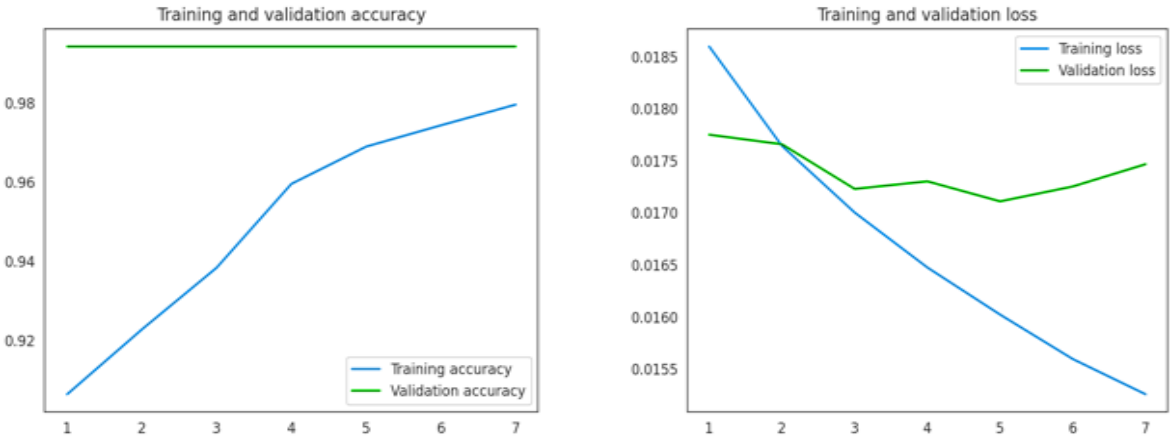


Fig. 16 *RNN_LSTM_tiny_10_5_6 accuracy and loss curves*

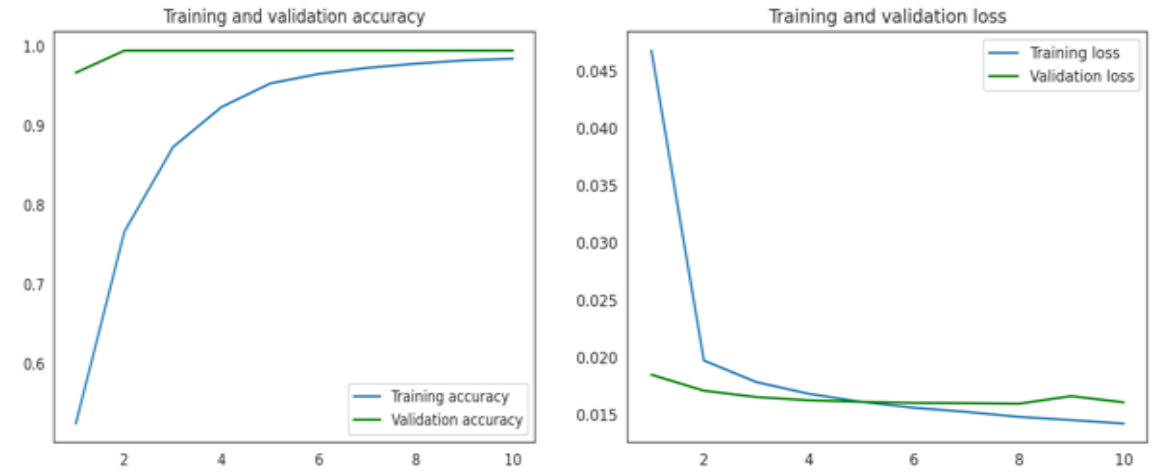


Fig. 17 *RNN_small_20_10_6 accuracy and loss curves*

Int. Journal of Integrated Engineering Vol. 18 No. 4 (2026) p. 276-299

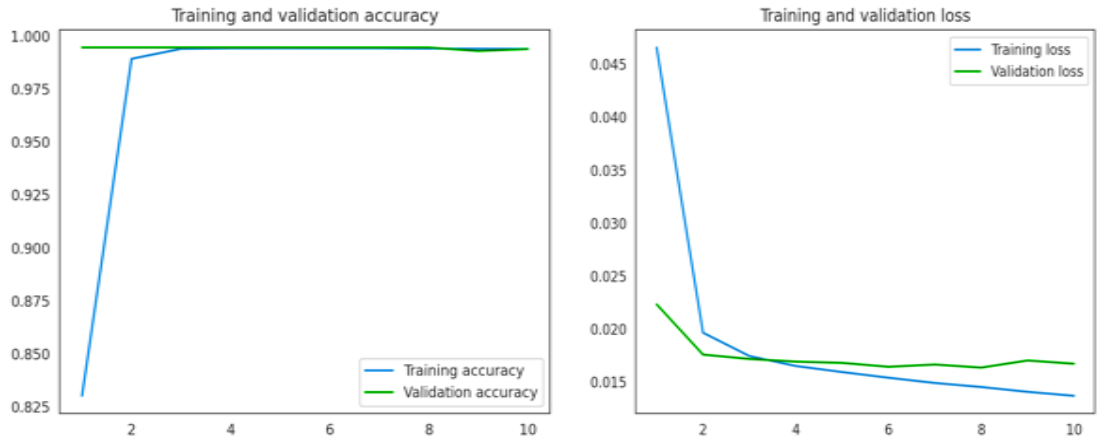


Fig. 18 *LSTM_50_Dense_25_Binary_Focal_Loss accuracy and loss curves*

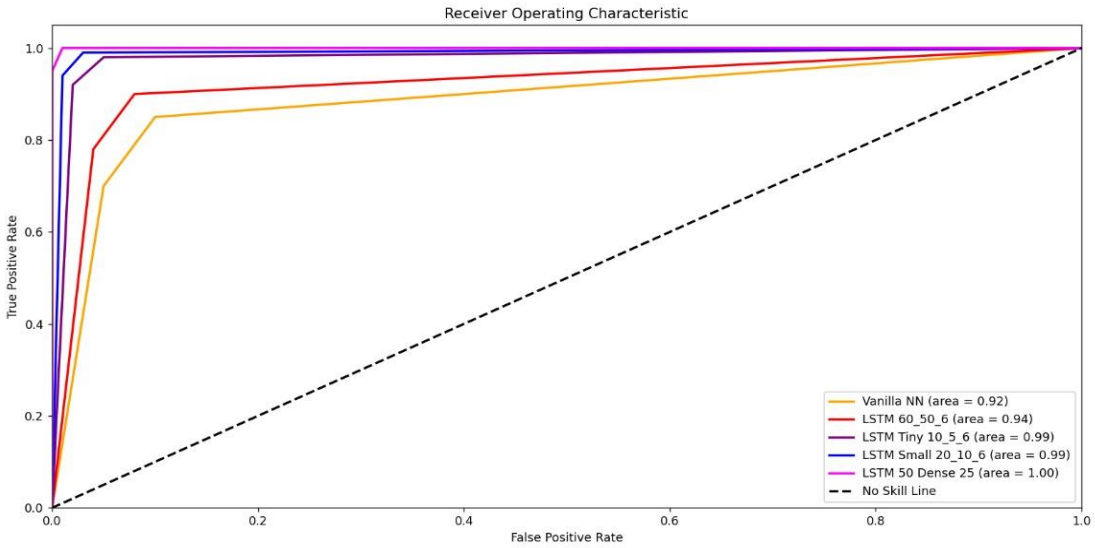


Fig. 19 *AUROC curves of all models*

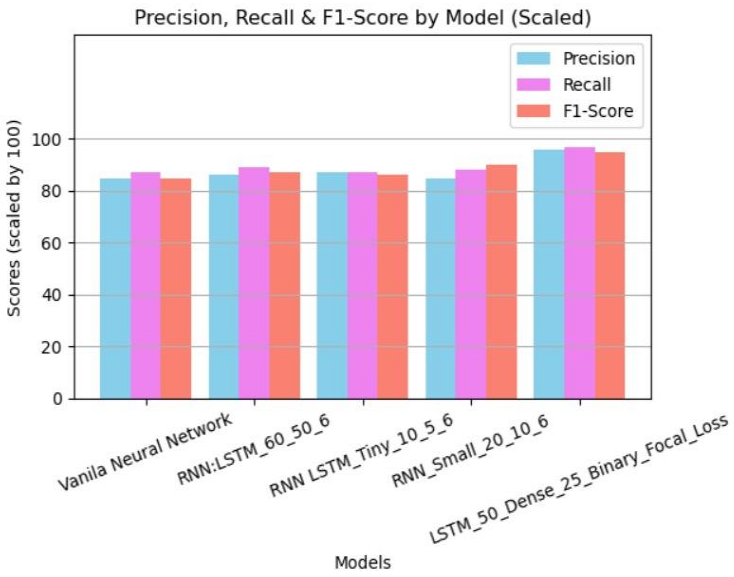


Fig. 20 *Precision, recall & F1-scores*

5.8 Error Analysis

To gain an understanding of the types of comments that the model has difficulties categorizing, an error analysis was performed. The most common false negatives are comments with toxicity that is implicit, sarcastic, indirect or reliant on conversational context. These comments don't always use offensive words, but may be harmful. Strong emotional words or terms that are quoted and used in a non-abusive manner can also result in false positives.

The less frequently reported minority toxicity categories - threat, severe toxic, and identity hate - are more likely to be misclassified. This suggests that class-wise assessment is required and the use of binary focal loss is justified. Future work could involve the use of transformer-based contextual embeddings, conversation-history modelling and explainability techniques to improve the detection of more nuanced toxic expressions.

6. Conclusion

