Toxic Comment Classification

Group Members: Apoorva Surendra Malemath, Ashir Bhavin Mehta

1. Introduction

Toxic comments have become a pervasive issue in today's online platforms and forums, posing a threat to the safety and well-being of users. Toxic comments can cause emotional harm, distress, and even lead to bullying or harassment. More than 4 in 10 Americans have experienced online harassment. [1] Toxicity may range from overt forms like abusive language and bullying to subtler methods. This behavior infiltrates virtually every corner of the internet, but can be especially pervasive in gaming, news, blogging, and social media. Safeguarding healthy discourse online ensures everyone has the ability to participate online. Toxic comments can be detrimental to a brand's reputation and discourage users from engaging with a company's online platforms. This process cannot be completed manually as the level of toxicity is relative to each person and moderating comments can be a time-consuming task, especially for large online communities. Thus, there is a need for automated techniques to detect toxic comments quickly. Natural Language Processing (NLP) has seen significant advancements in recent years, making it an ideal candidate for toxic comment classification. Previously, In the paper *Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning* [2], the authors compared performance of models that used no augmentation, unique words augmentation and synonym replacement. The proposed solution is an ensemble of three models: convolutional neural network (CNN), bidirectional long short-term memory (LSTM) and bidirectional gated recurrent units (GRU). Moreover, in Toxic Comment Classification [3], the authors demonstrated that the use of LSTM had a 20% higher true positive rate than the well-known Naive Bayes method. Thus, in this project we wish to draw inspiration from the previous work and explore various NLP models and techniques to develop an accurate and efficient model for toxic comment classification, contributing to research in NLP and providing a practical solution for online content moderation. The problem statement is floated on Kaggle by Jigsaw, Conversation AI. [4] It is a unit within Google that explores threats to open societies, and builds technology that inspires scalable solutions.

2. Problem Statement

We aim to develop a classification model that accurately classifies toxicity into 6 classes using an automated approach. This helps make the process quicker and less labor intensive for large online communities, and eliminates the human bias factor.

3. Dataset Overview

The dataset is sourced from Wikipedia comments in English with an average comment length of 384 characters. [4] The dataset comprises 159,571 unique comments. There are 6 types of toxicities namely, Toxic, Severe Toxic, Obscene, Threat, Insult and Identity Hate. From Fig 1 it is seen that we have imbalanced classes. Here one comment can have more than one toxicity type. 89% of the comments are clean comments, 4% comments have one toxicity associated and 0.02% of the comments have all toxicities.

From Fig 2, we can see the comment length distribution i.e. the average comment length is 394 characters, and majority of the comments have comment length in the range of 0 to 500 characters.

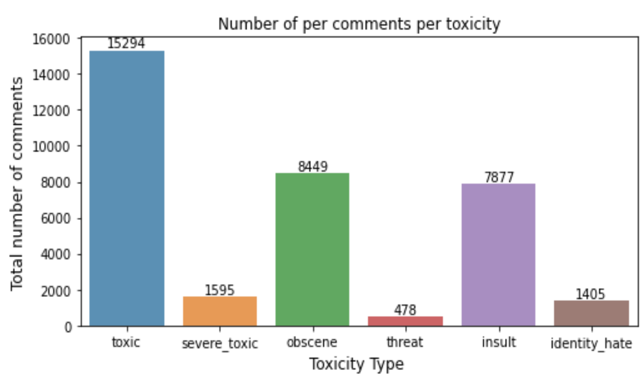

Fig 1: Distribution of Toxicity Types

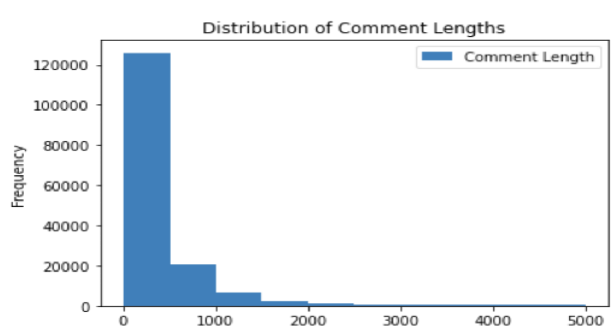

Fig 2: Trend in comment length

4. Methodology

In this section we discuss our methodology. Fig 3 depicts the elaborate workflow.

