Text Classification of Jigsaw Toxic Comments

Maganth Seetharaman
Department of EECS
University of California
Irvine, CA 92617
seetharm@uci.edu

Rahul R. Jois
Department of EECS
University of California
Irvine, CA 92617
rjois@uci.edu

1 Introduction

The digital landscape offers a platform for the exchange of ideas, yet it is often marred by the presence of toxic and harmful content. This reality poses a significant challenge, as online abuse and harassment can silence voices and hinder the richness of diverse dialogues. In response, online platforms frequently resort to restrictive measures, such as limiting or completely removing user comments, to maintain civility.

Addressing this challenge, our report presents an in-depth exploration of advanced classification models aimed at identifying and categorizing offensive online content. Our analysis focuses on six critical categories of toxicity: toxic, severe toxic, obscene, threat, insult, and identity hate. Utilizing a comprehensive dataset comprising 159,571 labeled and 153,164 unlabeled entries, we train and evaluate a variety of models, striving to foster a safer, more inclusive online environment.

Our exploration is structured into three distinct parts:

- **Part 1: Deciphering the Data**: This section delves into the dataset's intricacies, discussing our preprocessing strategies and the text embedding techniques we employed to enhance the predictive power of our models.
- **Part 2: A Spectrum of Models**: Here, we embark on an exploration of various classification models. Each model presents a unique approach and set of strengths in addressing the issue of online toxicity.
- **Part 3: Insights and Horizons**: In this concluding section, we present the outcomes of our model evaluations, highlighting the models that most effectively identify harmful content. We conclude by discussing potential future directions, focusing on how these models can be refined and applied to cultivate a respectful and constructive online discourse.

Embark with us on this critical journey to balance the scales between open expression and the imperative to shield online communities from the adverse effects of toxic content.

2 Data Preprocessing

2.1 Text Cleaning

Text cleaning is a crucial preprocessing step to ensure the quality and vectorization efficiency of the dataset. Our text cleaning procedures included:

- Removal of URLs and email addresses to reduce noise.
- Replacement of newline characters with a space to maintain text continuity.
- Elimination of punctuation from a defined list, which included commonly used symbols such as:

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~“”’¨«»®´·º½¾¿¡§£₤‘’
```

- Stripping of all unicode characters to standardize the text representation.

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)

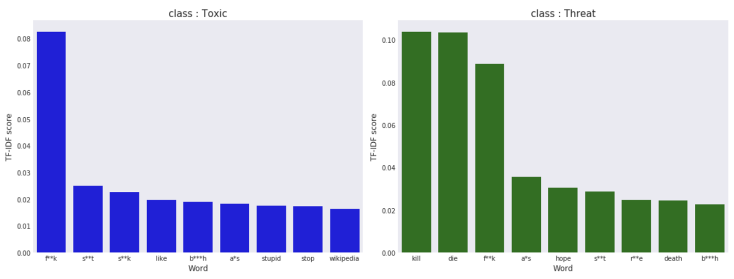


Figure 1: TF-IDF scores for select words within comments labeled Toxic or Threat.

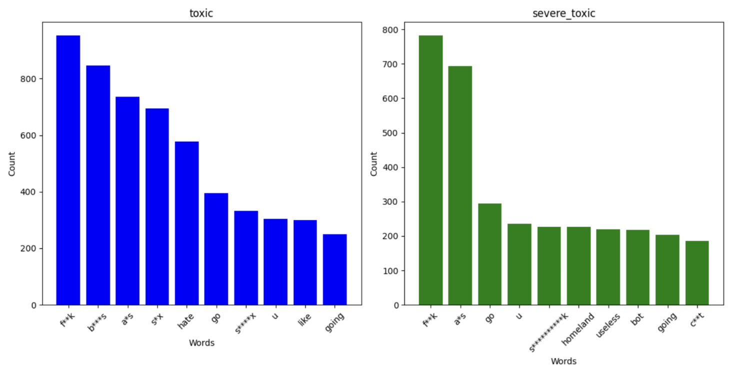


Figure 2: Distribution of token counts from Keras Tokenizer for comments labeled Toxic or Threat.

2.2 Vectorization

2.2.1 TF-IDF Embedding

The `TfidfVectorizer` from scikit-learn was employed to compute the Term Frequency-Inverse Document Frequency (TF-IDF) scores. These scores highlight the importance of words in relation to specific labels within the dataset. For instance in Figure 1, the term "kill" registers a higher TF-IDF score under the "Threat" label, underscoring its relevance. This TF-IDF scoring mechanism informs the feature importance for our logistic regression and Naive Bayes SVM (NB-SVM) baseline models.

2.2.2 Tokenization

Tokenization prepares text for model inputs. We utilized the Keras tokenizer for LSTM and HuggingFace's tokenizer for BERT, with the following configurations:

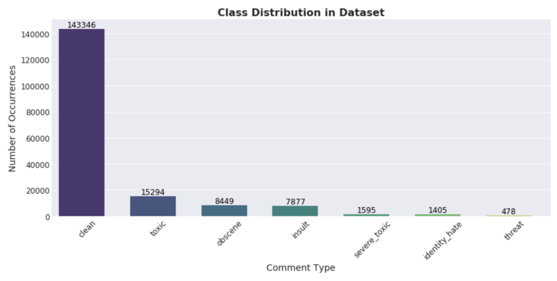
- The Keras tokenizer's vocabulary was capped at the top 20,000 words to encompass the most relevant terms.
- We limited sequence lengths to 200 characters, capturing over 85% of the comments, thereby maintaining sufficient context.
- Comments shorter than the maximum length were padded to ensure uniform input size.

3 Data Analysis

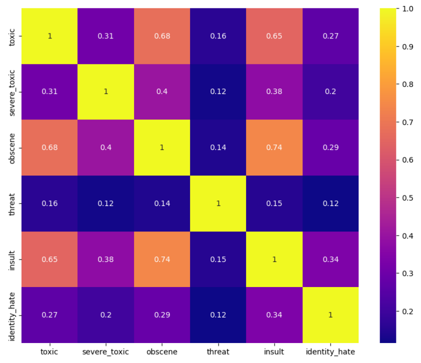
3.1 Dataset Distribution

The dataset exhibits an imbalanced distribution with a significant majority of 'clean' comments (143,346 instances) (refer fig. 3). The 'toxic' category follows with 15,294 instances, and subsequent categories like 'obscene' (8,449), 'insult' (7,877), 'severe_toxic' (1,595), 'identity_hate' (1,405), and 'threat' (478) are less frequent. Such an imbalance necessitates the use of specialized modeling techniques, such as resampling or class weighting, to ensure that minority classes are effectively represented.

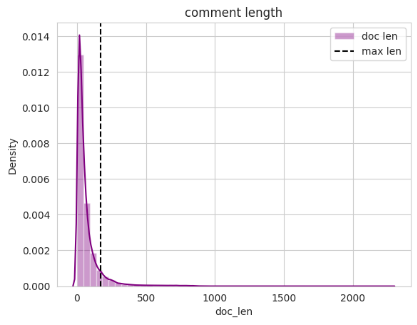
Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)



(a) Label Distribution



(b) Correlation between Toxicity Levels



(c) Comment Length Density

Figure 3: Dataset Analysis and Visualization

3.2 Correlation between Toxicity Labels

Correlation analysis within the dataset indicates a strong positive correlation between ’toxic’ and both ’obscene’ (0.6765) and ’insult’ (0.6475) categories, suggesting co-occurrence in comments. ’Severe_toxic’ shows moderate correlations with ’obscene’ and ’insult,’ whereas ’threat’ and ’identity_hate’ demonstrate lower correlation with other categories, pointing towards the complexity of toxic language and the need for models capable of discerning the nuances between different types of toxicity.

3.3 Comment Length Density

The density plot of comment lengths shows a peak at lower word counts, indicating that shorter comments are more prevalent. There is a noticeable decline in density as comment length increases, with the ’max_len’ parameter set to effectively encompass the majority of comments without excessive truncation. This parameter selection is instrumental for the LSTM model to perform optimally, given its reliance on sequential data for classification accuracy.

