Exploration of Effective Models for Detecting Hate Speech on Twitter

Lead Author: Xiang Gao; Co-authors: Mengkun Gao, Xiangwei Huang, Yiwei Zhang

1 Cover Letter

Summary of Feedback and Substantive Modifications Based on Three Groups' Reviews of the Project Titled "Exploring Effective Models for Detecting Hate Speech on Twitter":

Improving Dataset and Sample Size: One of the primary criticisms was the imbalance in our dataset, with significantly more tweets containing offensive language compared to those containing hate speech. This imbalance skewed model performance, particularly affecting the recall rate for hate speech detection by the logistic regression model. To address this issue, we recalculated the class weights within our models, ensuring that higher weights were assigned to the underrepresented hate speech instances. This adjustment, implemented by passing the class weight parameter in Keras, ensured the classifier focused more on the minority hate speech class, enhancing the model's fairness and accuracy. This modification also partially mitigates the overfitting concern raised by Jemin Vagadia's group. **Model Exploration and Performance Evaluation:** Zhenyu Du's group suggested that our model exploration was insufficient. In response, we expanded our model exploration to include BERT [1] combined with CNN and BERT combined with MLP. These additional models leverage BERT's contextual understanding capabilities alongside the classification strengths of CNN and MLP. This enhancement aims to improve our ability to accurately detect hate speech. During experiments, we used various evaluation metrics, including precision, recall, and F1 scores (both micro and macro). While Vishesh Munjal's group recommended incorporating the AUC-ROC [2] metric for a more comprehensive evaluation, we opted to use confusion matrices to demonstrate performance across different classes. We believe that our chosen metrics adequately cover all effective evaluations, thus we did not adopt the AUC-ROC recommendation. **Data Preprocessing and Visualization Analysis:** In response to Vishesh Munjal's group's suggestions, we provided detailed steps and information on data preprocessing in the methodology section, ensuring the model was trained on clean and standardized data. Regarding their suggestions on hyperparameter tuning, we included a range of tested values to aid in understanding the model optimization process in the appenidx. Additionally, while they suggested exploring extra features, we thoroughly explored all effective features available in our dataset, and the additional features they mentioned were not present in our data. For Zhenyu Du's group's suggestions, we enhanced model interpretability by incorporating confusion matrices and provided more detailed error analysis, including specific examples of misclassified tweets and potential reasons.

We believe these changes significantly enhance the robustness and effectiveness of our hate speech detection model. We appreciate the valuable insights provided by the three groups and hope these revisions meet their expectations.

2 Introduction and Hypotheses

2.1 Introduction

In today's interconnected digital landscape, social media platforms like Twitter play a pivotal role in shaping public discourse. These platforms, however, are not just channels for communication and information sharing; they also serve as arenas where harmful content, including hate speech, can proliferate. Hate speech, as defined in various legal and academic contexts, typically refers to language that targets or demeans a group based on attributes such as race, ethnicity, gender, or sexual orientation. This type of speech can incite violence, exacerbate social tensions, and harm individuals or communities. Unlike hate speech, offensive language, while potentially hurtful and crude, does not necessarily carry the same intent or capacity for significant social harm.

Given these distinctions, it is crucial for social media companies to effectively differentiate between hate speech, mere offensive language, and benign communications. This differentiation is not merely academic but has practical implications in content moderation, shaping user experience, and aligning with legal standards across different jurisdictions. The challenge, however, lies in the subtle nuances of language use, where the same word or phrase may be innocuous in one context but harmful in another.

Due to the vast volume of data, manually distinguishing these three categories of words is not only inefficient but also challenging to achieve with high accuracy. Our objective is to develop a classifier that not only operates at a higher speed but also surpasses human performance in capturing missing information and conducting a more comprehensive analysis of the data. This approach aims to enhance both the efficiency and accuracy of the classification process.

Additionally, we specifically employed methods combining BERT with CNN and MLP to improve our detection accuracy and address the issues of data imbalance and insufficient sample size. These advanced model combinations leverage BERT's contextual understanding along with CNN's and MLP's strengths in feature extraction and classification phase, significantly enhancing our model's performance in detecting hate speech. In summary, our experiments follow the pipeline illustrated in Figure 1.

2.2 Hypotheses and Objectives

Our project aims to develop an automated classification system capable of accurately categorizing tweets into three distinct categories:

- Hate Speech: This category includes tweets that contain language intended to degrade, intimidate, or incite violence against a group based on protected characteristics.
- Offensive Language: This category captures tweets

