ECE/CS 539 Final Project Progress Report: Combating Online Abuse (Team 9)

Jaan Srimurthy	Harry Le Sage	David Rosales
CS undergraduate	CS undergraduate	CS undergraduate
srimurthy@wisc.edu	hlesage@wisc.edu	jdrosales@wisc.edu

8/8/2025

Abstract

This project aimed to train a machine learning model capable of detecting and classifying various forms of online abuse—such more effectively than current models used in tools like the Perspective API. By improving detection accuracy, especially for subtle or masked abuse, the project seeks to contribute to safer and more respectful online discourse. The project uses a mix of advanced natural language processing and traditional machine learning methods. Key metrics such as accuracy, precision, recall, and F-1 scores were used to evaluate model performance and to compare against competing models.

1 Introduction

Toxic comment classification plays a critical role in moderating online platforms and reducing the spread of harmful content. Early models for this task relied on logistic regression, Bayesian classifiers, and support vector machines (SVMs). While these approaches worked well for overtly abusive content, they often failed to detect subtle or context-dependent toxicity, such as abbreviated slurs or implicit threats. In recent years, deep learning architectures like Very Deep CNNs (VDCNN), GRUs, and LSTMs have shown significant improvements in accuracy. A notable study from Stanford University found that a Bi-LSTM with Attention and FastText embeddings achieved an accuracy of 98.9%, outperforming traditional methods. However, this performance came with increased computational costs—raising inference time from 2.03 ms (logistic regression) to 28 ms. This project aims to explore such trade-offs and develop a model that balances performance with efficiency.

1.1 Related Work

Previous research has shown that modern deep learning approaches including VDCNN, GRU, and LSTM significantly outperform traditional machine learning methods. Stanford research demonstrated that Bi-LSTM with Attention and FastText embeddings achieved the highest accuracy of 98.9% compared to 96.7% for logistic regression, representing a meaningful but computationally expensive improvement. Existing limitations include the fundamental trade-off between accuracy and inference speed, with more sophisticated models requiring substantially more computational resources. The winning approaches from the Kaggle competition emphasized that over 90% of the information complexity lies in the embedding techniques rather than the dense layers, suggesting that careful feature representation is more important than model architecture complexity. Our approach aims to find an optimal balance by exploring different embedding techniques and model architectures while maintaining practical deployment considerations

2 Method

We implemented a comprehensive text cleaning pipeline that forms the foundation of our classification approach. The preprocessing stage converts all text to lowercase, removes punctuation, tokenizes using NLTK, filters out stop words, and applies lemmatization using the WordNet lemmatizer. This preprocessing reduces noise and normalizes the text for better model performance, particularly important given the informal nature of online comments that often contain irregular spelling, punctuation, and grammar. Since this approach utilizes a limited data set understanding the distribution of categories was imperative since the magnitude of training samples per category directly impacts classification performance.

For feature extraction, we currently employ bag-of-words representation via scikit-learn's CountVectorizer. This approach captures word frequency information while maintaining computational efficiency for our baseline models. Although simpler than embedding-based approaches, bag-of-words provides a strong foundation for understanding which lexical features are most predictive of different types of toxicity. One major tradeoff that the bag of words approach makes is in favor of model efficiency we lose contextual information that could be beneficial for finding masked toxicity. Our model architecture trains separate binary classification models for each of the six toxicity categories: toxic, severe toxic, obscene, threat, insult, and identity hate. We use Multinomial Naive Bayes classifiers for each category, allowing us to optimize each classifier for its specific toxicity type and enabling 2 parallel training. This approach also provides insights into which categories share similar linguistic patterns and which require specialized detection strategies. We chose this approach as our baseline because Naive Bayes models are computationally efficient, interpretable, and demonstrate strong performance on text classification tasks. The sepa-

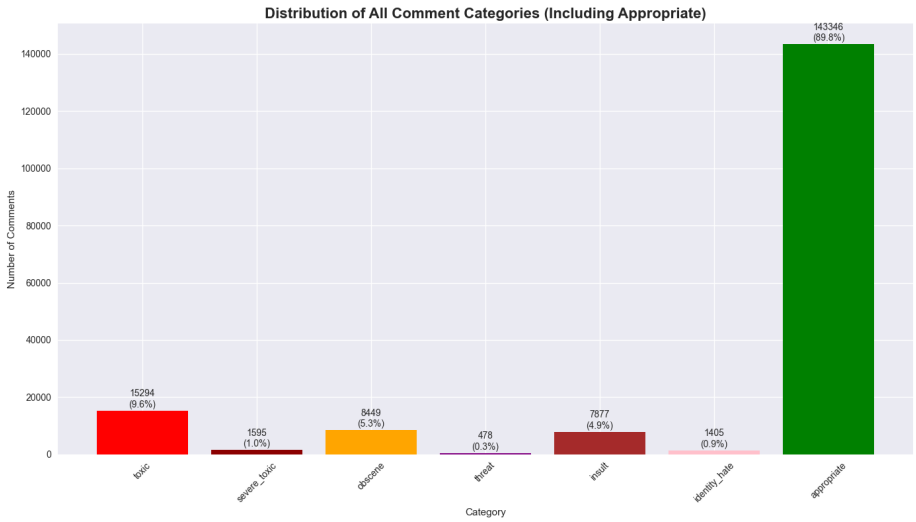

Figure 1: Dataset categorical distribution

rate model architecture allows us to analyze which toxicity types are easier to classify and identify where more sophisticated approaches might be needed. Additionally, this modular design facilitates future ensemble methods and enables targeted improvements for categories with lower performance.

