Project Machine Learning
— Milestone 1 —

[Imene Ben Ammar, Julian Dobler, Yannik Queisler]

November 24, 2024

Abstract

This project aims to fine-tune BERT for toxic comment classification, addressing the challenges of identifying harmful online content. Baseline methods such as logistic regression, support vector machines, and random forest were implemented and evaluated on the Civil Comments dataset as part of this milestone. These methods served as a reference for understanding the dataset and establishing performance benchmarks for BERT. Key findings revealed that logistic regression with preprocessing performed best among the baselines, achieving high-weighted F1 and AUC-ROC scores. However, precision and recall for the minority toxic class (C1) remained suboptimal. The dataset's significant class imbalance was identified as the primary challenge, emphasizing the need for advanced methods to improve performance. Future work will involve extending baseline evaluations to additional datasets (e.g., Jigsaw Toxicity and SST-2) and fine-tuning BERT for comparative analysis. The code for this project is publicly available on GitHub[1].

Contents

1 Introduction **1**

2 Datasets **1**
2.1 Civil Comments 1
2.2 Jigsaw Toxic Comment 1
2.3 Stanford Sentiment Treebank 1
2.4 Remarks 2
2.5 Data characteristics 2
2.6 Feature extraction 3
2.7 Typical queries 4

3 Baseline methods **5**

4 BERT **5**
4.1 Model 6
4.2 Prior approaches 6
4.3 Solutions 6
4.4 Limitations 6

5 Results **6**
5.1 Evaluation metrics 7
5.2 Findings 8

6 Discussion **9**
6.1 Task complexity 9
6.2 Future directions 9

A Visualizations

[1] `https://github.com/devWhyqueue/pml-bert`

1 Introduction

The exponential growth of social media platforms has led to unprecedented levels of online engagement, but it has also introduced significant challenges, particularly concerning the prevalence of toxic comments. Toxic comments have become a major issue, characterized by rudeness, disrespect, and a tendency to disrupt or drive participants out of discussions. [1]

Their presence exacerbates the problem of cyberbullying, with studies showing that approximately 15% of teens have experienced online bullying, which is strongly linked to psychological problems such as depression and anxiety. [2]

Effective moderation is essential, but manual approaches are not scalable given the sheer volume of user-generated content. This makes automated toxic comment classification a critical task for ensuring safer and more inclusive online spaces.

Machine learning (ML) and deep learning methods have emerged as promising tools to tackle this problem. Classical approaches like naive Bayes and support vector machines (SVMs) have been employed as benchmarks, while more sophisticated models such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and BERT have demonstrated considerable success. [1]

The focus of this report is on achieving the first milestone in this project, which involves implementing baseline methods for toxic comment classification. This milestone is designed to familiarize us with the datasets, establish performance baselines, create a prototype BERT model, and identify suitable evaluation metrics. The subsequent sections of this report will address these objectives in the order outlined, providing a foundation for further advancements in the project.

2 Datasets

In this report, we focus on three key datasets that are pivotal for understanding and evaluating models in toxic comment classification. These datasets provide a diverse range of annotations and textual contexts that allow us to benchmark various methods and analyze their effectiveness.

2.1 Civil Comments

The Civil Comments dataset comprises 1.8 million crowdsourced annotations across 280,000 comments sourced from the Civil Comments platform. Each comment is annotated with labels for toxic behavior, including subcategories such as obscene language, threats, and identity-based hate.

Labels are real numbers between zero and one. This means, that for classification tasks, a threshold is required to distinguish between positive and negative classifications. We chose the threshold to be 0.5, as this is a common practice with the Civil Comments dataset in the literature, ensuring consistency and comparability with prior studies. [3, 4]

Additionally, the dataset captures a variety of identity attributes, making it valuable for studying unintended biases in toxicity detection models. Given its comprehensive labeling and real-world application, this dataset is a prime candidate for developing and testing toxicity classification models. [3]

2.2 Jigsaw Toxic Comment

This dataset, released as part of a series of Kaggle competitions, includes comments from Wikipedia labeled for toxicity and specific subcategories like severe toxicity, obscene language, threats, insults, and identity-based hate. Each comment can belong to multiple subclasses at the same time. It has been widely adopted in the field and is frequently used in toxic comment classification research, as highlighted by Andročec in their systematic review. Its extensive use ensures comparability with previous studies and allows us to evaluate model performance in a well-established benchmark setting. [1, 5]

2.3 Stanford Sentiment Treebank

While primarily designed for sentiment analysis, the SST-2 dataset includes binary labels indicating whether a sentence from movie reviews is positive or negative. Though it does not directly address toxicity detection, SST-2 provides a relevant benchmark for binary text classification. Additionally, SST-2 was included in the original BERT paper, allowing for a direct comparison between our replication and the original results. Despite its conceptual differences from the toxicity detection domain, its inclusion ensures a broader perspective on our model's capabilities and alignment with existing literature.

2.4 Remarks

While our primary focus is on the datasets mentioned above, numerous other datasets are available for toxic comment classification. These include HateXplain, the Offensive Language Identification Dataset (OLID), the Hate Speech Offensive (HSOL) Dataset, and various Twitter Hate Speech datasets.

For the initial milestone of this project, we implement and evaluate baseline methods on the Civil Comments dataset. This allows us to establish a strong foundation for toxicity classification before integrating and analyzing models on the Jigsaw and SST-2 datasets in later phases.

