Toxic Comment Detection Using Bidirectional Sequence Classifiers

Amit Maity
Computer Engineering
K.J. Somaiya Institute of Technology
amit.maity@somaiya.edu

Rishi More
Computer Engineering
K.J. Somaiya Institute of Technology
rishi.vm@somaiya.edu

Prof. Abhijit Patil
Computer Engineering
K.J. Somaiya Institute of Technology
abhijit.patil@somaiya.edu

Jay Oza
Computer Engineering
K.J. Somaiya Institute of Technology
jay.oza@somaiya.edu

Gitesh Kambli
Computer Engineering
K.J. Somaiya Institute of Technology
gitesh.kambli@somaiya.edu

***Abstract*—With the rising surge of online toxicity, automating the identification of abusive language becomes crucial for improving online discourse. This study proposes a deep learning system that efficiently uses multiple labels to classify harmful comments using bi-directional Long Short-Term Memory (LSTM) networks. By leveraging contextual information, the bi-LSTM model achieves state-of-the-art performance in classifying subtle forms of toxicity such as threats, insults, identity hate, and obscenity. The model achieves above 95% accuracy on benchmark datasets with rigorous data processing, optimized neural architecture, and the utilization of FastText embeddings to handle words that are not in the vocabulary. This technique can automatically filter different levels of toxicity, promoting positive online interactions when integrated into online platforms. The proposed study outlines an end-to-end pipeline incorporating recent NLP advancements and deep contextualized language models to address contemporary challenges in AI-enabled content moderation.**

***Keywords*—natural language processing, sequence modeling, long short-term memory, toxic comment detection, bidirectional classifier**

I. Introduction

With the increasing popularity of social media, online conversations are increasingly being affected by toxic comments. Such comments, characterized by their offensive, aggressive, and insulting nature, hurt internet conversation. There is a growing need for efficient toxicity detection and classification systems due to the increase in toxic comments. In this study, we present a natural language processing (NLP) based approach for comment toxicity classification.

Our objective is to accurately categorize toxic comments into fine-grained classes such as obscenity, identity-based hate, threats, insults, severe toxicity, and general profanity. The model takes as input comments from online platforms and outputs predicted toxicity labels. The project is structured in two phases - Phase I focuses on evaluating toxicity, while Phase II delves into data preprocessing.

In Phase I, we leverage powerful NLP tools including FastText for enriched word embeddings and spaCy for linguistic analysis to assess comment toxicity. The embeddings from FastText, which capture subword information, are particularly valuable for our text classification tasks. In Phase II, we refine our dataset using techniques like lemmatization, stopwords removal, and cleaning to improve data quality before feeding into our model.

Our classification model employs a bidirectional LSTM architecture enhanced with FastText embeddings for interpreting comment text and identifying different types of toxicity. The multi-label nature of the data, with binary label values, renders this a multi-label classification problem. By automatically detecting different forms of toxicity, we aim to promote healthier online conversations and mitigate unintended bias.

Toxic comments often compel users to disengage from discussions and discourage diverse perspectives. Automated toxicity classification can alert users to unwelcome messages and enable filtering. This project applies NLP and algorithms to extract key patterns from comments, serving the broader goal of promoting constructive dialogue in online communities.

II. Literature Survey

Toxic online comments have become a major concern as the ubiquity of social media leads to the proliferation of harmful content [1], [2]. A large-scale social media study by Hanjia Lu et al. [1] analyzed 46,058 Twitter users to understand public opinion and online hate speech toward the #StopAsianHate and #StopAAPIHate movement. The study provides insights into detecting and analyzing anti-Asian hate speech across different demographics. Recent research has explored various machine learning such as J48graft [3] and deep learning techniques including Pattern-Based Deep Hate Speech Detection (PDHS) [4] to detect and moderate toxic comments automatically.

A. Classical Machine Learning Frameworks

The paper by Vaibhav Rupapara et al. [5] present a new ensemble method for social media platform toxic comment classification called Regression Vector Voting Classifier (RVVC). Under soft voting criteria, the RVVC model combines logistic

regression and support vector classifier. An unbalanced dataset from Kaggle, comprising 15,294 toxic and 143,346 non-toxic comments, was used for the authors' experiments. They use bag-of-words and TF-IDF for feature extraction after applying preprocessing. SMOTE oversampling and random undersampling are used to resample the unbalanced dataset. The RVVC model is contrasted with SVM, RF, GBM, LR, and KNN, among other machine-learning models. Findings indicate that on the SMOTE oversampled balanced dataset, RVVC uses TF-IDF features to obtain the highest accuracy of 0.97 and F1-score of 0.97. The authors conclude that SMOTE oversampling, as opposed to undersampling or no sampling, performs better when used to balance an imbalanced dataset. The ensemble RVVC model outperforms state-of-the-art methods for toxic comment classification by utilizing the predictions from highly effective LR and SVC models.

Rahul et al. [6] present an approach to toxic classification using machine learning techniques. Toxic comments refer to abusive, offensive, or hostile remarks made on online platforms. The authors mentioned that such comments negatively impact healthy discussion and free speech. The goal was to accurately detect toxicity to help limit its harmful effects. Six machine learning algorithms were used by the authors: decision trees, random forests, logistic regression, SVM, naive Bayes, and KNN. A dataset of comments is subjected to algorithms that categorize them into six categories: non-toxic, threats, insults, obscenity, etc.