- **Low Risk Level**

  As a low-level risk level, we understood the dataset and focused on the overlap between types of toxicities. Looking at the correlation coefficients in Fig 4, we can see that Obscene and Insult have the highest correlation i.e. 0.74, followed by Obscene and Toxic with 0.68 and Insult and Toxic with 0.65. We then looked at the confusion matrix of toxic comments with the other classes, and we observed that Severe Toxic comments are always Toxic and can be referenced from Appendix Table 1. We then built word clouds for each class to help visualize the actual text data as seen in the Appendix Fig 5 as a preliminary analysis of most common words of each category and to visually compare the text distribution in different classes.

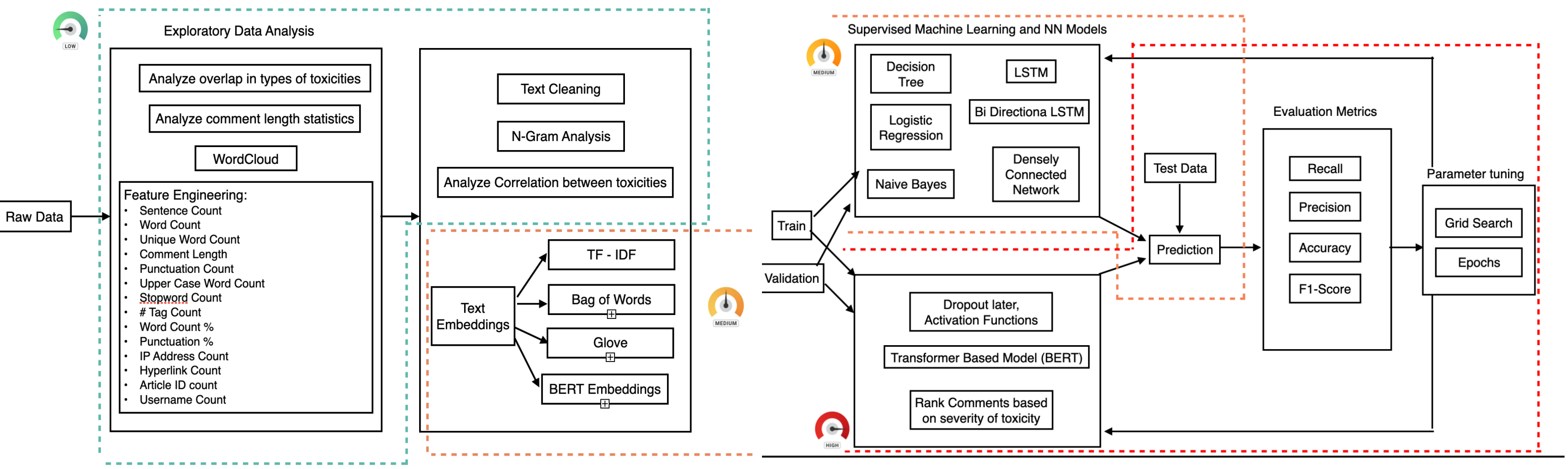

Fig 3: Implementation Flow Diagram

We then looked at the unique word count percentage across all the comments, and we observed that negative comments have fewer unique words in comparison to clean comments. From Appendix Fig 6, we can see that 60% of the comments with less than 40% unique word count are toxic in nature.

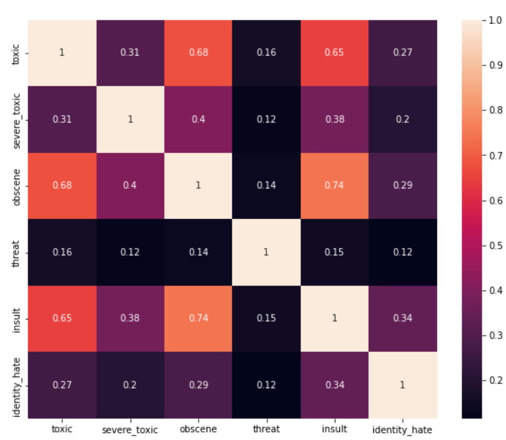

Fig 4: Correlation Coefficient between the target variables.

