Barcelona School of Economics

Advanced Methods in Natural Language Processing: Final Project

Data Science Master's Program - BSE

Alex Malo
Viktoria Gagua
Noemi Lucchi
Alejandro Delgado

Contents

1 Dataset Exploration and Baseline Analysis

(a) Bibliography and SOA

The toxicity classification task involves automatically identifying harmful, abusive, or toxic content in online text communications, representing a crucial binary classification problem for maintaining healthy digital discourse and protecting users from harassment.

Leading methodologies focus on transformer-based architectures (BERT, RoBERTa, DistilBERT) utilizing contextual embeddings with fine-tuning, multi-task learning frameworks, and adversarial training techniques to capture nuanced language patterns specific to toxicity detection.

BERT is serving as the foundational model that has spawned several optimized variants including **DistilBERT** (Sanh et al., 2019) [1], which achieves 40% parameter reduction and 60% speed improvement while preserving over 95% of BERT's performance, **ALBERT** (Lan et al., 2019) [2] that reduces parameters by 89% through layer sharing and embedding factorization, and **RoBERTa** (Zhuang et al., 2021) [3] which improves performance through enhanced tokenization and training methodologies.

(b) Dataset Overview

The Jigsaw Toxicity Classification dataset, sourced from HuggingFace (Arsive/ toxicity_classification_jigsaw), contains 25,960 samples of Wikipedia talk page comments with human annotations. The dataset exhibits a remarkably balanced class distribution with 13,727 non-toxic comments (52.9%) and 12,233 toxic comments (47.1%), but has relatively higher imbalance in subcategories of toxicity.

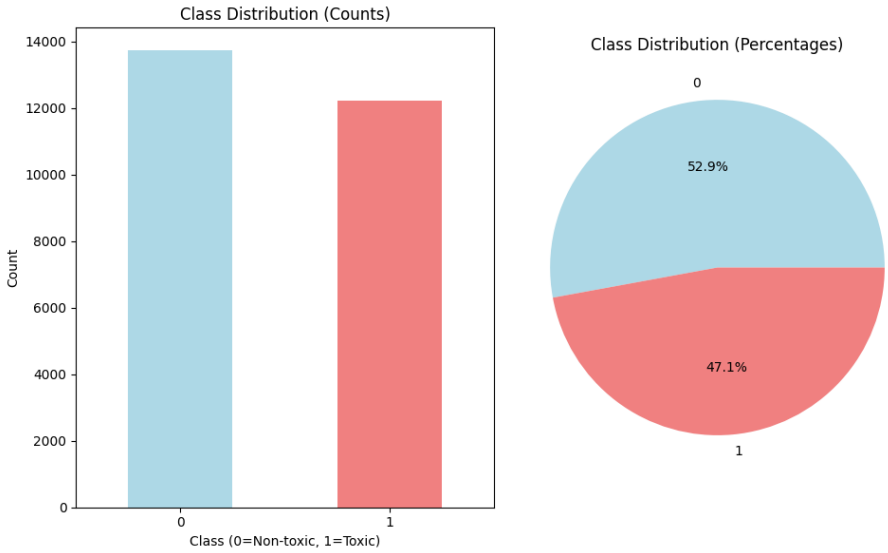

Figure 1: Class distribution showing balanced toxic vs non-toxic comments

(i) Text Characteristics Analysis

Statistical analysis reveals interesting patterns in comment structure. The average text length is 353.5 characters with a mean word count of 60.9 words per comment. The distribution shows a strong right skew, with most comments being relatively brief but extending to a maximum of 5,000 characters. This pattern reflects typical online discussion behavior where most interactions are concise, but detailed discussions occasionally occur.

Similar to character length, the word count follows a heavily right-skewed distribution with most comments containing around 29 words (median) but some extending to over 1400 words. The average of 61 words per comment indicates typical short-form social media communication patterns. This should be typical for online discussion platforms where brief interactions dominate but occasional detailed discussions also occur.

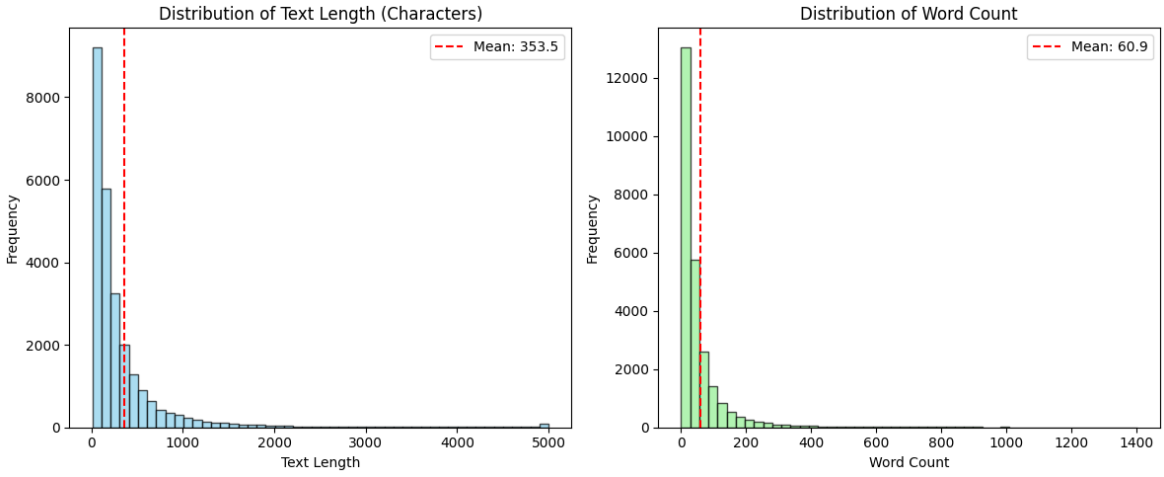

Figure 2: Distribution of text length and word count showing right-skewed patterns

(ii) Multi-Label Toxicity Categories

The dataset includes subcategory labels that reveal the different "nature" of online toxicity. Obscene content represents the largest category with 6,804 samples (26.21%), followed closely by insult content with 6,345 samples (24.44%). Severe toxic content accounts for 1,298 samples (5.00%), identity hate for 1,124 samples (4.33%), and threat represents the smallest category with 373 samples (1.44%).

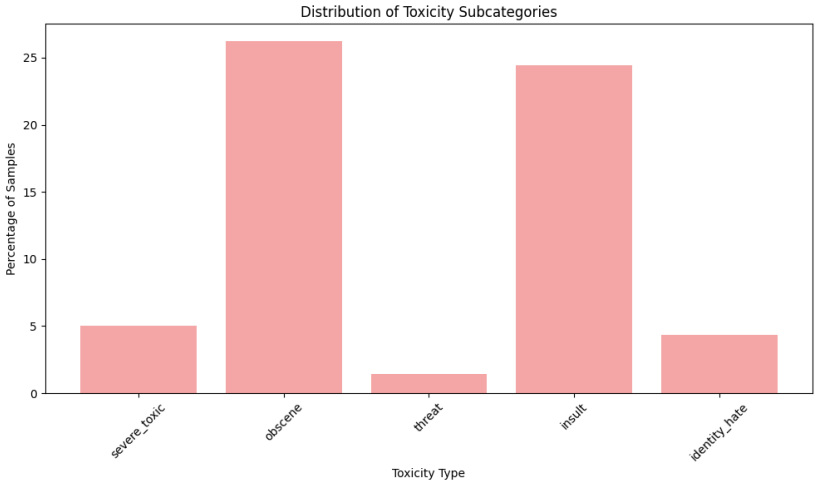

Figure 3: Multi-label toxicity Distribution

(iii) Toxic vs Non-Toxic Content Characteristics

Comparative analysis reveals significant differences between toxic and non-toxic content characteristics. Toxic comments average 293.0 characters compared to 407.4 characters for non-toxic comments, indicating that toxic users tend to write shorter, more aggressive outbursts. The most striking difference appears in exclamation mark usage, with toxic comments averaging 3.63 exclamation marks compared to 0.46 in non-toxic comments—a 7.8-fold increase indicating higher emotional intensity.

Caps ratio analysis shows toxic comments contain 11.5% uppercase characters compared to 4.5% in non-toxic comments, representing a 2.5-fold increase in "shouting" behavior. Question count is also higher in toxic comments (0.57 vs 0.45), often reflecting rhetorical or confrontational questioning patterns.

An additional insight from graph is lexical diversity's result that toxic comments exhibit only a slightly lower lexical diversity (0.865) compared to non-toxic comments (0.841). Lexical diversity reflects the proportion of unique words to total words in a text, serving as a proxy for vocabulary richness. The narrow gap between the two groups suggests that toxic users are not necessarily less articulate or less educated. In fact, both groups demonstrate similarly varied vocabularies, challenging common assumptions that associate toxic behavior with limited language ability. This highlights that toxic individuals may have access to a broad lexicon — they are simply using it in harmful ways. Consequently, vocabulary complexity alone is not a reliable feature for distinguishing toxic from non-toxic content, which is intuitive, but important to confirm through data.