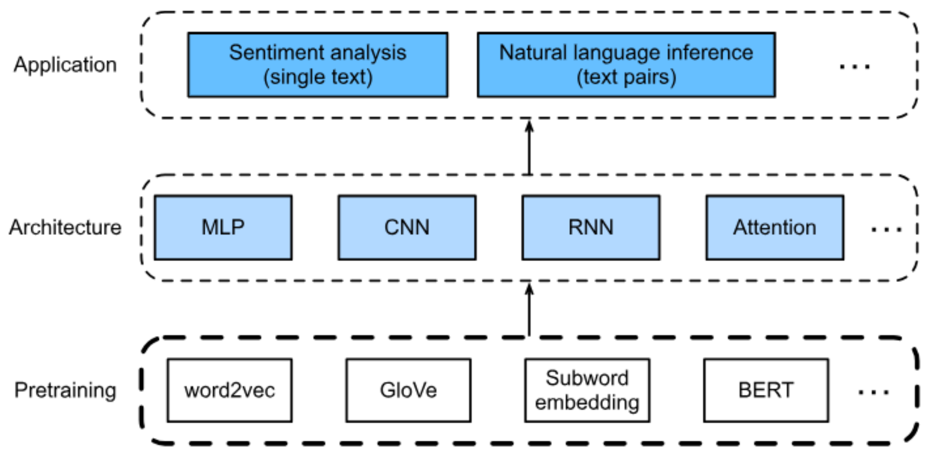


Figure 1: *Pipeline*

that use explicit or implicit derogatory language that is likely offensive to many users but does not meet the criteria of hate speech.

- Neither: Tweets that do not contain hate speech or offensive language fall into this category.

This distinction is vital for several reasons:

- Legal and Policy Compliance: Many regions have laws that specifically regulate hate speech, which can carry significant legal consequences. Social media platforms must navigate these regulations carefully to avoid legal liabilities and to enforce their community standards.
- Social Impact: Properly identifying and addressing hate speech can help mitigate its harmful effects on targeted groups, promoting a more inclusive and safe online environment.
- User Experience: Distinguishing between hate speech and merely offensive content allows platforms to implement more graded and context-sensitive moderation policies, enhancing user experience without overly censoring speech.

3 Methods

The methodology of our study on automated hate speech detection on Twitter is designed to address the complex challenges of classifying tweets into hate speech, offensive language, and neutral content. We implement a combination of machine learning techniques and statistical methods to develop a robust model that can accurately distinguish these categories, especially the Natural Language Processing, NLP. The following sections outline the specific computational and statistical methods used in our research:

3.1 Preprocessing Dataset

3.1.1 Data preprocessing

The data preprocessing phase is critical in our approach to hate speech detection on Twitter, ensuring the input text data is accurately prepared for the BERT model. Initially, we employed a BERT-specific preprocessing model from TensorFlow Hub [3], which systematically transforms raw text into numeric token IDs and arranges them into multiple tensors. This transformation is essential as it aligns with the input requirements of BERT models, facilitating efficient and standardized data handling. To illustrate, we began by extracting text and labels from our dataset, processing a sample tweet through the preprocessing model. This step involves converting the text into structured data formats, including ”input_word_ids”, ”input_mask”, and ”input_type_ids”, each playing a crucial role in the BERT model's input structure. The preprocessing model ensures consistency with BERT's documented requirements, maintaining integrity and uniformity across all text inputs.

Post-preprocessing, the structured text data was fed into the BERT encoder model. The encoder generates two types of outputs: ”pooled_output”, representing the aggregated sentence embedding, and ”sequence_output”, which includes embeddings for each token in the input sequence. These outputs are pivotal for subsequent classification tasks, providing robust and contextually rich representations of the input text.

Ensuring meticulous adherence to these preprocessing steps guarantees that our model receives high-quality, standardized input data, which is vital for achieving reliable and accurate performance in detecting hate speech. This detailed preprocessing pipeline not only enhances the reproducibility of our study but also contributes to the overall robustness of the model's performance.

3.1.2 Class Weight

Class weights [4] are an effective technique for addressing the challenge of imbalanced datasets during model training. When the distribution of classes in a dataset is skewed, with some classes being significantly more prevalent than others, the model tends to develop a bias towards the majority classes. This bias results in suboptimal performance, particularly for minority classes, as observed in our previous report. To combat this issue, class weights are introduced in the loss function of the model.

By assigning higher weights to the minority classes and lower weights to the majority classes, class weights ensure that the model places greater emphasis on correctly predicting the less frequent classes. This approach works by penalizing misclassifications of minority classes more severely than those of majority classes, thereby encouraging the model to focus more on accurately identifying instances of the minority classes. The weighted loss function adjusts the gradient updates during training, making the model more sensitive to the minority class examples and reducing the overall bias towards the majority classes.

Implementing class weights is particularly beneficial in our context, where the goal is to enhance the detection of hate speech within a dataset dominated by offensive but non-hateful language. By integrating class weights into our training process, we can significantly improve the model's performance on underrepresented classes, leading to a more balanced and fair classification outcome. This method not only enhances the model's ability to detect hate speech but also ensures that the evaluation metrics more accurately reflect the model's performance across all classes.

3.2 Enhanced Text Representation Methods

3.2.1 Global Vectors for Word Representation (GloVe) [5]

GloVe is a type of word embedding that uses matrix factorization techniques based on the co-occurrence probabilities across the whole text corpus to embed words into a geometric space. This space ideally captures the semantic and syntactic similarities between words based on their co-occurrence probabilities.