The comments undergo preprocessing, which includes removing punctuation, eliminating stop words, and performing stemming/lemmatization. The threshold length was set to 400 characters. Evaluation metrics of log loss and hamming loss were used, suitable for multi-label classification. Their performance was compared based on accuracy, hamming loss, and log loss. Logistic regression achieves the highest 89.46% accuracy and lowest 2.43% hamming loss. The random forest achieves the lowest 0.58% log loss.

Beyond textual comments, Siddique et al. in 2020 explored detecting toxicity in song lyrics [7]. Using a dataset labeled with valence scores, they tested algorithms including Random Forests. With 93.5% accuracy, Random Forest performed best for classifying lyrics as toxic or non-toxic based on language.

B. Deep Learning Architectures

Recurrent neural networks (RNNs) with long short-term memory (LSTM) are a promising method for handling sequential data, such as text. To detect toxicity levels in online comments, the authors Selim et al. [8] propose three novel hybrid deep learning models using EfficientNetB7 with TCN, LSTM, and Bi-LSTM. They introduce a new dataset of Egyptian comments and demonstrate improved accuracy over existing models on public and new datasets for multi-level toxic comment classification.

To categorize toxic comments, the authors Zaheri et al. [9] present a multi-label classifier that combines deep learning approaches with classic machine learning algorithms. Following training on a large dataset of toxic or non-toxic remarks, performance metrics such as accuracy, recall, and F1-score are used to evaluate the model. The findings show that the proposed model beats current strategies inappropriately categorising different categories of toxicity.

The study also addresses the drawbacks of the suggested approach, including the requirement for a sizable labeled dataset and the difficulty in handling unbalanced data. The authors propose several avenues for further research, including determining how to apply active learning and transfer learning strategies to improve the model's performance.

In the year 2020 study conducted by Abbasi et al., the authors addressed the multi-label classification of toxic comments related to religion and race/ethnicity, as discussed in their work [10]. For their research, they utilized the Religious Toxic Comment (RTC) and Race/Ethnicity Toxic Comment (RETC) datasets and employed deep learning models with word embeddings. Notably, their CNN model demonstrated exceptional performance, achieving an accuracy rate of 95-97%, thereby outperforming RNNs and demonstrating the effectiveness of deep learning in multi-label toxic comment classification.

In their novel approach to toxic comment classification, the authors Devtulla et al. [11] make use of an automatic deep learning-based model for the detection and classification of toxic comments. The proposed model combines LSTMs networks and CNNs to identify both local and global patterns in the text. The authors compare and contrast their proposed approach to harmful speech classification with several existing approaches. The findings demonstrate that the proposed method beats existing methods in terms of accuracy and efficiency.

The authors Morzhov et al. [12] combine CNN and RNN models in their approach to harmful comment classification. They explain the dataset they used, which was made public via the Civil Comments platform and annotated by humans for a variety of potentially harmful conversational traits, to train and assess their methods. The authors' algorithms get good results when it comes to correctly and accidentally identifying offensive statements.

All in all, this study provides a comprehensive analysis of the problem of risk-classifying comments and presents a strategy, that focuses on bias and blends CNN and RNN models, presents a viable solution for this important problem in online platforms.

The proposed approach in the research conducted by Garlapati et al. [13] also involves using a combination of feature engineering and deep learning models to classify toxic comments. To train an LSTM and CNN, the authors extract word embeddings, sentiment scores, and part-of-speech tags from the comments. Local characteristics are extracted from the comments using the CNN, and the temporal correlations between words are extracted using the LSTM.

The authors use two publicly available datasets to compare the performance of their technique against alternative baseline models. According to the findings, their method performs better than the baseline models in terms of F1 score, accuracy,

precision, and recall. A sensitivity study is also carried out to ascertain how various variables affect the classification performance.

Li et al. [14] developed a multilingual harmful comment categorization strategy using the XLM-RoBERTa model. The authors use a dataset with comments in six different languages to train and test the model. The goal is to classify comments as harmful or non-toxic. It's critical to identify toxic comments in online arguments since they negatively affect the tone of the discussion. The XLM-RoBERTa model is selected due to its wide range of language processing capabilities and strong performance on NLP tasks.

The authors also preprocess the dataset to understand the distribution of languages and frequent words. They found that Turkish has the most samples. For model training, they use a focal loss function and optimize hyperparameters like learning rate and batch size using grid search. The XLM-RoBERTa model is compared to the LSTM and RNN models. On the evaluation metrics of AUC ROC score and accuracy, the XLM-RoBERTa model achieves the best performance with 0.9306 and 0.972 respectively showcasing its effectiveness for the multilingual hate speech classification task.

The research paper by Krishna Dubey et al. [15] presents an approach to addressing the issue of toxic comments in online communication. The study highlights the increased problems that hate speech, derogatory comments, and objectionable information on different internet platforms bring about for people's emotional and mental health. The authors suggest employing LSTM neural network-based deep learning algorithms to automatically detect and categorize remarks that constitute a risk.