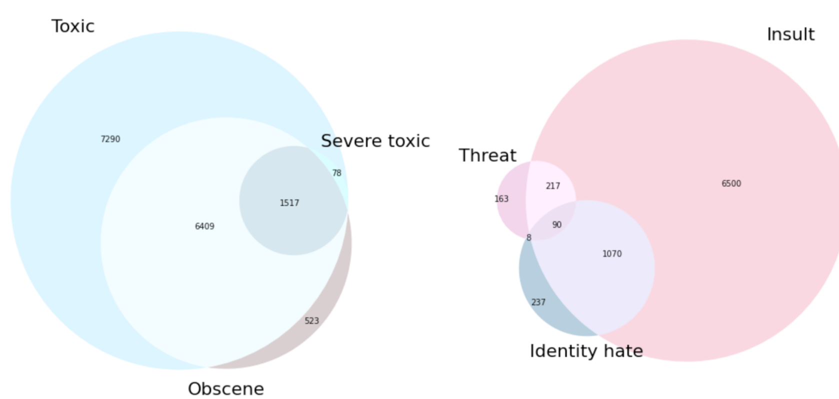

Fig 8: Venn Diagram to understand overlap between the classes

We then used venn diagrams as seen in Fig 8, to further understand the overlap between the classes, we built multiple diagrams with a unique combination of the toxic classes to understand the data better. Further, as we can see in Fig 9,generated bi-grams for each toxicity type and we observed that many Bi-Grams consisted of repetitive words. This validated our previous observation that the majority of the toxic comments have less than 40% unique words.

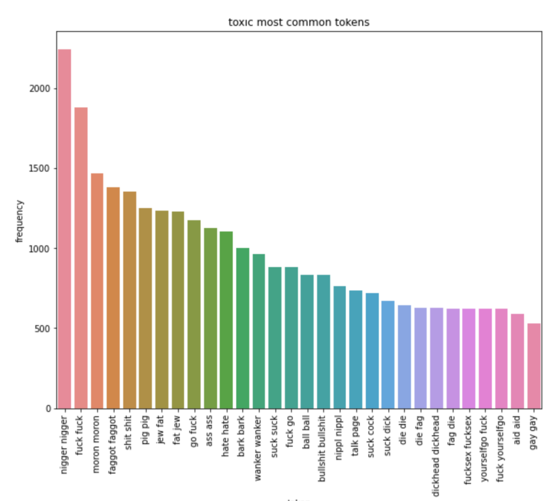

Fig 9: Bi-Gram frequencies in toxic comments

Feature Engineering: We generated 14 new features by picking up cues from the text before we clean it in order to preserve the essence of the data such as sentence count, word count, unique word count, text length, punctuation count, upper case count, stopword count, # Tag count, unique word count percent, punctuation percent, IP address count, hyperlink count, article id count and username mention count.

Text Cleaning: We then cleaned the text by converting to lowercase, removing line breaks, punctations, line breaks and stop words followed by stemming. Stemming is used to reduce the words to their base or root form, this helps to normalize the text and reduce inflectional forms of a word. It further standardizes the vocabulary being used.

We evaluated the above-mentioned by inspecting the plots and determining if they show expected trends and meet the initial hypothesis. As these tasks are exploratory in nature there is no direct validation metric to evaluate, we thus draw inferences from the data and this is exploratory in nature.

- **Medium Risk Level**
  At the medium risk level, we implemented different text embedding techniques like Term Frequency - Inverse Document Frequency (TF-IDF), Word-Vector (Continuous Bag of Words representation), GloVe embeddings and BERT Embedding. We then build model models such as Logistic Regression, Naive Bayes, Decision Trees, Dense Neural Network, LSTM and Bi-directional LSTM to predict the score for each toxicity, and determine the classes based on the score values. We compare the performance of Naive Bayes and LSTM, as in previous it was observed that LSTM performed better than Naive Bayes, however this was not the case here. We observed that Naive Bayes performed much better than LSTM. Also as we have multiple target columns, we individually fit each model for each of the individual classes. In addition we observed that one comment can belong to multiple classes, thus reassigning label values to have one target variable would not work in this case. We then compared the performance of the models using metrics like accuracy, precision, recall and F1-score. In this problem statement accuracy alone can be misleading as the classes are unbalanced, thus the additional metrics will help us to understand the model performance better. We will also dove deep into results to look closely at the metrics at each class level to help determine if the same classes continue to perform poorly across all models.