- Function: GloVe embeddings provide a dense representation of words where semantically similar words are mapped to proximate points in the vector space. This is particularly useful in understanding the context and subtle nuances in the usage of words, which is crucial for effectively distinguishing hate speech from offensive or neutral content.
- Usage: In our study, we utilize pre-trained GloVe embeddings to convert the text of each tweet into a fixed-size vector. This vector representation enables our machine learning algorithms to capture not only the presence of specific words but also the context provided by their relationships with other words in the space, enhancing the accuracy of the subsequent classification.

3.2.2 Term Frequency-Inverse Document Frequency (tf-idf) [6]

tf-idf is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. In the context of hate speech detection on Twitter:

- Function: tf-idf transforms the raw text data into a numerical form by reflecting how crucial a word is for understanding the content of a tweet compared to other tweets. It helps in distinguishing tweets with common language from those with specific, unusual terms that might indicate hate speech.
- Usage: We use tf-idf to vectorize tweets, creating a feature matrix where each row represents a tweet, and each column represents a term's tf-idf score across the tweet corpus. This numerical representation feeds into our classification models, assisting them in recognizing patterns associated with different categories of speech.

By leveraging these text representation methods, our study constructs a nuanced foundation for the advanced classification models discussed in section 2.2. This allows our models to better differentiate between hate speech, offensive language, and neutral content based on the textual characteristics of tweets.

3.3 Advanced Classification Models

We employ several classification models to evaluate their performance in distinguishing between the different types of speech. In addition to traditional machine learning models, we have incorporated advanced models such as BERT combined with CNN and BERT combined with MLP to enhance our classification capabilities:

3.3.1 Logistic Regression with Regularization [7], [8]

This model is used for its capability to handle multicollinearity and model complexity by penalizing the size of the coefficients, which helps in avoiding overfitting.

3.3.2 Support Vector Machines (SVMs) [9], [10]

SVMs are critical to our analysis due to their efficacy in high-dimensional spaces, which is common in text classification tasks like ours. To handle non-linear data relationships, we utilize SVMs with various kernels. The kernels we use include:

- RBF (Radial Basis Function): This kernel is particularly effective for handling complex, non-linear data separations.
- Linear: Although named similarly to linear SVMs, in this context, the linear kernel within a kernel SVM setting helps in handling non-linear properties of data in certain scenarios.
- Polynomial: This kernel allows for the modeling of interactions between features up to a specified degree, providing flexibility in capturing various data patterns.

These SVM configurations enhance our model's ability to accurately classify different speech types by effectively dealing with both the linear and complex non-linear characteristics of our data.

3.3.3 Random Forest [11]

Random Forest is another model used due to its robustness and excellent performance in classification tasks. By aggregating the decisions from multiple decision trees, it reduces the risk of overfitting and enhances the model's generalizability. This ensemble method is particularly useful for its ability to handle large data sets with numerous variables, making it an ideal choice for our application in hate speech detection.

3.3.4 Long Short-Term Memory Networks (LSTMs) [12]

LSTMs are a type of recurrent neural network (RNN) ideal for analyzing sequence data such as text. LSTMs are particularly suited for tasks where understanding the context and order of words is crucial. They are explored in our study to leverage their capability in capturing long-term dependencies and intricate patterns in tweet data, which is essential for recognizing contextual nuances and subtle indications of hate speech within textual streams.

By incorporating these advanced classification models, our study aims to robustly distinguish between hate speech, offensive language, and neutral content on Twitter, leveraging their distinct capabilities to improve prediction accuracy and reliability.

3.3.5 BERT [1] + CNN [13]

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art language representation model that captures deep contextual dependencies in text by processing it bidirectionally. By integrating BERT with CNN (Convolutional Neural Networks), we leverage BERT's contextual understanding with CNN's ability to extract local patterns and features. This combination involves passing BERT's output embeddings through convolutional layers, followed by pooling layers to highlight salient features, and finally into a fully connected layer for classification. This hybrid approach improves the detection of hate speech by capturing both broad contextual nuances and specific local patterns, resulting in a more robust model.

3.3.6 BERT + MLP [14]

BERT provides rich contextual embeddings crucial for understanding nuanced meanings in text. By combining BERT with an MLP (Multi-Layer Perceptron), we utilize these embeddings as high-quality input for the MLP's layers, which apply non-linear transformations to learn complex decision boundaries. The implementation involves feeding BERT's embeddings into fully connected layers of the MLP, culminating in a classification output. This approach enhances our model's ability to classify tweets accurately by leveraging deep contextual understanding and modeling complex relationships, leading to significant improvements in performance metrics like precision, recall, and F1-score.

These hybrid models demonstrate substantial advancements in the automated detection of hate speech, contributing to more effective and nuanced monitoring systems.