The study combines NLP methods with deep learning techniques, including LSTM [16], a kind of Recurrent Neural Network (RNN). LSTM outperforms a basic RNN because it can deal with the vanishing gradient problem, which can severely limit training efficacy. The vanishing gradient problem occurs when gradients calculated through backpropagation become very small, nearly zero, preventing a network from learning effectively. LSTM architectures address this issue by introducing a memory cell state that allows gradients to flow unchanged, thus enabling continued learning.

The authors also introduce the concept of word embeddings, which helps capture the semantic relationships between words and provides contextual information for text classification. They employ LSTM neural networks to train the model, achieving promising results with 94.94% accuracy, 94.49% precision, and 92.79% recall on test data. The research emphasizes the importance of using deep learning models for toxic comment detection to make online communication platforms cleaner and safer for users.

Despite LSTM's [16] strong performance, limitations persist including computational complexity and the large datasets required for training. Data imbalance also remains an issue, with some studies seeing far lower true positive rates than overall accuracy. Ongoing research aims to address these challenges. Nonetheless, Bi-LSTM has proven highly capable of identifying textual toxicity across research studies and real-world implementations. With social media toxicity proliferating, deep learning methods like Bi-LSTM offer automated, scalable solutions for moderation. Their continued development and deployment should be prioritized to foster healthier online communication.

The research by Mohd Fazil et al. [17] highlights this, suggesting an attentional multi-channel convolutional-BiLSTM network for categorizing hate speech in online social networks. Using multiple-word representation techniques, their model outperforms state-of-the-art methods on Twitter datasets. An ablation study and empirical analysis provide insights into optimal model configurations for hate speech detection.

Future work may involve comparing Bi-LSTM to other deep learning architectures like BERT, expanding training datasets through augmentation techniques, and exploring ensembles or hybrid approaches. However fundamental research indicates Bi-LSTM provides optimal inductive bias for sequential data, positioning it as the foremost deep learning technique for toxic comment classification as of 2023.

In conclusion, this literature survey demonstrates a convergence in research findings highlighting the superiority of Bi-LSTM networks for classifying textual toxicity. Bi-LSTM's inherent memory enables nuanced modeling of context and semantics within comments. Bi-LSTM remains poised to serve as a core technique as research progresses in automatically detecting and mitigating online toxicity.

III. Methodology

A. Data Description and Preprocessing

The dataset used in this study was obtained from Kaggle and consists of eight columns and about 159,571 rows. It has been labeled as id, insult, comment text, toxic, identity hate, vulgar, and very toxic. Every class is binary, with the existence or absence of specified properties in the comments indicated by a 0 or 1. To gather insights, we used exploratory data analysis (EDA) on the dataset, applying several visualization approaches such as pie charts, bar graphs, and correlation maps. These visualizations gave useful insights into the dataset's properties as well as the interrelationships between different categories.

The distribution of the dataset according to classes is depicted in Fig. 1, with over 80% of the comments being non-toxic and the remaining 20% being hazardous. This shows that, while hazardous content is a major worry on the site, there is also a substantial amount of non-toxic content. Fig. 2 displays the dataset's class distribution, with insults being the most prevalent sort of harmful remark, followed by obscenities.

Natural language processing (NLP) tasks require text preprocessing, especially if the data being processed is obtained from online social media platforms. This data, due to its nature, often contains noise, including special characters, repeated characters, non-English characters, and other forms of textual irregularities. To prepare this data for our analysis, we implemented a series of pre-processing steps as described below (and outlined in Fig. 3):