2.5 Data characteristics

To better understand the structure and content of the datasets used in this project, several key information were extracted. Furthermore, all visualizations can be found in the appendix A of this report.

First of all, invalid labels as well as null values were checked in all datasets and filtered out.

Dataset	Invalid Labels	Null Values	Total Rows
Jigsaw Toxicity	0	0	223,549
Civil Comments	0	0	1,999,514
SST-2	1821	0	68,221

Table 1: Invalid labels and null values in datasets.

2.5.1 Toxicity subtypes and sentiment analysis

1. **Toxicity subtype frequency:** The frequency of each toxicity subtype was visualized using bar plots. To this aim, the occurrences of comments labeled as toxic for each of the subtypes were counted. This provides insight into the prevalence of different types of toxicity within the datasets as shown in figure 1.
2. **Toxicity subtypes/sentiment percentage:** The percentage of comments falling under each toxicity subtype and sentiment label was visualized using pie charts as shown in figure 3. The analysis considered any comment that was classified under one or more toxicity subtypes as toxic. This provided an overview of the commonness of toxic comments compared to non-toxic ones as well as positive and negative sentiments in the dataset, highlighting the fact that all datasets suffer from class imbalance.

2.5.2 Textual analysis

1. **Word clouds:** Separate word clouds were generated to visually represent the most frequently occurring terms in comments labeled as toxic or non-toxic, as well as positive and negative sentiments. Comments categorized as toxic included those marked with any toxicity subtype label, while sentiment labels were categorized into positive or negative. An example of the Jigsaw Toxicity dataset is shown in figure 4.
2. **Word Frequency:** The most common words associated with each toxicity subtype/sentiment were identified and visualized using bar plots as shown in figure 2. As in the Civil Comments dataset, the toxicity labels are not binary but instead have continuous values to represent different degrees of toxicity, a threshold was applied to filter comments based on their toxicity score, selecting those that exceeded a threshold of 0.8. The top N (10) most frequent words for each toxicity label/sentiment were extracted by tokenizing and cleaning the text of comments that met the threshold. The resulting word frequencies were then plotted to show the most common terms for each category.
3. **Comment length:** To identify potential patterns between comment length and toxicity/sentiment in the datasets, the lengths of comments were analyzed. This included plotting histograms of the distribution of comment lengths, box plots to highlight the differences in comment length across the different categories (toxic vs. non-toxic and positive vs. negative), and bar plots to show the average comment lengths per category (see figure 5).

4. **Toxicity subtypes vs. comment length:** Scatter plots were created to explore the relationship between comment length and various toxicity subtypes in the Jigsaw and Civil Comments datasets. For each toxicity label, a scatter plot was generated with comment length on the x-axis and the toxicity level on the y-axis. These plots allow for the visualization of potential correlations between the length of comments and the increase in toxicity level and therefore provide insights into whether longer comments tend to be more toxic or if there is no clear relationship between comment length and toxicity. This plot is more useful in the Civil Comments datasets, as the toxicity values are real numbers between 0 and 1. Figure 6 illustrates an example of the comment length for the subclass threat of the Civil Comments dataset. It suggests that higher Threat levels are mainly linked to short comments, not exceeding 250.

2.6 Feature extraction

Multiple approaches exist to transform textual data into numerical vectors. They can be classified into several categories:

- **Frequency-based:** These methods rely on word or token frequencies to generate feature representations. Examples include BoW and TF-IDF. [6]
- **Embedding-based:** These techniques involve representing words, phrases, or documents as dense, low-dimensional vectors, typically learned from large corpora. Examples include word-level embeddings like Word2Vec, GloVe, and FastText, sentence-level embeddings like universal sentence encoder, and pre-trained contextual embeddings like BERT and GPT. [7, 8, 9, 10, 11, 12]
- **Statistical:** These methods use statistical models to generate feature representations. Examples include latent Dirichlet allocation (LDA) and latent semantic analysis (LSA). [13, 14]
- **Symbolic and graph-based:** These methods capture relationships between words or concepts based on symbolic or structural properties. Examples include n-grams and graph representations like DeepWalk and Node2Vec. [15, 16]

We selected TF-IDF as our vectorization method for the baseline because it aligns well with our goal of establishing a straightforward benchmark using classical machine learning algorithms. As a classical and widely accepted approach, TF-IDF is computationally efficient and easy to integrate into traditional models like logistic regression or SVMs, making it an ideal choice for initial experimentation. This rationale echoes Salton's foundational work in automatic text processing, which highlights the efficacy of frequency-based methods like TF-IDF for text analysis. By choosing TF-IDF, we ensure simplicity and reproducibility while focusing on baseline performance. [17]

In the case of BERT, embeddings are learned. As this project does not include training but only fine-tuning the model, we download the publicly available model weights. Note, that the BERT model cannot handle different shapes of features as it is a transformer model. Therefore, techniques like padding and truncating must be applied.

2.6.1 Method

Textual inputs are first (optionally) preprocessed and then transformed into numerical vectors using BoW and TF-IDF techniques. Finally, datasets are resampled as the toxic comment classification datasets show significant class imbalance.

