Toxic comment classification

Roshan Velpula B00802760 Xiaoman HU B00812030

1 Introduction

In the digital era, online platforms have become central to our social interactions, information sharing, and public discourse. However, this growth in digital communication has been accompanied by a surge in toxic online behavior. Toxic comments, ranging from subtle microaggressions to overt hate speech, not only stifle healthy discourse but also pose significant harm to individuals and communities. Traditional comment moderation methods often fall short in effectively managing the sheer volume and complexity of online interactions. Also, previous studies have revealed that those methods often inherent biases from their training data, this may lead to unfair or inaccurate tendencies when assessing real-world texts, particularly towards specific groups or topics.

2 Challenge Overview

This competition is designed to address the pressing need for advanced comment classification solutions that can reliably identify toxic comments and achieve high performance across all sub populations, rather than only focusing on the average performance across them. The core of this challenge lies in reducing biases related to comments referencing specific demographic groups.

2.1 Objectives

(1) Accurate Identification of Toxicity: Participants are expected to build models that can accurately determine the toxicity levels of comments, distinguishing harmful content from benign.

(2) Demographic Fairness: A critical aspect of this challenge is to ensure that the models are not biased against specific demographic identities. Models must be adept at recognizing and mitigating biases in comments that mention or relate to diverse demographic groups. This involves ensuring that the accuracy of toxicity detection is uniformly high across different groups, avoiding skewed performance favoring certain demographics.

2.2 Evaluation Criteria

The primary metric for evaluating submissions in this competition is the "Worst-group Accuracy." This metric is specifically chosen to address and quantify the biases in comment classification related to demographic identities.

(1) Demographic Group Segmentation: The competition dataset encompasses comments mentioning eight distinct demographic identities. Each of these identities will be divided into two categories based on the presence or absence of toxicity. For instance, comments related to the 'Black' demographic will be segmented into 'Black, Toxic' and 'Black, Not Toxic'.

(2) Accuracy Measurement: The model's performance will be evaluated separately for each of these 16 (8 demographics × 2 toxicity categories) groups. The accuracy for each group is calculated to understand how well the model performs in identifying toxic and non-toxic comments within each demographic segment.

(3) Final Evaluation Metric: The final evaluation of each model will be based on its lowest accuracy score across these 16 groups. This method ensures that the models are not just accurate on average but are also fair and effective across all demographic categories.

3 Data Description

Dataset	Description
Train_x **Val_x** **Test_x**	comments in string format
Train_y **Val_y**	**Target Column**: 'male', 'female', 'LGBTQ', 'christian', 'muslim', 'other_religions', 'black', 'white'
	Auxiliary Columns: 'Identity_any', 'severe_toxicity', 'obscene', 'threat', 'insult', 'identity_attack', 'sexual_explicit'
	Target Column: Binary label indicating toxic (1) or non-toxic (0)

Table 1: Data Overview

3.1 Target Distribution

The analysis revealed a significant imbalance in the target column, with a substantial majority of comments labeled as non-toxic.

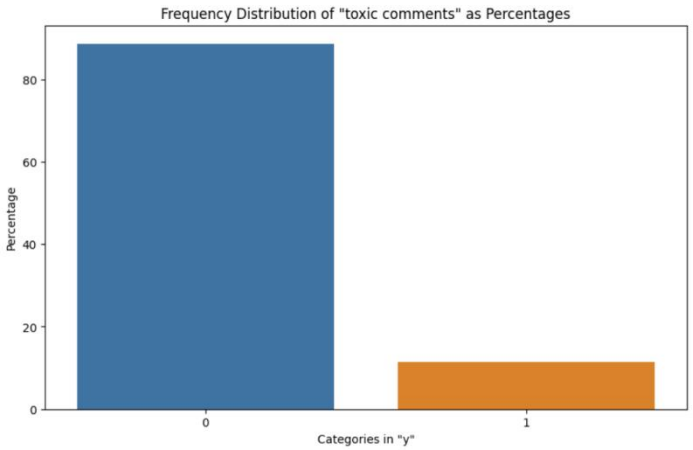

Figure 1: Frequency Distribution of Toxic Comments

This class imbalance poses a critical challenge for our classification task, as models trained on imbalanced datasets tend to be biased toward the majority class, potentially leading to suboptimal performance in identifying the minority class, in this case, toxic

comments. Addressing this imbalance is crucial to ensure that our model is capable of accurately capturing and distinguishing instances of toxicity amid the predominant non-toxic comments.

