Logistic Regression and DistilBERT for Classification of Social Media Data

Faizan Waheed
fawaheed
February 16, 2025

Abstract

*This report details the development of two machine learning models for the classification of comments into toxic or non-toxic across various social media platforms. The models include a fine-tuned **DistilBERT** transformer and a classical **Logistic Regression** model. The report covers dataset characteristics, preprocessing steps, model selection, evaluation metrics, and results. DistilBERT outperformed Logistic Regression on accuracy, precision, recall, and F1-score. The best-performing model's predictions were submitted as required.*

1. Introduction and Background

Toxicity detection in online discourse has become increasingly important due to the rise of online discussions on social media platforms, news forums, and public comment sections. Identifying and filtering out toxic comments can help create a more constructive and respectful digital space. However, toxic speech is often nuanced, context-dependent, and can involve sarcasm, coded language, or implicit threats, making it challenging to detect accurately.

The widespread presence of toxic comments has necessitated the development of automated tools capable of efficiently moderating online discussions. Manual moderation is labor-intensive and prone to bias, making machine learning (ML) a practical solution. Recent advancements in **Natural Language Processing (NLP)** have enabled the development of more sophisticated models that can better capture the intricacies of language. Traditional **rule-based approaches** have largely been replaced by **machine learning models** that learn patterns in textual data. Among these, **transformer-based deep learning models** such as **BERT and its variants (DistilBERT, RoBERTa, etc.)** have set new benchmarks in NLP tasks, including text classification.

This project focuses on **classifying social media comments as toxic or non-toxic** using two distinct machine learning approaches:

1. **Logistic Regression** - A lightweight, interpretable baseline model that uses statistical methods to determine the toxicity of comments based on TF-IDF features.
2. **Fine-Tuned DistilBERT** - A transformer-based deep learning model pre-trained on large textual data, capable of capturing contextual meanings of words and sentences.

The objective of this study is to evaluate and compare the performance of these models in detecting toxic comments. The comparison will help in understanding the trade-offs between deep learning models and traditional statistical learning models, considering factors such as **accuracy, computational efficiency, interpretability, and generalization to unseen data**.

By leveraging a dataset comprising **4,000 comments from Reddit, Twitter/X, and YouTube**, this study aims to determine which approach is best suited for the toxicity classification task. The best-performing model will be used to classify an unseen test dataset, and predictions will be submitted as required.

Definition of Toxic
"A rude, disrespectful, or unreasonable comment that is likely to make readers want to leave a discussion."

2. Dataset and Methods

To ensure a structured and rigorous approach to toxicity classification, this section outlines the dataset details, preprocessing steps, and the machine learning models employed in the study. The dataset subsection describes the nature and format of the data, while the methods subsection details the techniques used for data processing and model training

2.1 Dataset

The dataset consists of **4,000 comments** from platforms including **Reddit, Twitter/X, and YouTube**, with annotations indicating whether a comment is toxic. Each comment is accompanied by metadata such as parent comment, article title, and platform ID. The **test dataset** follows the same structure but lacks toxicity labels.

Each comment in the training dataset includes a **composite toxicity score**, obtained through multiple human annotations. This enables a more robust ground truth, reducing individual annotator bias. However, inconsistencies in annotations highlight the subjective nature of toxicity detection, making it a challenging task for automated models.

Training Dataset Details

- **Text**: The main body of the comment.
- **Parent Comment**: The preceding comment if the text is a reply; null otherwise.
- **Article Title**: The title of the article or post the comment is responding to.
- **Article URL**: The direct link to the associated news article or post.
- **Platform**: The social media site where the comment was posted (Reddit, Twitter/X, YouTube, etc.).
- **Platform ID**: A unique identifier associated with the comment.

- **Composite Toxicity**: A set of five human annotations indicating whether a comment is toxic (true) or non-toxic (false), along with the corresponding Amazon MTurk worker ID.

Example Data Entry

{ "text": "WTF, y'all never made MRE fart balloons in the stumps?\n\nFucking kids these days.",
"parent_comment": null,
"article_title": "Triangular UFO hovers over California military base in new footage",
"article_url": "https://www.dailymail.co.uk/news/article-12112321/Black-half-football-field-sized-triangular-UFO-hovers-California-military-base-video.html",
"platform": "reddit",
"platform_id": "jlcm021",
"composite_toxic": [[false, 74], [true, 323], [false, 1028], [false, 324], [true, 1068]]
}

Test Dataset Details

The **test dataset** follows the same structure as the training dataset, **except it does not include the composite_toxic field** since it is the target variable to be predicted. The goal is to use trained models to classify whether each test comment is toxic or not.

Key Differences Between Training and Test Datasets

- The **training dataset** has **human-labeled toxicity annotations** (composite_toxic field), which is used to train models.
- The **test dataset** only includes the text and metadata; toxicity labels are **absent** and must be predicted by the model.

2.2 Data processing

Data preprocessing is a crucial step to ensure the dataset is in an optimal format for machine learning models. This process involves text cleaning, feature extraction, and encoding labels to prepare the data for training.

The following preprocessing steps were performed for both models:

1. **Data Loading**
   - The dataset was loaded from a JSON file into a Pandas DataFrame to facilitate structured data handling.

2. **Label Processing**
   - Each comment had multiple human annotations indicating whether it was toxic or not.
   - The **majority vote approach** was used to assign a final toxicity label: if more than half of the annotators labeled a comment as toxic, it was considered toxic.
   - Labels were then encoded as binary values: 0 for non-toxic and 1 for toxic.