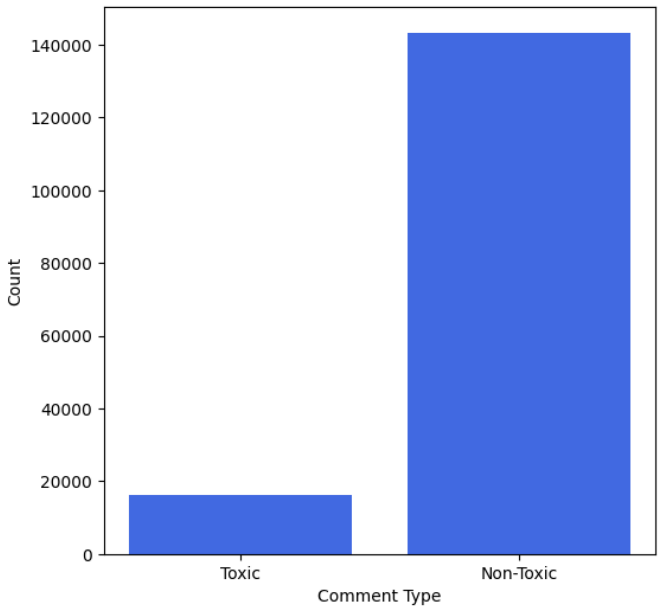

Fig. 1. Class Distribution Across the Dataset

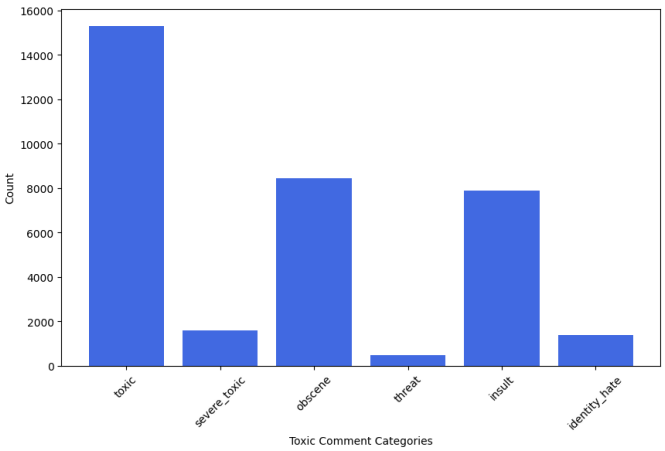

Fig. 2. Category-wise Distribution of the Dataset

- **Text Normalization**: The first step in the pre-processing pipeline is text normalization. All text is converted to lowercase to ensure uniformity. This step helps in treating uppercase and lowercase versions of words as equivalent.
- **Removing Pattern Text**: We maintain a dictionary of patterns and their corresponding target strings (RE_PATTERNS) to identify and replace specific patterns commonly found in online social media text. This step allows us to remove or replace specific patterns or symbols as needed.
- **Changing Repeating Characters**: Online text often includes consecutive repeating characters, which are reduced to a single instance of the character (e.g., ”coooool” becomes ”cool”). This step helps in reducing elongated words commonly found in social media text.

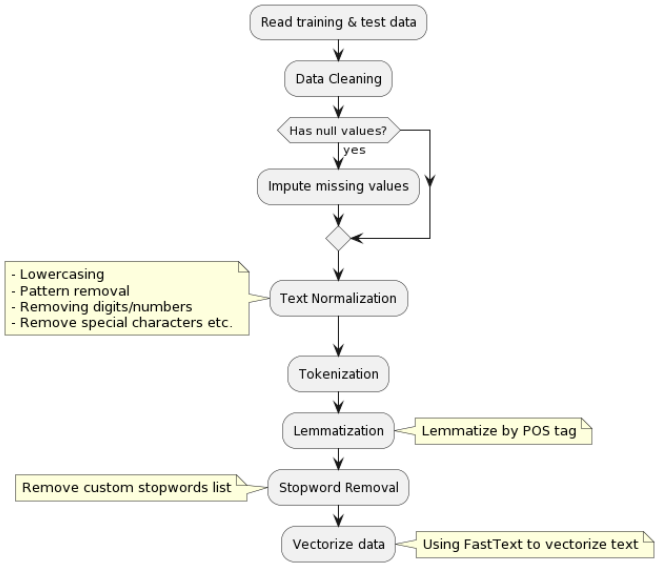

Fig. 3. Data pre-processing steps

- **Replacing ”\n” with a Space**: To ensure consistent formatting, newline characters (”\n”) are replaced with spaces. This step is particularly relevant when dealing with text that might contain line breaks or formatting.
- **Removing Non-Alphanumeric Characters**: A regular expression is used to eliminate non-alphanumeric characters, replacing them with spaces. This action helps in removing special characters, emojis, or symbols that may not be pertinent to our analysis.
- **Removing Digits**: All numeric digits (0-9) are removed from the text. In online social media data, numbers are often used for counts, timestamps, or other purposes that may not be relevant to our text analysis.
- **Replacing Multiple Consecutive Spaces**: To eliminate unnecessary whitespace, multiple consecutive spaces are replaced with a single space. This step ensures that text with excessive whitespace is standardized.
- **Removing Non-ASCII Characters**: Non-ASCII characters, such as emojis, special characters, or characters from languages other than English, are removed.

Lemmatization is a text-processing technique that simplifies words to their base or root form. We employ the WordNet Lemmatizer to lemmatize the textual data. The three steps in this technique were tokenization, lemmatization based on word parts of speech (POS), and reconstruction of the lemmatized text. The lemmatized text data, which lacked variations and inflections, was used for training and testing the model.

Online social media text usually has an abundance of terms and phrases that may not improve understanding or analysis of the content. To overcome this, we used the stop words module from the spaCy library’s English language package. In addition, we performed the following tasks to enhance the dataset fed into our model:

- **Stop Word Generation**: An iterative process resulted in

a comprehensive list of possible stop words. This list of possible stop words comprises both well-known English terms and words taken from the corpus of the book.

- **Stop Word Identification**: A subset of putative stop words was found through data analysis. Terms like "editor," "reference," "thank you," "work," and a few more were included in this subset.
- **Stop Word Augmentation**: A list of traditional stop words was supplemented with common stop words. Both conventional and domain-specific stop words were included in this expanded list.
- **Stop Word Removal**: The identified stop words were removed from the text data. This process involved iterating through the text and eliminating the occurrences of these words, ultimately enhancing the relevance and coherence of the text.

Subsequently, we employed tokenization to break down each comment into individual units, enabling the conversion of textual data into a numerical format that our model can interpret. To address variable text lengths, we performed padding, ensuring that all sequences conformed to a consistent maximum length. Sequences of numerical values obtained from the tokenization step were padded to match the maximum sequence length. Any sequences shorter than this length were padded with zeros, while longer sequences were truncated. The outcome of tokenization and padding was the creation of input data representations suitable for our deep learning model.

To convert preprocessed text into numerical vectors for machine learning models, FastText word embeddings were utilized. FastText was chosen for its ability to handle subword information and rare words effectively. Each token in the preprocessed text was replaced with its corresponding FastText word embedding vector. These embeddings played a crucial role in enabling the model to understand semantic relationships between words and effectively handle out-of-vocabulary words in the toxic comment detection process. Fig. 5 describes the system architecture of our toxic text classification model using a bidirectional LSTM (Bi-LSTM) network.