Bag of Words is a simple vectorization technique that represents text by counting the occurrence of words in a document. It creates a vocabulary of unique words and represents each document as a vector, disregarding word order. While simplistic, it is effective in capturing word frequency. [6]

Term frequency-inverse document frequency measures the importance of a word within a document relative to a corpus. It combines term frequency (TF) and inverse document frequency (IDF) to emphasize meaningful words while downweighting common ones. [6]

2.6.2 Preprocessing

Before transforming textual data into numerical representations, we (optionally) applied a series of preprocessing steps to clean and standardize the data:

1. **Lowercasing**: Converts all text to lowercase for consistency.

2. **Removing noise**: Eliminates URLs, mentions, hashtags, numbers, and punctuation that do not carry semantic meaning.

3. **Tokenization**: Splits text into individual tokens (words).

4. **Filtering**: Removes non-alphabetic tokens and stop words (e.g., “and,” “the”).

5. **Lemmatization**: Reduces words to their base form (e.g., “running” $\rightarrow$ “run”) to consolidate variations.

Similar preprocessing steps are also used in [2, 18].

2.6.3 Resampling

The Civil Comments and the Jigsaw Toxicity datasets have an imbalanced class distribution, with significantly fewer positive samples compared to negative samples. To address this, we used a resampling technique to create a more balanced dataset while ensuring the minority class (positive samples) is fully represented.

The resampling approach involves the following steps:

1. **Class identification**: Identify the indices of samples belonging to the positive and negative classes based on their labels.

2. **Minority class retention**: Retain all samples from the minority class (positive samples) to ensure complete representation.

3. **Majority class resampling**: From the majority class (negative samples), randomly sample a subset based on the desired proportion between positive and negative classes.

4. **Balanced subset creation**: Combine the retained positive samples and the resampled negative samples to form a balanced dataset.

This method ensures that the positive class is not underrepresented and that the desired class ratio is achieved, reducing the bias caused by class imbalance.

Resampling techniques are commonly employed in machine learning to handle imbalanced datasets. In this implementation, our approach can be categorized as undersampling the majority class, a straightforward and effective method for balancing datasets. More complex techniques exist like the Synthetic Minority Over-sampling Technique (SMOTE), which generates synthetic samples for the minority class but were discarded to keep the baseline simple. [19]

2.6.4 Feature intuition

In the context of toxic comment classification, good features should effectively capture the semantic and syntactic cues indicative of toxic behavior while minimizing irrelevant noise. Such features might include specific words, phrases, or patterns commonly associated with toxic language, such as offensive terms, negative sentiment indicators, or aggressive expressions. Additionally, good features should distinguish subtle differences between toxicity and non-toxicity, such as distinguishing between sarcastic humor and outright hostility. Contextual relationships between words are also essential; for example, the phrase ”not bad” has a different sentiment than the isolated word ”bad.” Hence, features that preserve context, capture linguistic nuances, and weigh terms based on their relevance within the corpus (e.g., through TF-IDF weighting) are well-suited for this task.

2.7 Typical queries

Toxic comment classification aims to identify harmful or offensive language in text. Understanding the types of questions humans ask about datasets like Civil Comments and Jigsaw Toxicity, as well as what ML can answer, provides critical context for this task.

2.7.1 Human questions

Humans typically seek to understand the data's structure and implications:

- **Data characteristics:** What types of toxicity (e.g., hate speech, profanity) are included? How balanced are the classes (toxic vs. non-toxic)?
- **Annotations:** How were labels assigned? Were multiple annotators involved, and how consistent were they?
- **Bias and ethics:** Does the dataset reflect demographic or cultural biases? What are the ethical implications of using it?

2.7.2 ML-answerable questions

ML models can address specific patterns and predictions:

- **Classification:** Is a comment toxic? If so, what type of toxicity does it exhibit?
- **Quantitative insights:** What patterns (e.g., word usage) correlate with toxicity? How well does the model generalize to unseen data?
- **Bias detection:** Are there biases in toxicity predictions across demographic groups?
- **Error analysis:** What are common false positives and negatives?

While ML can predict toxicity and detect patterns, it struggles with context-dependent nuances (e.g., sarcasm) and cannot address ethical concerns. Human interpretation remains essential for understanding intent and ensuring responsible application.

3 Baseline methods

For the first milestone, we implemented a baseline for the Civil Comments dataset. Since it poses a binary classification problem, we selected several popular machine learning models for this task: Multinomial naive Bayes, linear support vector machine, logistic regression, and random forest classifier. These models are well-known for their performance in text classification tasks. [20]

Multinomial naive Bayes is a probabilistic classifier rooted in Bayes' theorem. Particularly effective for text classification tasks, it relies on feature vectors representing frequencies or counts of words. By assuming that features are conditionally independent, this method simplifies computation and performs well with BoW representations. [21]

Linear support vector classifier, a model based on support vector machines (SVM), seeks the optimal hyperplane to separate data into two classes. Its linearity and efficiency make it a powerful choice for large-scale text datasets. [22]

Logistic regression predicts the probability of a binary outcome by employing the sigmoid function to map predictions to probabilities. Its simplicity and interpretability have made it a staple for text classification tasks. [23]

Random forest, an ensemble learning technique, combines multiple decision trees to enhance predictive accuracy. By averaging results from individual trees, it reduces overfitting and improves generalization. [24]

The models, preprocessing steps, and resampling parameters were optimized using grid search, a simple approach to hyperparameter tuning. Grid search exhaustively searches through a manually specified subset of hyperparameters to determine the best combination for a model.

