https://doi.org/10.62836/jcmea.v2i1.0005

Global Science Publishing

JCMEA

Journal of Computational Methods in Engineering Applications

https://ojs.sgsci.org/journals/jcmea

Article

An Integrated Model for Social Media Toxic Comments Detection: Fusion of High-Dimensional Neural Network Representations and Multiple Traditional Machine Learning Algorithms

Jiahuai Ma [1], Kaixian Xu [2], Yu Qiao [3] and Zhaoyan Zhang [4,*]

[1] *Department of Computer & Information Science, University of Florida, Gainesville, FL 32608, USA; maj2@ufl.edu*

[2] *Risk & Quant Analytics, BlackRock, Princeton, NJ 10001, USA; kx374@nyu.edu*

[3] *Khoury College of Computer Sciences, Northeastern University, Seattle, WA 98109, USA; qiao.yu1@northeastern.edu*

[4] *Beijing Kwai Technology Co, Ltd., Beijing 100085, China*

Abstract: Social media platforms have become pivotal for global communication and information exchange but are increasingly challenged by the proliferation of toxic comments. These comments, characterized by abusive, discriminatory, or harassing language, threaten user safety and well-being, necessitating efficient detection systems. This paper proposes a novel hybrid approach to detect social media toxic comments by combining the feature extraction capabilities of Long Short Term Memoery (LSTM) -based neural networks with multiple machine learning models, including Random Forest, Logistic Regression, and K-Nearest Neighbors. High-dimensional feature representations from the neural network are integrated with predictions from traditional classifiers, and Random Forest optimizes the output weights to maximize performance. Evaluated on a Kaggle dataset, the proposed model achieves an accuracy of 89.78% and outperforms individual models in handling the complexity of toxic comments. However, challenges such as overfitting, computational overhead, and interpretability remain. Future work aims to address these limitations through improved data augmentation, explainability methods, and more scalable architectures.

Keywords: component; multi-model fusion; social media toxic comments detection; machine learning

1. Introduction

Social media platforms have become essential for communication, social interaction, and information exchange across the globe [1–3]. However, with the increasing volume of user-generated content, the spread of harmful and offensive speech, such as toxic comments, has also risen significantly. Toxic comments [4,5], which can be abusive, threatening, discriminatory, or harassing in nature, pose serious challenges to online communities and platforms. Detecting and mitigating such toxic content is essential to create a safe and positive online environment for users.

Received: 29 September 2022; Accepted:28 October 2022.

* Corresponding: Zhaoyan Zhang (zhaoyanzhang9394@gmail.com)

The need for effective social media toxic comments detection has become more apparent as online platforms, such as Twitter, Facebook, and YouTube, continue to grow. Toxic content can have detrimental effects on individuals, communities, and the broader social media ecosystem. For instance, it can lead to emotional distress, bullying, or even encourage violence in extreme cases. Therefore, detecting these comments efficiently is crucial for enforcing platform policies, maintaining user well-being, and promoting healthy discourse online. Automating the process of identifying toxic comments allows platforms to scale their moderation efforts, which would be impossible to achieve manually due to the vast amount of content uploaded daily. Effective detection systems can help flag offensive content for review or immediate removal, minimizing the harmful impact of such comments.

Traditional approaches to toxic comment detection, such as rule-based or keyword-based filtering [6–8], have shown limited success. While these methods can catch certain patterns, they often lack the flexibility needed to handle the complexity and diversity of natural language. For instance, a rule-based system might struggle to identify sarcasm, contextual nuances, or hidden toxic meanings in a comment. Furthermore, they typically require constant manual updates and can lead to over-blocking or under-blocking content, which hampers their accuracy. Computer science models, particularly those based on handcrafted features, have been proposed as an improvement over these rule-based systems [9–12]. For instance, Feng et al. proposed a hybrid framework for effective malware detection based on advanced computation methods [13]. These models rely on statistical features such as word frequencies, part-of-speech tags, and sentiment scores to classify content [14–16]. While these methods offer better flexibility, they still fall short when dealing with complex linguistic patterns, long-range dependencies, and context-specific toxicity in social media comments.

With the advancement of artificial intelligence (AI), particularly deep learning, the landscape of toxic comment detection has changed dramatically. AI models [17–19], especially those leveraging natural language processing (NLP), have been shown to outperform traditional approaches due to their ability to automatically learn from vast amounts of data and extract complex features. Neural network models, particularly Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs), and Transformer-based models, have been successful in capturing the context and semantic meaning of words and phrases in text, which is vital for detecting nuanced toxic comments. Moreover, pre-trained models like Bidirectional Encoder Representations from Transformers (BERT) and Generative Pre-trained Transformer (GPT) have demonstrated exceptional performance in understanding and processing large-scale textual data. These models capture both local and global dependencies within the text, making them well-suited for tasks like sentiment analysis and toxic comment detection. Despite their success, these models often struggle with overfitting when trained on small datasets or when deployed in isolated systems that do not leverage external knowledge [20].

While AI models, particularly neural networks and traditional machine learning models, have been widely used for toxic comment detection, a major limitation of most existing approaches is their reliance on a single model. Single models, whether they are neural networks or machine learning classifiers like Random Forests [21–23], Support Vector Machines (SVMs) [24–26], or Logistic Regression [27,28], often fail to exploit the full potential of multiple algorithms. Each model has its strengths and weaknesses, which means a single approach might excel in some aspects (e.g, capturing non-linear relationships in text) while falling short in others (e.g., generalization across diverse toxic comments). Many studies have focused on the application of a single model, such as training deep learning models or machine learning classifiers independently. However, these models often fail to generalize well to all types of toxic comments, leading to reduced performance in real-world applications. In practice, the diversity and complexity of toxic comments demand a hybrid approach that can leverage the strengths of various algorithms.

