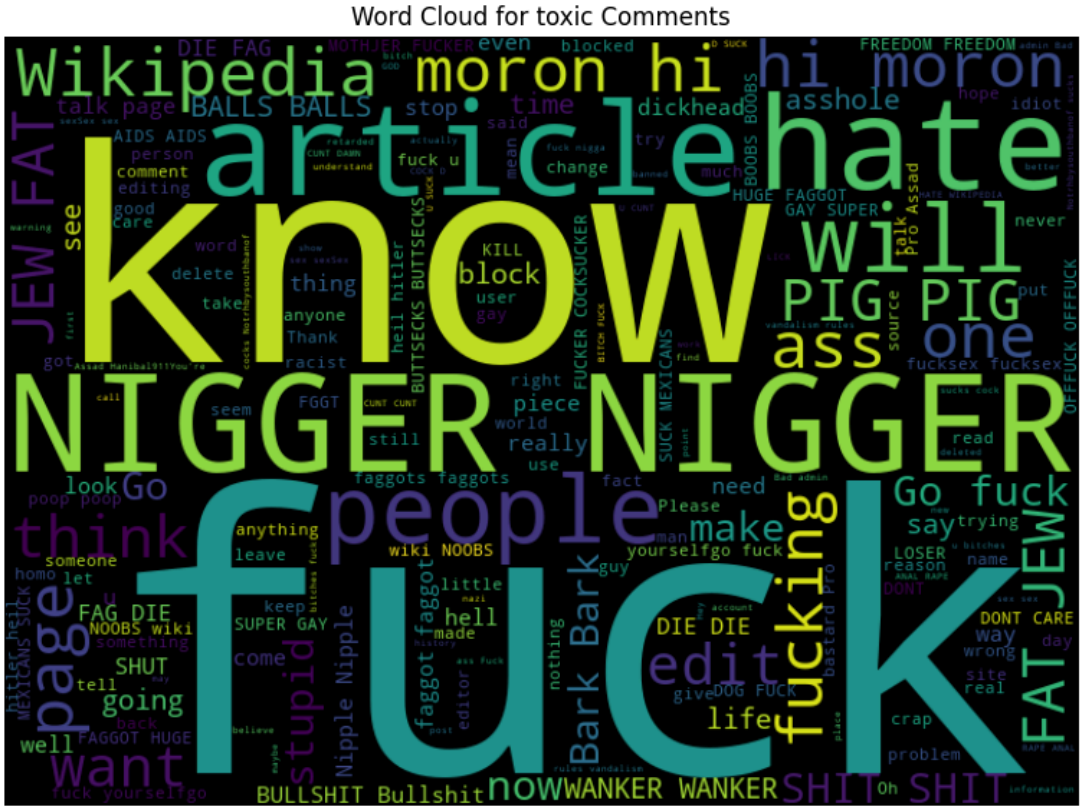
Group 43 - Rida Hummdan(11675845), Dushyant Bharadwaj(65143982), Ami Rajesh(76171988)