4 Classification

4.1 Logistic Regression (Baseline)

As a baseline, we employ a Logistic Regression model, a fundamental yet powerful algorithm in the realm of supervised learning. This model serves as a benchmark against which we can compare more complex algorithms. We preprocessed data as mentioned in the previous section and used the TF-IDF vectorization for getting the word vectors for training the model.

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)

4.1.1 Model Training

We implemented a Logistic Regression model using a One-vs-Rest classifier to manage the multilabel nature of our dataset. This approach treats each label as a separate binary classification problem. The choice of Logistic Regression is due to its simplicity, interpretability, and efficiency in dealing with binary and linear separable data. We experimented with different hyperparameters like regularization strength to prevent overfitting and optimized the solver for better performance.

4.1.2 Results and Evaluation

The Logistic Regression model, serving as our baseline, yielded insightful results: We evaluated the model using metrics appropriate for multilabel classification, such as precision, recall, F1-score, and accuracy. These metrics provided us with a holistic view of the model's performance across all labels.

	Test Accuracy	F1 Score	AUC Score	Precision	Recall
toxic	0.9376	0.9377	0.9572	0.9378	0.9376
severe_toxic	0.9934	0.9929	0.9760	0.9925	0.9934
obscene	0.9669	0.9649	0.9716	0.9643	0.9669
threat	0.9970	0.9963	0.9862	0.9961	0.9970
insult	0.9634	0.9600	0.9642	0.9593	0.9634
identity_hate	0.9902	0.9877	0.9754	0.9879	0.9902
Weighted Avg.	**0.9560**	**0.9546**	**0.9643**	**0.9543**	**0.9560**

Table 1: Model Evaluation Metrics of Logistic Regression

The model demonstrated reasonable accuracy in identifying certain categories of toxicity but showed limitations in others, possibly due to class imbalance and the inherent complexity of natural language.

4.2 Naive Bayes SVM

In order to find effective text classification methods, we explored the Naive Bayes-Support Vector Machine (NB-SVM) model, inspired by the insights from the paper “Baselines and Bigrams: Simple, Good Sentiment and Topic Classification.” This hybrid model combines the probabilistic foundations of Naive Bayes with the robustness of Support Vector Machines, making it particularly suitable for complex text classification tasks like toxic comment classification.

4.2.1 Model Description

In our implementation, we opted for Logistic Regression as a proxy for SVM, owing to its effectiveness in similar contexts. This combination aims to leverage the feature selection capabilities of Naive Bayes and the optimization strengths of SVM.

4.2.2 Implementation Steps

Our implementation of NB-SVM involved the following key steps:

1. **TF-IDF Vectorization**: We transformed the text data into TF-IDF vectors with an ngram range of 1 to 2. This allowed us to capture both the significance of individual words (unigrams) and their contextual relationships (bigrams).
2. **Naive Bayes Probability Computation**:
   For each label, we computed the log ratio of probabilities, $R = \log\left(\frac{P_{NB}(y=1)}{P_{NB}(y=0)}\right)$, where P_{NB} denotes the Naive Bayes probability. This ratio reflects the relative importance of each feature for the respective label.
3. **Recomputing TF-IDF Values with Naive Bayes**: The TF-IDF values were multiplied by the computed R values, effectively reweighting them in accordance with their importance in class differentiation. We trained a Logistic Regression model for each label using the recomputed TF-IDF.
4. **Tuning Regularization** C: We fine tuned the regularization parameter C, to identify which value would provide us with the best performing model. We finally chose the value: 0.5. Refer graph 4 for improvement of the model for each label for various values of C.

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)

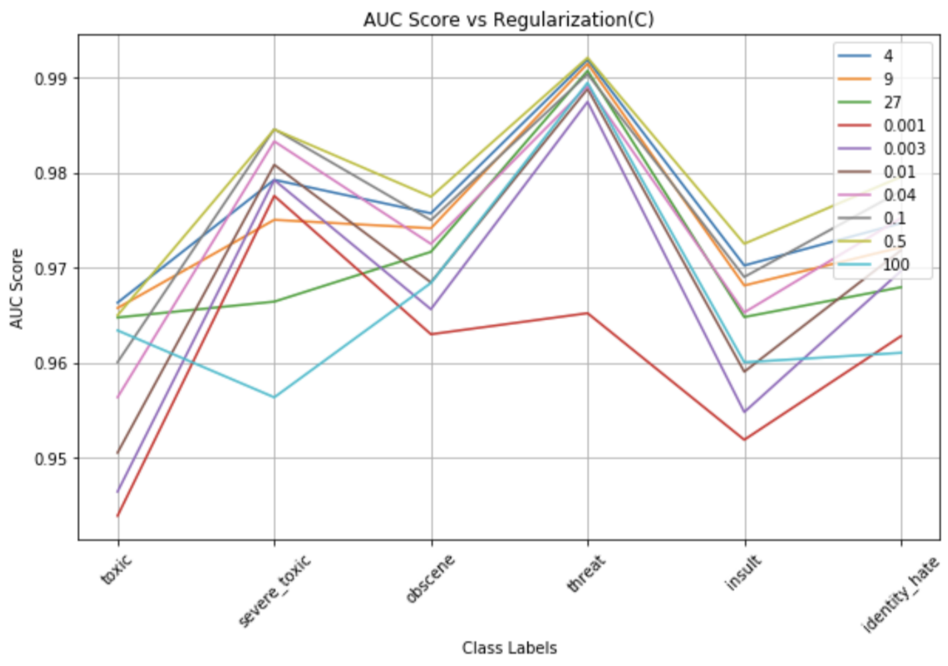


Figure 4: Tuning Regularization (C) over label vs AUC score

4.2.3 Results and Evaluation

The performance of the NB-SVM model was assessed using a suite of metrics, including precision, recall, F1 score, accuracy, and ROC-AUC. These metrics collectively provided a comprehensive view of the model's efficacy across various aspects of classification. Table 2 shows the test accuracy, AUC Score, F1 Score, Precision and Recall per label and also the weighted average.

	Test Accuracy (%)	AUC Score (%)	F1 Score (%)	Precision (%)	Recall (%)
toxic	93.35	96.63	69.00	94.21	93.35
severe_toxic	99.27	97.92	35.26	99.26	99.27
obscene	96.65	97.57	69.93	96.55	96.65
threat	99.72	99.19	44.85	99.66	99.72
insult	96.56	97.02	62.67	96.22	96.56
identity_hate	99.09	97.47	43.05	98.91	99.09
Weighted Avg.	**95.47**	**97.07**	**65.26**	**95.72**	**95.47**

Table 2: Model Evaluation Metrics for NBSVM

4.2.4 Conclusion

The NBSVM model showcases an effective way to combine the strengths of Naive Bayes and SVM in text classification. Our implementation for the toxic comment classification task underscores its potential in dealing with nuanced data. Future endeavors may include further model parameter tuning, exploring alternative feature extraction techniques, or integrating NBSVM with advanced models like neural networks for enhanced performance.

4.3 BERT

Continuing to look to improve our text classification capabilities, we adopted BERT (Bidirectional Encoder Representations from Transformers), a cutting-edge model in the field of natural language processing. BERT's approach, based on deep learning and leveraging Transformer architectures, offers a sophisticated understanding of language context and semantics, making it highly suitable for complex tasks like toxic comment classification. BERT represents a significant shift in how machines understand human language. Its core innovation is its bidirectional training, which

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)

enables the model to learn context from both the left and the right sides of a token within a sentence. This is a departure from traditional models that process text in a single direction (either left-to-right or right-to-left). By understanding the bidirectional context, BERT achieves a deeper and more nuanced understanding of language.

4.3.1 Preprocessing and Data Preparation

- We preprocess our text using BERT's tokenizer, which prepares the data into a format suitable for the model, including the addition of special tokens like [CLS] and [SEP].