This paper proposes a novel integrated model for social media toxic comment detection shown in Figure 1, which combines high-dimensional feature representations learned by neural networks with the power of multiple traditional machine learning algorithms. The proposed approach uses LSTM-based neural networks [29,30] to generate rich, high-level feature representations of text data. These representations are then combined with predictions from Random Forest, K-Nearest Neighbors (KNN) [31,32], and Logistic Regression, allowing the model to learn a more robust and comprehensive decision function.

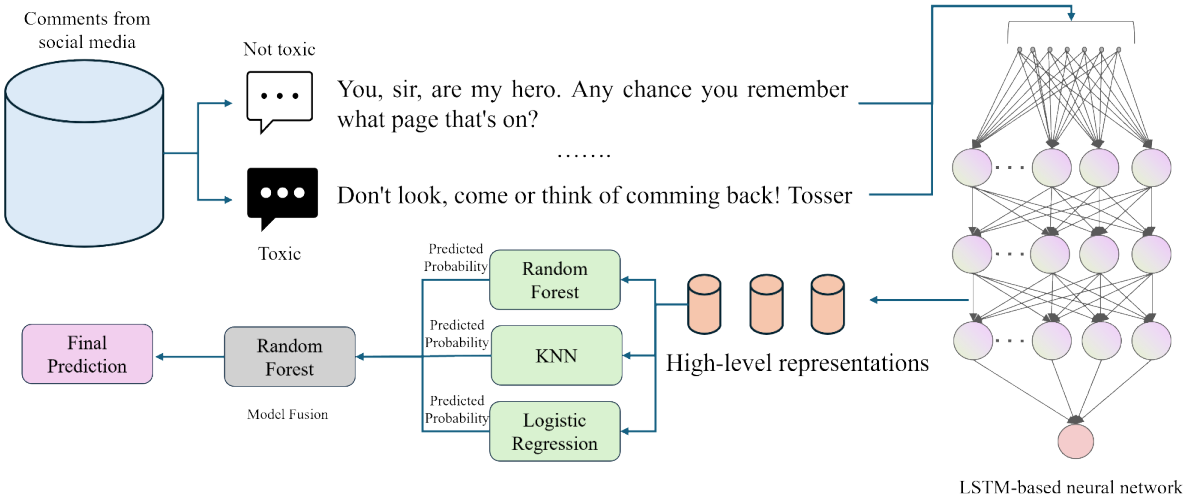

Figure 1. The workflow of the proposed model.

2. Literature Review

Toxic Comments Prediction

Aggression expressed through text is a multifaceted issue, which has attracted attention from various fields of study. The use of information technology to automatically identify aggressive language in texts has gained considerable interest. In this analysis, several types of aggression identified in existing literature are explored, such as hate [33], abusive language [34], toxicity [35], flaming [36], and hate speech [37]. Despite the variations among these forms of aggression, prior research provides valuable insights into strategies for identifying hostile interactions. Special emphasis is placed on the automatic identification of hate speech. For instance, Georgakopoulos et al. offer a concise, well-structured, and critical summary of the progress in automatic hate speech detection within natural language processing [37].

Various studies, especially those focusing on feature processing, have been conducted to improve the detection of toxic comments. Aggarwal and Zhai [38] explored how different transformations impact text classification by examining four distinct transformations and their combinations within the news and email domains. Their experimental results revealed that selecting the right combination of transformations could significantly enhance classification accuracy. Nobata and Tetreault [34] employed techniques such as normalizing numbers, replacing extremely long or unfamiliar words, and consolidating repeated punctuation into a single token. Haddadi et al. [39] discussed the importance of transformations in sentiment analysis, showing, through experiments with Support Vector Machines (SVM) on movie review data, that appropriate transformations and feature selection can lead to considerable improvements in accuracy. Their transformation methods included removing unnecessary whitespace, expanding abbreviations, stemming, eliminating stop words, and handling negations. On the other hand, other research tends to emphasize modeling approaches rather than focusing solely on transformations.

3. Method

3.1. Dataset Preparation

We received the dataset from Kaggle, which is focused on detecting toxic comments. The dataset includes several features, such as id, comment_text, toxic, severe_toxic, obscene, threat, insult, and identity_hate. The main goal of our analysis was to predict the toxicity of comments based on the text. We selected comment_text as the feature (X) and toxic as the target (y) for our prediction task. The original dataset contained 158,929 samples, which is a large number for analysis. Due to its size and the inherent challenges associated with handling such a large dataset, we selected a subset of 10,000 samples for our analysis. These samples were then split into training, testing, and validation sets. Specifically, 60% of the data was used for training, 30% for testing, and 10% for validation.

Since the data consists of textual information, we applied text vectorization using Term Frequency-Inverse

Document Frequency (TF-IDF) [40 – 42]. This technique converts the raw text data into numerical vectors, making it suitable for machine learning models. We limited the number of features to 10 through the vectorizer, which helps reduce the dimensionality of the data while preserving important information.

Additionally, the dataset exhibits a class imbalance issue shown in Figure 2, where the number of non-toxic comments greatly exceeds the number of toxic ones. This imbalance could lead to biased predictions, where the model might lean towards predicting "non-toxic" more often. To address this issue, we used Synthetic Minority Over-sampling Technique (SMOTE) [43, 44] to get the balanced data shown in Figure 3, which is a popular method for balancing datasets. SMOTE works by creating synthetic samples from the minority class, which helps improve the model's ability to generalize across both classes. This method significantly enhances the performance of models when dealing with imbalanced datasets, as it ensures the minority class is better represented during training.