4.2.1 Results

Word Embedding	Model Name	Evalution Metrics	Toxic Class	Severe_toxic Class	Obscene Class	Threat Class	Insult Class	Identity_hate Class	Overall Average Metric
TF-IDF	Decision Tree	Recall	0.737	0.626	0.787	0.624	0.711	0.641	0.688
		Precision	0.740	0.630	0.784	0.606	0.715	0.634	0.685
		Accuracy	0.910	0.986	0.956	0.995	0.946	0.987	0.963
		F1	0.739	0.628	0.785	0.614	0.713	0.637	0.686
Word2Vec	Logistic Regression	Recall	0.513	0.513	0.505	0.5	0.504	0.5	0.506
		Precision	0.792	0.737	0.751	0.499	0.688	0.496	0.661
		Accuracy	0.905	0.99	0.946	0.997	0.95	0.991	0.963
		F1	0.501	0.523	0.497	0.499	0.495	0.498	0.502
Glove	LSTM	Recall	0.489	0.316	0.586	0.296	0.489	0.308	0.414
		Precision	0.842	0.378	0.866	0.484	0.703	0.502	0.629
		Accuracy	0.942	0.988	0.973	0.997	0.964	0.991	0.976
		F1	0.612	0.282	0.689	0.143	0.562	0.276	0.427
	BiDirectional LSTM	Recall	0.494	0.215	0.588	0.276	0.519	0.200	0.382
		Precision	0.837	0.479	0.853	0.462	0.683	0.591	0.651
		Accuracy	0.942	0.990	0.972	0.997	0.964	0.992	0.976
		F1	0.614	0.228	0.685	0.133	0.577	0.202	0.407

Table 2: Model Evaluation (Medium Risk Level)

From the above table 2 we can see that across all the word embeddings method TF-IDF Embedding proved to be most efficient compared to Word2Vec and Glove Embedding. Moreover the Decision Tree Model was the best fit model with an average F1 Score across all the 6 classes is 0.68.

- **High Risk Level**
  At a high-risk level, we explored parameter tuning to help tune the models and compare results as seen in Table 2 and 3. We then built complex transformer based models such as BERT (Bidirectional Encoder Representations from Transformers). We tweaked the networks to various numbers of hidden layers, activation functions, and other hyperparameters.

From table 3 we can conclude that BERT embeddings used in a Sequential Neural Network with BERT Embeddings made of 3 hidden layers and 3 dropout layers using Relu activation function and sigmoid at the output layer.

We then ranked comments in order of severity of toxicity based on the score on our best performing model as mentioned above. We considered the sum of all 6 toxicity types, and observed majority were around 0 as the majority of the comments are clean the same can be seen in Fig 12 , and there were 200 comments that we ranked as highly toxic based on the threshold we considered on the basis of sum of toxicity score as seen in Fig 13.

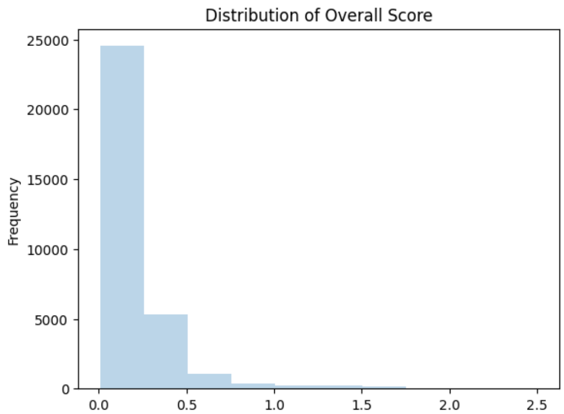

Fig 12: Distribution of Overall Score for all comments.

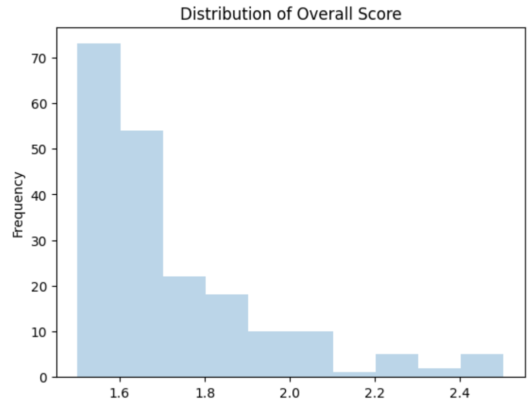

Fig 13: Distribution of Overall Score top 200 comments.