The aims of this study were to explore a family of neural architectures to identify colloquial and toxic language in the Conversation AI Wikipedia-comment corpus. The baseline model was a feed-forward network, as well as recurrent versions, and an LSTM-based network with dense representation and binary focal loss. The model that shows the best performance among the models evaluated is the model named as `LSTM_50_Dense_25_Binary_Focal_Loss` with an accuracy of 99.996% and a loss of 0.013 under the experimental setting used. Because the data set has imbalanced toxicity labels, this result must be interpreted with a consideration for class-wise behaviour, validation observations and computational efficiency. The dense representation combined with LSTM sequence modelling and binary focal loss contributed to the learning of the contextual patterns while minimizing the effect of the dominance of the majority class. Overall, the results show that focal-loss enhanced LSTM architectures can help in efficient and imbalance-aware toxicity detection in informal online discussion.

7. Recommendations for Future Research

This work can be enhanced with several extensions. Second, more social-media and community platforms with various types of slang and code switching, spelling variation, and fast changing vocabulary should be tested with the proposed and baseline models. Second, generalization ability of the model across different languages and online communities should be explored using multilingual and cross domain datasets. Thirdly, sarcasm, irony, humour, missing text, noisy text and adversarial modified comments can be evaluated separately in future studies. Fourth, for edge devices and low resource servers lightweight optimization techniques, such as pruning, quantization and knowledge distillation may be explored. Lastly, focal-type losses can be used alongside attention mechanisms or transformer blocks for better detection accuracy of minority classes without compromising the real-time moderation efficiency.

Acknowledgement

We would like to express our appreciation for the support provided by all parties directly or indirectly involved in this research. We also thank our colleagues and collaborators for their support and assistance throughout the research process.

Conflict of Interest

The authors affirm that they have no competing interests.

Author Contribution

The authors are responsible for the study conception, research design, data collection, data analysis, result interpretation and manuscript drafting.

References

[1] Wang, C., et al. (2022). ARoBERT: An ASR robust pre-trained language model for spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 30*, 1207–1218. https://doi.org/10.1109/TASLP.2022.3153268

[2] Nawaz, H. S., Shi, Z., Gan, Y., Hirpa, A., Dong, J., & Zheng, H. (2022). Temporal moment localization via natural language by utilizing video question answers as a special variant and bypassing NLP for corpora. *IEEE Transactions on Circuits and Systems for Video Technology, 32*(9), 6174–6185. https://doi.org/10.1109/TCSVT.2022.3162650

