Toxicity Guard: Implicit Words Bias Detection of LLM

Aaditya Ghosalkar

Zeqiang Ning

1 Introduction

Implicit hate speech refers to harmful language that conveys prejudice or hostility toward individuals or groups through indirect expressions such as irony or stereotypes (ElSherief et al., 2021). This type of language poses serious risks, including psychological harm and the reinforcement of hateful ideologies, which can negatively affect victims' mental health and perpetuate societal biases against minority groups .

Detecting implicit toxicity is crucial for addressing the broader consequences of hate speech, such as the incitement of violence on online platforms. Because implicit language is often culturally accepted and easily understood by in-groups, it can be used to subtly promote harmful ideologies. The nuanced nature of such language makes it particularly difficult to detect using traditional keyword-based methods (Schmidt et al., 2021).

Improving the detection of implicit toxicity is essential for enhancing content moderation systems, especially on social media platforms where the scale of user-generated content renders manual monitoring impractical. However, identifying implicit toxicity remains a significant challenge. Traditional approaches like keyword matching often fail to capture contextual cues, the intended audience, and authorial intent (Hartvigsen et al., 2022).

Many existing toxicity detectors mistakenly flag mentions of minority groups (e.g., Black Americans) as toxic without comprehending the semantic context, resulting in the under-detection of implicitly harmful content (Hartvigsen et al., 2022). Since implicit toxicity is typically expressed in subtle and indirect ways, its meaning is inherently difficult to detect. In this project, we aim to train the open-source model LLaMA-Guard to acquire the ability to detect implicit bias. Compared to current detection techniques, LLaMA-Guard holds promise for better understanding nuanced language and contextual subtleties (Inan et al., 2023). Thus, our work focuses on two main goals: (1) enhancing the model's ability to accurately identify toxicity related to minority groups, and (2) overcoming the limitations of existing detection methods by leveraging LLaMA-Guard's contextual understanding and reasoning capabilities.

2 Related work

2.1 LLaMA-Guard

LLaMA-Guard (Inan et al., 2023) is a safety-aligned language model developed by Meta for classifying prompts and responses as "safe" or "unsafe," using a predefined taxonomy of 13 risk categories. It builds on instruction-tuned LLaMA models and enforces structured outputs for interpretability. While effective in many safety-related tasks, studies have shown that its performance can degrade when category labels are modified, suggesting a reliance on surface patterns rather than true semantic understanding. But it lacks some ability to detect implicit bias, so in our project, we will add this one.

3 Problem Formulation

We formulate our task as a safety classification problem aimed at improving the detection of implicit bias in user inputs. Our work builds on LLaMA-Guard-3-1B, a safety-aligned LLM that classifies whether a given input is safe or unsafe. If marked unsafe, the model further assigns a fine-grained hazard taxonomy from a predefined taxonomy (e.g., S1: hate, S3: implict bias).

Formally, given a text input x, the model predicts a safety label $y \in \{\texttt{safe}, \texttt{unsafe}\}$. If $y = \texttt{unsafe}$, a hazard category $c \in C$ is assigned, where C denotes the set of predefined safety violation types.

Our goal is to improve the model's ability to correctly classify implicitly toxic content as either

safe or unsafe and enhance the model's capacity to accurately assign the appropriate safety taxonomy when the content is classified as unsafe. Accordingly, we fine-tune LLaMA-Guard-3-1B on datasets annotated for implicit toxicity-TOXIGEN, and restrict our evaluation to the prompt classification phase, assessing the model's capacity to detect subtle forms of bias in user-submitted prompts.

4 Dataset

4.1 TOXIGEN

Before starting preprocessing, it is important to understand how the TOXIGEN dataset is generated. TOXIGEN is a synthetic dataset automatically generated by the GPT-3 model, consisting of approximately 270,000 toxic and non-toxic statements about minority groups. Researchers designed specific prompts to guide GPT-3 in producing statements related to different demographic groups, and used an adversarial decoding algorithm called **ALICE** (Adversarial Language Imitation with Constrained Exemplars) to control the toxicity level of the generated text. Compared to datasets collected directly from the web, this approach enables better balance between toxic and non-toxic samples, and effectively generates a large number of implicitly toxic statements. This provides higher-quality training data for improving the performance and fairness of toxicity detection models.

4.2 Conversation format

As required by the LLaMA-Guard input format, each task must be structured as a **conversation**, where users and agents take turns. A conversation can be *single-turn*, consisting of one user message followed by a single agent response, or it can be *multi-turn*, involving multiple exchanges between the user and the assistant. Therefore, in our preprocessing step, we need to transform each TOXIGEN statement into a conversation format that aligns with this structure.