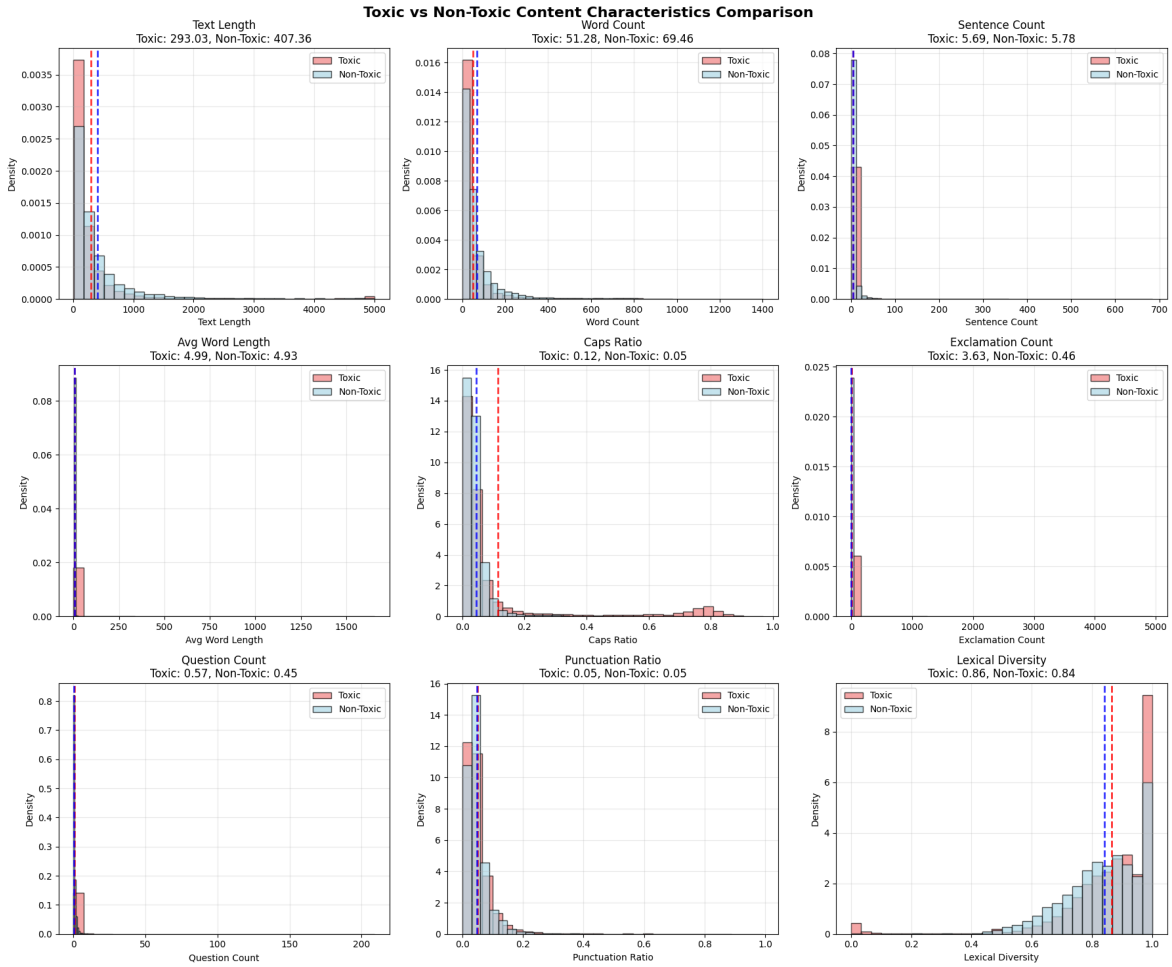

Figure 4: Comprehensive comparison of text characteristics between toxic and non-toxic comments

(iv) Overlap Analysis

Multi-label correlation analysis reveals significant relationships between toxicity categories. The strongest correlation exists between obscene content and insults (r=0.673), indicating that crude language frequently accompanies personal attacks. This pattern is visualized through Venn diagram analysis, showing 4,966 comments contain both obscene language and insults simultaneously, representing the classic "swearing while name-calling" behavior pattern.

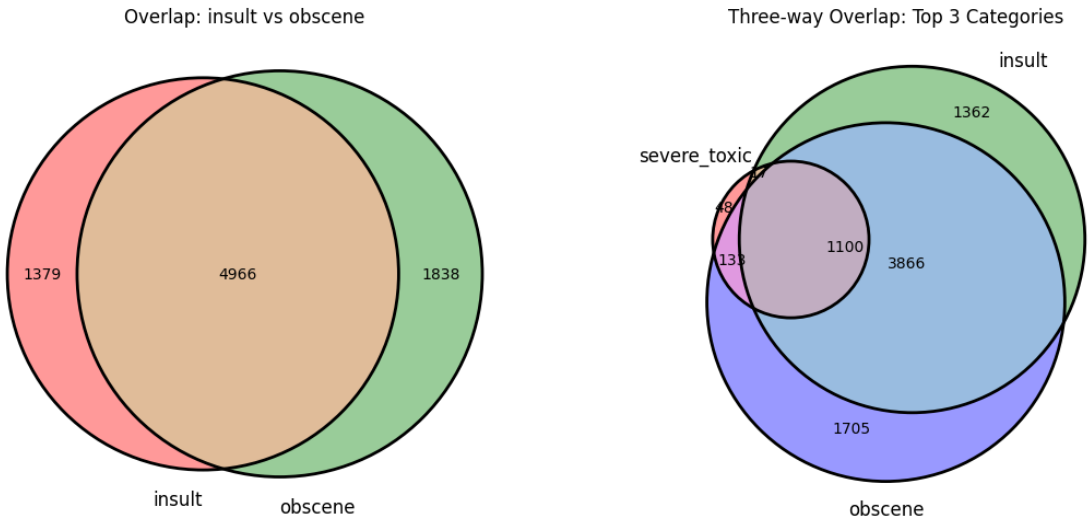

Figure 5: Subcategory Overlaps

Threats show very low correlations with other types of toxic behavior ($r \approx 0.10$), which likely reflects both the unique nature of threatening comments and the small amount of data available. Since threats make up only 1.44% of all toxic comments (373 examples), the small sample size limits our ability to find strong patterns. This makes it hard to tell whether threats are truly different from other toxic behaviors, or if the data is just too limited to show any connections.

Most toxic comments tend to exhibit multiple forms of toxicity rather than fitting into a single category. Analysis of the distribution reveals that around 4,500 comments carry no specific toxicity labels, approximately 2,500 are tagged with a single label, and about 5,500 are associated with multiple labels. This suggests that a significant portion of toxic content reflects more severe behavior, warranting higher priority in moderation efforts.

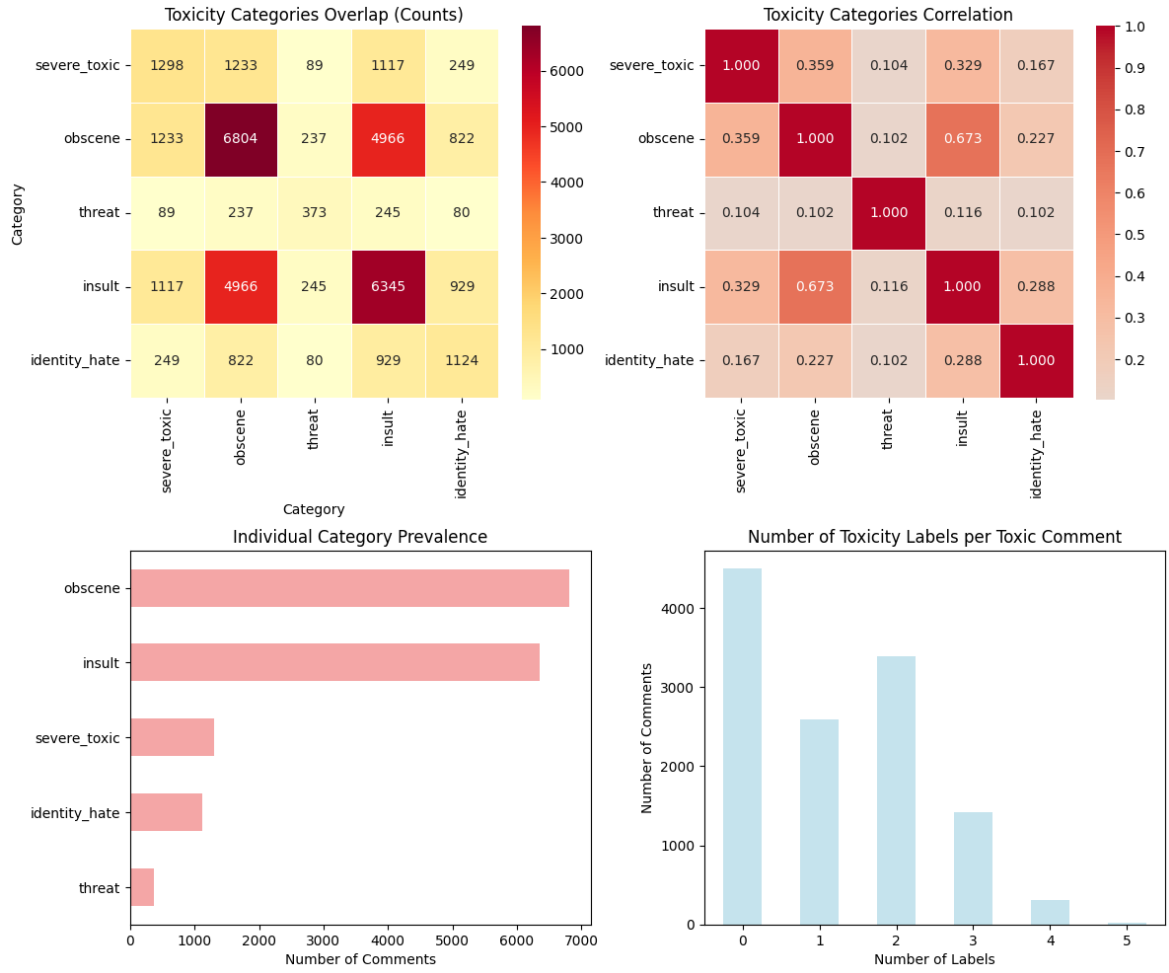

Figure 6: Multi-label toxicity analysis showing category overlap and correlation patterns

(v) Vocabulary Analysis

Word frequency analysis reveals stark vocabulary differences between categories. Toxic comments are dominated by explicit profanity, with "fuck" appearing 6,879 times as the most frequent word, followed by various slurs and aggressive terms. Non-toxic comments demonstrate collaborative vocabulary typical of Wikipedia discussions, featuring words like "articles" (1,558 occurrences), "information" (1,117 occurrences), and "sources," reflecting the platform's educational mission.