4 BERT

As part of the first milestone of this project, we implemented a prototype of the BERT-base model to address the task of toxic comment classification as described in the literature. [11]

While the baseline methods will be evaluated in this milestone to provide a foundational benchmark, evaluation of BERT and comparison with the mentioned baselines will be reserved for future milestones.

4.1 Model

The BERT-base model consists of a transformer-based architecture with 12 encoder layers, each featuring multi-head self-attention with 12 attention heads, a hidden size of 768, and a feed-forward network with an intermediate size of 3072. This implementation can be viewed on our GitHub repository. [25]

Its architecture is designed to capture deep contextual relationships in input sequences. For our implementation, we adopted the single-sentence classification approach, where the input comment is tokenized and augmented with a special [CLS] token, which serves as a pooled representation for the entire input sequence, which means that it contains all the necessary information of the sentence for classification. [11]

A classification head was added on top of the pre-trained BERT model. This head is a fully connected layer applied to the [CLS] token output from the final transformer layer, which output will serve for binary classification (toxic or non-toxic). To leverage prior knowledge, we initialized the model with pre-trained weights from HuggingFace and we will proceed to train the fine-tuning layer for the downstream task.

This architecture should enable the BERT model to achieve robust contextual understanding, and its performance will be compared against the baseline methods to evaluate its effectiveness in toxic comment classification.

4.2 Prior approaches

Historically, researchers applied classical machine learning approaches, such as SVMs and logistic regression, which relied on feature extraction methods like bag-of-words and TF-IDF. [26, 27]

While these methods provided early advancements, they were inherently limited in capturing semantic meaning and contextual dependencies within text. The introduction of word embeddings, such as Word2Vec and GloVe, alongside deep learning models like RNNs, LSTMs, and CNNs, marked significant progress. [2, 28, 29, 30]

These methods leverage pre-trained embeddings to incorporate some level of semantic understanding but remain to face challenges in effectively handling polysemy, capturing long-range dependencies, and understanding complex syntactic structures. These limitations resulted in less accurate performance in nuanced tasks like toxic comment detection. [11, 31, 32]

4.3 Solutions

The introduction of BERT brought transformative changes to NLP and addressed many of the limitations of earlier approaches. Prior transformer-based models used unidirectional processing, which limited the contextual understanding of a sentence. BERT's Transformer architecture enables bidirectional processing of text, allowing the model to consider both left and right contexts simultaneously. This capability provides BERT with a better understanding of language compared to traditional unidirectional models. Furthermore, BERT leverages pre-training on extensive corpora and fine-tuning on specific datasets, facilitating transfer learning and significantly improving performance on tasks such as toxic comment classification. [11, 33, 34]

4.4 Limitations

Despite its notable strengths, BERT is not without limitations. One primary concern is its computational intensity. Fine-tuning and deploying BERT requires substantial computational resources, which can be challenging in resource-constrained environments. Additionally, while BERT is pre-trained on extensive corpora, its performance in toxic comment classification heavily depends on the quality and diversity of the fine-tuning dataset. Biases present in training data can result in biased predictions, which is a critical challenge in fairness-sensitive tasks. Moreover, BERT's complexity often leads to overfitting when trained on small datasets, making it less effective in scenarios with limited labeled data. Another challenge arises from domain-specific language, such as internet slang and rapidly evolving toxic terms, which may require additional pre-training or customized tokenizers to ensure effective detection. [35, 36]

5 Results

This section presents the evaluation of our baselines' performances on the Civil Comments data set. The evaluation is conducted using multiple standard metrics to assess its effectiveness in classifying toxic and non-toxic comments. Note that this dataset poses a binary classification task. Therefore, category differences were no issue, and adaption of classic binary validation metrics was not necessary.

We include the metrics accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC). These metrics are widely used in the literature, allowing us to ensure that our results are comparable with existing studies. [1, 4, 37]

The results highlight the method's strengths and provide insights into areas requiring further improvement.

5.1 Evaluation metrics

The metrics are defined as follows.

Accuracy measures the proportion of all classification instances that are classified correctly:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{1}$$

Where TP, TN, FP, and FN represent the number of true positives, true negatives, false positives, and false negatives, respectively.

Precision indicates the proportion of predicted positive instances that are positive:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2}$$

Recall measures the amount of correctly positive classified instances proportionate to the true amount of positive classifications:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{3}$$

F1-score represents the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{4}$$

Finally, the area under the receiver operating characteristic curve (**AUC-ROC**) measures the ability of the model to distinguish between classes across all classification thresholds. It is computed as:

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR}), \tag{5}$$

where TPR (true positive rate) and FPR (false positive rate) are defined as:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{6}$$

We evaluate the performances using precision, recall, and F1-score specifically for the toxic class (Class 1) because it represents the primary focus in toxic comment classification. The ability to correctly identify and classify toxic comments is critical, as these instances typically have a disproportionate impact on user experience and platform safety. By focusing on these metrics for Class 1, we ensure the evaluation reflects the model's effectiveness in addressing the key challenge of identifying toxicity.