3.2 Auxiliary Columns Distribution

Among comments with auxiliary data, 'insult' appears to be the most prevalent category, followed by 'identity attack'. This distribution provides insights into the types of toxicity present in the dataset.

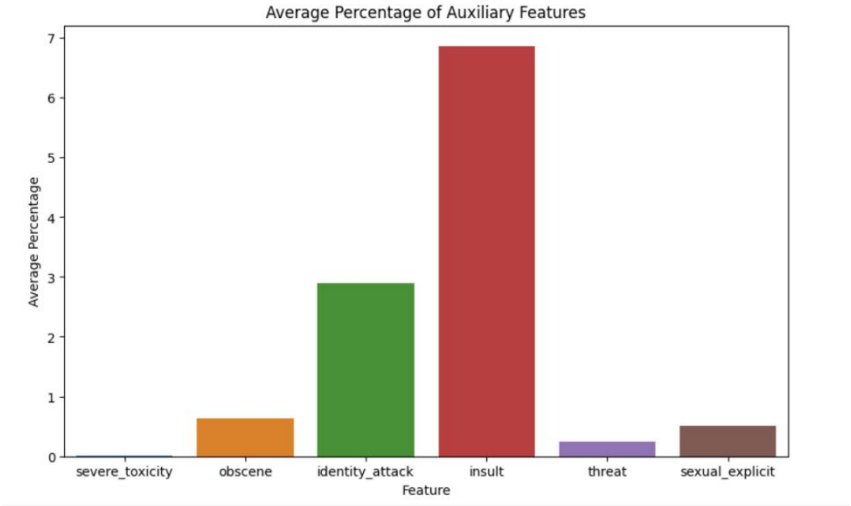

Figure 2: Average Percentage of Auxiliary Features

3.3 Demographics Distribution

Although the overall number of non-toxic comments is higher, a detailed examination at the demographic level shows a higher percentage of toxic comments for each demographic group. It's worth noticing that a substantial portion of comments labeled non-toxic lacks identity or demographic information.

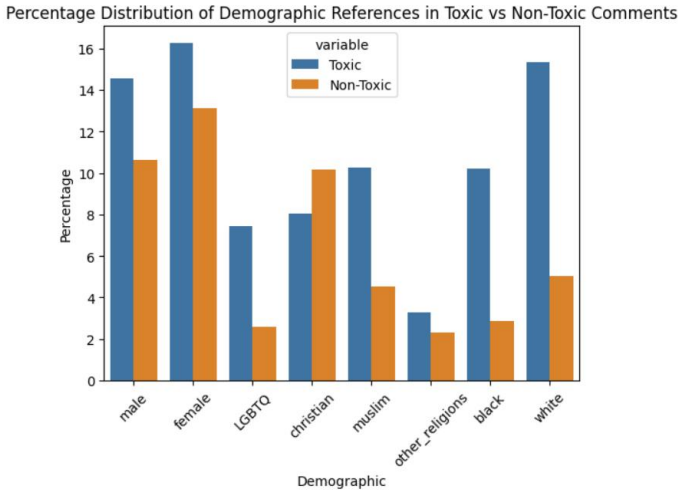

Figure 3: Distribution of Demographic References in Toxic vs Non-Toxic Comments

3.4 Word Cloud

The world cloud for toxic comments highlights a noteworthy presence of racial content,

indicating that a considerable portion of toxic comments is associated with issues related to race.

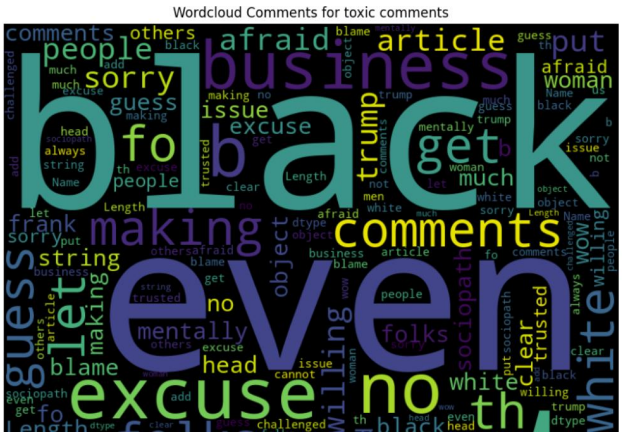

Figure 4: Woldcloud Comments for toxic comments

4 Model Selection

Our project employed two main models to address the intricacies of comment classification and bias mitigation: a BiDirectional Long Short-Term Memory (BiLSTM) network enhanced with an Attention mechanism, and a fine-tuned BERT model. Each model was selected for its unique strengths in processing textual data and its compatibility with our tailored loss functions.

