HateSpeech Detection using ML

Prithu Raj Singh Anish Jain Sarthak Kandpal Tushar Agrawal
IIIT Delhi
New Delhi, India
prithu22381@iiitd.ac.in, anish22077@iiitd.ac.in, sarthak22453@iiitd.ac.in, tushar22545@iiitd.ac.in

Abstract

The increasing prevalence of hate speech on social media platforms like Twitter, Instagram, and Reddit poses significant challenges to maintaining a safe and inclusive online environment. As platforms often struggle to enforce cyber laws consistently, automated detection and mitigation of harmful content are critical. Initially, we built and evaluated traditional machine learning models such as Logistic Regression, Naive Bayes, Perceptron, Decision Tree, Random Forest, and SVM using Word Vectorization techniques on the Twitter Sentiment Analysis Dataset from Kaggle. Expanding our scope, we tested the previously trained Random Forest model on the more extensive and complex Jigsaw dataset, revealing its limitations in handling intricate patterns in text. To enhance performance, we implemented advanced ensemble techniques like AdaBoost and hyperparameter-tuned AdaBoost models, which showed improved results. We also trained an ensemble of our 3 best performing models - RF, Perceptron and Logistic Regression. We further explored deep learning by designing a BiLSTM architecture, leveraging Keras Text Vectorization for preprocessing. Finally, to make the solution more accessible, we developed an interactive Gradio interface, enabling users to input comments and classify them as toxic or non-toxic effectively. Our work and results can be verified from
https://github.com/prithuchauhan/CSE343MLCourseProject

1. Introduction

In this study, we address the task of hate speech detection on social media platforms, with a focus on classifying tweets and comments as either toxic or non-toxic. Hate speech, a subset of toxic speech, specifically targets individuals, communities, or groups of people with discriminatory or harmful intent. To conduct this study, we utilize two publicly available datasets: the **Twitter Sentiment Analysis** dataset (Twitter Dataset) and the **Jigsaw Toxic Comment Classification Challenge** dataset (Jigsaw Dataset), both hosted on the Kaggle platform.

ID	Label	Tweet
9	1	@user lets fight against #love #peace
10	0	@user @user welcome here! i'm it's so #gr8!
11	0	i get to see my daddy today!! #80days #gettingfed

Table 1. Twitter Sentiment dataset example

The Twitter dataset comprises approximately 30,000 tweets, each labeled as either '1' or '0', where '1' indicates the presence of hate speech related to racism or sexism, and '0' represents non-offensive content. To expand the scope of this study, we also incorporate the larger and more complex Jigsaw dataset, which contains over 150,000 comments sourced from Wikipedia. These comments are binary labeled across six attributes: toxic, severe toxic, obscene, threat, insult, and identity hate. For the purpose of this study, we consolidate these attributes into a single binary column, `'is toxic'`, where a value of '1' indicates the presence of any one of the six features.

To extract meaningful features from the text, we employ both **CountVectorizer** and **TF-IDF Vectorizer**, combined with necessary preprocessing steps. Both datasets are split into an 80:20 train-test ratio to facilitate evaluation. Classical machine learning techniques such as Logistic Regression, Naive Bayes, Perceptron, Decision Tree, Random Forest, and SVM are implemented and evaluated using metrics such as Accuracy, F1 Score, and AUC-ROC, to account for the class imbalance.

To enhance model performance, we further explore advanced techniques such as ensemble learning, AdaBoosting, and hyperparameter fine-tuning. Additionally, we experiment with a deep learning architecture, specifically a BiLSTM model, to compare its performance with traditional machine learning models.