Toxic Comment Challenge Report

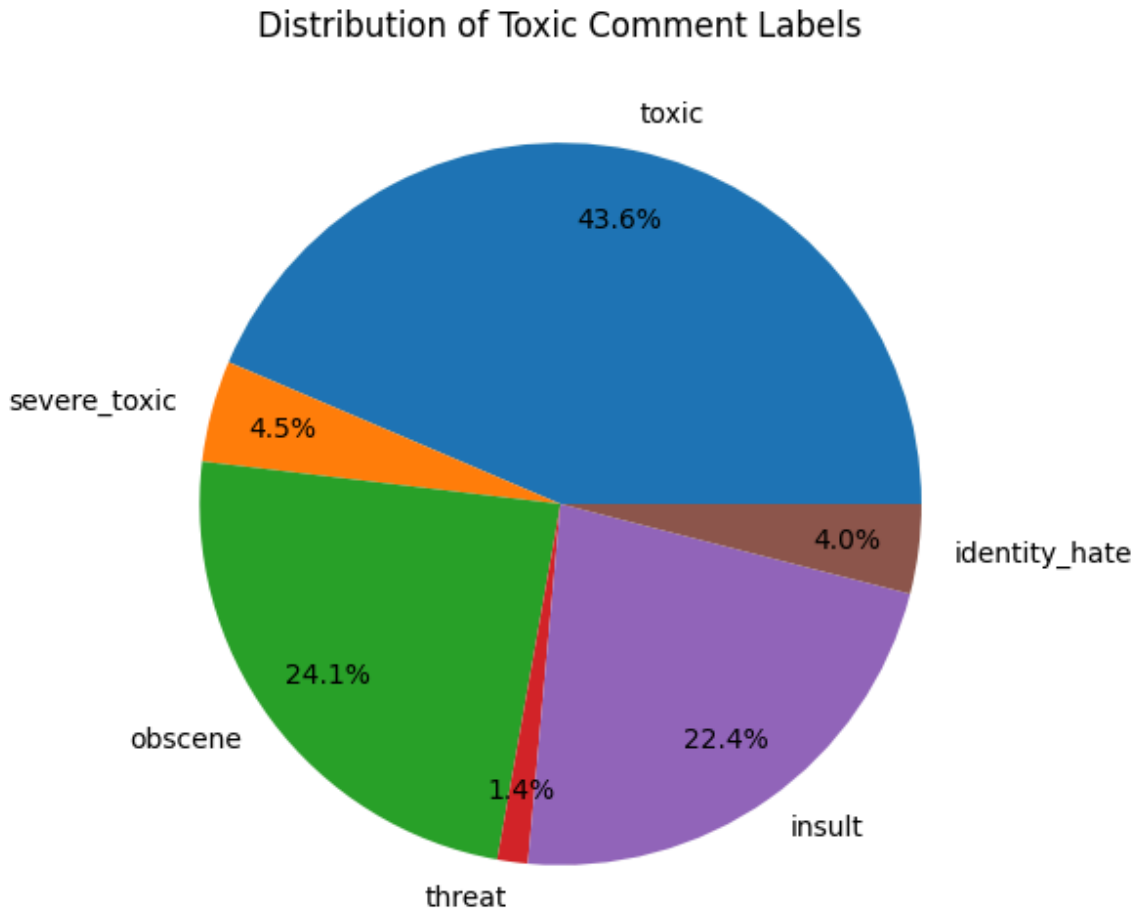


In this generation, a large part of interaction occurs online, especially among younger audiences. Because of this, it is crucial to foster positive online environments by addressing the prevalence of toxic comments on digital platforms. The Toxic Comment Challenge presents a multi-label classification problem focused on detecting and categorizing toxic behavior in online comments. This report will examine the application of machine-learning techniques to accurately classify comments into different toxicity categories: toxic, severe_toxic, obscene, threat, insult, and identity_hate.

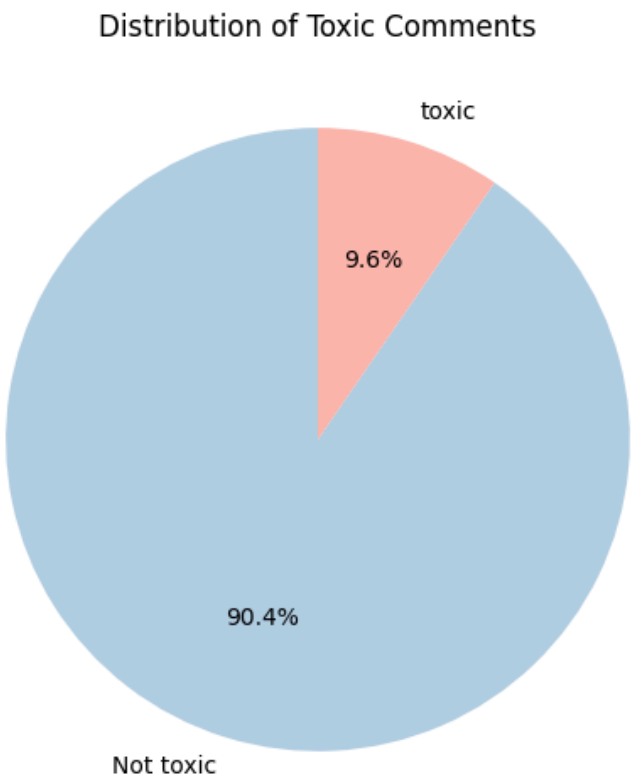
Data Exploration

To better understand the Toxic Comment Challenge dataset and its potential for classification, we processed and analyzed the raw data to explore key patterns and relationships. To start, we created a bar graph that provided an overview of how frequently each toxicity label