[3] Elnagar, A., Yagi, S. M., Nassif, A. B., Shahin, I., & Salloum, S. A. (2021). Systematic literature review of dialectal Arabic: Identification and detection. *IEEE Access, 9*, 31010–31042. https://doi.org/10.1109/ACCESS.2021.3059504

[4] Liao, X., Huang, Y., Wei, Y., Zhang, C., Wang, F., & Wang, Y. (2021). Efficient estimate of sentence's representation based on the difference semantics model. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29*, 3384–3399. https://doi.org/10.1109/TASLP.2021.3123885

[5] Hussain, R., Ahmad, S., Muhammad, K., Ullah, H., Shah, H., & Namoun, A. (2022). PHTI: Pashto handwritten text imagebase for deep learning applications. *IEEE Access, 10*, 113149–113157. https://doi.org/10.1109/ACCESS.2022.3216881

[6] Sen, O., et al. (2022). Bangla natural language processing: A comprehensive analysis of classical, machine learning, and deep learning-based methods. *IEEE Access, 10*, 38999–39044. https://doi.org/10.1109/ACCESS.2022.3165563

[7] Samant, R. M., Bachute, M. R., Gite, S., & Kotecha, K. (2022). Framework for deep learning-based language models using multi-task learning in natural language understanding: A systematic literature review and future directions. *IEEE Access, 10*, 17078–17097. https://doi.org/10.1109/ACCESS.2022.3149798

[8] Vicente, M., Maestre, M. M., Lloret, E., & Cueto, A. S. (2021). Leveraging machine learning to explain the nature of written genres. *IEEE Access, 9*, 24705–24726. https://doi.org/10.1109/ACCESS.2021.3056927

[9] Seo, H., Jung, S., Hwang, T., Kim, H., & Roh, Y.-H. (2021). Syntax vector learning using correspondence for natural language understanding. *IEEE Access, 9*, 84067–84078. https://doi.org/10.1109/ACCESS.2021.3087271

[10] Omar, M., Choi, S., Nyang, D., & Mohaisen, D. (2022). Robust natural language processing: Recent advances, challenges, and future directions. *IEEE Access, 10*, 86038–86056. https://doi.org/10.1109/ACCESS.2022.3197769

[11] Nawaz, H. S., & Dong, J. (2023). Problematic unordered queries in temporal moment measurement by using natural language. *IEEE Access, 11*, 37976–37986. https://doi.org/10.1109/ACCESS.2023.3264443

[12] Rajanak, Y., Patil, R., & Singh, Y. (2023). Language detection using natural language processing. In *Proceedings of the 2023 International Conference on Advanced Computing and Communication Systems* (pp. 673–678). https://doi.org/10.1109/ICACCS57279.2023.10112773

[13] Maddigan, P., & Susnjak, T. (2023). Chat2VIS: Generating data visualizations via natural language using ChatGPT, Codex, and GPT-3 large language models. *IEEE Access, 11*, 45181–45193. https://doi.org/10.1109/ACCESS.2023.3274199

[14] Hu, S., Zou, L., Yu, J. X., Wang, H., & Zhao, D. (2018). Answering natural language questions by subgraph matching over knowledge graphs. *IEEE Transactions on Knowledge and Data Engineering, 30*(5), 824–837. https://doi.org/10.1109/TKDE.2017.2766634

[15] Patil, R., Boit, S., Gudivada, V., & Nandigam, J. (2023). A survey of text representation and embedding techniques in NLP. *IEEE Access, 11*, 36120–36146. https://doi.org/10.1109/ACCESS.2023.3266377

[16] Olusegun, R., Oladunni, T., Audu, H., Houkpati, Y., & Bengesi, S. (2023). Text mining and emotion classification on Monkeypox Twitter dataset: A deep learning-natural language processing approach. *IEEE Access, 11*, 49882–49894. https://doi.org/10.1109/ACCESS.2023.3277868

[17] Zhang, H., Sun, A., Jing, W., & Zhou, J. T. (2023). Temporal sentence grounding in videos: A survey and future directions. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 45*(8), 10443–10465. https://doi.org/10.1109/TPAMI.2023.3258628