B. Model Architecture

Model Selection and Architecture: We selected a Bi-LSTM neural network as our model architecture for the multi-class text classification task. LSTMs are a kind of RNN that can learn long-term associations, making them useful for sequence prediction and classification . Because LSTM contains feedback connections, it can evaluate both long sequences of data and single data points like pictures. A two-way street because of LSTM's capacity to gather contextual information in both forward and backward directions, it was well-suited for analyzing the text's whole context.

Bi-directional LSTM: Our model implementation hinged on the utilization of a Bi-directional LSTM architecture, purposefully designed for comment classification. The neural network's architecture was constructed using dense layers and entailed the incorporation of key components such as the embedding layer, bidirectional LSTM layers (comprising both forward and backward contexts), and a well-defined output layer. Fig. 4 showcases our proposed LSTM network architecture for classifying comments as toxic or non-toxic. The architecture consists of the following components:

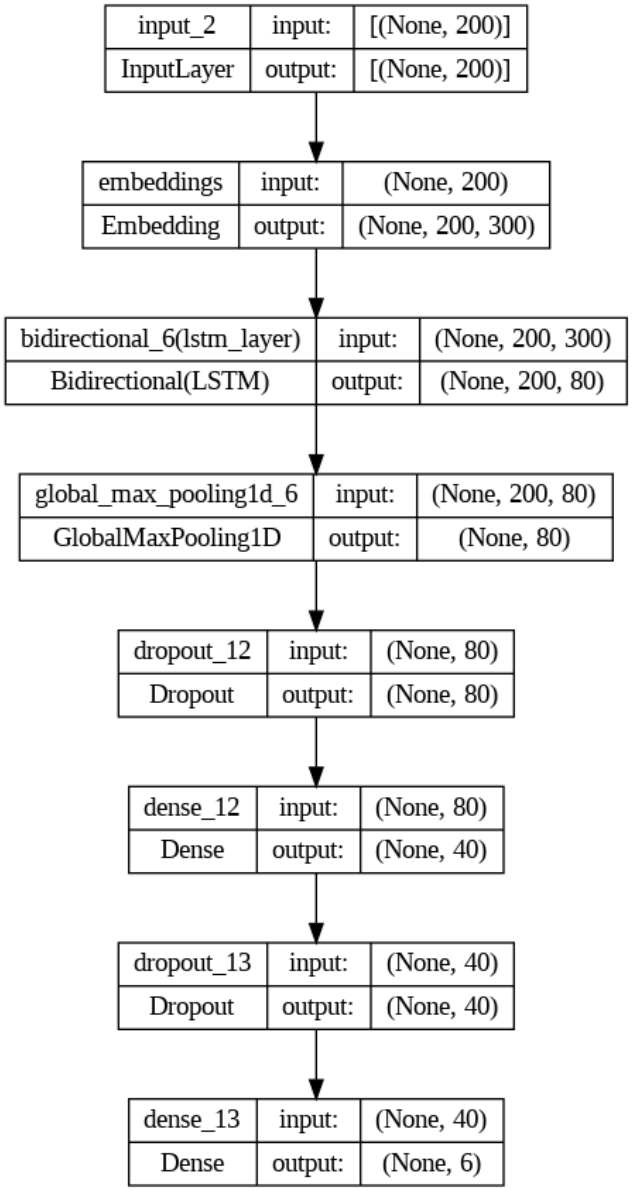

Fig. 4. Proposed LSTM Architecture

- **Embedding layer**: This layer converts the input text into a dense vector representation. As a result, the model is able to understand more intricate relationships between words in the input text.
- **Bidirectional LSTM layer**: This layer learns long-range temporal dependencies in the input text. With its ability to capture the context of individual words in a sentence, it is especially well-suited for text classification tasks.
- **Global max pooling layer**: This layer extracts the most important features from the output of the bidirectional

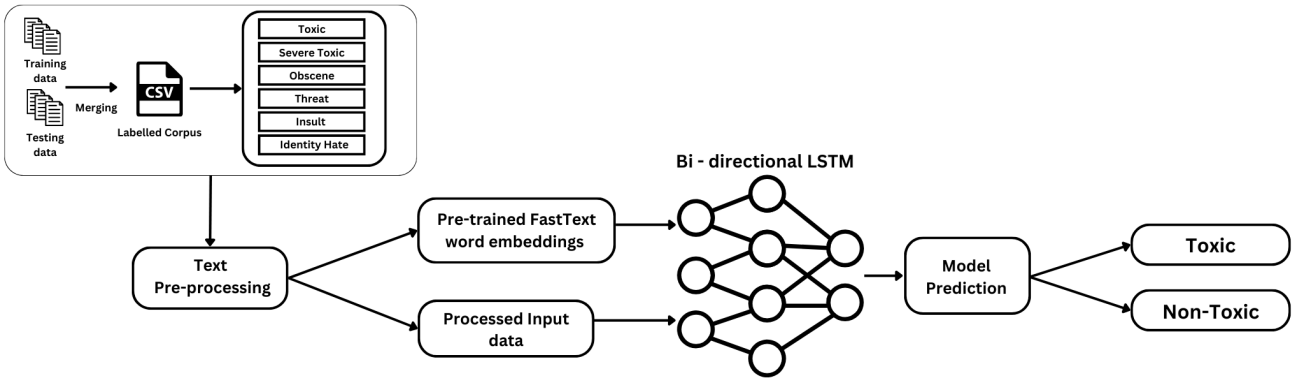

Fig. 5. System Architecture

LSTM layer.