3.4 Model Evaluation and Validation

To ensure the robustness and reliability of our models, we conduct extensive testing and validation:

3.4.1 5-Fold Cross-Validation [15]

We utilize 5-Fold Cross-Validation to assess the robustness and reliability of our models across different subsets of the data. This technique helps ensure that our findings are not biased by any particular partition of the dataset. More importantly, 5-Fold Cross-Validation is instrumental in comparing various hyperparameter combinations. By systematically applying different sets of hyperparameters across the folds, we can evaluate their impact on model performance. This process enables us to identify the most optimal combination of hyperparameters, which is crucial for enhancing the accuracy and efficiency of our models in detecting hate speech on Twitter. This method not only validates the generalizability of our models but also aids in fine-tuning them to achieve the best possible performance.

3.4.2 Performance Metrics

In our study, models are evaluated based on several key performance metrics: precision, recall, and the F1-score. These metrics help measure the accuracy of the models and their ability to minimize false positives (incorrectly labeling non-hate speech as hate speech) and false negatives (failing to identify actual hate speech). Special attention is given to the precision-recall trade-off, which is crucial in applications like hate speech detection, where both over-prediction and under-prediction can have serious social implications.

- F1-Score [16]: The F1-score is a harmonic mean of precision and recall, providing a single score that balances both the concerns of precision and recall in one number. It is particularly useful when the costs of false positives and false negatives are roughly equivalent. We mainly use two specific kinds of F1-Score.
- Micro F1-Score: Micro F1-score aggregates the contributions of all classes to compute the average score. This is calculated by summing up the individual true positives, false positives, and false negatives of the system for different classes and then computing the precision, recall, and F1-score using these sums. In our study, micro F1 is especially significant as it reflects the overall effectiveness of the model across all classes, making it a robust indicator of performance in the context of Twitter hate speech detection. We use micro F1 as our primary metric for assessing the accuracy of our models.
- Macro F1-Score: In contrast, the macro F1-score calculates the F1-score independently for each class but then takes the average, which treats all classes equally regardless of their frequency. This metric is useful for understanding how the model performs across different categories but can be misleading if the dataset is imbalanced, as it gives equal weight to the performance on rare and common classes.

By employing both micro and macro F1-scores, we can gain comprehensive insights into the model's performance across individual classes as well as the system's overall accuracy. This dual approach allows us to address the complexity of classifying tweets into hate speech, offensive language, and neutral content, ensuring that our models are not only accurate but also fair and balanced across different types of content.

3.4.3 Misclassification Analysis and Feature Impact

We analyze the rates of misclassification and the impact of different features on the model's predictions. This analysis helps in understanding which features are most predictive of hate speech and offensive language, and where the models may be making errors.

4 Results

4.1 Model Comparison

As discussed in Section 2, our study explores two different word embedding methods: sparse embedding using tf-idf and dense embedding with GloVe. Besides, we explored six machine learning models for tackling the task of hate speech detection, including logistic regression, Kernel SVM, random forest, LSTM combined with an embedding layer, BERT-MLP, and BERT-CNN. Additionally, for each method, we examined the model both with and without class weights to address the issue of imbalance. After conducting 5-fold cross-validation on the training set to select hyperparameters, we found the optimal parameters for each of the four models detailed in Appendix A.

Using the optimal hyperparameters detailed in Table 3, we evaluated the performance of logistic regression, Kernel SVM, and random forest models with tf-idf and GloVe word embeddings. As illustrated in Table 1, tf-idf consistently outperformed GloVe across all models, achieving higher micro-F1 and macro-F1 scores. This indicates that tf-idf word representation is more effective for hate speech detection. A potential reason for GloVe's underperformance is its pre-training on general datasets rather than on specific hate speech data, limiting its relevance and effectiveness for this task. Consequently, we will employ tf-idf word embedding for logistic regression, Kernel SVM, and random forest in subsequent experiments.

Furthermore, as shown in Table 2, we evaluated all six methods mentioned above with and without class weights to identify the most suitable model for this task. Our results indicate that among the models tested, BERT-CNN exhibited the highest micro-F1 score, demonstrating superior overall performance in detecting hate speech. Conversely, logistic regression with class weight achieved the best macro-F1 score, reflecting its effectiveness across all categories on average. Therefore, if our primary goal is to identify potential prohibited content such as hate speech and offensive language, BERT-CNN is the optimal choice due to its highest micro-F1 score. However, for more precise classification, logistic regression with class weight should be considered.

Additionally, we found that with class weights, the micro-F1 scores of machine learning methods like logistic regression, SVM, and random forest remained close to those without class weights, while their macro-F1 scores improved. However, for neural network methods like LSTM, BERT-MLP, and BERT-CNN, the micro-F1 scores significantly decreased. Consequently, in the following section, we will visualize these methods with class weights using confusion matrices and analyze the results.

4.2 Visualization

In this section, we first display the class distribution for the training and testing sets in Figure 2. As shown, the distribution is imbalanced, with numerous examples of offensive language but relatively few instances of hate speech and neither category. Therefore, we propose assessing the effectiveness of using class weights to mitigate the data imbalance problem.