Additionally, we include macro-averaged F1 and weighted-averaged F1 scores to provide a holistic evaluation across all comment classes. The macro-averaged F1 treats all classes equally, offering insights into how well the model performs across all types of comments, regardless of class imbalance. The weighted-averaged F1 adjusts for class distribution, ensuring that the evaluation accounts for the prevalence of non-toxic comments, which are the majority class.

Finally, overall accuracy is reported as a general performance metric for completeness, though its limitations in imbalanced datasets necessitate reliance on the other metrics for a more comprehensive understanding of the model's ability to detect toxic comments.

Table 2: Classifier performance on the Civil Comments validation dataset

Method	**Prep.**	**Pos. Prop.**	**Precision (C1)**	**Recall (C1)**	**F1 (C1)**	**Macro F1**	**Weighted F1**	**Acc.**	**AUC ROC**
Naïve Bayes	True	0.1	**0.86**	0.15	0.25	0.61	0.93	0.95	0.88
		0.25	0.58	0.39	0.47	0.72	0.94	0.95	0.88
		0.5	0.14	**0.85**	0.25	0.53	0.77	0.69	0.86
	False	0.1	0.83	0.18	0.29	0.63	0.93	0.95	0.87
		0.25	0.52	0.44	0.48	0.73	0.94	0.94	0.87
		0.5	0.14	**0.85**	0.24	0.52	0.76	0.68	0.86
Support Vector Machine	True	0.1	0.52	0.60	0.56	0.76	0.95	0.94	0.89
		0.25	0.37	0.72	0.49	0.72	0.92	0.91	0.90
		0.5	0.21	0.82	0.34	0.61	0.86	0.81	0.89
	False	0.1	0.51	0.59	0.55	0.76	0.94	0.94	0.88
		0.25	0.38	0.68	0.49	0.72	0.93	0.92	0.89
		0.5	0.25	0.76	0.37	0.64	0.88	0.85	0.88
Logistic Regression	True	0.1	0.67	0.43	0.52	**0.75**	**0.95**	**0.95**	**0.90**
		0.25	0.50	0.55	**0.53**	**0.75**	0.94	0.94	**0.90**
		0.5	0.22	0.78	0.34	0.62	0.87	0.82	0.88
	False	0.1	0.70	0.37	0.48	0.73	0.95	0.99	0.89
		0.25	0.56	0.47	0.51	0.74	0.94	0.95	0.89
		0.5	0.27	0.68	0.39	0.66	0.90	0.87	0.87
Random Forest	True	0.1	0.47	0.61	**0.53**	**0.75**	0.94	0.94	0.88
		0.25	0.37	0.73	0.49	0.72	0.92	0.91	0.90
		0.5	0.22	0.81	0.35	0.62	0.86	0.82	0.89
	False	0.1	0.45	0.61	0.52	0.74	0.94	0.93	0.87
		0.25	0.37	0.69	0.49	0.72	0.92	0.91	0.88
		0.5	0.22	0.75	0.34	0.62	0.87	0.83	0.85

5.2 Findings

Table 2 shows the results of the grid search over baseline methods and hyper-parameters pre-processing and proportion of positive samples. They reveal important insights into the performance of the classifiers and the challenges inherent in the task.

Logistic Regression, when combined with preprocessing and evaluated on positive sample proportions of 0.1 and 0.25, demonstrated the best performance among all tested methods. These configurations achieved the highest overall scores across critical metrics, including weighted F1, accuracy, and AUC-ROC. The evaluation on the test set confirmed these findings, with all three metrics reaching satisfactory levels ($\geq$ 0.9), indicating robust performance in distinguishing toxic and non-toxic comments. This consistency suggests that overfitting is not a significant concern in these configurations, further reinforcing their reliability and generalizability.

Despite the overall strong performance, the precision, recall, and F1 scores for the positive class (C1) remained suboptimal, peaking at only 0.53. This limitation suggests that the model struggles with effectively identifying and classifying toxic comments. The implications of these results highlight the need for further investigation into strategies to improve the model's sensitivity to toxic content, such as better handling of class imbalances, enhancing feature representation, or exploring alternative architectures.

Preprocessing had no significant or consistent impact across all metrics. While it slightly enhanced performance in certain configurations, the benefit was not universally observed across all classifiers or dataset splits. This suggests that preprocessing alone is insufficient to address the challenges posed by the dataset's inherent characteristics, such as class imbalance or feature sparsity.

The imbalance of the dataset emerged as the most critical factor influencing classifier performance. Higher positive proportions led to noticeable declines in precision and overall F1 scores, underscoring the challenges of effectively handling the minority toxic class. These results emphasize the importance of addressing imbalance, potentially through oversampling, undersampling, or class-weighted learning techniques, to achieve more reliable toxic comment classification.

6 Discussion

Logistic Regression, particularly with preprocessing and optimized positive sample proportions (0.1 and 0.25), proved to be the most reliable baseline, achieving high weighted F1, accuracy, and AUC-ROC scores, with minimal signs of overfitting as performance remained consistent across validation and test sets. This indicates that the model is learning generalizable patterns effectively. However, the approach struggles to identify toxic comments accurately, with precision, recall, and F1 scores for the toxic class (C1) remaining unsatisfactory, peaking at only 0.53.

Additionally, TF-IDF vectorization was found to be unsuitable, confirming its limitations in capturing the complex linguistic features needed for this task. Despite preprocessing providing marginal improvements in some configurations, it did not show consistent benefits across all metrics, highlighting the challenges posed by the dataset's significant class imbalance and the need for more sophisticated approaches to address these limitations.