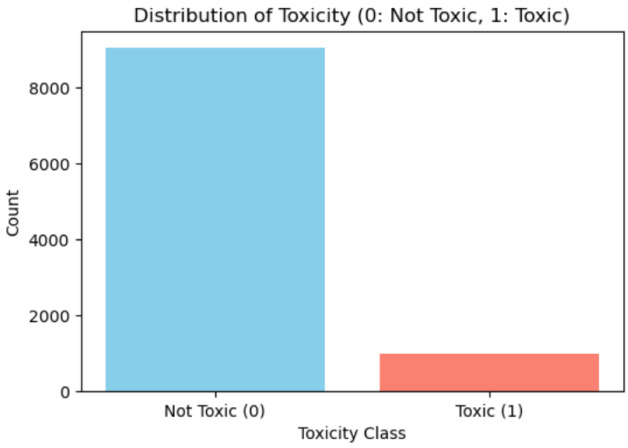

Figure 2. The original data distribution of the label.

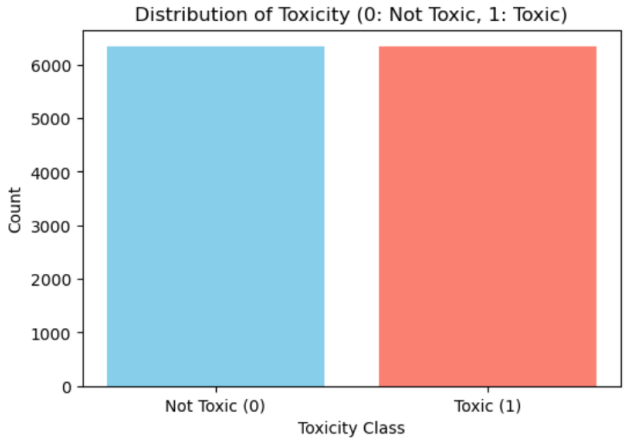

Figure 3. The balanced data distribution of the label processed by SMOTE.

3.2. The Introduction of Used Machine Learning Models

3.2.1. LSTM Model

LSTM [45, 46] is a type of RNN designed to better capture long-term dependencies in sequential data. Unlike traditional RNNs, LSTMs are equipped with a memory cell that allows them to maintain information over long periods of time. This makes them particularly effective for tasks involving sequential data, such as text, speech, and time series forecasting. The architecture of an LSTM is designed to overcome the vanishing gradient problem that traditional RNNs often face, allowing it to learn long-range dependencies.

In our model, we have built a LSTM-based architecture for binary classification. Here's a brief overview of its structure: (1) Embedding Layer: This is the first layer, which maps input sequences (such as words or tokens in text) into continuous vector representations of fixed size. The input data is transformed into embeddings,

which can then be used for further processing in the subsequent layers. (2) Bidirectional LSTM Layer: The next layer is a bidirectional LSTM, which processes the input sequence in both forward and backward directions. This helps the model capture information from both past and future states within the sequence, making it more powerful for sequential tasks. The LSTM units here have 16 memory units, and the activation function used is 'tanh', which is standard for LSTM networks. (3) Fully Connected Layer: After the LSTM layers, a fully connected layer with 8 neurons and ReLU (Rectified Linear Unit) activation is added. This layer helps the model learn more complex relationships between the features extracted by the LSTM layer. (4) Output Layer: The final layer is a single neuron with a sigmoid activation function, suitable for binary classification. It outputs a value between 0 and 1, which represents the probability of the input sequence belonging to the positive class (toxic comments in this case).

3.2.2. Random Forest

Random Forest [47] is an ensemble learning algorithm that builds a collection of decision trees to improve predictive accuracy and robustness. It is based on the principle of bagging, which involves training multiple individual models on different subsets of the data and then combining their predictions to get a more reliable result. Each decision tree in a Random Forest is trained on a random subset of the data, generated by sampling with replacement, known as bootstrap sampling. Additionally, when making splits at each node, Random Forest selects a random subset of features, rather than considering all features, which adds diversity to the trees and further helps reduce overfitting. Once the trees are built, their predictions are aggregated: for classification tasks, the majority vote is taken from all the trees, and for regression tasks, the predictions are averaged. This approach makes Random Forest highly robust, as it mitigates the risk of overfitting, making it a powerful tool for both classification and regression tasks, especially in complex and noisy datasets.

3.2.3. Logistic Regression

Logistic Regression [48] is a statistical model commonly used for binary classification tasks, where the goal is to predict one of two possible outcomes. Despite the name "regression," it is primarily used for classification problems. The model works by estimating the probability that a given input belongs to a particular class using a logistic (sigmoid) function. This function maps the input features to a value between 0 and 1, representing the probability of the positive class. Logistic Regression is based on a linear relationship between the input features and the log-odds of the target class, making it computationally efficient and easy to interpret. It is widely used in various applications such as medical diagnoses, marketing, and financial risk prediction, where the outcome is a binary decision, such as "yes" or "no".

3.2.4. K-Nearest Neighbors

K-Nearest Neighbors (KNN) [49] is a simple, non-parametric algorithm used for both classification and regression tasks. The idea behind KNN is straightforward: when making a prediction for a new data point, the algorithm finds the 'K' training samples that are closest to the point in question, using a distance metric like Euclidean distance. The class label (for classification) or the value (for regression) is then determined based on the majority class or average value of these nearest neighbors. KNN is particularly useful when the decision boundary between classes is irregular and complex, as it makes predictions based on local patterns in the data rather than assuming a global structure. However, KNN can be computationally expensive, especially with large datasets, as it requires calculating the distance between the new point and all points in the training set. Despite this, it remains a popular choice due to its simplicity and effectiveness, particularly in smaller datasets or when the data has clear local structures.