Interestingly, some words appear frequently in both categories (wordclouds), particularly "like," "don't," and "people." However, the context differs significantly - in toxic comments, these words often appear in aggressive constructions ("don't like you"), while in non-toxic comments they're used for constructive discussion ("like this approach" or "people might find this helpful").

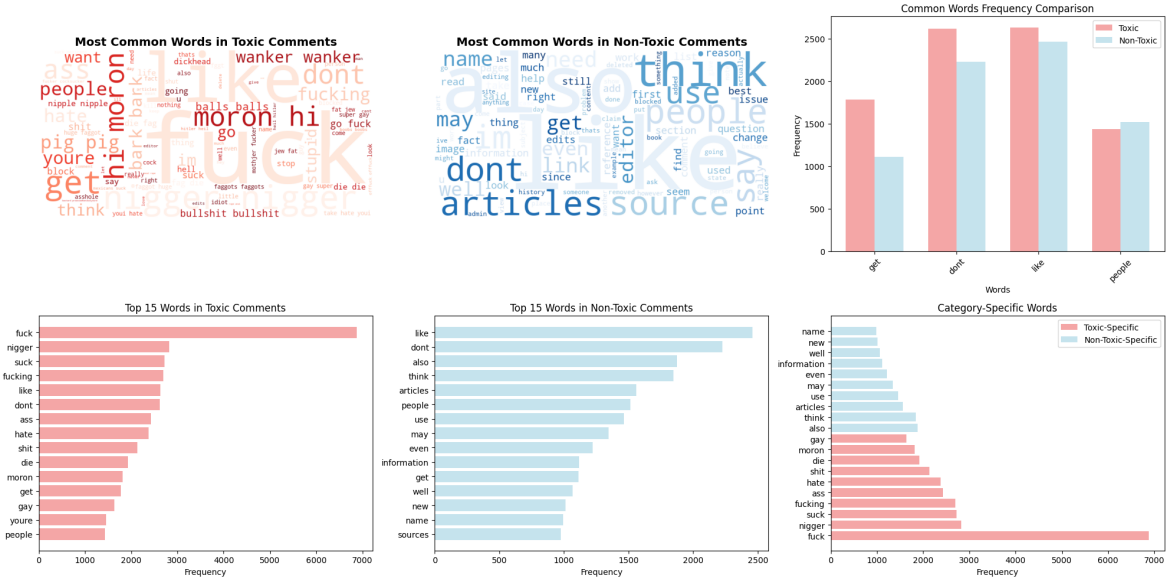

Figure 7: Word clouds and frequency analysis showing distinct vocabulary patterns

(c) Random Classifier Performance

(i) Theoretical Analysis

A random classifier serves as the fundamental baseline for any machine learning task. Given the balanced class distribution (52.9% non-toxic, 47.1% toxic), we can calculate theoretical performance metrics. For a random classifier that assigns labels with equal probability (50% for each class), the expected accuracy equals 50% regardless of class distribution, as it represents the probability of correct random guessing.

For binary classification with class probabilities $P(\text{non-toxic}) = 0.529$ and $P(\text{toxic}) = 0.471$, the theoretical precision for the positive class (toxic) equals the class prior probability: 0.471. The theoretical recall equals 0.5 (random 50% probability), yielding an F1-score of:

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = 2 \times \frac{0.471 \times 0.5}{0.471 + 0.5} = 0.485$$

(ii) Empirical Validation

Empirical simulation across 1,000 runs confirms theoretical predictions with remarkable consistency. The random classifier achieved 50.0% $\pm$ 0.003 accuracy, 47.1% $\pm$ 0.003 precision, 50.0% $\pm$ 0.005 recall, and 48.5% $\pm$ 0.004 F1-score. The small standard deviations demonstrate the stability of random performance and establish a reliable baseline threshold.

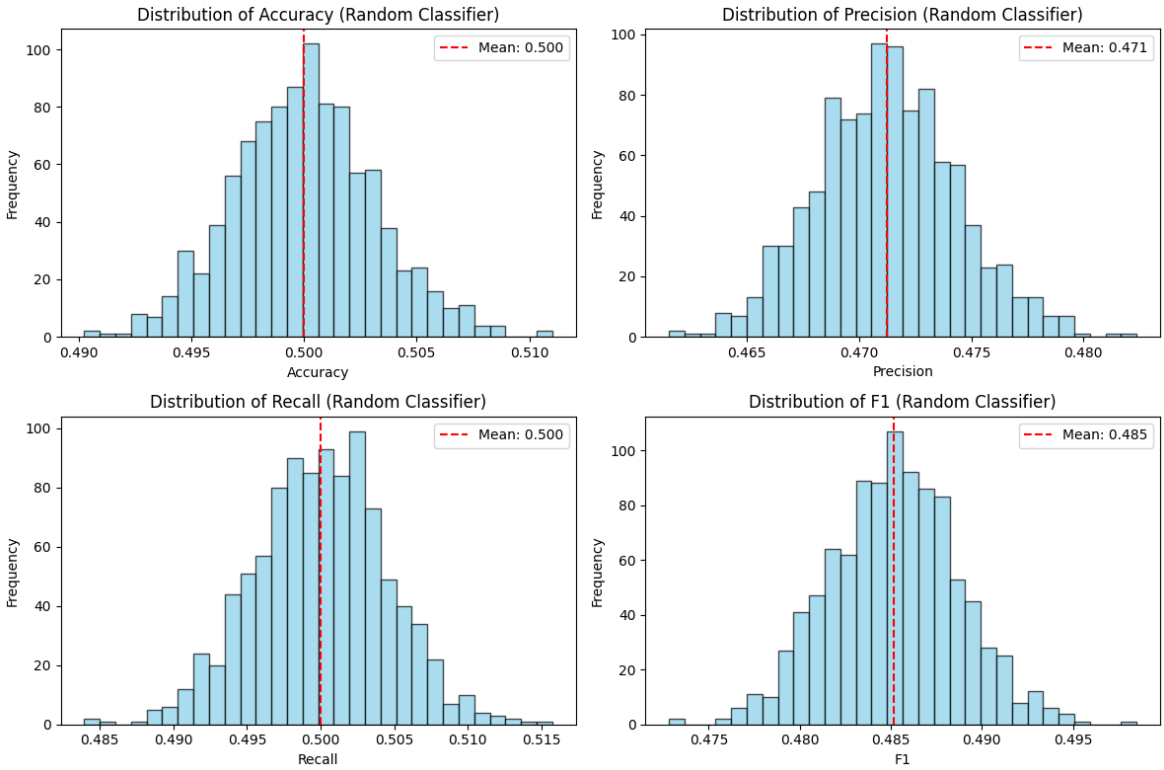

Figure 8: Distribution of random classifier performance metrics across 1,000 simulations

(d) Baseline Implementation

(i) Rule-Based Classifier Design

The rule-based baseline classifier employs a comprehensive feature engineering approach combining lexical patterns, linguistic indicators, and behavioral markers. The system implements weighted profanity scoring across three severity levels: mild words (weight=1) including "damn" and "stupid," moderate words (weight=2) such as "idiot" and "moron," and severe words (weight=4) including "kill" and "hate."

Pattern matching components detect personal attack constructions using regular expressions for phrases like "you are stupid" and direct threats such as "kill yourself." Text characteristic analysis examines caps ratio, exclamation count, repeated characters, and punctuation patterns as emotional intensity indicators.

The final toxicity score combines all features:

$$\text{Score} = \text{Profanity} + \text{Caps} \times 2 + \text{Exclamations} \times 0.5 + \text{Attacks} \times 3 + \text{Threats} \times 5$$

Comments exceeding a threshold of 3.0 are classified as toxic.

(ii) Performance Analysis

The rule-based classifier significantly outperforms the random baseline across most evaluation metrics. Most notably, it achieves substantially higher **precision** (84% vs. 47%), meaning that when the model predicts a comment as toxic, it is correct most of the time. However, this comes at the cost of **very low recall** (19% vs. 50%),

indicating that it misses a large portion of toxic content—even more than random guessing.