4.3.2 Model Training and Evaluation

- The model is fine-tuned on a subset of our dataset for efficient experimentation. We use the `bert-base-cased` model, offering a balance between size and performance.
- Key Parameters:
  - **Learning Rate**: Set at $2e^{-5}$, this rate is optimal for fine-tuning, allowing for gradual adjustments.
  - **Batch Size**: A moderate batch size of 32 is chosen, considering computational efficiency and memory constraints.
  - **Epochs**: The model undergoes training for 5 epochs, balancing the need for learning and avoiding overfitting.
  - **Loss Function**: `BCEWithLogitsLoss` is employed, suitable for our multi-label classification task.
- We employ a 5-fold cross-validation strategy, ensuring the robustness of our model's performance. Table 3 shows the corresponding values for all the labels
- Post-training, the model is evaluated on an unseen test set, using metrics such as accuracy and AUC.

	Test Accuracy (%)	AUC Score (%)	F1 Score (%)	Precision (%)	Recall (%)
toxic	95.31	97.49	95.57	96.87	94.31
severe_toxic	99.29	99.09	99.33	99.38	99.29
obscene	96.00	98.23	96.26	96.73	96.00
threat	99.74	99.57	99.71	99.70	99.74
insult	96.58	98.19	96.72	96.93	96.58
identity_hate	99.26	99.22	99.21	99.19	99.26
Weighted Avg.	**96.14**	**97.99**	**96.35**	**97.06**	**95.72**

Table 3: Model Evaluation Metrics for BERT

4.3.3 Conclusion

The use of BERT in our toxic comment classification project marks a significant advancement in applying advanced NLP techniques. BERT's deep understanding of textual nuances enhances our model's performance, setting a foundation for future explorations in model optimization and feature enhancement.

4.4 LSTM

Next, we worked on Long Short-Term Memory (LSTM) networks, a type of Recurrent Neural Network, to classify toxic comments in online platforms. LSTMs are adept at processing sequential data, making them ideal for natural language processing tasks where understanding context and dependencies in text is crucial. LSTMs stand out for their ability to maintain memory of previous inputs through internal states, enabling them to capture temporal dependencies in sequential data like text. This makes them particularly well-suited for text classification tasks.

4.4.1 Preprocessing and Data Preparation

- The dataset, comprising text comments and toxicity labels, undergoes preprocessing using Keras's `Tokenizer`. This tokenizer focuses on the top 20,000 most frequent words (`max_features`) to balance relevance and computational efficiency.
- Tokenized text is converted into sequences of uniform length (`maxlen` of 200 tokens) to ensure consistency in input size for the LSTM model.

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)

4.4.2 Model Architecture and Parameters

- **Architecture**: The model features an embedding layer, an LSTM layer with 20 units, a global max pooling layer, and dense layers with dropout (0.1 rate) for regularization. The architecture is designed to balance learning capability with computational efficiency.
- **Embedding Layer**: Converts words into 128-dimensional vectors, capturing semantic relationships in a computationally manageable space.