These results align with findings in the wider body of research on toxic comment classification. As observed by Gladwin et al., traditional techniques like TF-IDF for feature extraction often fail to capture the complex semantic and syntactic nuances necessary for this task, particularly in datasets with significant class imbalance. This reinforces the need for more advanced text representation techniques, such as word embeddings or transformer-based embeddings, to better capture the intricacies of online discourse. [37]

6.1 Task complexity

The Civil Comments dataset presents a unique combination of challenges that make the task of toxic comment classification non-trivial:

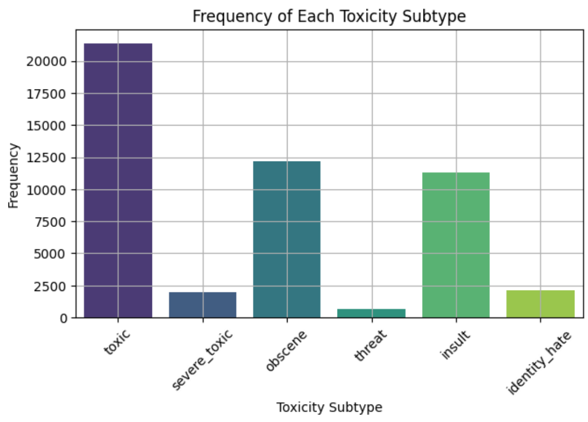
- **Class Imbalance:** Toxic comments represent a small fraction of the dataset, resulting in models that struggle to generalize well for the minority class (C1). As shown in our results, while overall accuracy and weighted F1 scores are satisfactory, precision and recall for the toxic class are not.
- **Linguistic Variability:** Toxic comments often involve subtle linguistic cues, sarcasm, or coded language, making them hard to detect with traditional methods.
- **Practical Implications:** Despite these challenges, achieving even moderate success in this task is critical, as the insights provided by the final model can help identify harmful content on online platforms and foster safer digital spaces.

Based on our findings, achieving satisfactory performance on the Civil Comments dataset is possible but requires careful handling of class imbalance and feature representation. In a business context, deploying a model with these capabilities could have significant value for content moderation on social media platforms, forums, and other online communities. However, improving precision and recall for toxic comments is essential before practical implementation to avoid misclassifications that could undermine user trust.

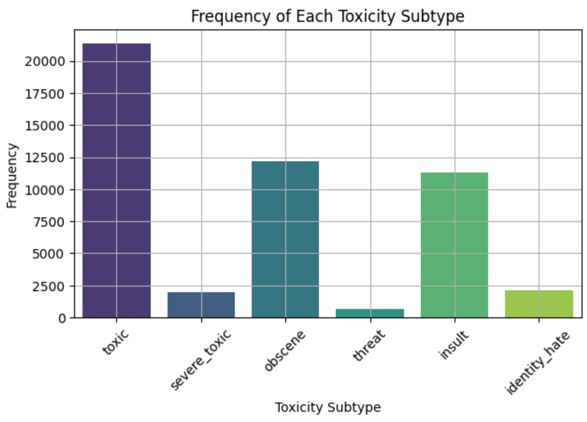
6.2 Future directions

Several promising avenues for future work can be explored to enhance toxic comment classification. First, applying the baseline methods to additional datasets, such as Jigsaw Toxicity and SST-2, would enable a broader evaluation of their effectiveness and provide a foundation for comparing the performance of BERT against the baselines across all three datasets. Furthermore, exploring a non-binary classification format, such as toxicity subtype classification, may yield deeper insights into the nuances of toxic language and offer more granular moderation capabilities. Finally, fine-tuning our pre-trained BERT model on these datasets and comparing its performance to the current baseline methods could provide more satisfactory results.

A Visualizations

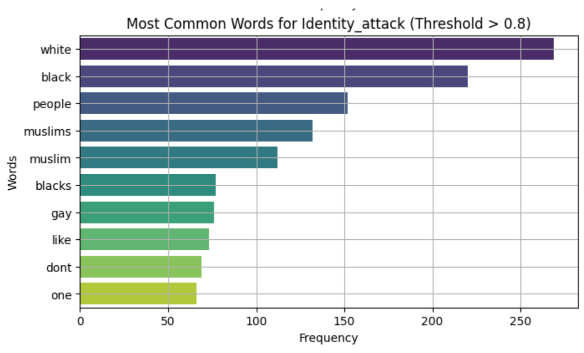


(a) The Civil Comments dataset.

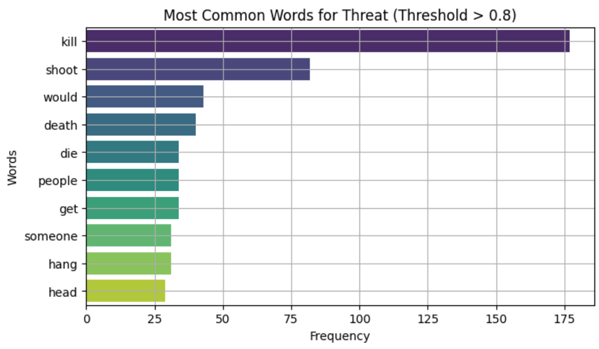


(b) The Jigsaw Toxicity dataset.