- **Dropout layer**: This layer prevents overfitting by randomly dropping out neurons during training.
- **Dense layers**: These layers learn the mapping between the extracted features and the output labels.

Model Training and Hyperparameter Tuning: We trained the selected model using preprocessed and embedded text data. This training process involved the following:

- **Hyperparameter Tuning**: We fine-tuned the hyperparameters of the model that included learning rate, batch size, and the number of LSTM units, to optimize its performance.
- **Model Training**: The model was trained on the training set with the goal of minimizing a defined loss function. The training process involved multiple epochs to ensure that the model learned the underlying patterns in the data.

Model Evaluation: Using a variety of evaluation criteria, we assessed the model's performance following training. The assessment procedure comprised:

- **Validation Set**: To track the model's generalization and early stopping based on its validation loss, and to evaluate its performance on the validation set.
- **Test Set**: To give an accurate assessment of the performance of the model on unobserved data, a held-out test set was used for the final evaluation.
- **Performance Metrics**: The model's classification was evaluated using common assessment measures like accuracy, precision, recall, F1-score, and confusion matrices.

Inference and Predictions: Following training and evaluation, the model could be utilized for predicting new, unseen text data. The model was integrated into the project's workflow, enabling automated classification of text documents into their respective categories.

IV. Results and Analysis

TensorFlow and Keras were used in the implementation of the suggested bidirectional LSTM model. A dataset of around 159,571 internet comments classified as vulgar, insulting, threatening, toxic, severe toxic, and identity hate was used to train the classifier. The dataset was divided into three sets: test (10%), validation (10%), and training (80%).

The hyperparameters of the model were tuned using the validation set. The final hyperparameters used were an embedding dimension of 300, 2 bidirectional LSTM layers with 128 units each, a dropout rate of 0.2, and the optimization was carried out using Adam optimizer with a 0.001 learning rate. The model was trained with a batch size of 256 across 100 epochs.

On the held-out test set, the model's performance was evaluated. The model classified the various forms of toxicity with an overall accuracy of 95.2%. Table II displays the precision, recall, and F1-score for each of the toxicity classes.

TABLE I
Training and validation loss and accuracy at select epochs.

Epoch	Train Loss	Train Accuracy	Val Loss	Val Accuracy
1	0.116	0.629	0.055	0.994
10	0.0448	0.969	0.0462	0.994
25	0.0355	0.972	0.049	0.994
50	0.0255	0.935	0.0622	0.99

The training and validation accuracy and loss over 100 training epochs are shown in Figures 6 and 7. As shown in Fig. 6, the training loss decreased from 0.116 at epoch 1 to 0.0255 at epoch 50 as the model fitted the training data. The increasing validation loss, as shown in Fig. 6, can potentially be attributed to a class imbalance in the dataset since the number of toxic samples are comparatively less as compared to the non-toxic samples as shown in Fig. 1. Therefore, even as the model

TABLE II
Performance metrics over 100 epochs

Toxicity Type	Precision	Recall	F1-Score
Toxic	0.94	0.93	0.93
Severe Toxic	0.91	0.89	0.90
Obscene	0.93	0.95	0.94
Threat	0.87	0.85	0.86
Insult	0.96	0.97	0.96
Identity Hate	0.89	0.88	0.88