Table 1: Performance Comparison: Random vs Rule-Based Classifier

Metric	Random	Rule-Based	Improvement
Accuracy	50.0%	60.3%	1.2×
Precision	47.1%	84.0%	1.8×
Recall	50.0%	19.0%	0.4×
F1-Score	48.5%	31.0%	0.6×

The confusion matrix highlights the classifier's conservative behavior. It correctly identifies 5,145 non-toxic comments and raises very few false alarms (169 false positives). However, it fails to detect 3,800 toxic comments (false negatives), capturing only 886 true toxic cases. This shows that while the classifier is highly precise, it tends to err on the side of caution, allowing many toxic comments to go undetected.

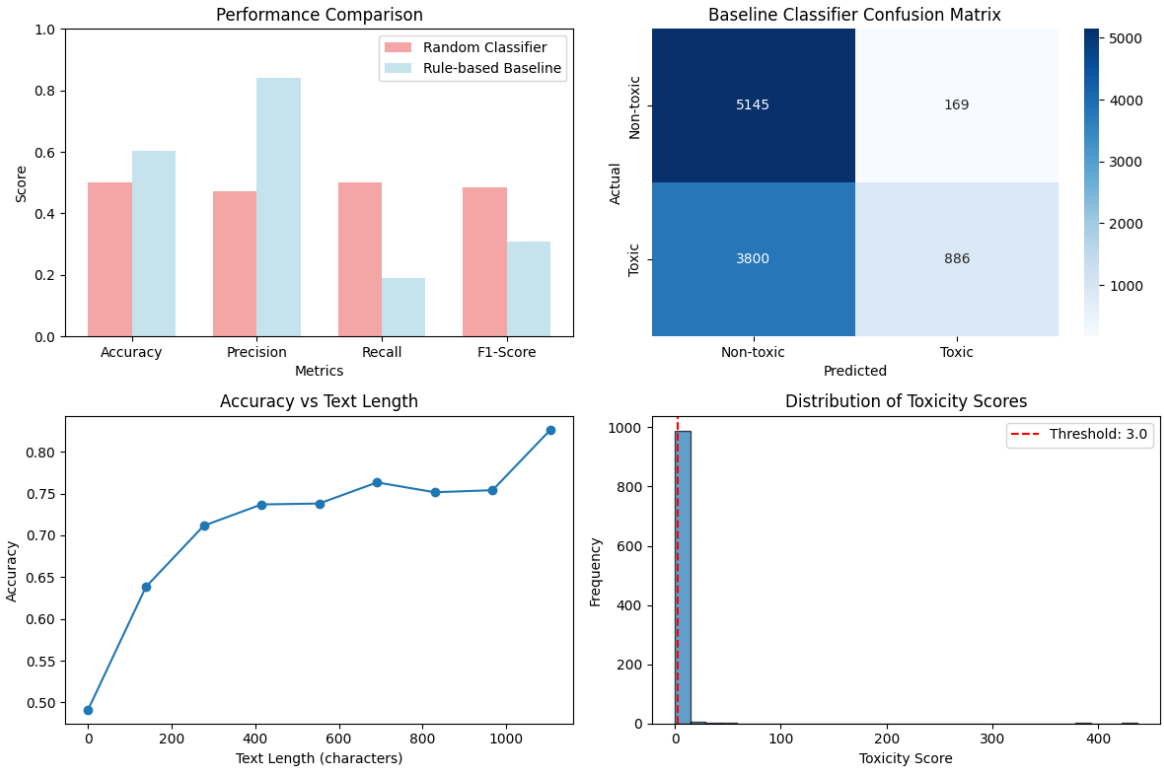

Figure 9: Baseline classifier performance analysis including confusion matrix and error patterns

An analysis of accuracy by text length reveals that the classifier struggles with very short comments, achieving only 49% accuracy. Performance improves significantly with longer inputs, peaking at 81% accuracy around 1,100 characters. This trend is expected, as longer comments provide more lexical and structural features for the rule-based system to evaluate.

The toxicity score distribution further illustrates the model's cautious approach. Most comments receive scores close to zero, and the applied threshold of 3.0 only flags the most overtly toxic content. This **right-skewed distribution** explains the low recall: by setting a high threshold, the classifier captures only the most explicit cases of toxicity—those containing multiple, clear toxic indicators—while overlooking more subtle or borderline instances.

(iii) Limitations

The system has trouble understanding context, so it often misses sarcasm or serious discussions about sensitive topics. It's also easy to trick by changing letters or symbols in words, and it can't keep up with new or changing ways people use language. Most importantly, it often fails to catch more subtle toxic comments that don't match clear patterns, which helps explain why its recall is so low.

2 Limited Data Learning Strategies

(a) BERT Model with Limited Data

We sampled 32 observations from the training set, deliberately deviating from the original data distribution to ensure effective learning across all toxicity categories. Specifically, we sampled 6 examples for each of the 3 more represented classes (toxic, obscene, and insult) and 7 examples for the 2 remaining highly unbalanced classes (severe_toxic and identity_hate). We opted for this balanced sampling methodology rather than following the initial distribution because the original dataset exhibits severe class imbalance - some minority classes would have had only 1 or 2 examples in a proportional 32-sample subset, which would have been insufficient for the model to learn meaningful patterns and would have resulted in suboptimal fine-tuning performance. This strategic oversampling of underrepresented classes ensures that our limited training data provides adequate representation for each toxicity category, enabling the model to develop robust classification capabilities across all labels despite the extremely limited data.

After defining our 32-examples balanced sample, we implemented DistilBERT-base-uncased, which is particularly well-suited for few-shot learning because it retains 97% of BERT's performance while being 60% smaller, reducing the risk of overfitting when training data is extremely scarce. The architecture consists of:

- **Input Layer**: We tokenized text with maximum length of 64 tokens and defined the input_ids and attention_masks as input through `tf.keras.layers.Input(shape=(max_length,), dtype='int32')`, which are required by the DistilBERT model - the input_ids represent the tokenized text converted to numerical IDs while attention_masks indicate which tokens to consider (1) and which to ignore (0, padding tokens).
- **DistilBERT Encoder**: Pre-trained transformer layers that process the input through `model(input_ids_in, attention_mask=input_masks_in)[0]` to generate contextual embeddings.

- **CLS Token Extraction**: We use `[:,0,:]` to extract only the [CLS] token representation for classification, as this token contains the aggregated representation of the entire sentence with 768 dimensions capturing the semantic meaning of the whole comment.
- **Dropout Layer**: Implemented with `tf.keras.layers.Dropout(rate=0.7)` where we used a dropout rate higher than usual, 0.7 instead of 0.5, since with only 32 examples the model would be extremely prone to memorizing the training data rather than learning generalizable patterns - this high dropout rate randomly sets 70% of neurons to zero during training, providing strong regularization.
- **Dense Layer**: With `tf.keras.layers.Dense(num_labels=5, activation='sigmoid')` where we used sigmoid activation function for multi-label classification, allowing the model to independently predict each of the 5 toxicity categories (toxic, severe_toxic, obscene, insult, identity_hate) since a comment can belong to multiple categories simultaneously.

We compiled the model with `binary_crossentropy` loss function appropriate for multi-label tasks, Adam optimizer with learning rate 2e-5 typical for BERT fine-tuning, and set a small batch size of 16 to ensure 2 weight updates per epoch (32 examples / 16 = 2 steps), which is necessary to learn given such limited data. Lastly, we limited training to 5 epochs to prevent the model from excessively memorizing the small training set.

The baseline BERT model trained on 32 examples achieved the following performance on the validation set:

Metric	Score (%)
Accuracy	78.95
Precision	39.48
Recall	50.00
F1-Score	44.12

Table 2: Baseline BERT Performance (32 examples)

Analysis of Results: The baseline results reveal both strengths and limitations of fine-tuning BERT with extremely limited data:

- **High Accuracy (78.95%)**: This appears promising but is misleading in the context of multi-label classification with imbalanced data. The model likely achieves high accuracy by predicting negative for most labels, which is correct for the majority class, but still doesn't mean the model is good.
- **Low Precision (39.48%)**: This indicates the model generates many false positives - when it predicts a comment belongs to a specific type of toxicity, it's wrong more than 60% of the time.
- **Moderate Recall (50.00%)**: The model catches exactly half of the actual toxic content.

- **F1-Score (44.12%)**: This metric combines precision and recall and, as already understood, the model works relatively good for being a starting point, but still needs improvements.

Considering that this model has been trained on only 32 examples, across 5 labels, we already appreciate this performance, although we acknowledge that this current model is achieving such high accuracy just predicting 0 most of the time. We need to further improve our model in order to well capture instances of toxicity.

(b) Dataset Augmentation

Among the different techniques that could be implemented to do dataset augmentation, we tried Zero-Shot Learning Classification and Back-Translation.

- **Zero-Shot Learning Classification.** Assuming access only to the initial labeled sample of 32 examples and a larger pool of unlabeled data, we employed a Zero-Shot Learning (ZSL) approach to classify additional instances and thus expand the training dataset. Specifically, we used ZSL to classify 1,000 unlabeled examples and applied a confidence threshold of 0.6, meaning that only those examples whose cosine similarity with their most probable label exceeded 0.6 were retained. Using this threshold, only 104 examples out of the 1,000 were accepted. These examples belonged to 3 out of the 5 target categories: *insult*, *obscene*, and *toxic*.

- **Back-Translation**: This methodology involves translating to another language the sample available and then translate back to English, since this would retain the initial meaning by slightly change the semantic of the sentence. To perform this task we chose Italian since it has different grammatical rules compared to English, and we thought this would have enriched the sample with enough diversity.