Figure 1: Toxicity frequency plots.

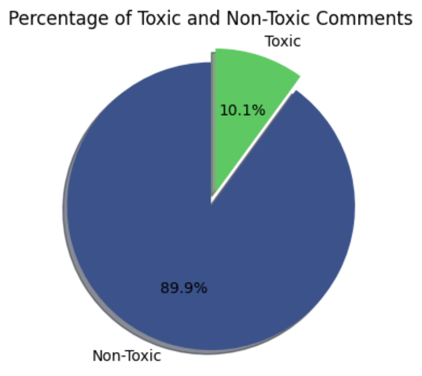


(a) identity_attack subtype.

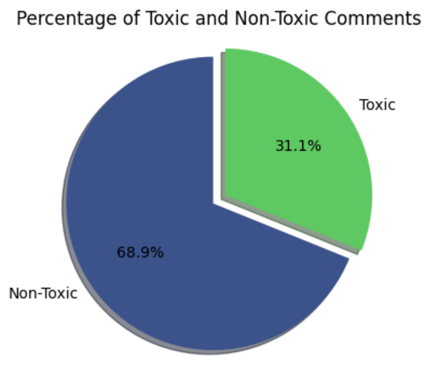


(b) Threat subtype.

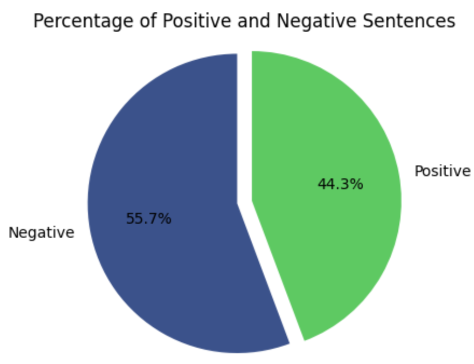
Figure 2: Example of word frequency plots for the Civil Comments datasets for two toxicity subtypes.



(a) The Jigsaw Toxicity dataset.



(b) The Civil Comments dataset.



(c) The SST-2 dataset.

Figure 3: Toxicity/Sentiment percentage.

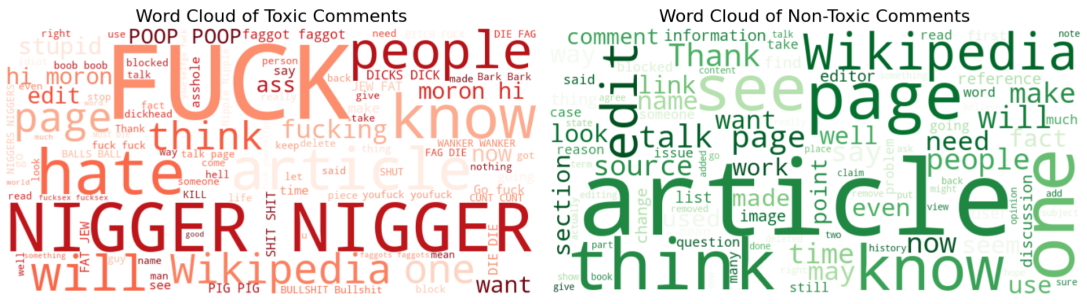
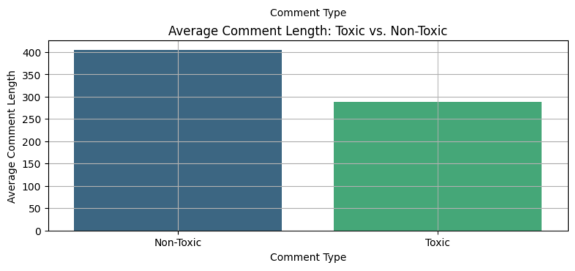
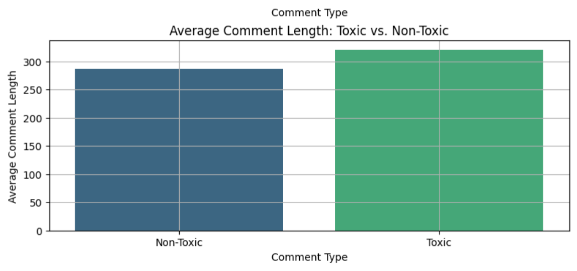


Figure 4: Example of Word Clouds for the Jisaw Toxicity dataset.



(a) The Jigsaw Toxicity dataset.



(b) The Civil Comments dataset.

Figure 5: Average comment length.