4.1 LSTM

(1) Model Introduction:

LSTM is a variant of recurrent neural networks designed specifically for handling sequential data. In comparison to traditional RNNs, LSTM effectively addresses the vanishing gradient problem in long sequence training by introducing memory cells and gate mechanisms. LSTM excels in capturing long-term dependencies within sequences, making it suitable for tasks such as text classification. The model adopts a Bidirectional LSTM architecture, incorporating both forward and backward information. This allows the model to comprehensively understand input sequences and capture contextual relationships.

(2) Code Implementation:

Typically, the LSTM model is constructed using the PyTorch deep learning framework by building a Bidirectional LSTM model. The forward propagation process of the model involves the processing of both forward and backward LSTM, ultimately yielding the model's output.

4.2 BERT

(1) Model Introduction:

BERT, or Bidirectional Encoder Representations from Transformers, stands as a robust pre-trained language model rooted in the Transformer architecture. It has been a

transformative development in natural language processing. Notably, BERT excels in processing large-scale text data, utilizing deep representation learning on input text to adeptly capture context and semantic relationships, rendering it highly suitable for tasks like text classification.

(2) Code Implementation:

In practical terms, the usage of the BERT model is primarily reflected in the definition of the '*Comment_classifier*' class, where we employ the '*AutoModel.from_pretrained*' function from the '*Hugging Face Transformers library*' to load the BERT model. In the forward propagation method of the model, we use the loaded model to encode input text, followed by a Dropout layer and a linear layer for the final prediction in the sentiment classification task.

4.3 Preprocessing Techniques

Preprocessing is a critical step in preparing our dataset for effective model training and evaluation. This process involves converting raw text data into a structured format that our deep learning models, namely LSTM and BERT, can interpret. Our preprocessing pipeline consists of two main stages: tokenization and vectorization.

(1)Tokenizing the Words

<table>
<tr><th>LSTM</th><th>BERT</th></tr>
<tr><td>Basic Cleaning: We begin by converting all text to lowercase and removing punctuations and symbols to standardize the dataset. This step helps in reducing the complexity and variability of the input data.</td><td rowspan="2">AutoTokenizer from HuggingFace: For BERT, specifically the RoBERTa model, we utilize the AutoTokenizer provided by HuggingFace. This tokenizer is designed to handle the tokenization specifics required by RoBERTa, including the addition of special tokens and adhering to a maximum length of 220 tokens.</td></tr>
<tr><td>Fixed-length Tokenization: The cleaned sentences are then tokenized into individual words. We fix the tokenized output to a length of 200 tokens for each sentence. Sentences shorter than 200 tokens are padded with zeros at the end. This uniform length is essential for batch processing in LSTM networks.</td></tr>
</table>

(2)Converting Tokens to Word Vectors

For **LSTM**:

GloVe Embedding: Once tokenized, we convert the tokens into word vectors using the GloVe (Global Vectors for Word Representation) embedding, specifically the 840B version with 300-dimensional vectors. GloVe embeddings provide a dense representation of words, capturing their meanings based on the global word-word co-occurrence statistics. These embeddings are integrated into our LSTM's embedding layer, transforming tokens into vectors that represent the semantic properties of each word.

For **BERT**:

Tokenizer Outputs: The BERT model benefits from its tokenizer, which directly provides input IDs and attention masks for each token. These IDs and masks are crucial for BERT to understand the input sentence structure and focus on relevant parts of the text during fine-tuning. The tokenizer effectively converts tokens into a numerical format that BERT can process, leveraging its pre-trained embeddings and attention mechanisms.