We tried both approaches, in order to compare their strengthens and weaknesses, specifically conditional on the task.

We anticipated that the Zero-Shot approach would be suboptimal for this task, given the nuanced nature of the classification problem. Specifically, the task involves distinguishing between subtle variations within a single thematic domain (i.e., different forms of toxicity), rather than separating clearly distinct topics or domains. As such, we expected that a pre-trained model, without any task-specific fine-tuning, would struggle to make reliable distinctions. Moreover, we expected the overall performance of this approach to be limited by the imbalanced label distribution in the newly classified data. Of the 104 additional examples retained through Zero-Shot classification, only three categories were represented (*insult*, *obscene*, and *toxic*), while the categories *severe_toxic* and *identity_hate* remained severely underrepresented, with only 7 examples each in the full dataset.

Conversely, we expected Back-Translation to be more effective, as it would have just doubled the sample, by adding properly-labeled examples.

After using these two techniques to increase the sample size, we use the same BERT model previously used to classify examples in the validation set. As follows we present and discuss results:

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Zero-Shot Classification	68.57	51.86	51.80	51.82
Back-Translation	77.06	40.67	48.89	43.76

Table 3: Augmentation Methods Comparison

Zero-Shot Augmentation:

- **Accuracy drops to 68.57%**: This suggests that the model is now making more positive predictions, leading to lower overall accuracy but potentially better minority class detection.
- **Precision increases to 51.86%**: This confirms that the model has learned to make more accurate positive predictions.
- **Recall slightly increases to 51.80%**: Suggesting that the model is able to detect more positive examples.
- **F1 increases to 51.82%**: This represents the most substantial improvement, demonstrating that zero-shot augmentation successfully enhanced the model's ability to identify toxic content.

Back-Translation: All metrics remain close to baseline results, suggesting back-translation provided limited benefit in this scenario. The marginal improvement can be attributed to several factors:

- The 32 back-translated examples are likely too semantically similar to the original ones. This means the method did not significantly increase data diversity or expose the model to substantially different linguistic patterns.
- The translation models may struggle with the informal, offensive language typical of toxic comments, potentially altering the toxicity levels during the translation process.
- Texts might be too short to benefit from additional diversity. Indeed, when comments are too short it might be hard to change the semantic while maintaining the same meaning.

Against our expectations, Zero-Shot performed well in this specific task, despite the very specific and subtle nature of toxicity classification. Conversely, Back-Translation proved to be basically ineffective in the specific application.

(c) Zero-Shot Learning with LLM

As LLM we chose Gemini, since it provides a free API tier, unlike Claude or ChatGPT. Specifically, we selected Gemini 2.0 Flash Lite as our model variant because it is smaller and more efficient than other available Gemini models, resulting in higher API rate limits that allow processing 30 requests per minute instead of the 10-15 requests typically allowed by other models provided by Gemini. By leveraging this LLM to classify the validation set, we overcame the fundamental limitation of having insufficient

labeled data to train a robust supervised model, instead harnessing the vast linguistic knowledge and understanding encoded in pre-trained LLMs to perform classification.

The quality of prompt engineering becomes crucial in this zero-shot scenario, as it represents the only mechanism we can manipulate to enhance model performance without additional training data. Our carefully crafted prompt includes explicit definitions of each toxicity category (toxic, severe_toxic, obscene, insult, identity_hate), clear classification instructions, and a structured output format to ensure consistent and parseable responses from the model.

Due to API rate limitations — specifically, a maximum of 30 requests per minute and 1,500 per day — we were unable to process the entire validation set, which consists of over 6,000 examples. Processing the full set would have required running the code for more than four days, including pauses to comply with both the per-minute and daily rate limits.

Additionally, for a subsequent task involving the generation of 100 examples with an LLM, we needed to remain within the daily quota. As a result, we restricted the validation process to 1,400 examples.

Nonetheless, we believe the results obtained are valid and would not differ substantially if the entire validation set were used. This belief is supported by two main considerations:

- During the development phase of the pipeline, we initially worked with a subset of only 100 examples. The results obtained with this small subset were essentially the same as those derived from the larger sample of 1,400 examples. Given that increasing the sample size from 100 to 1,400 did not significantly affect the results, it is reasonable to assume that extending validation to the full set of over 6,000 observations would likewise not lead to dramatic changes.

- The reduced validation set of 1,400 examples was constructed to match the distribution of the full validation set. This sampling strategy helps ensure that the results are reliable and not biased due to the selection of examples.

As follows we present and discuss results obtained from this methodology:

Metric	Score (%)
Accuracy	84.03
Precision	77.41
Recall	86.89
F1-Score	79.83

Table 4: Gemini Zero-Shot Classification Performance

- **Accuracy increases to 84.03%**: Higher than all fine-tuned models, demonstrating that the LLM has internal sophisticated understanding of toxicity patterns, thanks to the massive pre-training done on extremely large corpus.

- **Precision increases to 77.41%**: Again, higher than all fine-tuned models. This indicates that, among the labels assigned, Gemini is correct 77% of the

time. This indicates the LLM has learned nuanced boundaries between types of toxicity that our small fine-tuned models couldn't capture.

- **Recall increases to 86.89%**: Which means that Gemini successfully captures 87% of toxic content, among different labels.
- **F1-Score increases to 79.83%**: This high balance between accuracy and recall further demonstrates the power of pre-trained models trained on huge corpus, against models fine-tuned on limited data.

(d) Data Generation with LLM

This method represents an attempt to combine the benefits of LLM knowledge with traditional supervised learning. The approach involves using Gemini 2.0 Flash to generate realistic toxic comments with appropriate labels, and then fine-tune BERT on this sample (32 initial examples + synthetic data) and evaluate its performance on the validation set. For consistency with Zero-Shot Classification performed in task b), we generated 100 examples, using an optimized prompt that did the following:

- Describe the categories (same description as before);
- Indicate that each comment might fall into multiple categories;
- Specify the required output format.

As before, we asked to generated 30 examples at a time, with a 65 seconds pause in between, due to API constraints.

As follows we present and discuss results:

Metric	Score (%)
Accuracy	54.77
Precision	52.05
Recall	53.06
F1-Score	48.98

Table 5: BERT with Synthetic Data Performance

- **Significant Accuracy Drop (54.77%)**: The 24-point accuracy decrease compared to baseline suggests the synthetic data introduced substantial noise or distribution mismatch.
- **Moderate Precision/Recall**: While precision (52.05%) and recall (53.06%) are balanced, they're both mediocre, indicating the model learned imprecise decision boundaries.
- **F1-Score (48.98%)**: The F1-score slightly improved over the baseline model, but still remains low.

We believe that this approach may have failed due to two main reasons:

- **Stylistic mismatch between LLM-generated texts and real social media comments:** Texts produced by LLMs might differ from user-generated content that can be found in social media. This is mainly due to filters that LLMs typically have, which prevent them from generating highly offensive or inappropriate content. This leads to a discrepancy between the synthetic data and the real comments to be classified, making the additional examples less useful (or even counterproductive) for the fine-tuning process.
- **Lack of diversity in the generated examples:** The LLM-generated texts might be too similar to one another, offering little additional information beyond what was already captured in the original 32-example dataset (same problem observed when experimenting with Back-Translation). We believe this limited variation is partly related to the content filters applied to LLM outputs, which restrict the generation of certain types of content, leading to repetitive themes and reduced informational gain.

(e) Optimal Technique Application

We start by briefly summarizing previous findings:

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
BERT baseline	78.95	39.48	50.00	44.12
BERT Zero-Shot Classification	68.57	51.86	51.80	51.82
BERT Back-Translation	77.06	40.67	48.89	43.76
Gemini Zero-shot	**84.03**	**77.41**	**86.89**	**79.83**
BERT Synthetic Data	54.77	52.05	53.06	48.98

Table 6: Final Performance Comparison

We can clearly notice the dominance of the Zero-Shot Gemini. This suggests that when labeled data is scarce, leveraging pre-trained knowledge often outperforms attempts to fine-tune small models, even with augmentation techniques.

Additionally, our results demonstrate that not all data augmentation is beneficial:

- **Zero-shot augmentation succeeded** because it used a sophisticated model to label real (unlabeled) data with high confidence
- **Back-translation provided minimal benefit** because it created paraphrases without adding semantic diversity
- **Synthetic data generation failed** despite adding 100 examples, highlighting the importance of data quality over quantity

Lastly, we found that different approaches optimize different aspects of performance:

- **High accuracy models** (baseline BERT) often achieve this by conservative prediction strategies

- **Balanced precision-recall models** (augmented BERT with Zero-Shot) provide more reliable positive predictions
- **High-recall models** (Zero-Shot Gemini) excel at classifying types of toxicity content but may have more false positives

Given the predominance of Zero-Shot Gemini, we try to enhance the performance of this technique with the following improvements:

- **Implementation of a Larger Model:** Previously, we used Gemini 2.0 Flash Lite, and we have now switched to GPT-4.1 Mini. While the exact number of parameters for Flash Lite is not publicly available, we assume it is significantly smaller than GPT-4.1 Mini (which is estimated to have around 7 billion parameters). This assumption is based on the fact that Flash Lite is fast, requires low computational resources, and is available for free. An additional practical advantage of GPT-4.1 Mini is that it does not enforce strict rate limits per minute or per day, unlike Gemini Flash Lite, which had tighter API constraints that limited throughput during large-scale evaluations. On the other hand, GPT usage has a cost, which was very low for this specific task - approximately €0.50.
- **Improved Prompt:** We changed the definitions for the five toxicity categories to enhance clarity, particularly we made clearer the distinction between *toxic* and *severe toxic*. The prompt now explicitly states that the categories are not mutually exclusive and provides more precise instructions, emphasizing objectivity and awareness of tone, sarcasm, and implied meaning. We also added a system message (we specified the role as "You are a toxicity classification assistant") and set the temperature to 0 to ensure consistent and deterministic outputs.
- **Increased Sample Size:** In the Gemini zero-shot setup, we were constrained by API limits and used a sample size of 1,400 examples. As mentioned earlier, during our initial tests we used a sample of only 100 examples to speed up experimentation. When we later increased it to 1,400, we basically observed no difference in performance. In this round, we attempt to double the sample size again, to evaluate whether this change could have any measurable effect. However, based on our previous experience, we do not expect a major impact. Furthermore, since all improvements (i.e., larger model, better prompt, and increased sample size) are introduced simultaneously, we will not be able to isolate the individual contribution of each factor to the overall performance.