3.3. Multi-Model Fusion Strategy for Toxic Comment Prediction

In our research, we proposed a method that first uses a LSTM neural network to generate high-level representations from the data, and then feeds these representations into multiple traditional machine learning

models for further training and prediction. Specifically, the process is as follows: (1) First, we trained a neural network using the raw data. The neural network learns complex features from the data and generates high-level representations, which capture intricate patterns and relationships that may not be easily identified by simpler models. (2) Next, we extracted the high-level representations from the trained neural network and used these representations as new features. These new features were then fed into multiple traditional machine learning models—including Random Forest, KNN, and Logistic Regression—to train them and generate their respective predictions. (3) Finally, we fused the outputs from these machine learning models to make a final prediction. By combining the results from different models, we aimed to benefit from the strengths of each model and improve the overall performance, resulting in a more robust and accurate prediction.

4. Results and Discussion

4.1. The Performance of Different Machine Learning Models

Figure 4 shows the training curves for an LSTM model, detailing both accuracy and loss over the training epochs. On the left, the accuracy graph presents the training accuracy (solid blue line) and validation accuracy (dashed orange line) across 100 epochs. The training accuracy begins at around 55% and quickly stabilizes to just above 65%, showing minimal variance thereafter. The validation accuracy, after an initial spike, also stabilizes but fluctuates slightly more than the training accuracy, with values oscillating around 85%. The discrepancy between the training accuracy and validation accuracy is primarily due to the differences in data distribution caused by the balancing approach used during training. In the training set, the imbalance in class distribution was addressed by duplicating samples from the minority class to achieve a more balanced representation. However, this balancing method artificially altered the training data and did not reflect the true distribution of the dataset. In contrast, the validation and test sets were left unchanged to preserve their original class distributions, ensuring that the model's performance could be evaluated under real-world conditions. As a result, the validation set's accuracy is higher because it closely aligns with the ultimate goal of achieving optimal performance on unbalanced, real-world data. On the right, the loss graph depicts the training loss (solid green line) and validation loss (dashed red line) over the same epoch range. The training loss shows a steep decrease in the initial epochs and then levels off at around 0.66, indicating a plateau where further training shows little improvement in reducing loss. The validation loss, on the other hand, demonstrates significant variance, with sharp peaks and troughs, generally hovering around 0.67 to 0.71. This could suggest that the model is experiencing some overfitting to the training data, as evidenced by the higher and more volatile validation loss compared to the more stable and lower training loss.

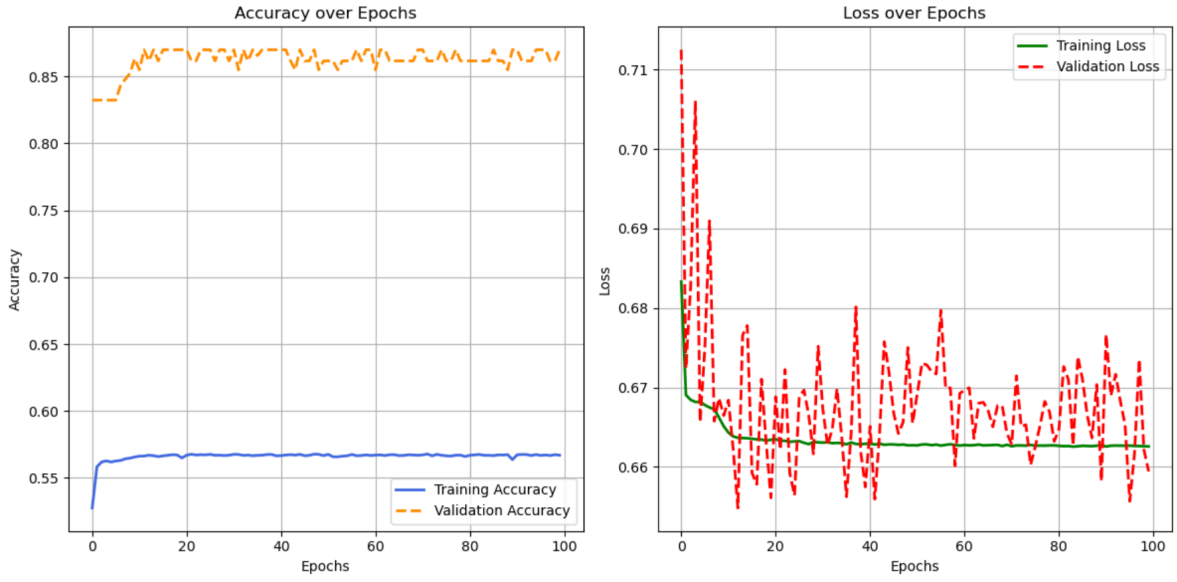

Figure 4. The training curve of the LSTM model.

The confusion matrices in Figure 5 provide insight into the classification performance of the various models and combinations. The standalone LSTM performs reasonably well in identifying non-toxic comments but struggles with toxic comments, leading to a noticeable number of false negatives. When features are passed to other machine learning models, improvements are observed. For instance, combining LSTM with Random Forest (LSTM + RF) slightly improves the detection of toxic comments, as fewer false negatives are observed. Similarly, the LSTM + LR model further enhances the balance between true positives and false negatives, indicating better handling of toxic samples. However, the LSTM + KNN combination shows weaker performance, particularly in identifying toxic comments, as reflected in the higher number of misclassifications.