4.4 Data Augmentation

To enhance our model's understanding and fairness in identifying toxic comments across diverse demographics, we augmented our dataset using external toxic comments sourced from Kaggle competitions like the Jigsaw Toxic Comment Classification Challenge. This approach broadened the variety of toxic expressions and contexts available for training.

Using our existing training data, we fine-tunined a BERT model to predict demographic identities mentioned within these external comments. This process allowed us to generate a synthetic dataset with identities labels. Incorporating this synthetic dataset into our training significantly improved the model's performance, particularly increasing the Worst Group Accuracy (WGA) for the LSTM model.

5 Loss Functions

The challenge of reducing bias in comment classification requires a tailored approach to loss functions. Throughout the course of our work, we experimented with various loss functions, identifying a few key approaches that significantly impacted our outcomes. This section discusses these approaches.

5.1 Weighted Binary Cross Entropy

(1)Class Weighting:

To address the class imbalance in our dataset, we implemented class weighting, by utilized the sci-kit learn module to compute class weights for toxic and non-toxic examples. These weights were based on the inverse proportion of their frequency in the dataset.The calculated weights were integrated into the *nn.BCEWithLogitsLoss* function in neural network models. While this method improved performance, it was insufficient in addressing biases at the demographic level.

(2)Identity Weighting:

To further refine our approach, we incorporated identity weighting, a new column named 'weights' was added to our dataset, these weights were dynamically updated based on both the identity and toxicity information of each comment, we calculated weights based on the discrepancy between the counts of toxic and non-toxic comments within each demographic group. This method allowed us to account for both identity and toxicity in our weighting scheme, these differential weights were then assigned to the comments associated with each identity category.

	Identity	Toxic_Count	Non_Toxic_Count	Total_Count	difference	difference_weight
0	male	4437	25373	29810	20936	1.423863
1	female	4962	31282	36244	26320	1.377052
2	LGBTQ	2265	6155	8420	3890	2.164524
3	christian	2446	24292	26738	21846	1.223931
4	muslim	3125	10829	13954	7704	1.811267
5	other_religions	1003	5541	6544	4538	1.442045
6	black	3111	6785	9896	3674	2.693522
7	white	4682	12016	16698	7334	2.276793

Figure 5: Identity Weighting

5.2 Custom Loss Function

In our continued efforts to reduce bias in comment classification, we introduced a custom loss function. This function is designed to address specific challenges inherent in our task, especially those related to subgroup fairness. Our primary objective was to ensure that the model performs equitably across various demographic subgroups, not just on the overall dataset. This loss function is created from the ‘Generalized AUC’ evaluation metric from the Jigsaw Toxic Comments classification Kaggle competition.

(1)Initialization:

Start by taking the input parameters: *predictions* (model outputs), *labels* (true labels of the comments), and *subgroups* (demographic group information for each comment).

(2)Mask Creation:

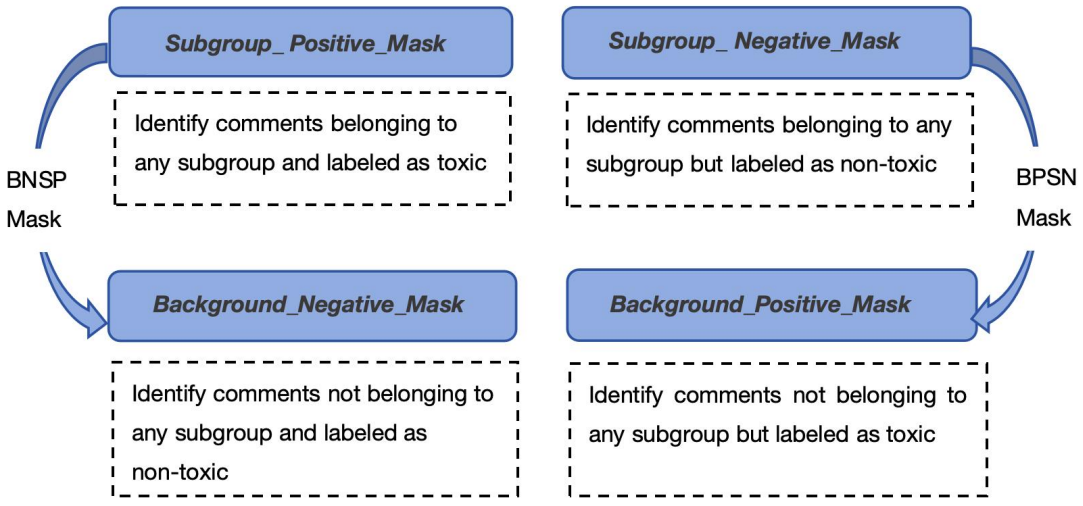

Figure 6: Mask Creation

(3)Binary Cross-Entropy Loss Calculation:

Binary Cross-entropy Loss	Compute the standard binary cross-entropy loss (*bce*) for each instance using the *predictions* and *labels*
Subgroup Component	Calculate the loss for comments in subgroups, emphasizing subgroup fairness
BPSN Component	Calculate the loss for the BPSN mask, addressing the subgroup-background dynamics
BNSP Component	Calculate the loss for the BNSP mask, further addressing subgroup-background interactions

Table 2: Binary Cross-Entropy Loss

Amplify the importance of each subgroup-based loss component by raising them to the power defined by the power parameter. This step emphasizes errors in subgroup classification, then average the standard binary cross-entropy loss with the three subgroup-based loss components to obtain the final loss value. Code implementation of this loss function can be found in our notebook.
This loss function worked the best for us when combined with BiDirectional LSTMs with Attention mechanism.

5.3 Focal Loss

A Focal Loss function addresses class imbalance during training in computer vision tasks like object detection.
We chose to use this for our case of text classification as the class imbalance is severe. Focal loss focuses on the examples that the model gets wrong rather than the ones that it can confidently predict, ensuring that predictions on hard examples improve over time rather than becoming overly confident with easy ones.
How exactly is this done? Focal loss achieves this through something called Down Weighting. Down weighting is a technique that reduces the influence of easy examples on the loss function, resulting in more attention being paid to hard examples. This technique can be implemented by adding a modulating factor to the Cross-Entropy loss.

$$Focal\ Loss\ =\ -\sum_{i=1}^{i=n}(1-p)^{\gamma}log(p)$$

Down-weighing well-classified examples: The term $(1-p)^{r}$ is key to how the focal loss works. For well-classified examples where (p) is high, (1-p) becomes smaller, and $(1-p)^{r}$ as it rises to the power it becomes even smaller. This effectively reduces the contribution of these examples to the total loss, allowing the model to focus more on difficult, misclassified examples.
Focusing on hard-to-classify examples: For examples that are hard to classify (where p is low). (1-p) approaches 1, and the loss term remains more significant. This encourages the model to focus on improving these examples, as their contribution to the total loss is more substantial compared to well-classified examples.

Using focal loss in finetuning our BERT model gave us the best results.

6 Model Implementation

6.1 BiDirectional LSTM with Attention Mechanism

(1)BiLSTM Layer: The BiLSTM architecture enables the model to capture context from both directions (forward and backward) of the sentence, providing a comprehensive understanding of the textual sequence. This bidirectional approach ensures that the model has access to all surrounding context when making predictions.

(2)Attention Mechanism: On top of the BiLSTM layer, we integrated an Attention mechanism. This mechanism allows the model to focus on specific parts of the input sequence that are more relevant to the task, essentially learning to weigh the importance of different words within a comment. The Attention mechanism is particularly useful in identifying key phrases or terms that might indicate toxicity, even in longer comments where critical information might be diluted.

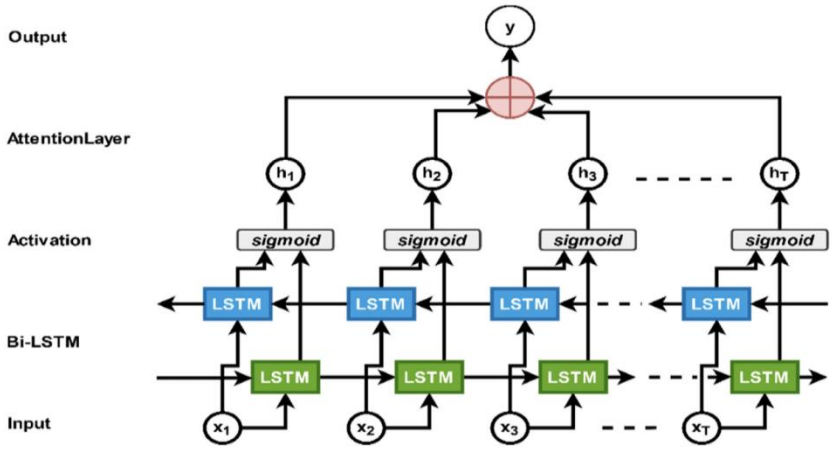

Figure 7: Attention Mechanism

(3)Custom Loss Function: To address the challenge of demographic bias in comment classification, we employed a custom loss function designed to enhance fairness and accuracy across demographic subpopulations. This loss function specifically targets the mitigation of biases by adjusting the model's focus towards underrepresented or frequently misclassified groups.

