Toxic Language Detection in Social Media

1st XinZhou Li
Computer Science
University of Calgary
Calgary, Canada
xinzhou.li@ucalgary.ca

2nd BoZheng Ma
Computer Science
University of Calgary
Calgary, Canada
bozheng.ma1@ucalgary.ca

3rd Zonglin Zhang
Computer Science
University of Calgary
Calgary, Canada
zonglin.zhang@ucalgary.ca

***Abstract*—The exponential growth of social media has resulted in an increased amount of harmful and abusive language online, leading to negative impacts on individuals and society. To address this issue, this paper proposes a novel approach to detect the toxic language in social media using a Bidirectional Long Short-Term Memory (BiLSTM) model. The proposed model leverages the advantages of BiLSTM to capture contextual and sequential information in texts and uses pre-trained word embeddings to represent text features. The proposed approach can be utilized by social media platforms to automatically detect and filter toxic content, thereby promoting a safer and healthier online environment.**

I. Introduction

The rapid rise of social media platforms has brought about numerous benefits such as easy communication, social networking, and dissemination of information. However, along with its benefits, social media has become a breeding ground for harmful and abusive language. Toxic language refers to words or phrases that are intended to offend, intimidate, or harm an individual or a group of people. The prevalence of toxic language on social media has become a serious concern, as it can negatively impact individuals' mental health, self-esteem, and well-being. Furthermore, toxic language can lead to the spread of hate speech, online bullying, and discrimination, which can have devastating consequences for society as a whole.

To address this issue, various approaches have been proposed to detect the toxic language in social media. Traditional approaches rely on keyword-based filtering and rule-based techniques, which are often ineffective in detecting subtle and nuanced toxic language. Moreover, these approaches can also result in false positives, which can harm the user's freedom of expression. Hence, there is a need for a more sophisticated and accurate approach to detecting toxic language in social media.

In recent years, deep learning-based approaches, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have shown promising results in natural language processing tasks, including sentiment analysis and text classification. Among these, Bidirectional Long Short-Term Memory (BiLSTM) models have gained popularity in modelling sequential data, as they can capture both past and future contextual information of a text. BiLSTM models have been successfully applied in various natural languages processing tasks, such as machine translation, text summarization, and sentiment analysis.

II. Literature review

Keyword-based filtering and rule-based techniques are traditional approaches used to detect the toxic language in social media. These approaches rely on a pre-defined list of keywords or rules to flag potentially toxic content. However, these approaches have limitations in detecting subtle and nuanced toxic language and can result in false positives, which can harm the user's freedom of expression.

Machine learning-based approaches have gained popularity in detecting toxic language in social media. In these approaches, a model is trained on a labelled dataset to predict whether a given text is toxic or not. We have considered various machine learning algorithms, such as logistic regression, support vector machines, and neural networks, for this task. However, these approaches have limitations in capturing the contextual and sequential information in texts.

Support Vector Machines (SVMs) are a type of supervised machine learning model that separates data into different categories using a hyperplane. SVMs have been widely used for NLP tasks, including toxic language detection in social media due to their ability to handle high-dimensional data and nonlinear relationships between features.

Logistic Regression is a type of linear model that is used for binary classification tasks. Logistic regression models the probability of a binary outcome using a logistic function. Logistic regression has been used for toxic language detection in social media, but its performance may be limited in scenarios where the data has complex relationships between features.

Neural Networks are a type of deep learning model that can learn complex and abstract features from raw input data. Neural networks have shown promising results in various NLP tasks, including toxic language detection in social media. However, neural networks require a large amount of data to train and are computationally expensive compared to SVMs and logistic regression.

While SVMs, logistic regression, and neural networks can all be used for toxic language detection in social media, SVMs and logistic regression may be suitable for scenarios where the data have simple relationships between features and where the amount of data is limited. On the other hand, neural networks may be more suitable for scenarios where the data has complex relationships between features and where a large amount of data is available.