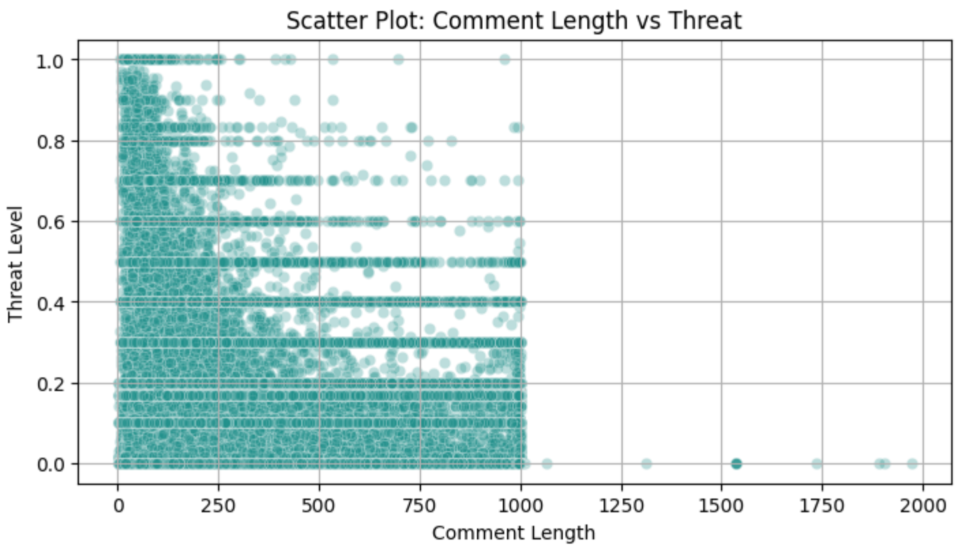


Figure 6: Toxicity subtype vs. comment length example for the Civil Comments dataset.

References

[1] D. Andročec, “Machine learning methods for toxic comment classification: A systematic review,” *Acta Universitatis Sapientiae, Informatica*, vol. 12, no. 2, pp. 205–216, 2020.

[2] S. Zaheri, J. Leath, and D. Stroud, “Toxic comment classification,” *SMU Data Science Review*, vol. 3, no. 1, p. 13, 2020. [Online]. Available: https://scholar.smu.edu/datasciencereview/vol3/iss1/13

[3] Jigsaw/Conversation AI, “Jigsaw unintended bias in toxicity classification,” https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification, 2019.

[4] C. Duchene, H. Jamet, P. Guillaume, and R. Dehak, “A benchmark for toxic comment classification on civil comments dataset,” *arXiv preprint arXiv:2301.11125*, 2023.

[5] Jigsaw/Conversation AI, “Toxic comment classification challenge,” https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge, 2017.

[6] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global vectors for word representation,” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[9] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” *Transactions of the Association for Computational Linguistics*, 2017.

[10] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. St. John, N. Constant, M. Guajardo-Céspedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, and R. Kurzweil, “Universal sentence encoder for english,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 169–174.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” *OpenAI Blog*, 2019.

[13] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[14] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[15] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014.

[16] A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 855–864.

[17] G. Salton, *Automatic Text Processing: The Transformation Analysis and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[18] M. Husnain, A. Khalid, and N. Shafi, “A novel preprocessing technique for toxic comment classification,” in *2021 International Conference on Artificial Intelligence (ICAI).* IEEE, 2021, pp. 22–27.

[19] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[20] G. M. Riduan, I. Soesanti, and T. B. Adji, "A systematic literature review of text classification: Datasets and methods," in *2021 5th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE).* IEEE, 2021, pp. 71–77.

[21] D. D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," in *Machine Learning: ECML-98*, 1998.

[22] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[23] J. M. Hilbe, *Logistic Regression*, 2011.

[24] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[25] Queisler, Dobler, and Ammar, "PML-BERT: Pre-trained transformers for machine learning," https://github.com/devWhyqueue/pml-bert, 2024, accessed: 2024-11-22.

[26] P. A. Ozoh, A. A. Adigun, and M. O. Olayiwola, "Identification and classification of toxic comments on social media using machine learning techniques," *International Journal of Research and Innovation in Applied Science (IJRIAS)*, vol. 4, no. 11, pp. 142–147, 2019.

[27] N. Chakrabarty, "A machine learning approach to comment toxicity classification," in *Computational Intelligence in Pattern Recognition: Proceedings of CIPR 2019.* Springer, 2020, pp. 183–193.

[28] R. Sharma and M. Patel, "Toxic comment classification using neural networks and machine learning," *International Advanced Research Journal in Science, Engineering and Technology*, vol. 5, no. 9, 2018.

[29] S. V. Georgakopoulos, S. K. Tasoulis, A. G. Vrahatis, and V. P. Plagianakos, "Convolutional neural networks for toxic comment classification," in *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, 2018, pp. 1–6.

[30] M. Anand and R. Eswari, "Classification of abusive comments in social media using deep learning," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC).* IEEE, 2019, pp. 974–977.

[31] A. Ezen-Can, "A comparison of LSTM and BERT for small corpus," *arXiv preprint arXiv:2009.05451*, 2020.

[32] K. L. Tan, C. P. Lee, K. S. M. Anbananthen, and K. M. Lim, "RoBERTa-LSTM: A hybrid model for sentiment analysis with transformer and recurrent neural network," *IEEE Access*, vol. 10, pp. 21 517–21 525, 2022.

[33] A. Radford, "Improving language understanding by generative pre-training," 2018.

[34] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[35] H. D. Lee, S. Lee, and U. Kang, "Auber: automated bert regularization," *Plos one*, vol. 16, no. 6, p. e0253241, 2021.

[36] D. Vucetic, M. Tayaranian, M. Ziaeefard, J. J. Clark, B. H. Meyer, and W. J. Gross, "Efficient fine-tuning of bert models on the edge," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 2022, pp. 1838–1842.

[37] I. Gladwin, E. V. Renjiro, B. Valerian, I. S. Edbert, and D. Suhartono, "Toxic comment identification and classification using BERT and SVM," in *2022 8th International Conference on Science and Technology (ICST).* IEEE, 2022, pp. 1–6.