In Figure 3, we present a detailed visualization of the confusion matrices for the six models evaluated without class weights. Similarly, in Figure 4, we display the confusion matrices for the same models with class weights. Each confusion matrix presents the classification performance across three categories of text: hate speech, offensive language, and neither. Besides, each number in the confusion matrix represents the proportion of total predictions for each actual class that were classified as a specific class by the model. This layout allows us to assess not only the overall accuracy of each model but also its ability to distinguish between these categories. As a result, we can gain insights into the specific strengths and weaknesses of the models in differentiating between hate speech, offensive language, and neither.

5 Discussion

Based on Figure 3, our models demonstrate robust performance in identifying offensive language, with all models correctly identifying over 95% of such texts. Furthermore, they perform well in distinguishing text that is neither offensive nor hateful, with logistic regression, Kernel SVM, BERT-MLP, and BERT-CNN models correctly identifying around 90% of these instances, and random forest and LSTM models achieving about 80% accuracy.

However, a significant challenge remains in differentiating hate speech from offensive language. The best-performing logistic regression model only correctly identifies 31% of hate speech instances, with the majority misclassified as offensive language. Despite implementing pre-trained language models like BERT, the accuracy for the hate speech class remains low. These advanced models primarily contribute to more precise classification of offensive language, which is the majority class. Based on Figure 2, this issue largely stems from the data imbalance—offensive language occurs more frequently than hate speech in the dataset, impacting the models' training and performance.

After implementing class weights in all our models, we observed several notable findings. Compared to models without class weights, all six models showed improved accuracy in the minority class. For instance, the accuracy for the hate speech class increased to over 50% for logistic regression and LSTM, and models using BERT improved to exceeding 70%. However, these improvements came at the cost of decreased accuracy in the offensive language class. The more complex the model, the greater the negative impact on this majority class. Specifically, logistic regression accuracy dropped by 4%, LSTM by 9%, BERT-MLP by 13%, and BERT-CNN by 21%. One potential reason for this trend is that in more complex models, increasing the weights of the minority class may exacerbate overfitting issues more significantly than in simpler models. As a result, there is a trade-off between model complexity and overall performance. While class weights can help complex models generalize better to the minority class, they significantly impact performance on the majority class.

Upon deeper analysis of the data, we discovered that the

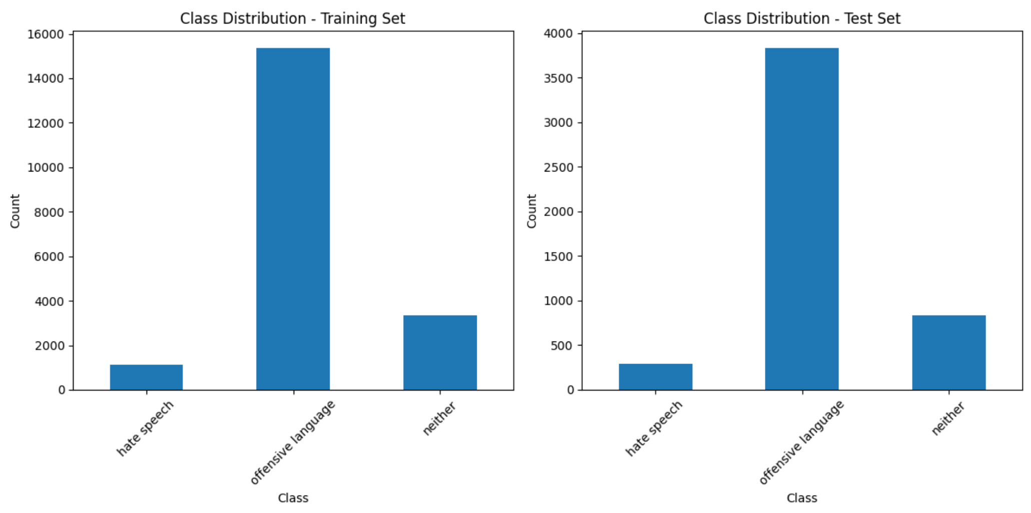


Figure 2: *Class Distribution*

Method	Embedding	Micro-F1 score	Macro-F1 score
Logistic Regression	tf-idf	0.9011	**0.7275**
	GloVe	0.8418	0.5470
Kernel SVM	tf-idf	**0.9052**	0.7060
	GloVe	0.8465	0.5270
Random Forest	tf-idf	0.8899	0.6501
	GloVe	0.8235	0.4704