On the other hand, deep learning models can learn features directly from the raw input data, without the need for feature engineering. Deep learning models can capture complex and abstract features, such as the relationships between words and the context in which they are used. It typically requires a large amount of data to train and is computationally more expensive than machine learning models. However, deep learning models have shown better performance than machine learning models in various NLP tasks, including toxic language detection in social media.

For deep learning-based approaches, we have considered models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). They have shown promising results in detecting toxic language in social media. CNNs can capture local patterns in texts, whereas RNNs can capture the contextual information and dependencies between words.

CNNs are mainly used for image processing, but they can also be applied to text by treating each word as a pixel. The model slides a fixed-size window over the text and applies convolution operations to extract local features. While CNNs can capture local patterns in texts, they may not capture long-range dependencies between words, which are important for NLP tasks such as toxic language detection in social media.

RNNs, on the other hand, are designed to handle sequential data and can capture contextual information and dependencies between words. In RNNs, each input is processed based on the previous inputs, and the output of the network is influenced by the previous inputs. However, standard RNNs suffer from the vanishing gradient problem, which limits their ability to capture long-range dependencies.

In recent years, Bidirectional Long Short-Term Memory (BiLSTM) models have gained popularity in modeling sequential data, including natural language processing tasks. BiLSTM models can capture both past and future contextual information and dependencies between words. For example, Badjatiya et al. (2017) proposed a BiLSTM-based approach for detecting hate speech on Twitter, which outperformed existing approaches with an F1-score of 0.76.

In summary, various approaches have been proposed to detect toxic language in social media, including keyword-based filtering, rule-based techniques, machine learning-based approaches, and deep learning-based approaches. BiLSTM models have shown promising results in capturing the contextual and sequential information in texts, making them a suitable choice for detecting toxic language in social media.

III. Dataset

In order to train the Bidirectional Long Short-Term Memory (BI-LSTM) model, an initial step involves the collection and pre-processing of relevant data. A dataset comprising social media posts was procured from Kaggle, with each post classified into one of six categories based on content: toxic, severe toxic, obscene, threat, insult, and identity hate. Subsequently, two additional columns were generated from the original file, delineating the data as either toxic or non-toxic. The text data was pre-processed through a series of operations, including text cleaning, which entailed the use of the Pandas library to remove punctuation and standardize lowercase lettering. Data entries labeled as -1 in the toxic column were deemed non-toxic and allocated to the ”non-toxic” column, while those labeled as 0 were categorized as toxic. The remaining five classifications were excluded from further analysis. The new file will display toxic comments as 1, and non-toxic as 0.

IV. Architecture of Model

We employ a deep learning model encompassing several layers, including an embedding layer, a Bidirectional Long Short-Term Memory (BI-LSTM) layer, and a dense output layer. Initially, the embedding layer transforms input text data into numerical vectors. The BI-LSTM layer subsequently processes these sequences and learns the semantic context. Finally, the dense output layer relies on the BI-LSTM layer's output to predict the toxicity of the text, thus yielding the result. As for the attention mechanism, we are in the process of determining implementation specifics and assessing its necessity.

The BI-LSTM model architecture entails the following components:

Embedding Layer: Input text data sequences are converted into numerical vector sequences through pre-trained GloVe embedding, which supplants the initial random Embedding layer. The embedding layer then transforms these vectors into a dense representation of size (1, 300) before relaying it to the subsequent layer.

BI-LSTM Layer: This layer, a recurrent neural network tailored to capture textual context and meaning, accepts the embedding layer's output and processes it sequentially while updating its internal weight matrix at each stage. Comprising multiple cells, the BI-LSTM layer possesses weights and biases that are learned during training. These parameters are utilized to compute the new internal weight matrix and BI-LSTM cell output at every time step.

Dense Output Layer: The BI-LSTM layer's output is directed to a dense output layer, a fully connected neural network layer responsible for generating the ultimate prediction of text toxicity. Employing a sigmoid activation function, the dense output layer compresses the output between 0 and 1 (toxic and non-toxic) to produce a probability value for each input.