For instance, a single toxic or benign statement from the TOXIGEN dataset should be wrapped into a conversation where the statement is placed under the `"user"` role, and a placeholder response (e.g., `"[AGENT]"`) is included for the assistant. This enables LLaMA-Guard to evaluate the input in context and assign appropriate safety labels.

4.3 Data Preprocessing

We now describe how to convert the TOXIGEN dataset into the conversation format required by LLaMA-Guard. The original dataset contains multiple fields, and we select the following for our task:

- **Input.prompt**: the prompt used in demonstration-based prompting;
- **Input.text**: the generated statement from GPT-3;
- **Input_prompt_label**: indicates whether the prompt itself is toxic (1 = toxic, 0 = benign);
- **Answer.intent**: a human-annotated score (1–5) for the toxicity level of the generated text;
- **Answer.refTarget**: the type of toxicity towards minority groups: either direct or indirect.

Our processing strategy is as follows:

- **Prompt safety labeling**: If Input_prompt_label = 1, the prompt is labeled as unsafe; otherwise, safe.
- **Text safety labeling**: If Answer.intent is bigger than 3, the generated text is labeled as unsafe; otherwise, safe.
- **Type of unsafe content**: If Answer.refTarget = "direct", the unsafe text is classified as hate; if "indirect", it is classified as implicit bias.

Based on this, we construct a structured dataset with three columns:

Statement	**Label**	**Safety Risk Category**
Generated text	safe / unsafe	hate / implicit bias / N/A

Table 1: Final structured format before converting to conversation

We then convert each row into a conversation format. The statement becomes the **user** message, and the combined label and safety category become the **assistant** response.

4.4 Balancing the Data

In the safety taxonomy, the categories of implicit bias and hate are often difficult to distinguish due to their semantic overlap. Upon analyzing the processed TOXIGEN data, we observed a significant class imbalance: the number of samples labeled as hate was much smaller than those labeled as implicit bias.

To address this issue, we employed demonstration-based prompting. Specifically, we provided the language model(GPT-4o) with several example statements that clearly exhibit hateful content. These examples typically include explicit language and direct attacks, while also mentioning minority groups by name. Based on these demonstrations, the model was prompted to generate similar but distinct statements, thereby augmenting the dataset with more diverse and representative hate samples. This strategy helps mitigate class imbalance and improves the robustness of toxicity classifiers across different subtypes of harmful language.

5 Experiment

5.1 Experiment Setup

We use the LLaMA-Guard-3-1B model as our base and apply the LoRA (Low-Rank Adaptation) method for efficient fine-tuning. We experiment with two sets of hyperparameters, as shown in Table 2. Each configuration varies the number of epochs, batch size, and learning rate in order to explore performance under different training conditions.

Epoch	Batch Size	Learning Rate
5	4	2e-5
3	6	3e-5

Table 2: Fine-tuning hyperparameter settings

For evaluation, we compare the base model and the fine-tuned model on the same test set. Improvements in implicit bias detection indicate that fine-tuning enhances the model's sensitivity and classification performance for subtle toxic content.

5.2 Configuration in LLaMA-Guard

Our classification task is defined through a structured prompt system in LLaMA-Guard. The task instructs the model to determine whether a given user message is *safe* or *unsafe* based on a predefined safety policy. The prompt consists of four main components:

- **Task Type:** The system prompt clearly specifies the model's role as a safety classifier and defines the expected output format.
- **Policy:** We adopt the original 13-category safety taxonomy from LLaMA-Guard and introduce an additional category for *implicit bias* to better capture subtle and indirect harmful language.
- **Conversation Input:** Each input is structured as a user message in conversation format, derived from our processed TOXIGEN dataset.
- **Output Format:** The model is required to output "safe" or "unsafe" on the first line. If the message is unsafe, it must output the corresponding category code(s) on the second line. And then we add some brief explanation.

This structured prompt design enables consistent and interpretable safety classification, while also ensuring the model distinguishes between explicit and implicit forms of harm.

5.3 Experimental Analysis: Safe vs. Unsafe Classification

Table 3 presents the classification results for the *safe*/*unsafe* prediction task across the base model and two fine-tuned versions. We report standard evaluation metrics, including accuracy, precision, recall, and F1 score for both classes.