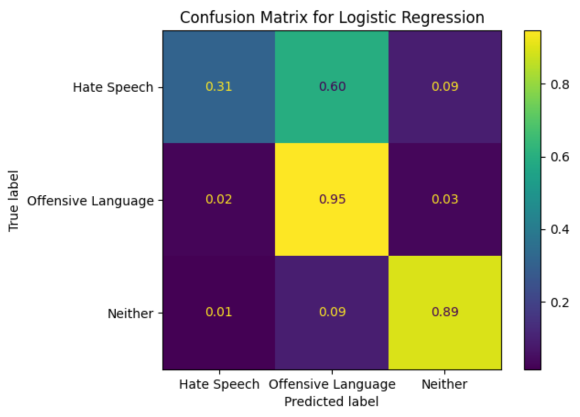
Table 1: *Micro-F1 score and Macro-F1 score comparison for different word embeddings*

nuanced use of language on social media poses additional challenges to our classification efforts. For example, tweets containing explicit racial or homophobic slurs, such as ”@JuanYeez shut yo beaner ass up sp*c and hop your f*ggot ass back across the border little n*gga” or ”@eBeZa: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot,” are straightforwardly categorized as hate speech. Conversely, a tweet like ”Why no boycott of racist ’redskins’? #Redskins #ChangeTheName,” which uses a racial slur but in a context advocating against racism, underscores the complexity of text classification. Complications also arise from tweets that utilize derogatory language in culturally specific or casual contexts, such as ”@SmogBaby: These h*es be l*yin to all of us n*ggas” and lyrics from rap songs like ”these hoes ain’t loyal.” These cases highlight the necessity for models to meticulously navigate cultural and contextual layers to accurately classify language. As a result, misclassifications are often due to the presence of specific keywords that can be misleading without proper contextual analysis, as seen in the tweet ”When you realize how curiosity is a b*tch #CuriosityKilledMe.” Additionally, infrequent forms of hate speech, such as ”Every slant in #LA should be deported. Those scum have no right to be here. Chinatown should be bulldozed,” present detection challenges due to their rarity.

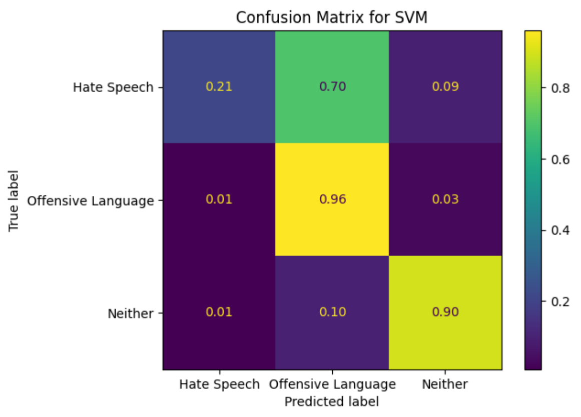
Moreover, our results in Table 1 indicate that dense word embeddings are not always superior to sparse embeddings, especially when applied to topics distinct from the original training data. Shallow neural networks, like a one-layer LSTM, often do not outperform traditional machine learning methods. Given their complexity, LSTMs may lead to overfitting, particularly in data-scarce categories like hate speech, where logistic regression might offer better generalization.

Moving forward, we propose several potential improvements to refine our approach and enhance the efficacy of our models in detecting hate speech:

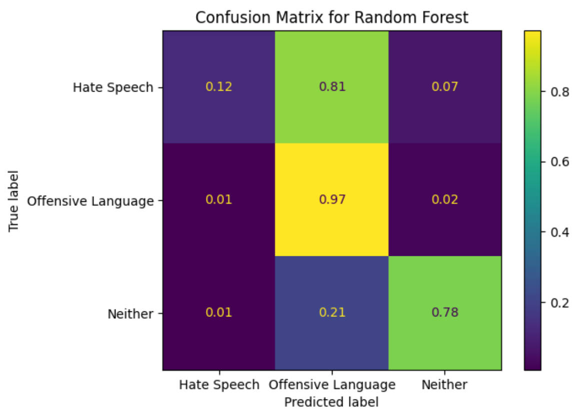
1. **Exploring Alternative Method to Balance the Dataset:**
   - **Acquisition of More Hate Speech Samples:** By collecting a larger corpus of hate speech examples, we can provide our models with a richer and more diverse set of training data, enhancing their ability to recognize such instances.
   - **Bootstrapping Techniques:** Utilizing bootstrapping to artificially augment the number of hate speech instances in our dataset could help mitigate the current imbalance, allowing for more effective training and generalization.
2. **Enhancing Word Embedding Techniques:**
   - **Hybrid tf-idf and GloVe Embedding:** By integrating the specificity of tf-idf, which emphasizes unique terms in documents, with the dense representation capabilities of GloVe, we aim to create a hybrid embedding that captures both the importance of specific terms and the contextual richness of the text. This approach involves adjusting the embeddings based on their tf-idf scores, which could lead to a more nuanced understanding of