ID	Comment	toxic	severe toxic	obscene	threat	insult	identity hate
000113f07ec002fd	Hey man, I'm really not trying to edit war. It's just that this guy is constantly removing relevant information.	0	0	0	0	0	0
0001b41b1c6bb37e	I can't make any real suggestions on improvement - I wondered if the section statistics should be later on, or a subsection of types of accidents.	0	0	0	0	0	0
0001d958c54c6e35	COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK	1	1	1	0	1	0

Table 2. Jigsaw Toxic Comment Dataset Example

2. Literature Review

1. **Text Data Vectorization & Classification Algorithms Of Machine Learning - *Kozhevnikov et al. (2020)*** [1]

   This paper explores methods for text vectorization and classification, analyzing their advantages and limitations. It examines classical algorithms like Support Vector Machines, Naive Bayes, Logistic Regression, Decision Trees, and K-Nearest Neighbors, along with neural networks. Techniques such as Bag-of-Words, TF-IDF, and Word2Vec are highlighted for text vectorization. The study concludes with a discussion on widely used classification metrics, providing a comprehensive overview of text classification approaches.

2. **CoARL: A Reinforcement Learning Framework for Intent-Conditioned Counterspeech Generation** ***Shaktar, Prof. and Tanmoy, S. (2024)*** [2]
   In this paper, the authors present a novel framework called CoARL for generating non-toxic counterspeech in response to hate speech. The framework employs a combination of multi-task instruction tuning and reinforcement learning with AI feedback to produce intent-conditioned counterspeech that aligns with the context of the original hate speech. The model is trained using the COBRACORPUS dataset and optimized to ensure the generated responses are both effective and non-toxic. This paper introduced us to the concept of Counterspeech generation.

3. Dataset : EDA and Preprocessing

We used the Twitter Sentiment Analysis dataset, publicly available on the Kaggle platform and provided by Analytics Vidhya. The dataset consists of approximately **30,000 tweets** labeled as offensive (1) or non-offensive (0). The tweets include hashtags and special characters, some of which may not be interpreted correctly under UTF encoding. This issue was addressed during preprocessing. Usernames were replaced with "@user" to ensure privacy. The dataset contains three columns: ID, Label, and Tweet. The ID column, being redundant, was dropped during preprocessing. The Jigsaw Toxic Comment dataset, provided by Google and Alphabet Inc., consists of **159571 comments** sourced from Wikipedia and labeled for toxic, severe toxic, insult, threat, obscene and identity hate. We consolidate these six attributes into **one binary 'is toxic' column** which serves as our label column. We follow similar preprocessing steps for comments as we did with the tweets.

3.1. Class Imbalance

The Twitter dataset consists of 93% non-offensive tweets and 7% offensive tweets, while the Toxic Comment Dataset has 90% non-toxic and 10% toxic comments, indicating a clear imbalance. To address this, we utilized the F1 Score and AUC-ROC, which are more sensitive to class imbalances, in addition to the Accuracy metric.

3.2. Length of Tweets

For the Twitter dataset, the number of tokens between the 25th and 75th percentiles ranges from approximately 9 to 17 tokens, with the median being around 13 tokens. The maximum tweet length in the dataset is less than 35 tokens, with a few outliers present beyond 30 tokens. The Jigsaw dataset has longer comments with median being around 58 tokens. The longest tweet length is less than 1000 tokens.

3.3. Exploring Hashtags

We also explored the hashtags used in the tweets as they also carry a potential to be used for classification. **71.64%** of the total tweets used hashtags to convey information. The Jigsaw dataset does not have any hashtags.

3.4. Preprocessing

We did the following Data preprocessing steps before vectorization:-

Dropping duplicates, Emoji Removal, Punctuation and Number Removal, Lowercasing, Stop Word Removal, Username Placeholder Removal, Lemmatization

Original Tweet	@user Let's fight against #love #peace !!!
After Preprocessing	let fight against love peace

Table 3. Example of Tweet Preprocessing

4. Methodology

4.1. Data Splitting

The preprocessed dataset was split into an 80:20 train-test ratio for both the datasets. The Twitter test set had around **6000 samples** while the Jigsaw test set had approximately **30000 samples**.

4.2. Feature Extraction Using Vectorizers

We used both Count Vectorizer and TF-IDF Vectorizer initially in training and evaluating on the Twitter dataset. For Jigsaw, we only used the TF-IDF Vectorizer to vectorize tokens. For BiLSTM model, we use the TextVectorizer from Keras.