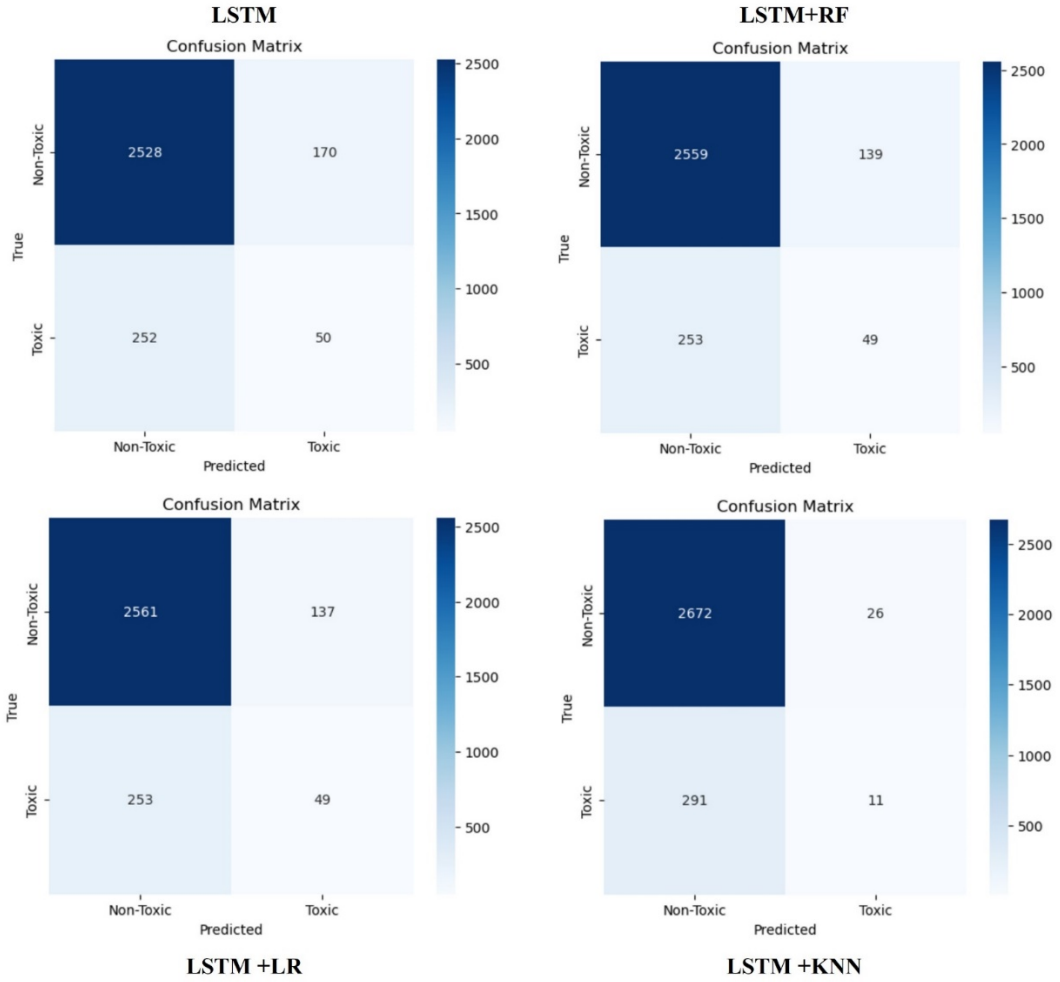

Figure 5. The confusion matrix of different models.

Table 1 and Figure 6 further quantify the performance across several metrics. The recall of the fusion model is comparable to other combinations like LSTM + RF and LSTM + LR, indicating consistent identification of toxic comments. Importantly, the fusion model attains the highest accuracy of 0.8978, significantly outperforming individual models and partial fusions. This demonstrates its robustness and effectiveness in handling the complexities of social media comment classification. While the F1-score remains modest at 0.2070, it is the highest among all models, highlighting the fusion model's ability to balance precision and recall effectively.

Overall, these results underscore the potential of the proposed approach in the domain of toxic comment detection. The integration of LSTM for feature extraction and the fusion of traditional machine learning models allows the system to capture the nuanced characteristics of social media comments. The final optimization step using Random Forest ensures that the model benefits from the complementary strengths of individual classifiers, leading to superior overall performance. This approach provides a promising direction for tackling the challenges of toxic comment detection in social media, where the balance between precision, recall, and

accuracy is critical for real-world applications.

Table 1. The performance of different models in the testing dataset evaluated by various metrics.

Model Name	Precision	Recall	Accuracy	F1-Score
LSTM	0.2273	0.1656	0.8593	0.1916
LSTM + RF	0.2606	0.1623	0.8693	0.2000
LSTM + LR	0.2634	0.1623	0.8700	0.2008
LSTM + KNN	**0.2973**	0.0364	0.8943	0.0649
LSTM + RF + LR + KNN	0.2677	**0.1688**	**0.8978**	**0.2070**