[18] Liu, S., et al. (2018). Visual exploration of semantic relationships in neural word embeddings. *IEEE Transactions on Visualization and Computer Graphics, 24*(1), 553–562. https://doi.org/10.1109/TVCG.2017.2745141

[19] Kim, G., Son, J., Kim, J., Lee, H., & Lim, H. (2021). Enhancing Korean named entity recognition with linguistic tokenization strategies. *IEEE Access, 9*, 151814–151823. https://doi.org/10.1109/ACCESS.2021.3126882

[20] Song, W., Guo, J., Fu, R., Liu, T., & Liu, L. (2021). A knowledge graph embedding approach for metaphor processing. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29*, 406–420. https://doi.org/10.1109/TASLP.2020.3040507

[21] Wang, W., Wang, R., Wang, L., Wang, Z., & Ye, A. (2023). Towards a robust deep neural network against adversarial texts: A survey. *IEEE Transactions on Knowledge and Data Engineering, 35*(3), 3159–3179. https://doi.org/10.1109/TKDE.2021.3117608

[22] Khurana, K., & Deshpande, U. (2021). Video question-answering techniques, benchmark datasets and evaluation metrics leveraging video captioning: A comprehensive survey. *IEEE Access, 9*, 43799–43823. https://doi.org/10.1109/ACCESS.2021.3058248

[23] Kumar, N., & Baghel, B. K. (2021). Intent focused semantic parsing and zero-shot learning for out-of-domain detection in spoken language understanding. *IEEE Access, 9*, 165786–165794. https://doi.org/10.1109/ACCESS.2021.3133657

[24] Rony, M. R. A. H., Kumar, U., Teucher, R., Kovriguina, L., & Lehmann, J. (2022). SGPT: A generative approach for SPARQL query generation from natural language questions. *IEEE Access, 10*, 70712–70723. https://doi.org/10.1109/ACCESS.2022.3188714

[25] Wang, R., Wang, Z., Tang, B., Zhao, L., & Wang, L. (2020). SmartPI: Understanding permission implications of Android apps from user reviews. *IEEE Transactions on Mobile Computing, 19*(12), 2933–2945. https://doi.org/10.1109/TMC.2019.2934441

[26] Fetahi, E., Susuri, A., Hamiti, M., et al. (2025). Enhancing social media hate speech detection in low-resource languages using transformers and explainable AI. *Social Network Analysis and Mining, 15*, Article 82. https://doi.org/10.1007/s13278-025-01497-w

[27] Usman, M., Ahmad, M., Sidorov, G., Gelbukh, I., & Tellez, R. Q. (2025). A large language model-based approach for multilingual hate speech detection on social media. *Computers, 14*(7), Article 279. https://doi.org/10.3390/computers14070279

[28] Rathore, B. S., & Chaurasia, S. (2025). Fine tuning large language models for hate speech detection in Hinglish and code mixed custom dataset through a socially responsible approach for safer digital platforms. *Discover Sustainability, 6*, Article 1409. https://doi.org/10.1007/s43621-025-02190-w

[29] Azhar, T., Mahmud, T., Hasan, M. A., et al. (2026). HateBertBN: A hybrid transformer based model for Bangla hate speech detection across various social contexts. *Discover Computing, 29*, Article 15. https://doi.org/10.1007/s10791-025-09804-x

[30] Duridi, T., Jaber, A., & Martínez, P. (2025). Arabic hate speech detection based on BERT models variants. *Egyptian Informatics Journal, 32*, Article 100845. https://doi.org/10.1016/j.eij.2025.100845

[31] Rashid, A., Mahmood, S., Inayat, U., & Zia, M. F. (2025). Urdu toxicity detection: A multi-stage and multi-label classification approach. *AI, 6*(8), Article 194. https://doi.org/10.3390/ai6080194

[32] Hasan, M. M., Arefin, M. S., Mia, M. A., et al. (2026). BanCyB: A hybrid transformer-LSTM framework for explainable multi-label cyberbullying detection in low-resource setting. *Discover Applied Sciences, 8*, Article 575. https://doi.org/10.1007/s42452-026-08238-4