Model configurations, such as the number of layers, and input and output channels, dictate specific model properties. By adjusting the layers, hidden units, and additional hyperparameters, the model's performance in detecting toxic language can be optimized. However, discovering the ideal configuration necessitates experimentation and hyperparameter fine-tuning. Our implementation will involve a single BI-LSTM layer with 256 hidden units and a single dense output layer with one output channel. The model input comprises a sequence of 300-

dimensional GloVe vectors, with the output being a binary classification (toxic or non-toxic).

V. Training & Evaluation

In our training process, we employ two distinct optimizers. The first optimizer, Adaptive Moment Estimation (ADAM), is a popular optimization algorithm that efficiently adapts the learning rate for each parameter. ADAM combines the benefits of two gradient descent methodologies: Adaptive Gradient Algorithm (AdaGrad), which maintains a per-parameter learning rate, and Root Mean Square Propagation (RMSProp), which employs an adaptive learning rate based on recent magnitudes of the gradients. By quickly adjusting the weight matrix upon reaching over-fitting, ADAM enables a more robust convergence.

Following pre-processing, the model is trained on the labeled dataset, and its performance is evaluated using a variety of metrics, which are crucial for obtaining a comprehensive understanding of the model's effectiveness. These metrics include:

Accuracy: A measure of the proportion of correctly classified instances out of the total instances. While this metric is simple and easy to interpret, it may not provide an accurate representation of model performance when dealing with imbalanced datasets.

Precision: The proportion of true positive predictions out of all positive predictions. This metric is particularly important in scenarios where false positives have a significant impact, as it quantifies the model's ability to correctly identify toxic content while minimizing false alarms.

Recall: The proportion of true positive predictions out of all actual positive instances. This metric is crucial when the cost of false negatives is high, as it measures the model's ability to identify all toxic content without omitting any.

F1 score: The harmonic mean of precision and recall, which provides a balanced measurement of both metrics. The F1 score is particularly useful in cases of imbalanced datasets, as it ensures that both false positives and false negatives are taken into consideration when evaluating model performance.

VI. Result

TABLE I
Table 1 shows the accuracy of training and testing

Training accuracy	**Test accuracy**
0.9704122875293185	0.9744148173084926

Table 1.1 shows the accuracy of our training and the accuracy of our model for the test set, which shows that our model is not yet in the over-fitting stage and therefore its prediction accuracy is still very high.

TABLE II
Table 2 shows the confusion matrix

	Relevant	NonRelevant
Retrieved	**TP = 12133**	**FP = 3161**
Not Retrieved	FN = 926	TN = 143351

This matrix shows the details of the output of the test dataset. From the matrix, it is easy to see that our TN has 143351 texts, the reason that causes that is we treated all the other categories of text to be non-toxic. That may cause the model has some potential problem.

TABLE III
Table 3 Precision, Recall, F1-measurement

Precision	**Recall**	**F1-measure**
0.7933176409	0.9290910483	0.8558529647

The value of recall(0.92) is a little bit higher than precision(0.79), this unbalance of precision and recall causes the pretty low of the value of the F1-measure(0.85).

VII. Discussion

In this section, we discuss the performance of our proposed model for toxic language detection and compare it with the 3rd place solution in the Kaggle Jigsaw Toxic Comment Classification Challenge. Our model is based on a BiLSTM architecture, whereas the Kaggle team employed a combination of feature engineering approaches, algorithms, and embeddings, including logistic regression, GRUs, LSTMs, and stacking layers. The team achieved an impressive 98.10% accuracy on the public leaderboard and scored between 0.99270 and 0.99308 on their local cross-validation. Our model's performance was evaluated using precision, recall, F1-measure, training accuracy, and test accuracy.

A. Model Comparison

Our BiLSTM model achieved a precision of 0.7933, recall of 0.9291, F1-measure of 0.8559, training accuracy of 97.04%, and test accuracy of 97.44%. Although our model has a lower overall accuracy compared to the Kaggle team's solution, there are several notable differences that may provide insight into the pros and cons of each approach.