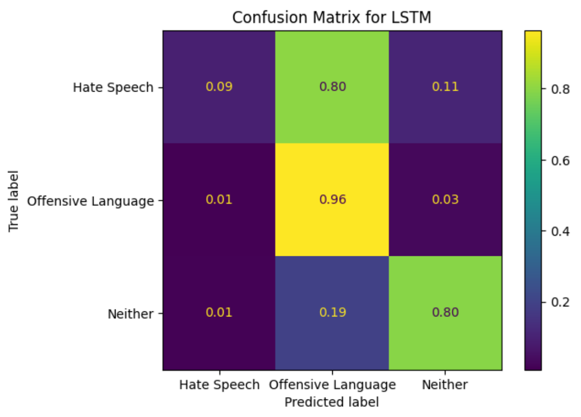
(a) *Logistic Regression*



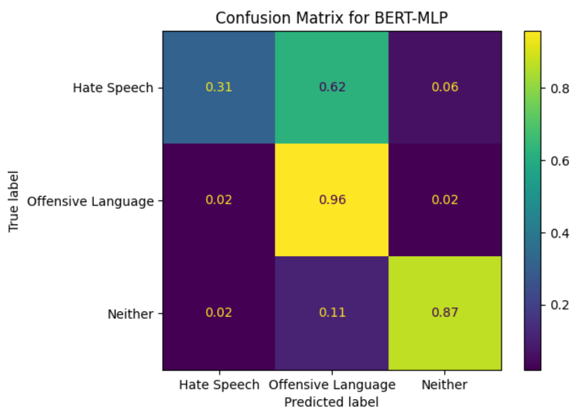
(b) *Kernel SVM*



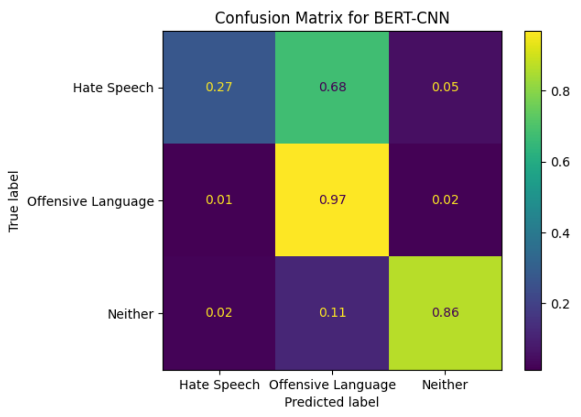
(c) *Random Forest*



(d) *LSTM*

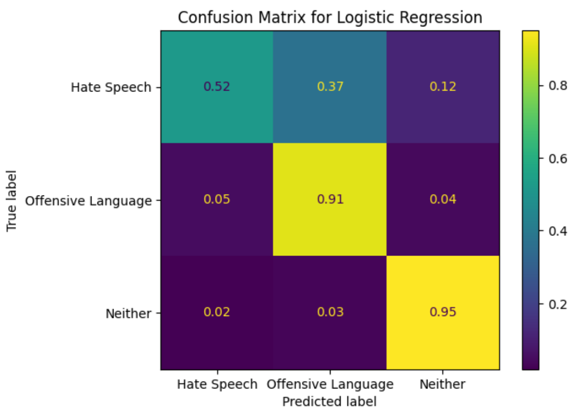


(e) *BERT-MLP*

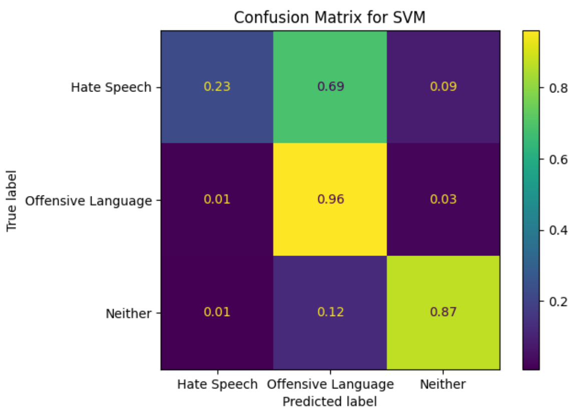


(f) *BERT-CNN*

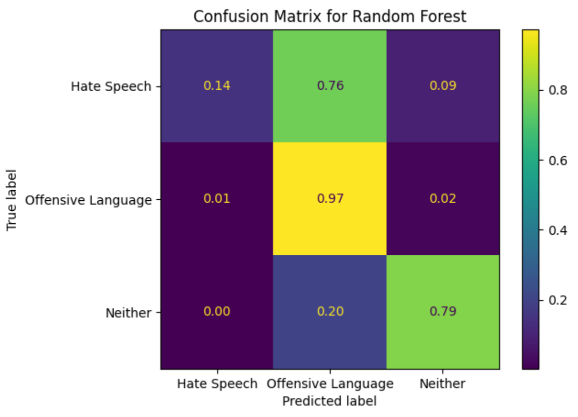
Figure 3: *Confusion Matrices for all six methods without class weight*