4.3.1 Results

Word Embedding	Model Name	Evalution Metrics	Toxic Class	Severe_toxic Class	Obscene Class	Threat Class	Insult Class	Identity_hate Class	Overall Average
BERT Embedding	Sequential	Recall	0.908	0.989	0.947	0.997	0.952	0.992	0.964
		Precision	0.888	0.978	0.937	0.994	0.933	0.983	0.952
		Accuracy	0.908	0.989	0.947	0.997	0.952	0.992	0.964
		F1	0.875	0.984	0.922	0.996	0.929	0.987	0.949
	BERT Model	Recall	0.838	0.986	0.907	0.997	0.908	0.988	0.937
		Precision	0.831	0.979	0.902	0.994	0.908	0.985	0.933
		Accuracy	0.838	0.986	0.907	0.997	0.908	0.988	0.937
		F1	0.835	0.983	0.904	0.996	0.908	0.987	0.935

Table 3: Model Evaluation (High Risk Level)

5. Conclusion

Building such a classification system that can predict the probability of different types of toxicities in comments could help the content moderators more efficiently and accurately identify and remove the harmful content from online platforms. Moreover, these systems can also provide analytics of the toxic comment which can help in designing policies and strategies in preventing online toxicity. Furthermore, analyzing these comments can also help in flagging users who repeatedly are making toxic comments. By doing so various platforms can take action to prevent such behavior and ensure a safer and more inclusive environment for all users. Moreover, from the results we conclude that the best performing models are the Sequential Model and the BERT Model with BERT Embeddings. BERT's transformer architecture enables it to pay attention to important parts of the text, which effectively process long sequences of text. Furthermore BERT embedding proved to be the best word embedding method as it is pre-trained on a large corpus of text data, that makes it more effective at capturing a wide range of linguistic patterns and thus this pre-training allows the model to perform well.

Please find the link to our code repository [5] and final presentation [6] under the references section.

References:

[1] The Toxicity Issue - https://jigsaw.google.com/the-current/toxicity/

[2] Mai Ibrahim; Marwan Torki; Nagwa El-Makky, *Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning*

[3] Sara Zaheri, Jeff Leath, David Stroud, *Toxic Comment Classification*

[4] Toxic Comment Classification Kaggle Dataset - https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data

[5] Code Repository: https://github.com/ashirm1999/Capstone-Project/tree/main/Phase%202

[6] Final Presentation: https://drive.google.com/file/d/1b4R1b0D_b1qZOXPw3X7vNsbuj7j-U6QW/view?usp=sharing

Appendix:

1. Exploratory Data Analysis:

- The below table summarizes the relationship between the toxic comment and every other toxicity type.

	severe_toxic		obscene		threat		insult		identity_hate	
	0	1	0	1	0	1	0	1	0	1
toxic										
0	144277	0	143754	523	144248	29	143744	533	144174	103
1	13699	1595	7368	7926	14845	449	7950	7344	13992	1302

Table 1 : Confusion matrix of Toxic comments with other classes

- The below image showcases the word cloud for a few classes. Here the words appearing frequently appear with a larger font.

Fig 5 : Word Clouds for Toxic and Threatening class.

- As we previously summarized the negative comments have fewer unique word count, we can further look at the distribution of unique word count in Fig 7. Majority of the comments falling above 60% unique word count are clean comments.

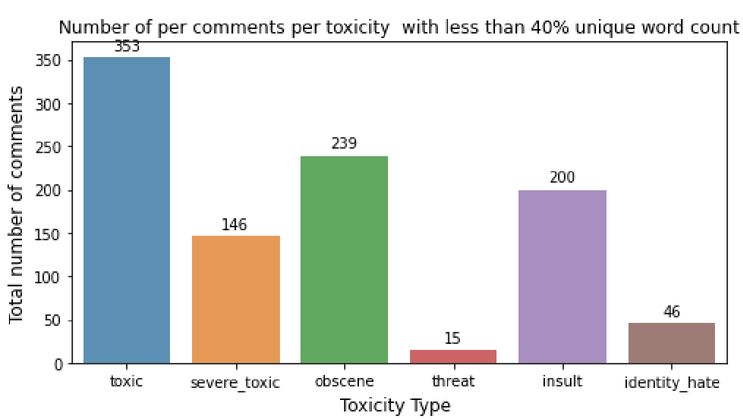

Fig 6: Comment count based on unique words across classes.