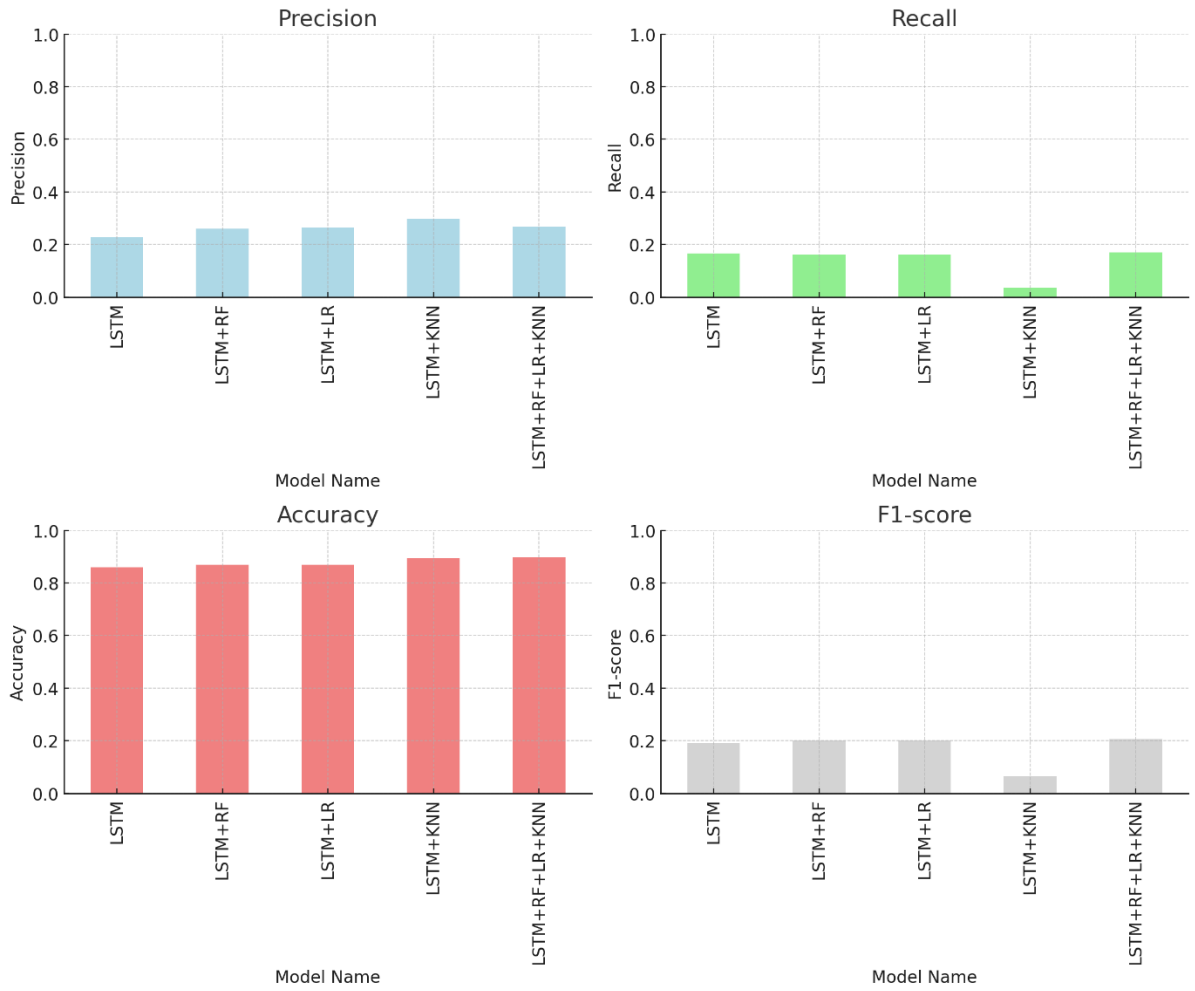

Figure 6. The performance of different models in the testing dataset.

4.2. The Influence of n_estimators on the Model Performance

Figure 7 illustrates the effect of the number of estimators (n_estimators) in the Random Forest on the test accuracy of the proposed fusion model for social media toxic comments detection. Since Random Forest is responsible for determining the optimal weights for combining outputs from multiple machine learning models, the parameter n_estimators, which represents the number of trees in the forest, plays a critical role in influencing the final performance. As shown in the figure, the test accuracy exhibits a sharp increase as n_estimators grows from 10 to 200, indicating that a higher number of trees significantly enhances the model's ability to generalize. The accuracy peaks at around 0.90 when n_estimators reaches 200, suggesting that this is the optimal setting for the fusion model in the given task. Beyond this point, as the number of estimators increases to 300 and 500, the test accuracy stabilizes, showing minimal variation and maintaining a high level of performance. This

stabilization suggests that adding more trees beyond a certain point does not provide significant additional benefits and could lead to diminishing returns in terms of computational efficiency.

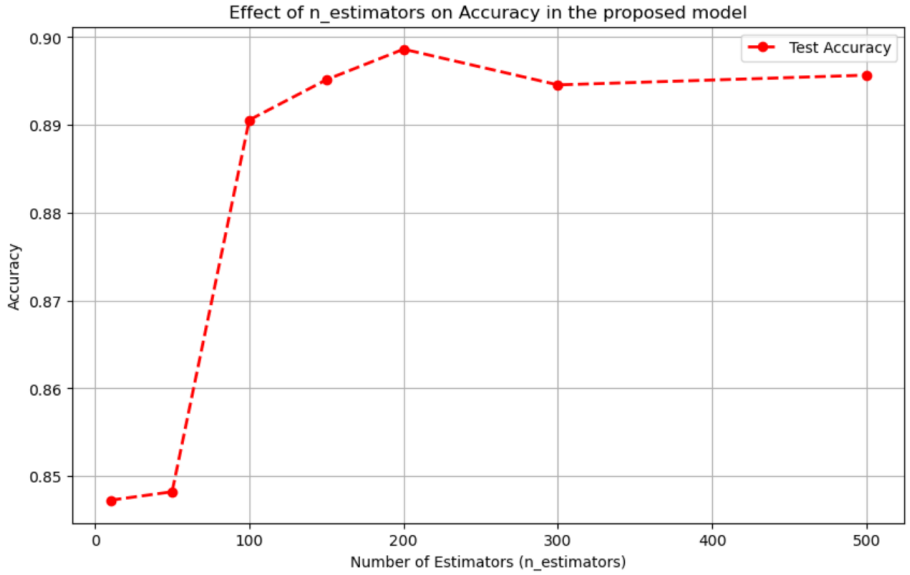

Figure 7. The influence of n_estimator on performance of the proposed model.

4.3. Discussion

While the proposed fusion model combining high-dimensional neural network representations with multiple traditional machine learning algorithms has demonstrated promising results in social media toxic comments detection**,** there are still notable limitations that warrant discussion. (1) Overfitting Risks**:** The integration of multiple models and the use of Random Forest for weighting may inadvertently introduce overfitting to the training data, particularly when the feature set from the LSTM is high-dimensional. This could limit the generalizability of the model when applied to new datasets. (2) Complexity and Interpretability**:** The fusion process, which involves multiple layers of model integration and optimization, adds significant complexity to the system. While Random Forest provides some interpretability by outputting feature importances, the overall model remains challenging to interpret, especially in high-stakes applications where transparency is critical. (3) Computational Costs: Training the LSTM for feature extraction and subsequently combining predictions from multiple machine learning models increases computational overhead. This may limit the model's scalability and deployment in resource-constrained environments.