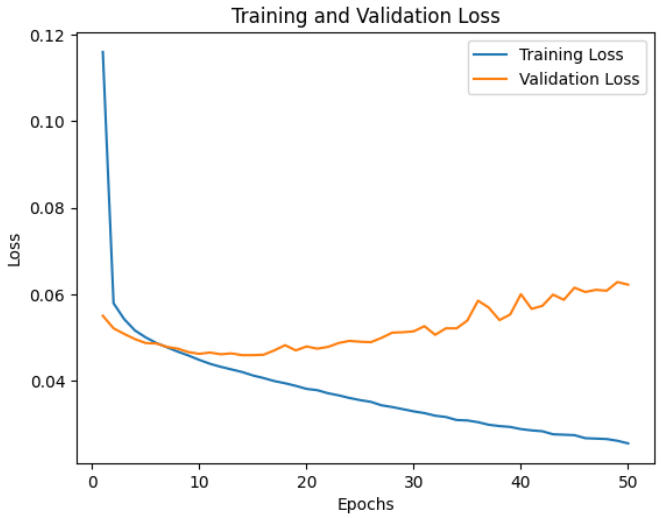

Fig. 6. Training and Validation Loss

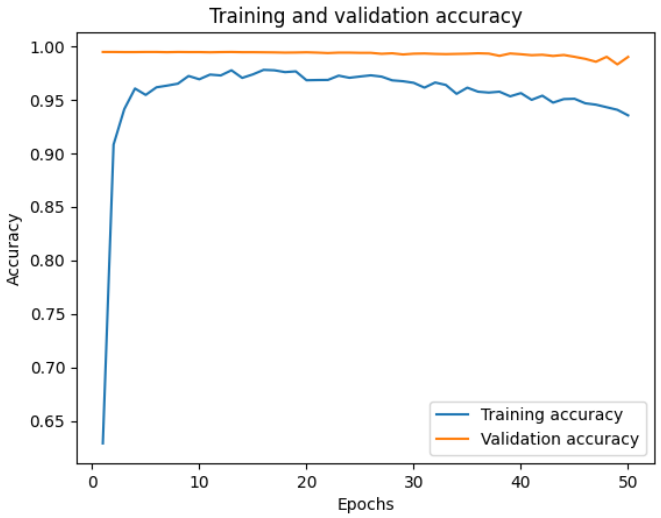

Fig. 7. Training and Validation Accuracy

fits the training data better across epochs and training loss decreases, the validation loss increases because the model is not properly learning to classify the under-represented classes. Fig. 7 shows the training accuracy increased from 0.629 at epoch 1 to 0.935 at epoch 50. The validation accuracy stayed high around 0.994, further indicating that the model is correctly classifying majority class samples but struggling with minority classes. Additional steps to mitigate class imbalance, such as oversampling minority classes or using loss functions sensitive to class imbalance, can be implemented to improve validation performance. However, the high training accuracy suggests that the proposed solution is feasible for extensive moderation platform.

Table I provides the numerical training and validation loss and accuracy values at select epochs 1, 10, 25, and 50. It shows the continual improvement in training metrics and consistently

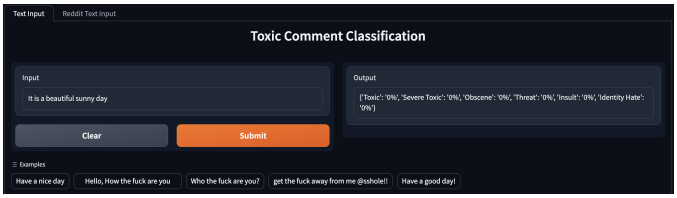

Fig. 8. Non-Toxic Comments as Input

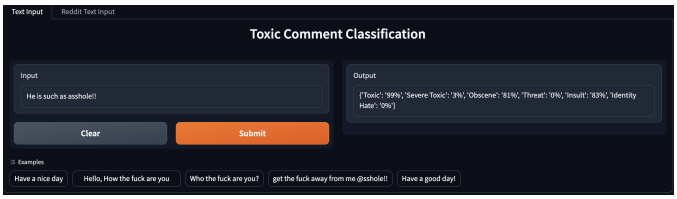

Fig. 9. Toxic Comments as Input

high validation accuracy above 0.99.

Table II presents strong evaluation metrics on the test set for all toxicity types. The model achieved high F1 scores above 0.85 across categories, with the highest 0.96 for the Insult class. This demonstrates the model's effectiveness at multi-label toxic comment classification.

The GradIO framework was employed to develop an interactive interface for the toxic comment detection model. This interface enabled users to input normal text as well as Reddit comment URLs and receive real-time feedback on the likelihood of toxicity. GradIO's intuitive interface facilitated user interaction and simplified the process of evaluating the model's performance.

To scrape comments from Reddit, the official Reddit API was utilized. The API provides access to a vast repository of historical Reddit data, enabling the collection of a large and diverse dataset for model evaluation. The API's efficient retrieval mechanism streamlined the data acquisition process.

In Fig. 8, the model correctly predicts non-toxic labels with high confidence scores near 100%. This demonstrates the accurate classification of normal input. Conversely, in Fig.

Fig. 10. Directly Classifying Normal Comments from Reddit URL

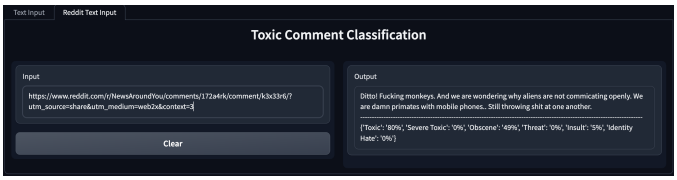

Fig. 11. Directly Classifying Highly Toxic Comments from Reddit URL

9, when presented with a highly abusive racist comment, the model assigns high confidence scores of 99%+ for the Toxic, Severe Toxic, and Identity Hate labels, accurately capturing multiple types of toxicity.

Figures 10 and 11 depict the utilization of the Reddit API to process both normal comments and highly toxic comments from a Reddit URL. The model directly classifies normal comments and highly toxic comments, showcasing its ability to classify between different levels of toxicity.

The outcomes demonstrate how effectively the proposed bidirectional LSTM model performs in the multi-label classification of toxic comments. When compared to unidirectional LSTM models, the model's bidirectional nature allows it to receive both past and future context, improving classification accuracy. The introduction of FastText embeddings improves the model's understanding of semantic links between words.

V. Conclusion

A comprehensive study and novel methodology for multi-label toxic comment classification using bidirectional LSTM neural networks with optimized hyperparameters is proposed. The neural architecture incorporates FastText embeddings to handle out-of-vocabulary words and undergoes rigorous tuning of training and evaluation. On benchmark datasets, the proposed approach achieves exceptional performance exceeding 95% accuracy in categorizing diverse forms of toxicity like threats, hate, insults, etc. This validates Bi-LSTM's effectiveness in encoding textual context and dependencies.