CountVectorizer: This method converts the text into a sparse matrix of word counts. Each tweet is represented as a vector, where each element corresponds to the frequency of a particular word in that tweet. The vector serves as the input features for the machine learning models.

TF-IDF Vectorizer: This technique converts the text into a matrix based on the **Term Frequency-Inverse Document Frequency**. It assigns weights to each word in the text based on its importance relative to other tweets. Words that are frequent but less informative are given lower weights.

4.3. Model Training and Evaluation

For Twitter Dataset, we used six classical machine learning models—**Logistic Regression**, **Naive Bayes**, **Perceptron**, **Decision Tree**, **Random Forest**, and **SVM (Support Vector Machine)**—to create baselines for hate speech detection. The models were tested on the held-out 20% of the dataset. To evaluate the models, we used metrics such as **Accuracy**, **F1 Score**, and **AUC-ROC**. These metrics were chosen to ensure that both majority and minority classes were considered, given the class imbalance in the dataset.

For the **Jigsaw Dataset**, we trained 5 classical ML models - **Random Forest, Naive Bayes, Perceptron, Logistic Regression, Decision Tree**. Further, we explored ensemble models like **Adaboosting,** and a **Voting Classifier** - an ensemble of three models - Random Forest, Perceptron and Logistic Regression. We also applied **Random grid search** for **hyperparameter tuning** Adaboost model. Finally, we shifted to a Deep learning model, specifically a **6 layered BiLSTM model** to see if capturing the sequential and contextual meaning could lead to better results. Due to resource constraints, we trained the BiLSTM model on only **10% of the original training set for 3 epochs**.

5. Results and Analysis

5.1. Performance Comparison Across Models on Twitter Dataset

Random Forest consistently performs better than other models in terms of Accuracy and F1 Score while having a respectable ROC-AUC score. **Logistic Regression and SVM** are other better performing models with higher ROC-AUC score than Random Forest using both Count and TF-IDF Vectorizer but slightly lesser Accuracy and F1 Score. **Decision Tree** delivers the worst AUC-ROC scores across both vectorization methods indicating its reduced ability to distinguish between the two classes compared to other models when data is imbalanced. All scores have been summarised in Table 4 and Table 5.

5.2. Performance Across Vectorizers on Twitter Dataset

Models generally perform better on **CountVectorizer** compared to TF-IDF on **Accuracy** and **F1 Score**. **Random Forest** and **SVM** perform well with both vectorizers, but their AUC-ROC scores improve slightly with **TF-IDF**, suggesting better ability to capture class separability when term weights are considered. **Naive Bayes**, in contrast, suffers a performance drop with TF-IDF, likely due to its reliance on the independence assumption, which works better with raw word frequencies rather than weighted importance.

There are no **AUC-ROC scores** for perceptron because it does not calculate probability estimates.

Model	Accuracy	F1 Score	AUC-ROC
Random Forest	0.962	0.957	0.934
Naive Bayes	0.959	0.953	0.937
Perceptron	0.954	0.953	N/A
Logistic Regression	0.958	0.953	0.953
SVM	0.956	0.947	0.943
Decision Tree	0.947	0.946	0.784

Table 4. Results with CountVectorizer (Twitter Dataset)

5.3. Performance on Jigsaw Toxic Comment Dataset

Voting Classifier (0.78), Random Forest (0.76), Logistic Regression (0.75), and **tuned AdaBoost (0.74)** show

Model	Accuracy	F1 Score	AUC-ROC
Random Forest	0.963	0.958	0.942
Perceptron	0.954	0.953	N/A
SVM	0.956	0.947	0.953
Decision Tree	0.947	0.947	0.801
Logistic Regression	0.947	0.933	0.950
Naive Bayes	0.936	0.909	0.922

Table 5. Results with TF-IDF Vectorizer (Twitter Dataset)

Model	Accuracy	F1 Score	ROC AUC
Random Forest	0.9573	0.7633	0.8330
Naive Bayes	0.9483	0.6771	0.7644
Perceptron	0.9368	0.7047	N/A
Logistic Regression	0.9574	0.7514	0.8136
Decision Tree	0.9427	0.7117	0.8333
Voting Classifier	0.9599	0.7752	0.8359
AdaBoost	0.9505	0.7125	0.7967
Tuned AdaBoost	0.9529	0.7400	0.8227
BiLSTM	0.9500	0.7222	0.9331