Metric	Score (%)
Accuracy	86.60
Precision	79.88
Recall	88.42
F1-Score	82.56

Table 7: GPT-4.1 Mini Performance

We observed a slight improvement across all evaluation metrics, although the overall gains remained relatively modest. Nonetheless, given that these enhancements required

minimal time and effort to implement, and the total API cost was only around €0.50, we consider this strategy worthwhile. From a cost-effectiveness perspective, even small improvements in performance can justify low-effort interventions.

It is also important to acknowledge that the scope of improvements we could implement was inherently limited by the nature of the approach. When relying exclusively on LLMs for direct classification in a Zero-Shot setting, opportunities for user-driven optimization are constrained. Without access to model internals or the ability to fine-tune, our influence is largely limited to prompt engineering and input sampling strategies. Therefore, we interpret the modest gains not as a failure of the approach, but as a natural consequence of working within the constraints of a Zero-Shot LLM pipeline.

An alternative strategy would be to fine-tune the LLM on the corpus of interest (in this case, examples annotated with the five toxicity labels) and then use the fine-tuned model to classify additional unlabeled data. This approach would likely lead to substantial improvements in performance, as the model could better capture the specific linguistic patterns and contextual nuances of the task.

However, such a strategy is not always feasible. Many LLMs do not support user-side fine-tuning, and even when technically possible the process requires significant computational resources and time. Without having access to such strategy to improve the model, the possibilities to enhance the performance of a Zero-Shot classification with an LLM are limited.

3 Full Dataset Training

In this section, we perform a progressive fine tuning on DistilBERT models on increasing fractions of the dataset, to study the relationship between dataset size and performance. We also incorporate the data augmentation techniques (developed in part 2) into the smaller datasets, to asses its impact on different size levels.

In order to evaluate correctly the training with different percentages of the full dataset, we use a stratified sampling technique by which we keep the same proportion of labels — the one in the full dataset— for each percentage. This allows us to keep all experiments consistent, and later apply weighted loss functions.

Given the multi label classification nature of our problem, we use a Binary Cross Entropy **BCEWithLogitsLoss** function. This loss function differs from the more common **Cross Entropy Loss** because it treats each label prediction as an independent binary classification task, applying a separate sigmoid function to each output and computing binary cross-entropy loss per label. In contrast, Cross Entropy Loss assumes that each input belongs to exactly one class out of many and applies a softmax over all output logits, which is appropriate for single-label, multi-class problems. Therefore, BCEWithLogitsLoss is more suited for a multi-label classification problem like this one, where each comment can be simultaneously classified under multiple toxicity sublabels (e.g., both toxic and obscene at once).

The metrics we measure our model's performance on are the following:

- **Accuracy**: Measures hoe many of the total predictions are actually correct.
- **Precision**: Measures the percentage of the predicted positives are correct.
- **Recall**: Measures the proportion of actual positives that are identified correctly.

- **F1 Score**: The harmonic mean of Precision and Recall.

Given the extreme class imbalance in some of the dataset's labels, using plain Accuracy can be misleading, as predicting negatively every single instance for a class with only 5% positive passages would lead to a 95% Accuracy. Therefore, our most relevant metric is the F1 score.

The baseline results for the DistilBERT model are the following:

Class	Precision	Recall	F1 Score
`toxic`	0.2703	0.0033	0.0065
`severe_toxic`	0.0493	1.0000	0.0940
`obscene`	0.3461	0.6739	0.4573
`insult`	0.0000	0.0000	0.0000
`identity_hate`	0.0299	0.5573	0.0568
Overall (macro avg)	**0.4075**	**0.3621**	**0.3639**

Table 8: Baseline performance of DistilBERT

As we can observe, the classifier behaves pretty oddly. It classifies all texts as *severe_toxic*, given that it has a 100% Recall value but a very low Precision one, which matches exactly with its positive frequency. The other rare label (*identity_hate*) also gets an extremely low Precision. Even the *Toxic* label, the most balanced one, gets an unusually low performance. Therefore, it is clear that there is much room for improvement.

(a) Methodology Analysis

The three main loss functions analyzed are the weighted and unweighted version of the **Binary Cross Entropy Loss**, and the **Focal Loss**. To compare results, we do not only look at the overall metrics but at a detailed per-class level analysis. We compared both variants of the Binary Cross Entropy Loss for small size datasets, and then later tested the Focal Loss using 50% of the available dataset, comparing it against the best performing of the two others.

Below are some comparative results between the different methodologies.

1% Dataset

The first clear observation is a slight drop in the F1 score for the toxic label (from 0.8770 to 0.8446). This is likely a side effect of using the weighted loss: the model is now less biased toward the most frequent class and redistributes some of its representational capacity to minority classes.

Most notably, the rare classes show significant improvement. With weighted loss, these classes reach F1 scores of 0.4006 and 0.2324, respectively. This change is driven primarily by a boost in recall, indicating that the model is now at least predicting some positives for these labels.

However, precision for these classes remains very low—indicating high false positive rates. This suggests that while the model is encouraged to identify minority labels, it's still not confident or accurate enough to do so reliably.

Label	Metric	Unweighted BCE	Weighted BCE
toxic	Precision	0.8812	0.9082
	Recall	0.8728	0.7894
	F1-Score	0.8770	0.8446
severe_toxic	Precision	0.0000	0.2603
	Recall	0.0000	0.8688
	F1-Score	0.0000	0.4006
obscene	Precision	0.7621	0.6940
	Recall	0.7907	0.8300
	F1-Score	0.7761	0.7559
insult	Precision	0.6716	0.6019
	Recall	0.7056	0.8165
	F1-Score	0.6881	0.6930
identity_hate	Precision	0.0000	0.1446
	Recall	0.0000	0.5916
	F1-Score	0.0000	0.2324

Table 9: Performance comparison of unweighted and weighted binary cross-entropy loss using 1% of the training dataset. Weighted BCE improves performance on minority classes.

Another insightful observation is the "staircase effect" in F1 scores: there's a visible trend where more frequent labels get systematically better results. This pattern supports the hypothesis that larger training sets will further improve rare class performance - both by providing more examples and by reinforcing the reweighted loss-, so we conclude that the weighted loss function seems to yield very positive results.

10% Dataset

Again, the Weighted loss function does show noticeable improvements, particularly for the rare classes.

- **severe_toxic** achieves an F1 score of 0.53 , with a notably high recall of 0.63 , indicating the model is now correctly identifying a significant portion of the rare positive cases.
- **identity_hate** also improves to an F1 of 0.58 , with recall above 0.62 , which is among the best values obtained for this label so far.

Meanwhile, the performance of more frequent classes such as 'toxic', 'obscene', and 'insult' remains stable and high, demonstrating that reweighting the loss does not negatively impact their classification for larger datasets (recall that overall F1 score saw a small drop when using the weighted loss for the 1% dataset)

These results confirm that applying a weighted BCEWithLogitsLoss helps the model pay closer attention to rare classes without sacrificing accuracy on common ones. Among all the augmentation and tuning strategies tested at 10%, **weighted loss yields the most effective improvement**, especially when balanced performance across all classes is a priority.

Label	Metric	Unweighted BCE	Weighted BCE
toxic	Precision	0.8883	0.8712
	Recall	0.9252	0.9341
	F1-Score	0.9064	0.9015
severe_toxic	Precision	0.5823	0.4593
	Recall	0.4531	0.6344
	F1-Score	0.5097	0.5328
obscene	Precision	0.8415	0.8002
	Recall	0.8415	0.8845
	F1-Score	0.8415	0.8402
insult	Precision	0.7575	0.6894
	Recall	0.7473	0.8358
	F1-Score	0.7523	0.7556
identity_hate	Precision	0.7027	0.5488
	Recall	0.3969	0.6221
	F1-Score	0.5073	0.5832

Table 10: Performance comparison of unweighted and weighted binary cross-entropy loss using 10% of the training dataset. Weighted BCE enhances recall and F1 for underrepresented labels.

However, we still observe significant differences in Precision and Recall values for the rare classes, which suggests we could modify the threshold for the logit value at which we decide whether an instance is predicted to be positive or negative.

It is also interesting to observe that, despite consistent improvements in F1 score and other evaluation metrics across epochs, the validation loss does not consistently decrease. At first glance this might seem counterintuitive, yet it is not necessarily something to worry about, as the two values measure slightly different behaviors.