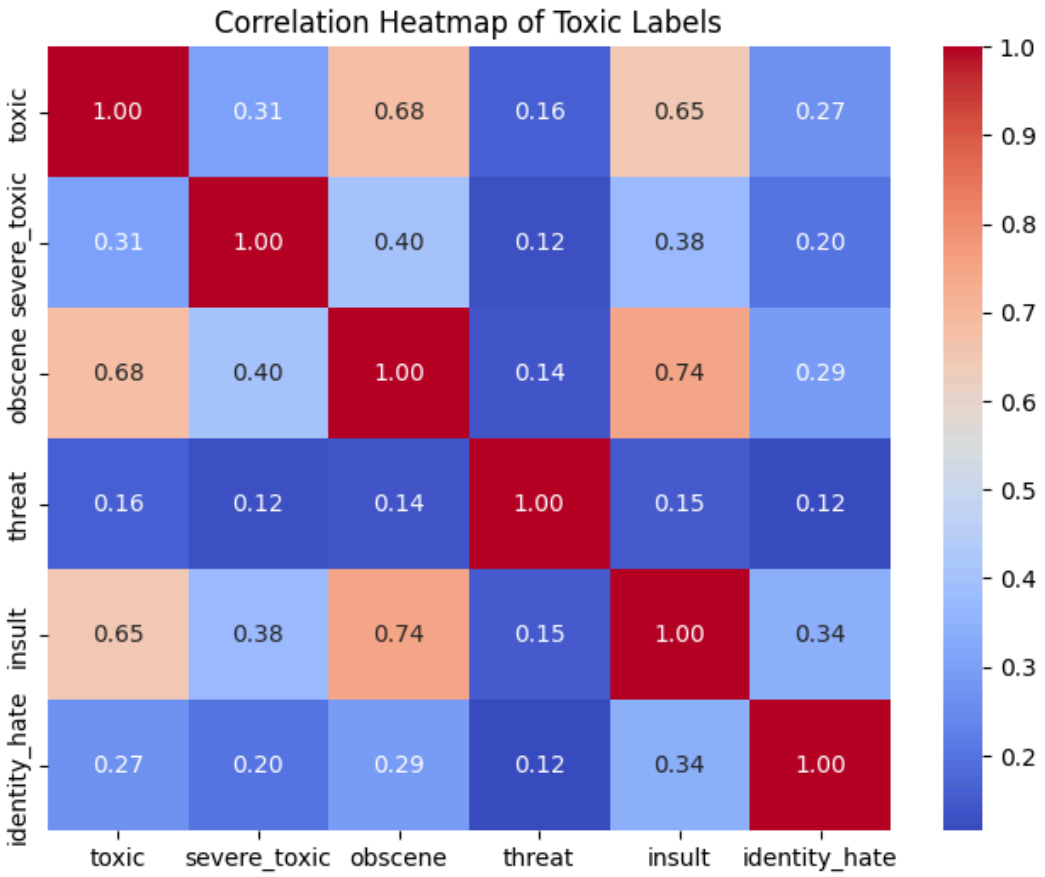
appears across the data set. The "toxic" label ended up being the most common, occurring in 9.6% of all comments. Labels such as "obscene" and "insult" were also relatively frequent, whereas "severe_toxic", "threat", and "identity_hate" were rare.