Table 6. Results on Jigsaw Dataset

strong performance in terms of F1 scores and accuracy. However, The **BiLSTM model (0.933)** outperforms all **others (0.83)** by a margin when comparing ROC-AUC scores indicating its capability to differentiate between toxic and non-toxic comments. While **Naive Bayes** and **Decision Trees** and **AdaBoost** performed well in accuracy, their F1 scores suggest that they may not be as effective in handling class imbalance or detecting the minority toxic class. Its easy to notice that **Hyperparameter optimized** AdaBoost outperforms Un-tuned **AdaBoost**. Another expected observation is that **Voting Classifier** delivers better results than all three of its constituent models which shows that combining multiple models can significantly boost performance as it leverages strengths of all models.

6. Conclusion

6.1. Future Scope

Real World Deployment : Integrate a tool to automatically flag hate speech on popular social media platforms like Reddit, providing real-time monitoring and mitigation.

6.2. Learnings from the Project

- **Introduction to NLP and Feature Extraction**: This project served as a gateway into the world of **Natural Language Processing (NLP)**. Through tasks like **vectorization** and exploring methods such as **CountVectorizer**, **TF-IDF**, we understood how text can be transformed into numerical features. This feature extraction process was foundational in enabling our machine learning models to ”understand” language.
- **Hatespeech Detection and its Prevalence on Social Platforms :** In recent years, the prevalence of hate speech on social media platforms like Twitter, Reddit, Instagram, and Facebook has risen significantly, driven by anonymity, unregulated communication, and often polarized discussions. This calls for automatic detection models which we train and evaluate using classical ML techniques.
- **Metrics beyond Accuracy** : A crucial realization was the significance of using metrics other than **accuracy** when working with **imbalanced datasets**. By prioritizing metrics like **F1 Score** and **AUC-ROC**, we learned how to better evaluate our models in terms of handling minority classes, a vital insight when dealing with real-world datasets that often suffer from imbalance.

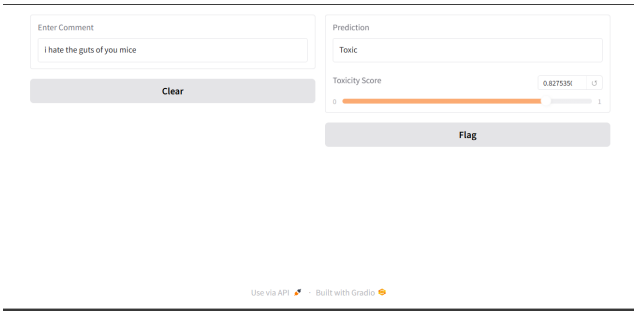

Figure 1. User Interface using Gradio

6.3. Challenges Faced

The major challenge we faced was dependency errors which made it impossible to load our models downloaded from Colab in our local systems and we had to retrain them to get inferences for our Gradio interface.

6.4. Final Deliverable

A working user interface developed using Gradio which inputs comments and labels it as toxic or non-toxic with probability scores.

6.5. Individual Contribution

- **Anish Jain** : Proposal, Literature Review, Result Analysis.
- **Prithu Raj Singh** : Model Training and Report Preparation.
- **Sarthak Kandpal** : EDA, Preprocessing and Slides Preparation,
- **Tushar Agrawal** : Future Scope and Slides preparation.

References

[1] P. P. J. et al., “Hate speech detection using machine learning,” *Proceedings of the Seventh International Conference on Communication and Electronics Systems (ICCES 2022)*, 2022 (cit. on p. 2).

[2] P. Shaktar and S. Tanmoy, “Coarl: A reinforcement learning framework for intent-conditioned counterspeech generation,” *Journal of NLP and Ethics*, vol. 56, no. 4, pp. 123–140, 2024 (cit. on p. 2).