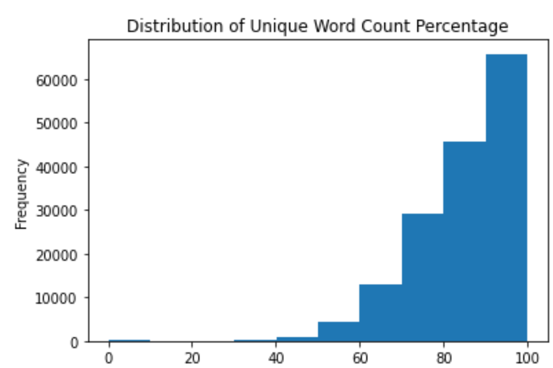

Fig 7: Distribution of comments based on unique word counts.

- From Fig 10, we drew inferences by looking at the median comment length, average comment length, minimum comment length and average number of unique words across the toxicity types. For instance, the average comment length of threat is the highest whereas the median comment length of clean comment is the highest. Further we could see that the average number of unique words are lower in negative comments. These insights were further helpful generating new features.

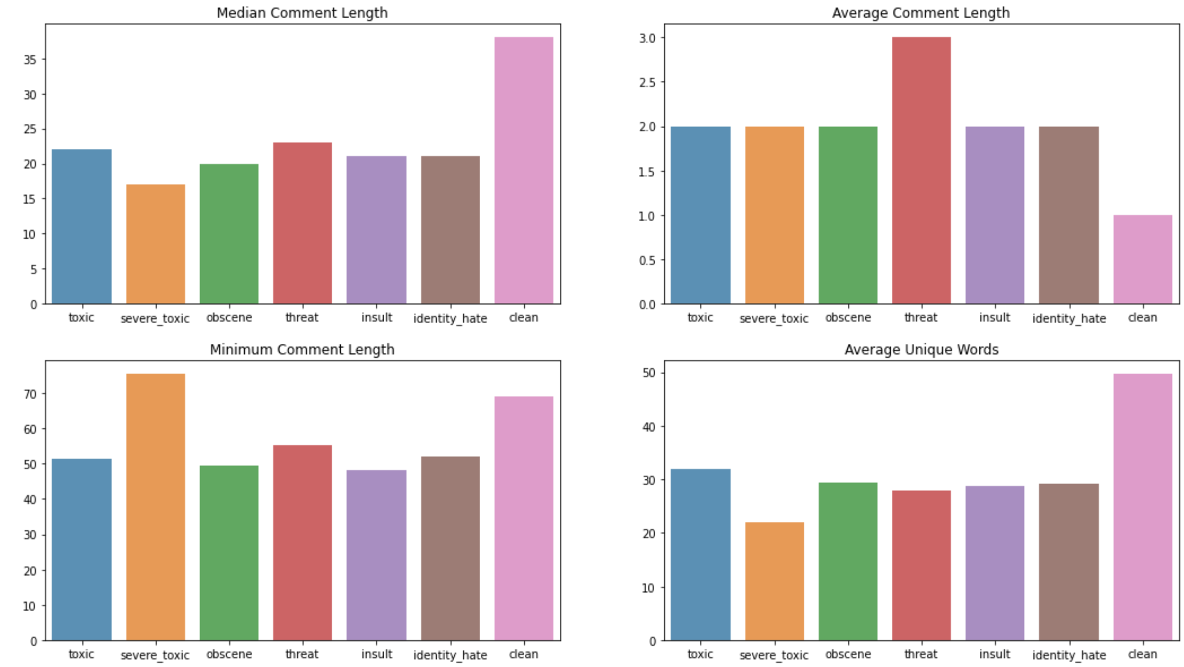

Fig 10: Comment length statistics across different toxicity types.

2. Excellent Failure

We observed that LSTM and Bi-Directional LSTM performed very poorly for the dataset, the same can be witnessed from the result table 4 in the appendix section.