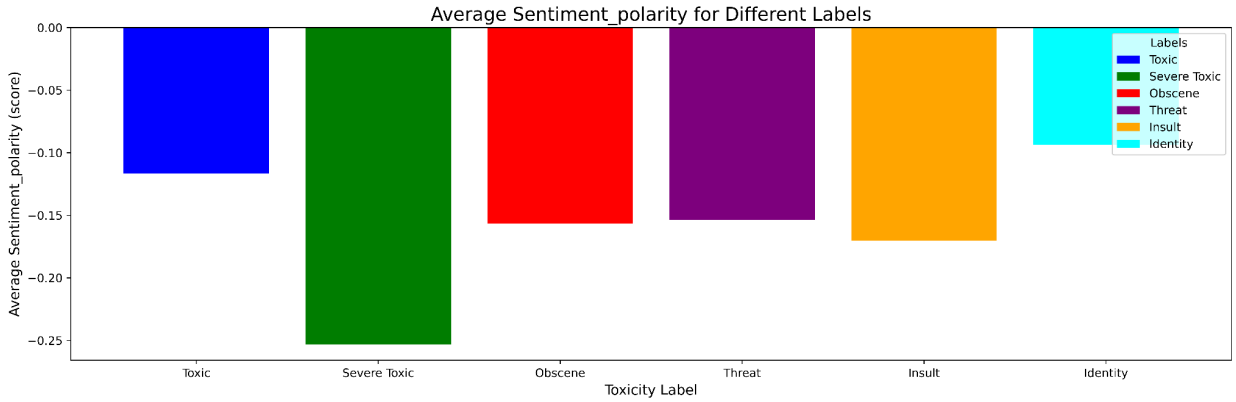
After that, we complemented the percentage breakdown bar graph with a pie chart that illustrates the distribution of toxic labels in the data set.



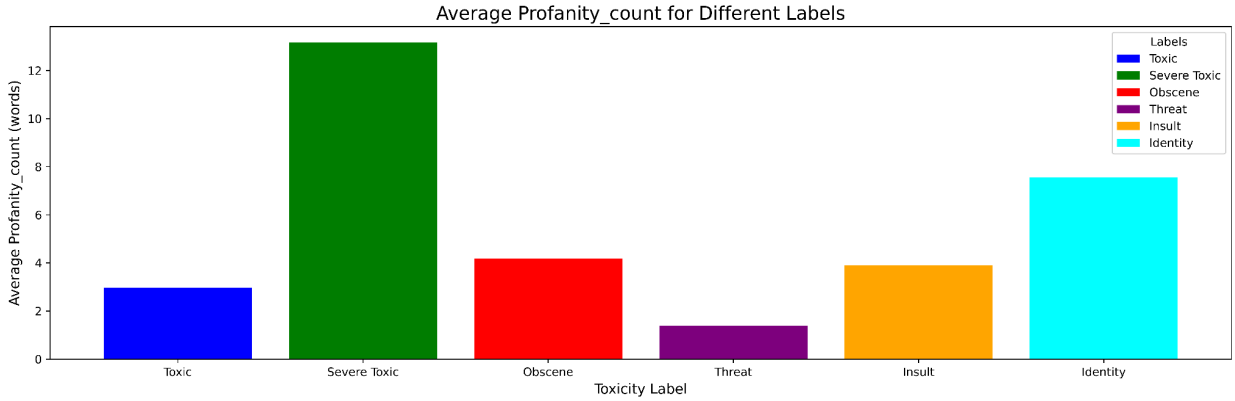
Once we understood how frequently each label occurred, we sought to establish relationships between them to better capture their dependencies and overlaps. To accomplish this, we created a correlation heat map of toxic labels. The heatmap revealed several interesting insights about the connections between the different toxicity labels. There was a strong positive correlation between the “toxic” and “obscene” labels, suggesting that content marked as “toxic” often contains obscene language as well. The same holds for the labels “insult” and “obscene” and “obscene” and “insult”. In contrast, the “threat” label was more distinct, with lower correlations to the other toxicity types. This suggests that comments classified as threats tend to be distinct and separate from other forms of toxic language.



After gaining insights into the relationships between the toxicity labels, we shifted our focus to analyzing the content of the comments. This involved us examining specific textual features such as the use of capital letters, profanity, and exclamation marks. To further understand the language used in each toxicity label, we conducted a sentiment analysis of the comments.