6.2 Fine-Tuning BERT with Focal Loss

(1)Pre-Trained BERT Model: BERT’s pre-trained models are leveraged for their deep understanding of language nuances, derived from training on vast amounts of text data. By fine-tuning BERT for our specific task, we benefit from its advanced language representation capabilities.

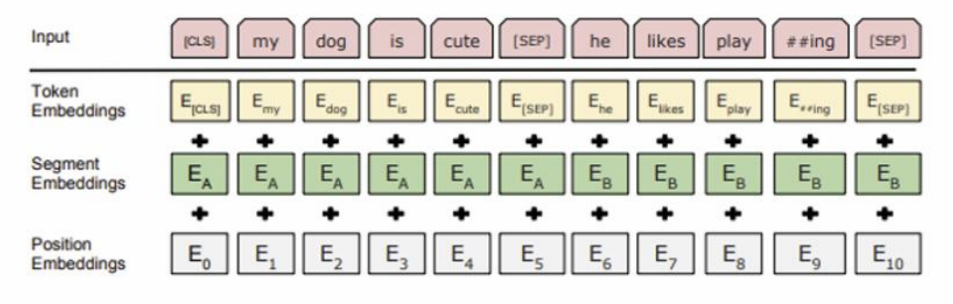

Figure 8: BERT Model

(2)Focal Loss Integration: To counteract the class imbalance inherent in toxic comment classification, we fine-tuned BERT using Focal Loss. This choice was motivated by Focal Loss's effectiveness in prioritizing the learning from hard-to-classify examples and its ability to adjust the focus towards minority classes, thus ensuring a more balanced and nuanced model performance.

(3)Model Adaptation: Fine-tuning involved adjusting the final layers of BERT to align with our binary classification task, while the Focal Loss function was applied to refine the model's sensitivity to the class imbalance and subtleties in toxic versus non-toxic comment distinctions.

7 Results

Below is the summary of results for the models and various loss functions we tested. The results represent the WGA we secured on the validation set.

	BiLSTM + Attention (Glove)	**BERT Finetuning (RoBERTa)**
Weighted loss function	77.5%	82.44%
Custom Loss function	78.2%	79%
Focal Loss	-	83.45%

An innovative approach was taken with the LSTM model by incorporating external toxic comments. By predicting the identities of these comments using a BERT model, we generated a synthetic dataset that significantly boosted the LSTM model's WGA to 81% on the validation set. This approach underscores the potential of augmenting training data to improve model robustness and fairness. However, the performance of the BERT model remained unchanged with this augmentation.

For the final test predictions, we employed an ensemble strategy combining two of our best-performing models:

BERT with Identity weighted Loss

BERT with Focal Loss

By averaging the output logits from both models and applying a sigmoid function to obtain the final labels, we achieved a public leaderboard score of 82.9% WGA. This ensemble method leveraged the strengths of both models, resulting in a balanced and

robust classifier capable of delivering high accuracy while maintaining fairness across diverse demographic groups.

8 Conclusion and Future work

In this report, we aimed to address two main challenges in comment classification mission: accurately identify toxic comments and reduce biases associated with specific demographic identities. Through data analysis, we identified a imbalance in the demographic distribution of comments, leading us to design sophisticated loss functions and model adjustments for fair and accurate performance across various demographic subgroups. By introducing models such as RoBERTa and BiDirectional LSTM, we try to enhance performance and mitigate biases through methods like class weighting, custom loss functions, and focal loss. Overall, our approach demonstrated excellent performance on the validation set, particularly in terms of Worst-group Accuracy. Future work could involve further optimizing the models, improving loss functions, and exploring additional preprocessing and data augmentation techniques to enhance model generalization and robustness.