F1 computes the scores for the binary predictions, after applying the 0.5 threshold; this means that, for each label, if the probability of it being positive is above 0.5, the sigmoid function turns it into a 1, and the F1 score only "sees" that binary 1 or 0 classification.

In contrast, the binary cross-entropy loss takes the probability into account. This means that a wrong prediction at probability 0.99 is penalized far more than another one at 0.51. Therefore, what we learn from the loss function increase is that the model steadily becomes more confident as epochs advance, but this does not mean that results become worse. To analyze that, we look at the per class metrics.

Focal Loss

Looking at results in Table 11 , using a **Focal loss does not seem to improve results** over our Weighted Binary Cross Entropy loss. Most classes see marginal differences between losses, except for *identity_hate*, for which there is a stronger gap. However, the focal loss shows a more unbalanced relation between Recall and Precision for this class (0.81 - 0.72 , while BCE gets 0.72 - 0.74). Therefore, we do not consider the performance gains to be significant, thus we keep our weighted BCE as

Label	Metric	Focal Loss	Weighted BCE
toxic	Precision	0.9323	0.9137
	Recall	0.9578	0.9634
	F1-Score	0.9449	0.9379
severe_toxic	Precision	0.6667	0.6188
	Recall	0.6562	0.7000
	F1-Score	0.6614	0.6569
obscene	Precision	0.8896	0.8702
	Recall	0.9117	0.9286
	F1-Score	0.9005	0.8984
insult	Precision	0.8458	0.8115
	Recall	0.8339	0.8724
	F1-Score	0.8398	0.8408
identity_hate	Precision	0.8120	0.7201
	Recall	0.7252	0.7366
	F1-Score	0.7661	0.7283

Table 11: Comparison of classification performance using Focal Loss vs. Weighted BCE at 75% dataset size. Focal Loss improves recall and F1 on most minority classes.

the best performing model.

(b) Dataset Augmentation Techniques

Overall, none of the three data augmentation techniques lead to significant performance gains. Results remain largely similar to the baseline yet they tend to be slightly noisier, and rare classes still remain mostly undetected. The only exception is a non-zero F1 score for the severe_toxic class using the LLM-generated data, marking a small but notable improvement.

The Back-translated technique uses the same examples in the original 1% dataset, with minor semantic modifications. This does not expose the model to new or insightful instances, which limits its utility in our context. This leads us into thinking that this technique does not align very well with our necessities, as we require very different and varied examples.

Regarding the Zero-Shot labeled approach, a key limitation is that its performance is capped by the quality of the classifier used to assign labels, which reduces its added value compared to acquiring genuinely new labeled examples. Seeing that results have not significantly improved, the limitations of this method seem to prevent it from being useful.

In contrast, the LLM synthetically generated dataset seems to be the only one with a slightly positive effect. LLMs offer more contextually nuanced and diverse samples, making them a potentially promising tool when no additional labeled data is available. While the improvements seem to be limited, this has proved to be the only one capable of enhancing rare class detection and therefore overall performance.

Even at the 10% training-set size, none of the three augmentation strategies produces a substantial lift over the baseline. Overall F1 for the frequent labels (toxic,

obscene, insult) fluctuates by at most ±0.02, indicating that the extra data neither helps nor harms those classes. The rare labels show only marginal movement: severe_toxic varies around 0.50 F1 and identity_hate remains in the 0.47 – 0.51 range, for any augmentation method. Therefore, we can conclude that the additional examples introduce a bit of noise but do not inject enough truly insightful signal to increase performance. The takeaway from the 1% experiments therefore holds: these automatic augmentation methods deliver limited value once even a modest amount of real data (10 %) is available. Therefore, training with the remaining percentages of the dataset does not also incorporate these techniques, already proven to be much less helpful that acquiring new data.

(c) Learning Curve

Below are two different plots that allow to observe the evolution of F1 score both for the overall result, and stratified per classes.

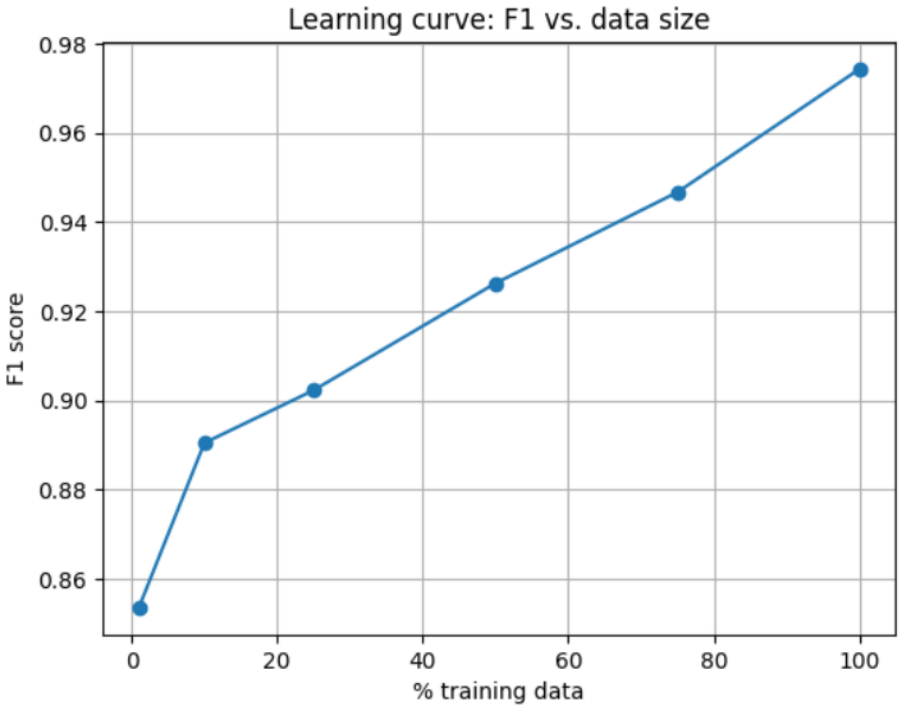

Figure 10: Learning Curve for the F1 score against different data sizes

These results confirm an early hypothesis (suggested after the 1% baseline) that the primary bottleneck in model performance was simply the limited amount of training data. As shown in the learning curve above, F1 score improves steadily and consistently as we increase the dataset size, with no signs of saturation up to 100

The smooth growth also suggests that the model has not yet fully plateaued, implying that even more data could perhaps further improve performance. Nonetheless, reaching an F1 score of 97.4% with 100% of the available data already represents an extremely strong result.

Looking deeper into the per-class metrics, we observe that **precision and recall are now highly aligned** , even for rare classes. This balance indicates that the model has

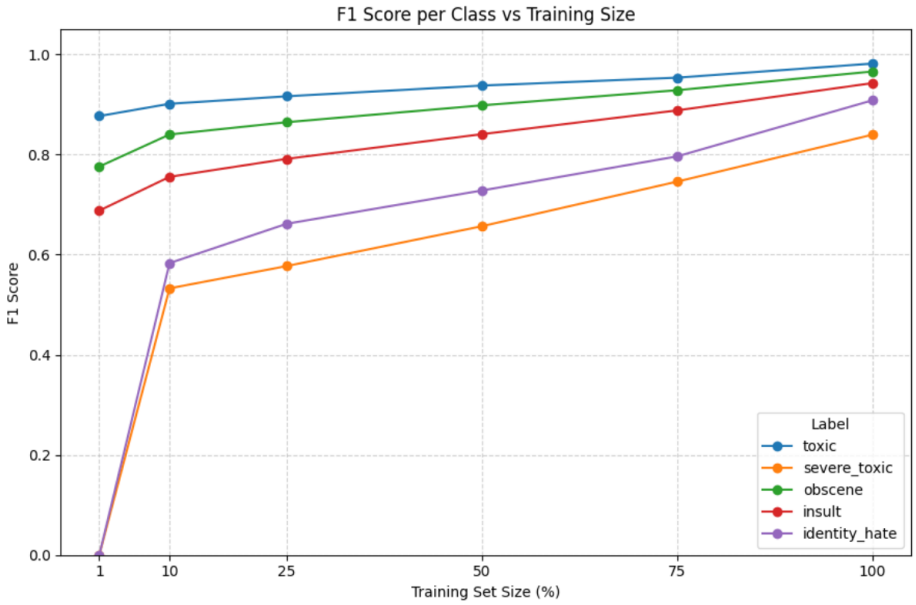

Figure 11: Learning Curve for the F1 score, stratified per class

learned not only to detect positives reliably, but also to avoid excessive false positives. This is largely thanks to the **weighted loss function**, which helps the model better handle label imbalance during training.

Overall, this experiment validates both the effectiveness of fine-tuning DistilBERT on task-specific data, and the value of simple, principled adjustments like weighted loss over more complex data augmentation schemes.

4 Model Distillation/Quantization

(a) Distill/Quantize best-performing model

To quantize our best-performing model, we used the official PyTorch quantization toolkit (torch.quantization). Specifically, we applied dynamic quantization, which reduces model size and inference time by converting weights from 32-bit floating point to lower-precision integers (8-bit).

Unlike static quantization, dynamic quantization determines the activation scale factors at runtime, based on the actual range of observed data. According to the PyTorch documentation (PyTorch, 2020) [5], this allows for better preservation of signal fidelity, as the quantization parameters adapt to the inputs during inference.