Next, we experimented with the target columns i.e. we have 6 target classes, we assigned a new label based on the unique combination of labels that we see in these label values i.e. we will have 64 combinations. Our data consisted of 41 unique such combinations. In Fig 11 we can see the distribution of the new labels generated excluding the clean comments. The model performed poorly for Naive Bayes as seen in the results that are documented in table 4 . We see that Logistic Regression and Decision trees performed fairly well even in this scenario, however logically this approach might not always work well as this depends on the dataset being used and the relationship between the types of toxicities observed. For the dataset that we used, we had previously observed that there exists strong correlation between the toxicity types, thus it seems like our models seem to perform well as they are able to understand this correlation to make better predictions.

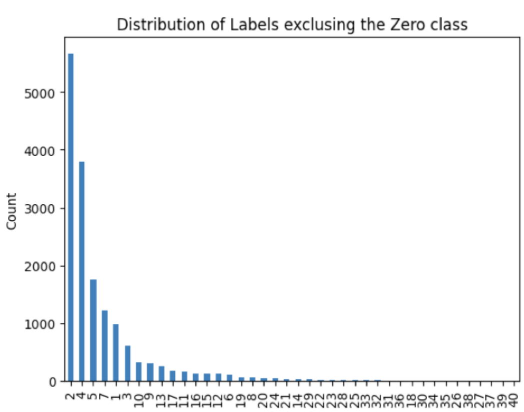

Fig 11: Distribution of new labels generated

Word Embedding	Model	Accuracy	Precision	Recall	F1-Score
TF-IDF	Logistic Regression	0.898	0.819	0.817	0.855
	Naive Bayes	0.355	0.891	0.355	0.505
	Decision Tree	0.867	0.864	0.867	0.866
BERT Embedding	Logistic Regression	0.897	0.816	0.897	0.856
	Naive Bayes	0.002	0.839	0.002	0.003
	Decision Tree	0.81	0.824	0.81	0.817

Table 4: Model Results (Excellent Failure)

3. Medium Risk Level Results

Word Embedding	Model Name	Evalution Metrics	Toxic Class	Severe_toxic Class	Obscene Class	Threat Class	Insult Class	Identity_hate Class	Overall Average Metric
TF-IDF	Decision Tree	Recall	0.737	0.626	0.787	0.624	0.711	0.641	0.688
		Precision	0.740	0.630	0.784	0.606	0.715	0.634	0.685
		Accuracy	0.910	0.986	0.956	0.995	0.946	0.987	0.963
		F1	0.739	0.628	0.785	0.614	0.713	0.637	0.686
	Logistic Regression	Recall	0.548	0.512	0.715	0.500	0.503	0.504	0.547
		Precision	0.901	0.636	0.933	0.499	0.675	0.591	0.706
		Accuracy	0.912	0.990	0.967	0.997	0.950	0.991	0.968
		F1	0.564	0.520	0.784	0.499	0.493	0.506	0.561
	Naive Bayes	Recall	0.611	0.594	0.622	0.588	0.619	0.639	0.612
		Precision	0.539	0.575	0.525	0.509	0.523	0.505	0.529
		Accuracy	0.505	0.981	0.507	0.970	0.509	0.597	0.678
		F1	0.430	0.584	0.397	0.511	0.395	0.387	0.451
Word2Vec	Decision Tree	Recall	0.565	0.546	0.553	0.514	0.553	0.52	0.542
		Precision	0.562	0.544	0.55	0.51	0.548	0.518	0.539
		Accuracy	0.844	0.981	0.906	0.993	0.91	0.982	0.936
		F1	0.563	0.545	0.551	0.512	0.55	0.519	0.540
	Logistic Regression	Recall	0.513	0.513	0.505	0.5	0.504	0.5	0.506
		Precision	0.792	0.737	0.751	0.499	0.688	0.496	0.661
		Accuracy	0.905	0.99	0.946	0.997	0.95	0.991	0.963
		F1	0.501	0.523	0.497	0.499	0.495	0.498	0.502
	Naive Bayes	Recall	0.597	0.584	0.606	0.571	0.606	0.617	0.597
		Precision	0.534	0.567	0.522	0.507	0.52	0.504	0.526
		Accuracy	0.491	0.981	0.492	0.969	0.495	0.579	0.668
		F1	0.42	0.574	0.387	0.507	0.386	0.379	0.442
BERT Embedding	Decision Tree	Recall	0.842	0.980	0.904	0.994	0.909	0.982	0.935
		Precision	0.850	0.981	0.912	0.995	0.917	0.984	0.939
		Accuracy	0.842	0.980	0.904	0.994	0.909	0.982	0.935
		F1	0.846	0.980	0.908	0.994	0.913	0.983	0.937
	Logistic Regression	Recall	0.904	0.989	0.948	0.997	0.951	0.992	0.964
		Precision	0.884	0.985	0.931	0.994	0.930	0.983	0.951
		Accuracy	0.904	0.989	0.948	0.997	0.951	0.992	0.964
		F1	0.862	0.984	0.923	0.996	0.929	0.987	0.947
	Naive Bayes	Recall	0.387	0.516	0.369	0.319	0.347	0.339	0.380
		Precision	0.865	0.984	0.924	0.995	0.930	0.986	0.948
		Accuracy	0.387	0.516	0.369	0.319	0.347	0.339	0.380
		F1	0.473	0.670	0.488	0.481	0.466	0.497	0.513

