CAS2105 Homework 6: Mini AI Pipeline Project
(Topic: Toxic Comment Detection)

Jiyun Jung (2024149021)

1 Introduction

In this project, I designed and implemented an AI pipeline to perform **Toxic Comment Detection**. The primary objective was to classify user comments into binary categories: **Toxic** (harmful, obscene, threatening) or **Non-toxic** (neutral, constructive).

To understand the effectiveness of modern AI in content moderation compared to traditional methods, I implemented and evaluated two distinct approaches:

1. A **Naïve Baseline** that uses a simple keyword filter based on a "blacklist" of offensive words.
2. An **AI Pipeline** leveraging a pre-trained RoBERTa model fine-tuned for toxicity detection.

I evaluated both methods on a **balanced subset** of the `Civil Comments` dataset (50 toxic and 50 non-toxic samples). Using a balanced dataset allows for a fair comparison by preventing the baseline from achieving artificially high accuracy simply by predicting the majority class. This project focuses on the end-to-end process of problem definition, model selection, and result analysis using robust metrics such as Accuracy and F1-score.

The key aspects of this project are:

- **Task:** Toxic Comment Detection (Binary Text Classification).
- **Motivation:** Online platforms suffer from harassment and hate speech. Automated detection is critical for maintaining healthy online communities without relying solely on manual moderation.
- **Goal:** To demonstrate how pre-trained Transformer models outperform simple keyword filters, especially in detecting nuanced or obfuscated toxicity.

2 Task Definition

In this section, I define the specific problem addressed in this project, based on the real-world challenge of content moderation [1].

- **Task Description:** The objective is to perform **binary classification** on user-generated comments to determine if they are **Toxic** (rude, disrespectful, or unreasonable) or **Non-toxic**.
- **Motivation:** Online harassment effectively silences voices and makes platforms unwelcoming. Manual moderation is slow and unscalable. An automated system can help maintain civil discourse by quickly flagging harmful content.
- **Input / Output:**
  - **Input:** A text string containing a user comment from an online discussion.
  - **Output:** A binary label (1 for Toxic, 0 for Non-toxic).

- **Success criteria:** The primary metrics for success are **Accuracy** and **F1-score** on a held-out, balanced test subset of the Civil Comments dataset. F1-score is particularly important to ensure the model captures toxic comments (recall) rather than just predicting the majority class.

3 Methods

This section describes the two distinct approaches implemented: a heuristic keyword filter and a deep learning-based pipeline.

3.1 Naïve Baseline

I implemented a simple dictionary-based filtering method, often referred to as a "blacklist" approach.

Blacklist Keyword Filter

- **Method description:** I defined a list of common offensive keywords (e.g., "stupid", "idiot", "shut up", "kill"). The function checks if the input text contains any of these substrings. If a match is found, the comment is predicted as Toxic (1); otherwise, it is predicted as Non-toxic (0).
- **Why naïve:** This method is considered naïve because it relies on **exact string matching**. It fails to understand the intent or context of the comment. It cannot handle typos, obfuscations (e.g., using symbols), or toxicity expressed without using specific bad words.
- **Likely failure modes:**
  - **Obfuscation:** Users often bypass filters by altering spellings (e.g., "sh*t", "morons" not in list).
  - **Implicit Toxicity:** Insults can be conveyed through sarcasm or hateful phrasing without using explicit profanity (e.g., "You are not very smart").
  - **False Positives:** Words on the blacklist might be used in a non-toxic context (e.g., quoting someone else or discussing the word itself).

3.2 AI Pipeline

I designed an improved pipeline using a pre-trained Transformer model from the Hugging Face library, specifically fine-tuned for toxicity detection.

RoBERTa Toxicity Classifier

- **Models used:** I utilized the `s-nlp/roberta_toxicity_classifier`. This is a RoBERTa (Robustly optimized BERT approach) model [2] that has been fine-tuned on the Jigsaw Toxic Comment dataset.
- **Pipeline stages:**
  1. **Preprocessing (Tokenization):** The raw input text is truncated to a maximum length of 512 characters to satisfy the model's input constraints. The text is then split into sub-word units using the RoBERTa tokenizer.
  2. **Inference:** The token IDs are passed through the Transformer layers to generate contextualized representations of the sentence.

3. **Decision:** The model outputs confidence scores for the labels "toxic" and "neutral". I implemented a threshold logic where the final prediction is **Toxic (1)** if the model predicts "toxic" or the probability score exceeds 0.5, and **Non-toxic (0)** otherwise.

- **Design choices and justification:**
  - **Choice of Architecture:** I chosen RoBERTa because it generally outperforms standard BERT on text classification tasks due to its optimized pre-training procedure.
  - **Transfer Learning:** Instead of training a model from scratch (which requires massive data and compute), I used a model already fine-tuned on a similar task. This approach allows for high performance with zero training cost.
  - **Handling Context:** Unlike the baseline's keyword matching, the Transformer model uses self-attention mechanisms to understand the context. This allows it to detect implicit toxicity (e.g., insults without swear words) and ignore benign uses of flagged words.

4 Experiments

4.1 Datasets

I utilized a subset of a public dataset available on the Hugging Face Hub to evaluate the pipeline.