(b) Performance and Speed Comparison

Speed

The inference test was done on the validation data (6490 rows) we can see that the quantized model is approximately 34% faster.

Table 12: Inference Time Comparison Between FP32 and INT8 Model

Model	Time per Batch (ms)	Relative Speedup
FP32 (Full Precision)	452.98	1.00x
INT8 (Quantized)	339.20	**1.34x**

Aggregated Performance Comparison

Table 13: Performance Comparison: FP32 vs INT8 Model

Metric	FP32 (Full Precision)	INT8 (Quantized)
Accuracy	0.9826	0.9660
Precision	0.9728	0.9638
Recall	0.9750	0.9326
F1 Score	0.9739	0.9471

Per-Class Performance Comparison

Table 14: Per-Class Evaluation Metrics: FP32 vs INT8 Models

Label	FP32 (Full Precision)			INT8 (Quantized)		
	Precision	Recall	F1-score	Precision	Recall	F1-score
`toxic`	0.9784	0.9842	0.9813	0.9825	0.9268	0.9539
`severe_toxic`	0.8302	0.8406	0.8354	0.9027	0.3187	0.4711
`obscene`	0.9593	0.9704	0.9648	0.9416	0.9564	0.9490
`insult`	0.9410	0.9416	0.9413	0.9418	0.8922	0.9163
`identity_hate`	0.9129	0.9198	0.9163	0.9667	0.3321	0.4943

Overall, the performance loss from quantization is minimal, while achieving a 30–40% speedup in inference time on CPU. However, it's worth noting that recall dropped significantly for the severe_toxic and identity_hate labels, even though precision remained relatively strong. This likely stems from the fact that these classes are underrepresented in the dataset, comprising only 5.00% and 4.33% of the samples, respectively. As a result, the simplification introduced by quantization may have led to the model losing some of the finer-grained patterns needed to correctly identify these rare cases.

(c) Analysis and Improvements

By filtering cases where the full-precision model correctly predicted the labels severe_toxic and identity_hate but the quantized model failed, we visualize the differences in predicted probabilities in the following graph:

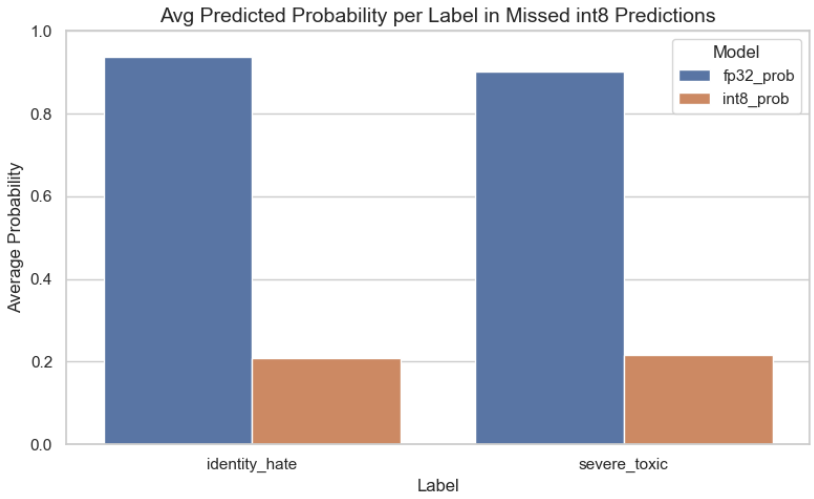

Figure 12: Average Predicted Probability per Label in Missed INT8 Predictions

To gain deeper insight into why the quantized model struggles with these examples, we will examine the five comments with the largest probability differences for each label. This analysis aims to uncover the factors driving the discrepancy in predictions between the two models.

Severe Toxic Discrepancy

Table 15: Top 5 `severe_toxic` Samples with Largest Prediction Probability Differences

Comment Text	FP32	INT8	Absolute Diff
Dumb Liberals. Liberals suck. I will beat you at tanks anytime you swastika wearing freak. Sieg Heil.	0.9857	0.0075	0.9782
Thanks for being late to the party but I have added no new articles since yesterday. While you have all been very impolite and overly protective, I have lost interest in torturing you, so you can go fuck yourself.	0.9778	0.0155	0.9623
wateva minga mc donalds worker u r a fuckin paedofile leave ant and dec alone just cause ur not famous u fuckin ugly bitch fuck ur hole family	0.9479	0.0023	0.9455
Fuck you all. This site is full of stuck up cunts, who for some unknown reason think being a part of wikipedia promotes them to President of the cyberspace. Sad pricks, go fuck yourselves. I won't be returning to this site. So off you all go to self congratulate yourselves.	0.9758	0.0364	0.9395
i DONT KNOW HOW THIS THING WORKS. BUT I'LL TELL YOU THIS. VERTIGO ACID. CALL ME FAMILY FLUFF. STOP THREATENING ME YOU TIGHT DRY LITTLE PUSSY. I OWN THIS FING CITY. AND I'LL OWN YOUR FING PUNK ASS IF YOU DONT STAY OUT OF MY BUSINESS. CAPICHE?	0.9857	0.0570	0.9286

The quantized model demonstrates difficulty accurately identifying uncommon or highly toxic language, which can be attributed to the reduction in precision inherent to quantization. For instance, comments like "Dumb Liberals ... swastika wearing freak. Sieg Heil." and "I DONT KNOW HOW THIS THING WORKS... I OWN THIS FING CITY..." contain informal, slang-heavy, or less common expressions. Quantized models typically struggle to generalize effectively on these out-of-distribution or atypical inputs due to the loss of representational fidelity caused by reduced numerical precision.

Identity hate Discrepancy

Table 16: Top 5 `identity_hate` Samples with Largest Prediction Probability Differences

Comment Text	FP32 Probability	INT8 Probability	Absolute Difference
sabu is a violent piece of crap dis guy is so violent. he hits people with steel chairs, triple jump moonsaults ppl thru tables. he could put som1 thru a burning table which could kill them. he was taught by his uncle, and he is from bombay, india.	0.9943	0.0020	0.9923
ur so immature fr3nzi3, im 17 and ur not evn debating. ur a 6 year old baby and im like an adult, nobody likes you. anyway, everybody doesn't like any fat assholes like u. grown-ups fight like a man, i fight using words like swearin' like eminem dos in is songs. u don't evn have a brain, shit wat a crazy guy.	0.9919	0.0113	0.9806
LOL, haha, you loser Wobbs, yeah maybe I am Epf.....NOT. Although he shares some views with me on race and other things, hes some dumb Italian-British mut from Canada that actually annoys me. I have used other accounts on here, but with other anon. accounts, not as a registered user (which is allowed by Wikiped(...)	0.9695	0.0018	0.9676
How dare you call my contribution spam!!! I am a Kurd and I made a list of kurdish dictionaries. you bloody turkish nationalist and atrocity committing bone breaking Nazi. watch out folk this slimy Turk is trying to censor the internet this is not undemocratic Turkey here, no prison cells in wikipedia you stupid Turk! And you buggers want membership to the EEC	0.9922	0.0349	0.9573
Listen you piece of shit, pencil necked twerp that you are. Nowhere did I make any further attacks to anyone. How about you get a fuckin life rather than getting a little stiffy from banning people. I couldn't give a slight fuck about your(...)	0.9592	0.0035	0.9557

For identity hate a similar thing is happening, these comments are highly charged with specific language against ethnic or religious groups (e.g., "Kurd, "Turkish nationalist," "christian piece of shit"). The full model picks up these signals confidently, while quantization likely blunts or loses sensitivity to the key hateful keywords or phrases.

Potential Improvements or Further Research Directions

For suggested improvements, Quantization-Aware Training (QAT) provides a compelling alternative to post-training quantization by integrating quantization effects

directly during the training phase. This method helps the model develop representations that are resilient to reduced numerical precision, thereby preserving higher accuracy—particularly for subtle or rare toxic cases that often degrade most after post-training quantization.

Although knowledge distillation was not utilized in this work, it remains a valuable avenue for enhancement. Advanced distillation techniques transfer not only the final outputs but also intermediate representations such as hidden layer activations or attention maps. Employing a larger, full-precision teacher model to guide a smaller or quantized student model enables the student to capture more nuanced patterns. This deeper mimicry can improve classification performance, especially on complex or subtle examples.

In our approach, quantization was selectively applied only to the linear layers, leaving other components at full precision. While this selective quantization yields computational speedups, some performance degradation was still observed. To address this, exploring mixed-precision quantization—where critical layers like embeddings or attention modules remain in higher precision (e.g., FP16 or FP32) while less sensitive layers are quantized—could better balance computational efficiency with accuracy retention.

5 References

[1] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. arXiv preprint arXiv:1910.01108, 2019.

[2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. arXiv preprint arXiv:1909.11942, 2019.

[3] Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. *A robustly optimized BERT pre-training approach*. arXiv preprint arXiv:1907.11692, 2021.

[4] Corentin Duchene, Henri Jamet, Pierre Guillaume, and Reda Dehak. *A benchmark for toxic comment classification on Civil Comments dataset*. arXiv preprint arXiv:2301.11125, 2023.

[5] PyTorch Team. *Dynamic Quantization — PyTorch Tutorials 2.2.0+cu121 documentation*. Available at: https://docs.pytorch.org/tutorials/recipes/recipes/dynamic_quantization.html, 2023.