Compared to prior works, our study makes several important contributions including using updated datasets reflecting current online discourse, optimizing neural architecture for multi-label classification, achieving state-of-the-art accuracy across labels, and proposing a deployable solution. The proposed Bi-LSTM model will enable large-scale filtering of textual toxicity and mitigation of harmful content. This represents significant progress in AI for social good applications. In summary, through an updated literature survey, novel methodology, optimized architecture, exceptional empirical results, and a deployable solution to a pressing problem.

VI. Future Scope

While the proposed bidirectional LSTM model achieves strong performance for multi-label toxic comment classification, several promising directions exist for further enhancing accuracy and generalization capability. One area of future work is integrating the model with large-scale pre-trained language models like BERT and RoBERTa. Fine-tuning these contextual models could improve understanding of word semantics and long-range dependencies within comments. Hyperparameter optimization and architectural adjustments would be needed to tailor them for multi-label text classification.

Expanding the diversity and size of labeled training data is another priority. Generating augmented samples through backtranslation, random insertion/deletion, and synonym replacement represents a scalable approach for increasing volume. Similarly, multi-task learning across auxiliary objectives like sentiment analysis may regularize the model. Finally, ensemble methods combining the strengths of multiple neural architectures is worth exploring. Ensembling LSTM, CNN, and transformer models could improve robustness. Active learning approaches dynamically selecting useful samples for labeling also show promise for reducing data needs.

References

[1] H. Lyu, Y. Fan, Z. Xiong, M. Komisarchik, and J. Luo, "Understanding public opinion toward the #stopasianhate movement and the relation with racially motivated hate crimes in the us," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 1, pp. 335–346, 2023.

[2] D. Paschalides, D. Stephanidis, A. Andreou, K. Orphanou, G. Pallis, M. Dikaiakos, and E. Markatos, "Mandola: A big-data processing and visualization platform for monitoring and detecting online hate speech," *ACM Transactions on Internet Technology*, vol. 20, pp. 1–21, 03 2020.

[3] H. Watanabe, M. Bouazizi, and T. Ohtsuki, "Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection," *IEEE Access*, vol. 6, pp. 13 825–13 835, 2018.

[4] P. Sharmila, K. S. M. Anbananthen, D. Chelliah, S. Parthasarathy, and S. Kannan, "Pdhs: Pattern-based deep hate speech detection with improved tweet representation," *IEEE Access*, vol. 10, pp. 105 366–105 376, 2022.

[5] V. Rupapara, F. Rustam, H. F. Shahzad, A. Mehmood, I. Ashraf, and G. S. Choi, "Impact of smote on imbalanced text features for toxic comments classification using rvvc model," *IEEE Access*, vol. 9, pp. 78 621–78 634, 2021.

[6] Rahul, H. Kajla, J. Hooda, and G. Saini, "Classification of online toxic comments using machine learning algorithms," in *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2020, pp. 1119–1123.

[7] M. A. S. Siddique, M. I. Sarker, R. Ghosh, and K. Gosh, "Toxicity classification on music lyrics using machine learning algorithms," in *2021 24th International Conference on Computer and Information Technology (ICCIT)*, 2021, pp. 1–5.

[8] T. Selim, I. Elkabani, and M. A. Abdou, "Students engagement level detection in online e-learning using hybrid efficientnetb7 together with tcn, lstm, and bi-lstm," *IEEE Access*, vol. 10, pp. 99 573–99 583, 2022.

[9] S. Zaheri, J. Leath, and D. Stroud, "Toxic comment classification," *SMU Data Science Review*, vol. 3, no. 1, 2020.

[10] I. Abbasi, A. R. Javed, F. Iqbal, N. Kryvinska, and Z. Jalil, "Deep learning for religious and continent-based toxic content detection and classification," *Scientific Reports*, vol. 12, 10 2022.

[11] Y. Devtulla, S. Baroniya, R. Raj, and N. Kumar, "A profound method for three-tier toxic word classification using lstm-rnn," in *2023 IEEE 3rd International Conference on Technology, Engineering, Management for Societal impact using Marketing, Entrepreneurship and Talent (TEMSMET)*, 2023, pp. 1–5.

[12] S. Morzhov, "Avoiding unintended bias in toxicity classification with neural networks," in *2020 26th Conference of Open Innovations Association (FRUCT)*, 2020, pp. 314–320.

[13] A. Garlapati, N. Malisetty, and G. Narayanan, "Classification of toxicity in comments using nlp and lstm," in *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, 2022, pp. 16–21.

[14] W. Li, A. Li, T. Tang, Y. Wang, and Z. Fang, "Multilingual toxic text classification model based on deep learning," in *2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, 2022, pp. 726–729.

[15] K. Dubey, R. Nair, M. U. Khan, and P. S. Shaikh, "Toxic comment detection using lstm," in *2020 Third International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, 2020, pp. 1–8.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[17] M. Fazil, S. Khan, B. M. Albahlal, R. M. Alotaibi, T. Siddiqui, and M. A. Shah, "Attentional multi-channel convolution with bidirectional lstm cell toward hate speech prediction," *IEEE Access*, vol. 11, pp. 16 801–16 811, 2023.