5. Conclusion

Toxic comments on social media can lead to emotional distress and disrupt online discourse, making their detection a pressing issue. Traditional methods like keyword-based filtering lack the sophistication to handle the nuanced nature of toxic language, such as sarcasm and contextual toxicity. While machine learning approaches improve flexibility, they still struggle with long-range dependencies and complex linguistic patterns. Deep learning models, particularly LSTMs and Transformer-based architectures, have revolutionized toxic comment detection by leveraging their ability to capture semantic and contextual relationships within text. Despite their success, reliance on single models often leads to reduced generalizability and suboptimal performance. To overcome these challenges, this study introduces a hybrid fusion model that combines neural network-extracted features with the strengths of Random Forest, Logistic Regression, and KNN. By optimizing the weighting of individual model outputs with Random Forest, the proposed system achieves higher accuracy and a better balance between precision and recall. However, the integration of multiple models introduces computational complexity and risks of overfitting. Future research will explore advanced explainability techniques, lightweight architectures, and data augmentation strategies to enhance robustness and scalability, paving the way for more effective toxic comment detection systems in diverse online environments.

Funding

Not applicable.

Author Contributions

Conceptualization, J.M.; writing—original draft preparation, J.M., K.X., Y.Q., and Z.Z. All of the authors read and agreed to the published the final manuscript.

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

Not applicable.

Conflicts of Interest

The authors declare no conflict of interest.

References

1 Weller K. Trying to Understand Social Media Users and Usage: The Forgotten Features of Social Media Platforms. *Online Information Review* 2016; **40**(**2**): 256–264.

2 Bucher T, Helmond A. The Affordances of Social Media Platforms. In *The SAGE handbook of Social Media*; Sage Publishing: New York, NY, USA, 2018; Volume 1, pp. 233–254.

3 Van Dijck, J, Poell, T. Social Media Platforms and Education. In *The SAGE Handbook of Social Media*; Sage Publishing: New York, NY, USA, 2018; pp. 579–591.

4 Hosseini H, Kannan S, Zhang B, *et al.* Deceiving Google's Perspective Api Built for Detecting Toxic Comments. *arXiv* 2017, arXiv:1702.08138.

5 Zaheri S, Leath, J, Stroud, D. Toxic Comment Classification. *SMU Data Science Review* 2020; **3**(**1**): 13.

6 Saeed HH, Ashraf MH, Kamiran F, *et al.* Roman Urdu Toxic Comment Classification. *Language Resources and Evaluation* 2021; **55**: 971–996.

7 Brassard-Gourdeau É, Khoury R. Impact of Sentiment Detection to Recognize Toxic and Subversive Online Comments. *arXiv* 2018, arXiv:1812.01704.

8 Gémes K, Recski G. TUW-Inf at GermEval 2021: Rule-Based and Hybrid Methods for Detecting Toxic, Engaging, and Fact-Claiming Comments. In Proceedings of the GermEval 2021 Shared Task on the Identification of Toxic, Engaging, and Fact-Claiming Comments, Duesseldorf, Germany, September 2021; pp. 69–75.

9 Dai W. Safety Evaluation of Traffic System with Historical Data Based on Markov Process and Deep-Reinforcement Learning. *Journal of Computational Methods in Engineering Applications* 2021; **1**: 1–14.

10 Yu L, Li J, Cheng S, *et al.* Secure Continuous Aggregation in Wireless Sensor Networks. *IEEE Transactions on Parallel and Distributed Systems* 2013; **25**(**3**): 762–774.

11 Xiong S, Yu L, Shen H, *et al.* Efficient Algorithms for Sensor Deployment and Routing in Sensor Networks For Network-Structured Environment Monitoring. In Proceedings of the 2012 IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 1008–1016.

12 Zhu D, Gan Y, Chen X. Domain Adaptation-Based Machine Learning Framework for Customer Churn Prediction Across Varing Distributions. *Journal of Computational Methods in Engineering Applications* 2021; **1**: 1–14.

13 Feng Z, Xiong S, Cao D, *et al.* Hrs: A Hybrid Framework for Malware Detection. In Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics, San Antonio, TX, USA, 4 March 2015; pp. 19–26.

14 Xiong S, Li J, Li M, *et al.* Multiple Task Scheduling for Low-Duty-Cycled Wireless Sensor Networks. In Proceedings of the 2011 IEEE INFOCOM, Shanghai, China, 10–15 April 2011; pp. 1323–1331.

15 Yu L, Li J, Cheng S, *et al.* Secure Continuous Aggregation via Sampling-Based Verification in Wireless Sensor Networks. In Proceedings of the 2011 IEEE INFOCOM, Shanghai, China, 10–15 April 2011; pp. 1763–1771.

16 Li J, Xiong S. Efficient Pr-Skyline Query Processing and Optimization in Wireless Sensor Networks. *Wireless Sensor Network* 2010; **2**(**11**): 838.

17 Xiong S, Li J. An Efficient Algorithm for Cut Vertex Detection in Wireless Sensor Networks. In Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems, Genoa, Italy, 21–25 June 2010; pp. 368–377.

18 Xiong, S, Li, J. Optimizing Many-to-Many Data Aggregation in Wireless Sensor Networks. In *Advances in Data and Web Management, Proceedings of the Asia-Pacific Web Conference 2009, Suzhou, China, 2–4 April 2009*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 550–555.