Metrics/Model	Fine-tuned 1	Fine-tuned 2	Base Model
Accuracy	0.5830	0.5965	0.4879
Precision (safe)	0.4075	0.4161	0.3995
Recall (safe)	0.9642	0.9705	0.7326
F1 (safe)	0.5729	0.5824	0.5171
Precision (unsafe)	0.9669	0.9736	0.8344
Recall (unsafe)	0.4273	0.4437	0.3225
F1 (unsafe)	0.5927	0.6096	0.4326

Table 3: Performance on safe vs. unsafe classification

The results demonstrate that both fine-tuned models outperform the base model across all metrics. Notably, Fine-tuned 2 achieves the best overall performance, with improvements in both recall and precision for the *unsafe* class, indicating enhanced detection of harmful content.

Metric / Model	FT-1	FT-2	Base
Accuracy	0.7265	**0.8391**	0.4237
Precision (S3)	0.9674	**0.9875**	0.9972
Recall (S3)	0.4972	**0.6611**	0.2808
Precision (S1)	0.6998	**0.7988**	0.5606
Recall (S1)	**0.9947**	0.9928	0.6872

Table 4: Performance on safety categories: S1 = implicit bias, S3 = hate

However, we observe that overall accuracy remains relatively low (below 60%). This is expected, as the task is not a traditional classification problem—rather, it involves subjective judgments and nuanced semantic understanding. In particular, many implicit bias examples lie close to the semantic boundary between acceptable and harmful language, making them difficult to distinguish with high certainty.All models exhibit high recall but relatively low precision in the safe class, indicating that the models are able to correctly identify most safe statements (low false negative rate). However, the low precision suggests that some unsafe statements are mistakenly classified as safe. In contrast, the unsafe class shows consistently high precision across all models, meaning that when a statement is predicted as unsafe, it is usually correct. Nevertheless, the recall for the unsafe class remains low, indicating that many harmful statements are still not being detected.

These findings highlight the importance of fine-tuning for domain-specific safety tasks and the limitations of accuracy as a standalone metric for evaluating performance in nuanced safety classification scenarios.

5.4 Experimental Analysis:Safety Category Classification

We evaluate model performance on two specific safety categories: S1 (implicit bias) and S3 (hate), as shown in Table 4. Fine-tuned models significantly outperform the base model across all metrics, particularly in recall, indicating better coverage of unsafe content.

Fine-tuned models significantly outperform the base model in recall for both categories, indicating better coverage of subtle or explicit unsafe content. In particular, the recall for S1 (implicit bias) is extremely high (above 0.99) across both fine-tuned models. This suggests that the models are highly sensitive to this category, though it may be due to overfitting or memorization of patterns associated with S1 labels rather than deep semantic understanding.

The precision for S3 (hate) is consistently high across all models, but the base model suffers from low recall (0.2808), meaning it misses many hate-related statements. Fine-tuned 2 boosts recall to 0.6611 while maintaining nearly perfect precision (0.9875), striking a better balance between accuracy and sensitivity.

Overall, these results demonstrate that fine-tuning not only improves binary safety detection but also enhances the model's ability to distinguish between different types of unsafe content. However, the extremely high recall for S1 warrants further investigation into whether the model is truly understanding bias-laden content or merely associating surface-level features with category codes.

Conclusion

In this project, we improve the ability of a safety-aligned language model to detect implicit bias and hate speech by fine-tuning LLaMA-Guard-3B on the TOXIGEN dataset. We preprocess the data into conversation format, extend the safety taxonomy with a new implicit bias category, and use demonstration-based prompting to balance the dataset.Experimental results show that fine-tuning improves classification performance, especially in detecting subtle and harmful content. However,to ensure that the model truly understands the semantics of conversations rather than merely memorizing category codes, we replaced and modified the implicit bias category label during evaluation. As a result, the model's performance on safety category recognition significantly dropped, suggesting a strong reliance on label memorization. Future work will focus on improving the model's generalization and robustness to such variations.

References

Mai ElSherief, Caleb Ziems, David Muchlinski, Vaishnavi Anupindi, Jordyn Seybolt, Munmun De Choudhury, and Diyi Yang. 2021. Latent hatred: A benchmark for understanding implicit hate speech. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 345–363, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A large-scale machine-generated dataset

for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326, Dublin, Ireland. Association for Computational Linguistics.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. 2023. Llama Guard: LLM-based Input-Output Safeguard for Human-AI Conversations.

Thomas Schmidt, Katrin Dennerlein, and Christian Wolff. 2021. Emotion classification in German plays with transformer-based language models pretrained on historical and contemporary language. In *Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 67–79, Punta Cana, Dominican Republic (online). Association for Computational Linguistics.