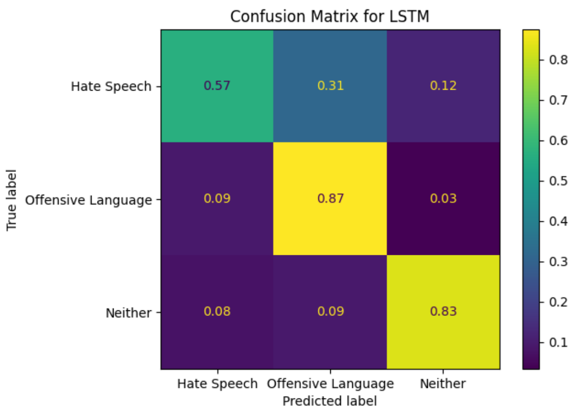
(a) *Logistic Regression*



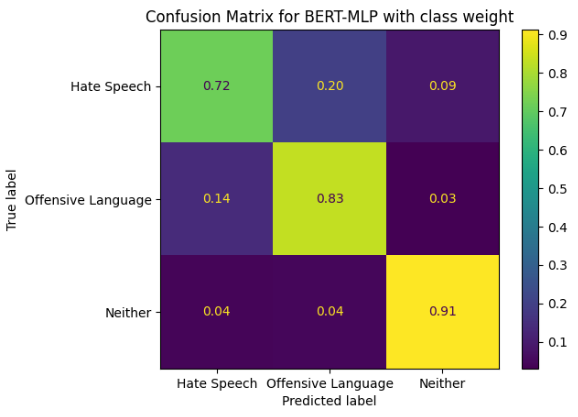
(b) *Kernel SVM*



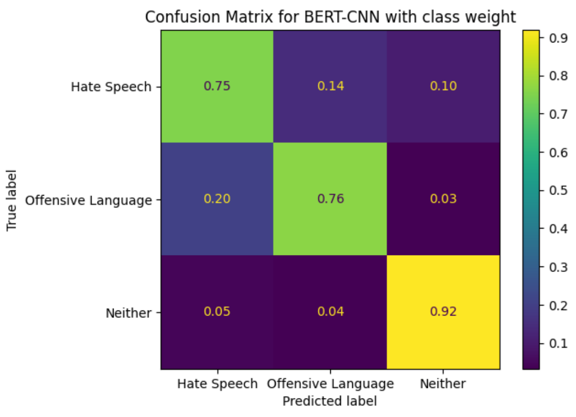
(c) *Random Forest*



(d) *LSTM*



(e) *BERT-MLP*



(f) *BERT-CNN*

Figure 4: *Confusion Matrices for all six methods with class weight*

Method	Class weight	Micro-F1 score	Macro-F1 score
Logistic Regression	w/o	0.9011	0.7275
	w/	0.8913	**0.7548**
Kernel SVM	w/o	0.9052	0.7060
	w/	0.9020	0.7066
Random Forest	w/o	0.8899	0.6501
	w/	0.8931	0.6645
LSTM	w/o	0.8844	0.6314
	w/	0.8487	0.7024
BERT-MLP	w/o	0.9052	0.7330
	w/	0.8374	0.7204
BERT-CNN	w/o	**0.9108**	0.7304
	w/	0.7898	0.6867

Table 2: *Comparison of Micro-F1 and Macro-F1 scores for various methods (w/ indicates methods with class weight, w/o indicates methods without class weight)*

the textual context and its semantic nuances.

By implementing these enhancements, we aim to further improve the precision of our hate speech detection models. These advancements will not only aid in better identifying and classifying hate speech but also in understanding the broader context within which potentially harmful language is used, thus contributing to more effective and nuanced automated systems for monitoring and addressing hate speech on social media platforms.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve.," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[3] A. Puff, S. Gross, M. Chrzan, *et al.*, "Tensorflow hub: A library for reusable machine learning modules," *arXiv preprint arXiv:1705.01066*, 2018.

[4] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[5] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[6] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[7] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, 1996.

[8] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, pp. 273–297, 1995.

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[11] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[12] A. Graves and A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[13] Y. LeCun, B. Boser, J. S. Denker, *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[14] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[15] B. Efron, "Estimating the error rate of a prediction rule: Improvement on cross-validation," *Journal of the American statistical association*, vol. 78, no. 382, pp. 316–331, 1983.

[16] A. Roshdi and A. Roohparvar, "Information retrieval techniques and applications," *International Journal of Computer Networks and Communications Security*, vol. 3, no. 9, pp. 373–377, 2015.

Appendix

A Hyperparameter Results

The table below demonstrates the hyperparameter selection results for logistic regression, Kernel SVM, random forest, and LSTM models used in hate speech detection. Due to the large size of the BERT model, which makes it difficult to train on GPUs with limited memory, we followed the hyperparameter recommendations provided in the original paper to do the hate speech detection task [1].

Model	**Hyperparameter**	**Range**	**Value**
Logistic Regression	C (regularization strength)	[0.1,10]	5
	Penalty	['l_1','l_2']	'l_1'
Kernel SVM	C (regularization strength)	[0.1,10]	1
	Gamma (kernel coefficient)	[0.1,1]	1
	Kernel	['linear', 'poly', 'rbf']	'linear'
Random Forest	n_estimators	[100,300]	300
	max_depth	['None', 10, 20]	'None'
LSTM	Dropout rate	[0.1,0.5]	0.3
	Embedding dimension	[100,300]	200

Table 3: *Best hyperparameters for different models*