Table 5: Model Evaluation (Medium Risk Level)

4. High Risk Level Results

Word Embedding	Model Name	Evalution Metrics	Toxic Class	Severe_toxic Class	Obscene Class	Threat Class	Insult Class	Identity_hate Class	Overall Average Metric
TF-IDF	LSTM	Recall	0.108	0	0.024	0	0.009	0	0.024
		Precision	0.643	0	0.504	0	0.5	0	0.275
		Accuracy	0.908	0.99	0.946	0.997	0.95	0.991	0.964
		F1	0.178	0	0.044	0	0.015	0	0.040
	BiDirectional LSTM	Recall	0.114	0	0.008	0	0.011	0	0.024
		Precision	0.62	0	0.524	0	0.475	0	0.270
		Accuracy	0.908	0.99	0.946	0.997	0.95	0.991	0.964
		F1	0.187	0	0.014	0	0.018	0	0.040
Word2Vec	LSTM	Recall	0.115	0	0.017	0	0.01	0	0.024
		Precision	0.62	0	0.57	0	0.429	0	0.270
		Accuracy	0.908	0.99	0.946	0.997	0.95	0.991	0.964
		F1	0.189	0	0.032	0	0.017	0	0.040
	BiDirectional LSTM	Recall	0.171	0.011	0.018	0	0.006	0	0.034
		Precision	0.536	0.857	0.573	0	0.63	0	0.433
		Accuracy	0.906	0.99	0.946	0.997	0.95	0.991	0.963
		F1	0.254	0.014	0.033	0	0.011	0	0.052
Glove	LSTM	Recall	0.489	0.316	0.586	0.296	0.489	0.308	0.414
		Precision	0.842	0.378	0.866	0.484	0.703	0.502	0.629
		Accuracy	0.942	0.988	0.973	0.997	0.964	0.991	0.976
		F1	0.612	0.282	0.689	0.143	0.562	0.276	0.427
	BiDirectional LSTM	Recall	0.494	0.215	0.588	0.276	0.519	0.200	0.382
		Precision	0.837	0.479	0.853	0.462	0.683	0.591	0.651
		Accuracy	0.942	0.990	0.972	0.997	0.964	0.992	0.976
		F1	0.614	0.228	0.685	0.133	0.577	0.202	0.407
BERT Embedding	Sequential	Recall	0.908	0.989	0.947	0.997	0.952	0.992	0.964
		Precision	0.888	0.978	0.937	0.994	0.933	0.983	0.952
		Accuracy	0.908	0.989	0.947	0.997	0.952	0.992	0.964
		F1	0.875	0.984	0.922	0.996	0.929	0.987	0.949
	LSTM	Recall	0.117	0.000	0.014	0.000	0.010	0.000	0.024
		Precision	0.651	0.000	0.571	0.000	0.533	0.000	0.293
		Accuracy	0.908	0.989	0.948	0.997	0.952	0.992	0.964
		F1	0.175	0.000	0.020	0.000	0.012	0.000	0.034
	BERT Model	Recall	0.838	0.986	0.907	0.997	0.908	0.988	0.937
		Precision	0.831	0.979	0.902	0.994	0.908	0.985	0.933
		Accuracy	0.838	0.986	0.907	0.997	0.908	0.988	0.937
		F1	0.835	0.983	0.904	0.996	0.908	0.987	0.935

Table 6: Model Evaluation (High Risk Level)