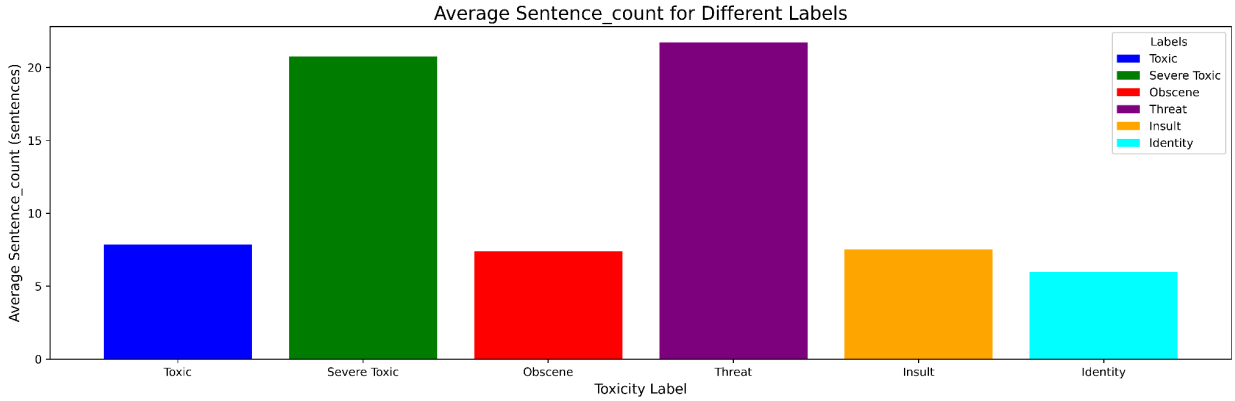


The bar graph shows the average sentiment polarity for each label, revealing key patterns in the data. Labels such as Severe Toxic and Threat showcased the most negative sentiment, which reflects the intensive negativity associated with these labels. This insight highlights how varying levels of toxicity impact the sentiment distribution and helps to understand the type of words used in each label. After analyzing the sentiment polarity across labels, which revealed varying degrees of negativity among labels. We wanted to explore the use of profanity in comments.



Labels like Severe Toxic show significantly higher levels of profanity compared to others, aligning with strong negative sentiment profiles. However, Threat label comments, despite being harmful, show lowest profanity indicating that threats may rely more on implied meanings rather than explicit offensive language.

We also examined features like sentence count and word count, among others.



This transition allowed us to gain a more comprehensive understanding of the data set, which was critical in shaping our feature selection.

Classifiers

The classifiers that we chose were XGBoost, Long Short-Term Memory (LSTM), and pretrained models.

XGBoost is a powerful tree-based gradient-boosting ensemble method. Gradient boosting typically uses decision trees as its base learner. XGBoost builds on this concept by introducing optimizations to traditional gradient-boosting decision trees (GBDT). “With XGBoost, trees are built in parallel, instead of sequentially like GBDT. It follows a level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set.” (NVIDIA). We started the XGBoost training process with preprocessing, where we cleaned the text by converting it to lowercase, removing special characters, and eliminating stopwords to reduce noise. The cleaned text was then transformed into dense sentence embeddings, which prepared the data for XGBoost. We also split the dataset into 80% training data and 20% testing data to accommodate the computational requirements of our chosen models while effectively utilizing the size of the dataset.

Initially, we trained the model without taking care of the class imbalance and the model performed an average of 98.2%. To look into the possibility of improving the model accuracy furthermore, we applied SMOTE to the training data, which would generate synthetic samples for minority classes like threat and identity_hate.

An important thing to note is that for XGBoost and the rest of our classifiers, we trained a separate model for each label, which turned the multi-class classification problem into a binary

classification. Although this approach resulted in significant drawbacks in computation resources, we decided to prioritize accuracy and take this path.

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) designed to process sequential data and address the challenges of traditional RNNs. Traditional RNNs struggle with long-term dependencies because they rely on a single hidden state, which makes retaining information over long sequences difficult. LSTMs solve this by using a more complex memory structure, including gates that control what information is kept, updated, or forgotten. The training process of LSTM and XGBoost was similar in that we performed text preprocessing, handled class imbalance with SMOTE, and treated each label as a separate binary classification task.