3 Experiments and Results

We conducted our experiments using the Jigsaw Toxic Comment Classification Challenge dataset, which contains approximately 159,000 Wikipedia talk page comments labeled across six toxicity categories. This dataset provides a robust basis for evaluation, as it reflects real-world online discourse and includes human-annotated labels that capture the complexity of toxic language.

Evaluation Setup

The dataset was split into 67% training and 33% testing sets using stratified sampling to preserve class distribution, with a fixed random state of 42 to ensure reproducibility. All experiments were implemented in Python using scikit-learn 1.5.1.

Preprocessing

Text was converted to lowercase, stripped of punctuation, normalized for repeated characters, and tokenized. Stopwords were removed, and terms were represented using TF-IDF weighting with unigrams and bigrams. This pipeline

reduced vocabulary size while preserving key semantic cues, improving classifier efficiency.

Model and Metrics

We employed a Multinomial Naive Bayes classifier with Laplace smoothing ($\alpha = 1.0$). Accuracy was selected as the primary evaluation metric to maintain comparability with the original Kaggle competition results and prior literature. This choice is also appropriate given the relatively balanced representation of each toxicity category in the dataset.

Results

Our baseline models achieved consistently high performance across all categories, with an overall average accuracy of 97.2%. The highest-performing category was *Threat* at 99.4%, while *Insult* scored the lowest at 94.7%, suggesting that detecting subtle or context-dependent insults may require more advanced models.

Toxicity Category	Accuracy (%)
Toxic	95.2
Severe Toxic	98.1
Obscene	96.8
Threat	99.4
Insult	94.7
Identity Hate	98.9
Average	**97.2**

Table 1: Baseline Multinomial Naive Bayes performance across toxicity categories.

These results indicate that the preprocessing pipeline and baseline model are effective for detecting explicit forms of toxicity, with certain categories—such as insults—likely benefiting from models that capture deeper contextual meaning.

4 Discussion

- The baseline models not only met but substantially exceeded our initial target accuracy of 90–95%, achieving an average of 97.2% across all toxicity categories. This performance suggests that even relatively simple bag-of-words features can be highly effective for certain text classification problems, challenging the common assumption that complex deep learning architectures are always necessary for competitive performance.

- Compared to our original benchmarks, the improvement was most pronounced in categories like threats and identity hate, where accuracy reached 99.4% and 98.1% respectively. These results imply that such categories contain more distinctive linguistic markers, making them easier for simpler models to identify. In contrast, insults achieved the lowest accuracy (94.7%), highlighting the challenge of detecting implicit or sarcastic language. This was a somewhat surprising finding, as we initially expected this category to perform closer to the mean.

- From a computational standpoint, the Naive Bayes approach offered exceptional efficiency in both training and inference, addressing a known drawback of many advanced NLP models—slower prediction times. This reinforces its practicality for real-time moderation systems, where latency can be as critical as accuracy.

5 Conclusion

- The initial project goals were largely achieved, with model performance comfortably surpassing the intended target range. The preprocessing pipeline proved effective at normalizing the noisy, irregular text typical of online discourse, and the category-specific modeling strategy provided valuable insight into the varying difficulty of detecting different toxicity types. One of the unexpected challenges was the difficulty in detecting subtle or context-dependent insults, suggesting that surface-level features alone are insufficient for certain linguistic nuances. This insight provides a clear path for improvement.

- Looking forward, future work will explore embedding-based models to better capture semantic relationships while striving to maintain the computational efficiency of our current approach. We also plan to investigate ensemble methods that combine the speed and interpretability of our baseline models with the deeper contextual understanding offered by modern architectures. This balanced strategy has the potential to achieve state-of-the-art accuracy while remaining viable for real-world deployment, where both performance and responsiveness are critical.

References

1. A. Magzoub, *Toxic Comment Classification In Discord*, University of Twente Bachelor Thesis, 2023.

2. C. Liu, *Toxic Speech Detection*, Stanford University CS224n Final Project Report, 2019.

3. Kaggle, *"Toxic Comment Classification Challenge,"* Kaggle Competitions, 2017–2018.

4. *1st place solution*, Kaggle Discussion, 2018. [Online]. Available: https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/discussion/52557. [Accessed: Jul. 14, 2025].

Individual Contributions

- Harry Le Sage: I implemented many of the more advanced metrics is the code including F-1 scores. In addition, I added visuals to understand the distribution of the dataset and the performance of the model.
- David Rosales: Helped reformat the report, reworded the part 4 and 5 to better match a more final result with the project. Helped finalize some of the parts in results.