- **LSTM Layer**: The 20-unit layer is tailored to process sequential data effectively, balancing complexity and learning capacity.
- **Dense Layers and Activation**: Includes 'relu' activation for intermediate processing and 'sigmoid' for the final output, appropriate for binary classification in multi-label contexts.
- **Optimizer and Loss Function**: Utilizes the Adam optimizer for adaptive learning and binary cross-entropy loss, suitable for multi-label classification tasks.
- **Input Parameters**: Limits vocabulary to the top 20,000 words and pads sequences to a maximum length of 300 tokens, ensuring relevance and uniform input size.

4.4.3 Training and Evaluation

The model is trained for 5 epochs with a batch size of 32, a setup that balances training speed and model performance, aiming to minimize overfitting while ensuring adequate learning. We tuned parameters such as lstm layer units, dense layer units and `max_len`.

We selected the following values for these parameters:

- LSTM layer units: 20
- Dense layer units: 70
- Max Length of tokens: 300

The graphs associated to our tuning improvement can be found at fig 5

The results of test data evaluation can be found at table 4.

	Test Accuracy (%)	AUC Score (%)	F1 Score (%)	Precision (%)	Recall (%)
toxic	94.16	95.80	94.53	94.95	93.16
severe_toxic	99.27	98.57	99.27	99.27	99.27
obscene	96.05	97.11	96.16	96.31	96.05
threat	99.68	98.16	99.60	99.58	99.68
insult	96.24	96.60	96.17	96.10	96.24
identity_hate	99.09	97.25	98.92	98.91	99.09
Weighted Avg.	**95.58**	**96.50**	**95.74**	**95.94**	**95.16**

Table 4: Model Evaluation Metrics for LSTM

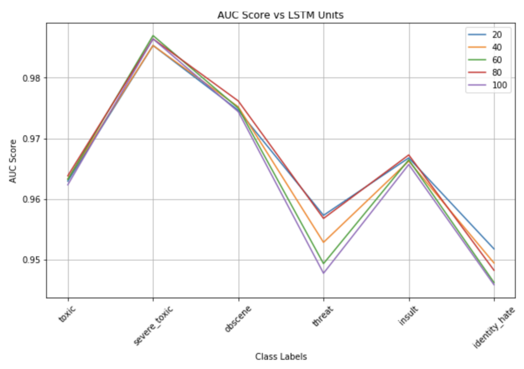
4.4.4 Conclusion

The LSTM model, with its strategically chosen parameters, presents a robust approach for classifying toxic comments. Its proficiency in handling sequential text data makes it a valuable tool in this domain. Future enhancements might include further parameter optimization, exploring additional feature engineering techniques, or investigating alternative sequence model architectures for improved performance.

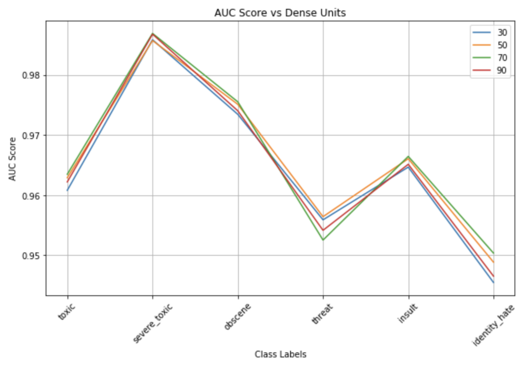
4.5 Ensemble Averaging

We implemented ensemble of our individuals models. The individual models were trained separately and stored. The prediction was computed finding the average of prediction probability of each model and determining the class for the input. Let f_i is the individual learners and x is the input, then,

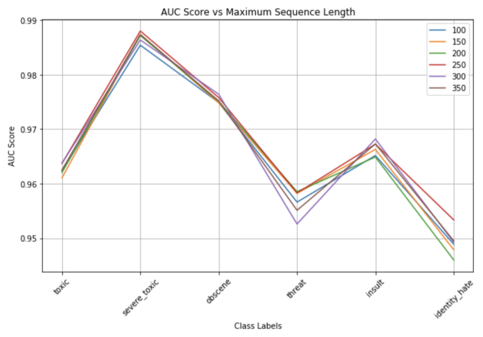
Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)



(a) LSTM layer units for AUC vs labels



(b) Dense layer units for AUC vs labels



(c) Max length of tokens for AUC vs labels

Figure 5: Dataset Analysis and Visualization

$$f(x) = \mathbb{1}[\frac{1}{n}\sum_i f_i(x)] \tag{1}$$

The results of the ensemble averaging can be found in table 5:

	Test Accuracy (%)	AUC Score (%)	F1 Score (%)	Precision (%)	Recall (%)
toxic	94.10	97.90	95.32	96.61	94.10
severe_toxic	99.03	98.98	99.19	99.40	99.03
obscene	96.08	98.16	96.31	96.69	96.08
threat	99.72	99.07	99.67	99.73	99.72
insult	96.38	97.89	96.60	97.12	96.38
identity_hate	99.26	99.14	99.34	99.39	99.26
Weighted Avg.	**95.60**	**98.07**	**96.23**	**97.00**	**95.60**

Table 5: Model Evaluation Metrics of Ensemble Averaging

5 Conclusion