B. Pros of our Model

Simplicity: Our model relies on a single architecture, the BiLSTM, which requires less time and effort to implement and maintain compared to the complex ensemble of models employed by the Kaggle team.

Interpretability: With a single architecture, it is easier to understand the decision-making process and identify areas for improvement. In contrast, ensemble methods with multiple layers of stacking can become less transparent and challenging to interpret.

Generalizability: Our model's simplicity may lead to better generalizability, as it relies on fewer features and assumptions about the data. This can be beneficial when dealing with new or changing datasets.

C. Cons of our Model

Lower Accuracy: Our model achieved a lower test accuracy (97.44%) compared to the Kaggle team's solution (98.10%). This might suggest that our model is less effective at identifying the toxic language in this specific context.

Lower F1-measure: Our model's F1-measure of 0.8559 indicates that there is room for improvement in balancing precision and recall. A higher F1 measure, as seen in the Kaggle team's results, may be more desirable for certain applications.

D. Pros of the Kaggle Team's Model

Higher Accuracy: The Kaggle team's model achieved a higher accuracy —(98.10%) compared to our model (97.44%), which suggests that their approach may be more effective at identifying toxic language.

Robustness: By employing multiple algorithms and feature engineering approaches, the Kaggle team's model can potentially benefit from the strengths of each individual method, leading to a more robust overall solution.

E. Cons of the Kaggle Team's Model

Complexity: The ensemble of models, stacking layers, and various feature engineering approaches used by the Kaggle team can result in increased complexity, making it harder to implement, maintain, and interpret.

Overfitting: With a more complex model and multiple layers of stacking, there is a risk of overfitting to the training data, which could reduce the model's generalizability to new or unseen data.

In summary, our BiLSTM model provides a simpler and more interpretable solution to toxic language detection, although it achieves a lower accuracy compared to the Kaggle team's ensemble approach. Future work could focus on improving the precision and recall balance of our model or explore the incorporation of additional features and techniques from the Kaggle team's approach to enhance performance while maintaining simplicity and interpretability.

VIII. Conclusion

In this paper, we proposed a novel approach for detecting toxic language in social media using a Bidirectional Long Short-Term Memory (BiLSTM) model. The model was designed to capture contextual and sequential information in texts, leveraging pre-trained word embeddings for text representation. Our proposed approach outperformed traditional keyword-based filtering and rule-based techniques, which often fall short in detecting subtle and nuanced toxic language.

Our BiLSTM model achieved a precision of 0.7933, recall of 0.9291, F1-measure of 0.8559, training accuracy of 97.04%, and test accuracy of 97.44%. Although our model's accuracy was lower than the 3rd place solution in the Kaggle Jigsaw Toxic Comment Classification Challenge, the BiLSTM model offers several advantages, such as its ability to effectively capture the sequential nature of the text and learn complex features from raw input data.

While our model shows promise in detecting toxic language in social media, there are potential areas for improvement and further exploration. In future work, we aim to investigate the inclusion of an attention mechanism to improve the model's ability to focus on relevant parts of the input text. Additionally, further fine-tuning of model hyperparameters and experimenting with ensemble techniques may yield improved performance.

Overall, the proposed approach holds significant potential for use by social media platforms to automatically detect and filter toxic content, thereby fostering a safer and healthier online environment.

References

[1] *Jigsaw Toxic Comment Classification Challenge Data*. Kaggle, https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data

[2] *Kaggle Jigsaw Toxic Comment Classification Challenge Discussion*. Kaggle, https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/discussion/52762

[3] *NLP Tutorial*. GitHub, https://github.com/graykode/nlp-tutorial

[4] *PyTorch Text Issue*. GitHub, https://github.com/pytorch/text/issues/1350

[5] *Machine Translation and Sentiment Analysis with LSTM*. Jiqizhixin, https://www.jiqizhixin.com/articles/2017-07-24-2

[6] *OpenAI Chat*. OpenAI, https://chat.openai.com