However, the main difference is how we had each model process the text data. While we used sentence embeddings for XGBoost to convert text into numerical features, for LSTM, we applied a tokenizer to map words to numeric indices and padded the sequences to ensure uniform input lengths. This approach allowed the LSTM to directly analyze sequential patterns in the data, leveraging its architecture to capture dependencies across the text.

The pre-trained model we used for toxicity classification was DistilBERT, a variant of BERT designed for sequence classification tasks. DistilBERT is pre-trained on extensive datasets, learning to understand word context through masked language modeling. It has the ability to handle linguistic complexity and capture deep contextual meanings.

Results

XGBoost Classifier Test Accuracy

XGBoost achieved excellent performance across all labels, even for minority classes like threat and identity_hate. This demonstrates its robustness and ability to handle imbalance datasets effectively when we train the model with SMOTE balancing and parameter tuning.

	Label	Accuracy
1	toxic	0.9505248316
2	severe_toxic	0.9864640451
3	obscene	0.9707347642
4	threat	0.9967413442
5	insult	0.9666301112
6	identity_hate	0.9903180323

LSTM Classifier Test Accuracy

LSTM are computationally intensive and achieved strong results with an average accuracy of 90.2% approximately. However, it underperformed compared to other two models. This shows that LSTM are sensitive to dataset characteristics despite SMOTE balancing. To improve in the future, we would like to consider using fine-tuning architecture with additional layers.

	Label	Accuracy
1	toxic	0.8380072066
2	severe_toxic	0.9013316622
3	obscene	0.9135202883
4	threat	0.9195676014
5	insult	0.8663637788
6	identity_hate	0.8810590631

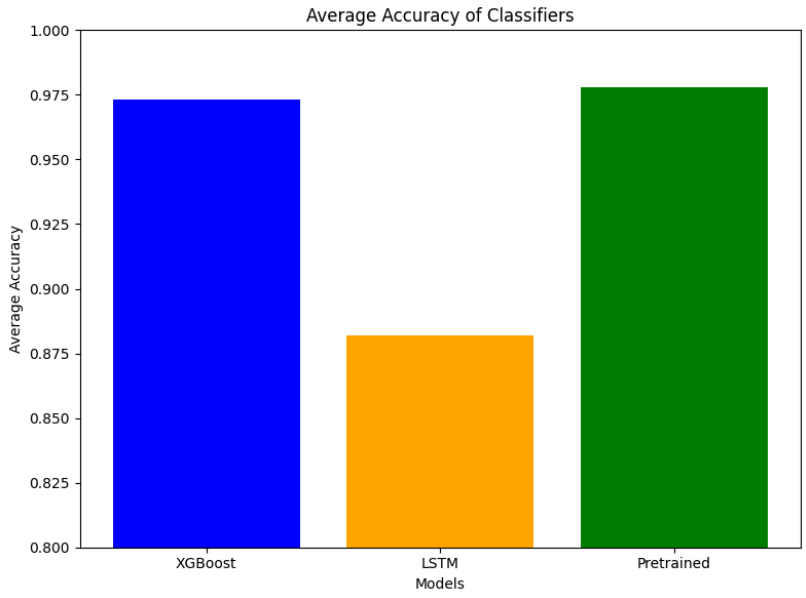
Pre-trained Classifier Test Accuracy

DistilBERT demonstrated the best performance overall, with exceptional results across all labels. This emphasizes the effectiveness of transformer-based architectures in capturing context and handling them even with imbalanced data.

	Label	Accuracy
1	toxic	0.9637474542
2	severe_toxic	0.9900047
3	obscene	0.9814193953
4	threat	0.99699201
5	insult	0.9640921197
6	identity_hate	0.9916340279

Conclusion

The bar graph shows the average accuracy of the three classifiers. The pretrained model outperformed the other classifiers.



This project highlights the outstanding result of transformer-based architectures for text classification tasks. In addition, we would like to work on exploring ensemble methods combining XGBoost and pretrained models to further boost performance.

Contributions

Ami and Dushyant worked on exploratory data analysis and model training. Rida worked on writing the report and visualizations.

References

"What Is XGBoost?" *NVIDIA Data Science Glossary*, www.nvidia.com/en-us/glossary/xgboost/. Accessed 13 Dec. 2024.