This project explored various machine learning models for a multi-label classification problem, each offering unique strengths. Below is a summary of our findings and conclusions based on the performance metrics of each model.

The Logistic Regression model showed solid performance across all metrics, with particularly high scores in the 'severe_toxic', 'threat', and 'identity_hate' categories. It demonstrated a balanced performance in terms of Precision and Recall, indicating a reliable prediction capability across labels. However, it exhibited slightly lower effectiveness in categories like 'toxic', which could be due to the model's linear nature.

NB SVM excelled in specific categories, especially 'severe_toxic', 'threat', and 'identity_hate', mirroring the performance seen in LR. However, the model showed limitations in F1 Scores for some labels such as 'toxic' and 'obscene', suggesting challenges in balancing Precision and Recall for these categories. This could be due to the model's sensitivity to the class imbalance.

BERT, a more advanced model, showed outstanding performance across all metrics, outperforming other models in most categories. Its strength in understanding context and nuances in language is evident from its high scores, particularly in AUC, indicating its robustness in handling false positives and negatives.

LSTM's performance was noteworthy, especially in handling sequential data, which is crucial in text analysis. It showed consistent results across all categories, with a slight edge in 'severe_toxic' and 'threat'. This model's ability to capture temporal dependencies in text data was evident in its metrics. The Ensemble model, combining predictions from various models, provided a well-rounded performance. It leveraged the strengths of individual models, leading to improved overall accuracy and balance in Precision and Recall. This model was particularly effective in averaging out the individual weaknesses of each model, leading to a robust classification system.

In conclusion, while each model has its unique strengths, the Ensemble approach provided the most balanced and effective solution for our multi-label classification problem. BERT and LSTM showed superior performance in individual categories, highlighting the benefits of advanced models in text classification. .

6 Contributions

6.1 Rahul's Contributions

- **BERT Model Development**: Implemented, trained and evaludated the BERT model, ensuring its robustness in understanding and classifying textual data.
- **Ensemble Techniques**: Implemented the ensemble model to get predictions from all the models and compute the final prediction of the ensemble.
- **Data Analysis and Preprocessing**: Analysed the dataset and determined various parameters associated with the dataset and provided graphs to visualize the dataset

6.2 Maganth's Contributions

- **Data Preprocessing**: Implemented cleaning the comments and and tokenizer for the models.
- **Baseline Logistic Regression (LR)**: Developed the baseline logistic regression model, setting a benchmark for evaluating other models.
- **Long Short-Term Memory (LSTM) Model**: Implemented and tuned the LSTM model.
- **Naive Bayes SVM (NB SVM)**: Implemented and tuned NB SVM model.

Project Group 4, Maganth Seetharaman (64409266) Rahul R. Jois (35483737)