19 Wang H, Li J, Xiong S. Efficient Join Algorithms for Distributed Information Integration Based on XML. *International Journal of Business Process Integration and Management* 2008; **3**(**4**): 271–281.

20 Zhou Z, Wu J, Cao Z, *et al.* On-Demand Trajectory Prediction Based on Adaptive Interaction Car Following Model with Decreasing Tolerance. In Proceedings of the 2021 International Conference on Computers and Automation (CompAuto), Paris, France, 7–9 September 2021; pp. 67–72.

21 Rigatti SJ. Random Forest. *Journal of Insurance Medicine* 2017; **47**(**1**): 31–39.

22 Biau G, Scornet E. A Random Forest Guided Tour. *Test* 2016; **25**: 197–227.

23 Breiman L. Random Forests. *Machine Learning* 2001; **45**: 5–32.

24 Wang H, Hu D. Comparison of SVM and LS-SVM for Regression. In Proceedings of the 2005 International Conference on Neural Networks and Brain, Beijing, China, 13–15 October 2005; Volume 1, pp. 279–283.

25 Vishwanathan SVM, Murty MN. SSVM: A Simple SVM Algorithm. In Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290), Honolulu, HI, USA, 12–17 May 2002; Volume 3, pp. 2393–2398.

26 Jakkula, V. Tutorial on Support Vector Machine (SVM). *School of EECS, Washington State University* 2006; **37**: 3.

27 LaValley MP. Logistic Regression. *Circulation* 2008; **117**(**18**): 2395–2399.

28 Hosmer DW, Jr., Lemeshow S, Sturdivant RX. *Applied Logistic Regression*; John Wiley & Sons: Hoboken, NJ, USA, 2013.

29 Yu Y, Si X, Hu C, *et al.* A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Computation* 2019; **31**(**7**): 1235–1270.

30 Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D: Nonlinear Phenomena* 2020; **404**: 132306.

31 Peterson LE. K-Nearest Neighbor. *Scholarpedia* 2009; **4**(**2**): 1883.

32 Imandoust SB, Bolandraftar M. Application of k-Nearest Neighbor (knn) Approach for Predicting Economic Events: Theoretical Background. *International Journal of Engineering Research and Applications* 2013; **3**(**5**): 605–610.

33 Tarasova Z, Khlinovskaya Rockhill E, Tuprina O, *et al.* Urbanisation and the Shifting Of Boundaries: Contemporary Transformations in Kinship and Child Circulation amongst the Sakha. *Europe-Asia Studies* 2017; **69**(**7**): 1106–1125.

34 Nobata C, Tetreault J, Thomas A, *et al.* Abusive Language Detection in Online User Content. In Proceedings of the 25th International Conference on World Wide Web, Montreal, QC, Canada, 11–15 April 2016; pp. 145–153.

35 Hanson R. *Foul Play in Information Markets*; George Mason University: Fairfax, VA, USA, 2004

36 Waseem Z, Thorne J, Bingel J. Bridging the Gaps: Multi Task Learning for Domain Transfer of Hate Speech Detection. In *Online Harassment*; Springer: Cham Switzerland, 2018; pp. 29–55.

37 Georgakopoulos SV, Tasoulis SK, Vrahatis AG, *et al.* Convolutional Neural Networks for toxic Comment Classification. In Proceedings of the 10th Hellenic Conference on Artificial Intelligence, Patras, Greece, 9–12 July 2018; pp. 1–6.

38 Aggarwal CC, Zhai C. A Survey of Text Classification Algorithms. In *Mining Text Data*; Springer, Boston, MA, USA, 2012; pp. 163–222.

39 Cha M, Haddadi H, Benevenuto F, *et al.* Measuring User Influence in Twitter: The million Follower Fallacy. In Proceedings of the International AAAI Conference on Web and Social Media, Washington, DC, USA, 23–26 May 2010; Volume 4, pp. 10–17.

40 Aizawa A. An Information-Theoretic Perspective of tf-idf Measures. *Information Processing & Management* 2003; **39**(**1**): 45–65.

41 Ramos J. Using tf-idf to Determine Word Relevance in Document Queries. In Proceedings of the First Instructional Conference on Machine Learning, Los Angeles, CA, USA, 23 – 24 June 2003; Volume 242, pp. 29–48.

42 Qaiser S, Ali R. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications* 2018; **181**(**1**): 25–29.

43 Junsomboon N, Phienthrakul T. Combining Over-Sampling and under-Sampling Techniques for Imbalance Dataset. In Proceedings of the 9th International Conference on Machine Learning and Computing, Singapore, 24–26 February 2017; pp. 243–247.

44 Karthik MG, Krishnan MM. Hybrid Random Forest and Synthetic Minority over Sampling Technique for Detecting Internet of Things Attacks. *Journal of Ambient Intelligence and Humanized Computing* 2021; 1–11.

45 Zhao J, Huang F, Lv J, *et al.* Do RNN and LSTM Have Long Memory?. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 11365–11375.

46 Zhao Z, Chen W, Wu X, *et al.* LSTM network: A Deep Learning Approach for Short-Term Traffic Forecast. *IET Intelligent Transport Systems* 2017; **11**(**2**): 68–75.

47 Cutler A, Cutler DR, Stevens JR. Random Forests. In *Ensemble Machine Learning: Methods and Applications*; Springer, New York, NY, USA, 2012; pp. 157–175.

48 Speelman D. Logistic Regression. *Corpus Methods for Semantics: Quantitative Studies in Polysemy and Synonymy* 2014; **43**: 487–533.

49 Batista GE, Silva DF. How k-Nearest Neighbor Parameters Affect Its Performance. In Proceedings of the Argentine Symposium on Artificial Intelligence, Mar del Plata, Argentina, 24–25 August 2009; pp. 1–12.