Civil Comments Dataset

- **Source:** I used the `google/civil_comments` dataset [3]. This dataset originated from the Jigsaw Unintended Bias in Toxicity Classification Challenge and contains public comments from news websites.
- **Total examples:** I constructed a **balanced subset** of 100 examples, consisting of exactly **50 toxic** and **50 non-toxic** samples.
  - *Note on Balancing:* In initial tests with random sampling, the dataset was heavily skewed towards non-toxic comments, allowing the Naïve Baseline to achieve high accuracy (88%) simply by predicting "Non-toxic". Constructing a balanced set ensures a fair evaluation of the model's true ability to detect toxicity.
- **Train/Test split:** Since the AI pipeline relies on a pre-trained model for inference (without fine-tuning), no training split was required. The entire balanced subset of 100 samples was treated as a **Test Set** for evaluation.
- **Preprocessing steps:**
  - **Label Conversion:** The original dataset provides a continuous toxicity score (0.0 to 1.0). I converted this to a binary label: **Toxic (1)** if the score was ≥ 0.5 and **Non-toxic (0)** otherwise.
  - **Truncation:** Input texts were truncated to a maximum of 512 characters to fit the input constraints of the RoBERTa model.

4.2 Metrics

I selected two quantitative metrics to evaluate the performance of the models:

- **Accuracy:** This measures the overall ratio of correct predictions. While useful, it can be misleading if the dataset is imbalanced.
- **F1-Score:** I also measured the F1-score, which is the harmonic mean of precision and recall. This metric is particularly important for toxicity detection because we want to ensure the

model actually *catches* toxic comments (high recall) rather than just being conservatively correct (high precision but low recall).

4.3 Results

The experimental results demonstrate that the AI Pipeline significantly outperformed the Naïve Baseline, particularly on the balanced dataset used in this experiment.

Quantitative Results

As shown in Table 1, the Naïve Baseline achieved an Accuracy of only 0.61 and a significantly low F1-Score of 0.40. This indicates that the keyword filter failed to detect a large portion of toxic comments (low recall). In contrast, the AI Pipeline achieved robust performance with 0.86 Accuracy and 0.84 F1-Score.

Table 1: Performance Comparison on Balanced Test Set (50 Toxic / 50 Non-toxic)

Method	**Accuracy**	**F1-Score**	**Note**
Naïve Baseline	0.61	0.40	Keyword Filter
AI Pipeline	**0.86**	**0.84**	Pre-trained RoBERTa

Qualitative Analysis (Failure Cases)

The following examples highlight specific cases where the **Baseline failed** (predicted Non-toxic) but the **AI Pipeline succeeded** (predicted Toxic).

- **Case 1 (Missing Keyword):** "They are terrorists pure and simple...."
  - *Baseline:* Predicted Non-toxic (Incorrect). The specific term "terrorists" was not included in the limited manual blacklist, despite its hateful usage in this context.
  - *AI Pipeline:* Predicted Toxic (Correct). The model correctly identified the derogatory nature of labeling a group as terrorists.
- **Case 2 (Limited Vocabulary):** "I honestly cannot decide if these guys are complete morons..."
  - *Baseline:* Predicted Non-toxic (Incorrect). The insult "morons" was not in the fixed `bad_words` list, illustrating the limitation of dictionary-based approaches.
  - *AI Pipeline:* Predicted Toxic (Correct). The pre-trained model recognizes a vast vocabulary of insults, including "morons".
- **Case 3 (Contextual Toxicity):** "Their ridiculous band photo has me wanting to drink white wine..."
  - *Baseline:* Predicted Non-toxic (Incorrect). The sentence contains no explicit swear words but uses mocking language ("ridiculous") combined with a sarcastic tone.
  - *AI Pipeline:* Predicted Toxic (Correct). The model understood the mocking tone and context of the comment.

5 Reflection and Limitations

Your Reflection

Through this project, I observed a significant performance gap between the rule-based heuristic and the pre-trained Transformer model. The most critical insight gained was the importance of **experimental design** and data distribution. Initially, the Naïve Baseline achieved a misleadingly high accuracy (88%) on random data simply by predicting the majority class (Non-toxic). However, after I constructed a **balanced test set** (50 Toxic / 50 Non-toxic), the baseline's true weakness was revealed (Accuracy dropped to 61%, F1-Score to 0.40), while the AI pipeline maintained robust performance (Accuracy 0.86, F1-Score 0.84).

This experience demonstrated that **F1-score** is a far superior metric than Accuracy for detection tasks, as it penalizes models that fail to retrieve the target class (low recall). While the AI model worked better than expected in detecting implicit toxicity (e.g., context-dependent insults like "terrorists"), the project was limited by the small sample size (100 examples) and the lack of a fairness evaluation. If I had more time, I would conduct a **bias analysis** to ensure the model does not unfairly flag neutral mentions of specific identity groups (e.g., race or religion) as toxic, which is a known limitation of pre-trained language models.

References

[1] Kaggle. Jigsaw toxic comment classification challenge. https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge, 2018. Accessed: 2025-12-09.

[2] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. Base architecture for the AI Pipeline.

[3] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of the The Web Conference 2019*, pages 491–500, 2019. Source of the Civil Comments dataset.