3. **Train-Validation Split**
    - To ensure fair model evaluation, the dataset was split into **80% training** and **20% validation** while maintaining class balance (stratified sampling).
    - This ensures that both toxic and non-toxic comments are well-represented in each dataset.

Preprocessing for Logistic Regression

1. **Text Cleaning**
    - Special characters, punctuation, and extra whitespace were removed.
    - All text was **lowercased** to ensure uniformity.

2. **Feature Engineering (TF-IDF Vectorization)**
    - Text was transformed into numerical features using **TF-IDF (Term Frequency-Inverse Document Frequency)**.
    - The following parameters were used:
        - max_features=20000 (use the top 20,000 most important words)
        - ngram_range=(1,3) (consider single words, bigrams, and trigrams)
        - stop_words="english" (remove common words like "the" and "is")
        - min_df=3, max_df=0.9 (filter words appearing too rarely or too frequently).

3. **Model Training**
    - Logistic Regression was trained using **L2 regularization (Ridge regression)** to prevent overfitting.
    - **Class weights were balanced** to handle potential imbalances between toxic and non-toxic comments.

Preprocessing for DistilBERT

1. **Tokenization**
    - Since transformer models like DistilBERT require tokenized input, each comment was converted into tokens using DistilBertTokenizer.
    - Tokenization includes **padding** (to standardize input length) and **attention masks** (to differentiate actual words from padding).
2. **Dataset Formatting**
    - A **custom dataset class** was created in PyTorch to format the inputs appropriately for training.
    - The dataset includes:
        - input_ids: Tokenized representation of the comment.
        - attention_mask: Identifies actual words versus padding.
        - labels: Encoded toxicity labels.

3. **Batch Processing**
   - The processed data was grouped into batches using PyTorch's DataLoader to optimize memory use during training.

These preprocessing steps ensured that both models received optimized input representations, improving their ability to accurately classify toxic comments.

2.3 ML methods

This section describes the machine learning models used in this study and their fundamental principles. The goal was to explore two distinct approaches to text classification: a classical statistical model (Logistic Regression) and a deep learning-based model (DistilBERT)

Logistic Regression (Classical Machine Learning Approach)

Logistic Regression is a traditional linear model used for binary classification. It models the probability that an instance belongs to a particular class using the **sigmoid function**:

$$P(y = 1 \mid x) = \frac{1}{1 + e^{-(wTx+b)}}$$

Where:

Where:

Where:

- ***P(y=1/x)*** *→ The probability that the comment is toxic.*
- **x** *→ The input feature vector representing the comment in numerical form.*
- ***w*** *→ The learned weights associated with each feature.*
- ***b*** *→ The bias term that helps adjust the decision boundary.*
- ***ee*** *→ The mathematical constant (approximately 2.718), which is the base of the natural logarithm.*

This function produces a probability between **0 and 1**. A threshold (typically **0.5**) is applied:

- If P(y=1|x)>0.50, the comment is classified as **toxic**.
- Otherwise, it is classified as **non-toxic**.

Implementation Steps:

1. **Text Preprocessing**:
   - Text was converted to lowercase and cleaned by removing special characters.
   - Tokenization was performed using **TF-IDF (Term Frequency-Inverse Document Frequency)** to convert text into numerical vectors.
2. **Feature Engineering (TF-IDF Vectorization)**:
   - The following settings were used:
     - max_features=20000: Uses the top 20,000 most frequent words.

- stop_words="english": Removes common English stop words.
- ngram_range=(1,3): Extracts unigrams, bigrams, and trigrams.
- sublinear_tf=True: Applies logarithmic scaling of term frequency.
- min_df=3: Ignores words appearing in fewer than 3 documents.
- max_df=0.9: Ignores words appearing in more than 90% of documents.

3. **Transformation Process:**
   - `vectorizer.fit_transform(train_df['text'])`: Learns vocabulary from training data.
   - `vectorizer.transform(val_df['text'])`: Transforms validation data using the same vocabulary.
4. **Model Parameters**:
   - max_iter=1000: Number of iterations to ensure convergence.
   - C=0.0001: Controls regularization strength (lower values increase regularization).
   - solver="saga": Optimized solver for large-scale text classification.
   - class_weight="balanced": Adjusts weights for imbalanced classes.
   - penalty="l2": Uses **L2 regularization** to prevent overfitting.
5. **Model Results:**
   - **Accuracy:** Measures the proportion of correct predictions.
     Training: 80.56%, Validation: 75.62%.
   - **Precision**: Measures how many predicted toxic comments are actually toxic.
     Training: 60.95%, Validation: 52.60%.
   - **Recall:** Measures how many actual toxic comments were correctly identified.
     Training: 67.48%, Validation: 49.27%.
   - **F1 Score:** Balances precision and recall to provide a single performance measure.
     Training: 64.05%, Validation: 50.88%.
   - **Log Loss:** Measures how well the predicted probabilities align with actual labels, where lower values indicate better probability calibration.
     Training: 0.6931, Validation: 0.6931.
6. **Model Performance:**
   - The model shows **moderate performance** but exhibits signs of **overfitting**, as indicated by the gap between training and validation scores.
   - Accuracy remains **relatively high (75.62%)** on validation data, suggesting that the model is making correct predictions most of the time. However, the drop from training accuracy (**80.56%**) indicates that the model **does not generalize well** to unseen data.
   - Precision on validation (**52.60%**) is lower than training (**60.95%**), meaning the model **misclassifies a significant number of non-toxic comments as toxic (false positives)**. This could be problematic for real-world applications where precision is critical, such as content moderation.
   - Recall on validation (**49.27%**) is notably lower than training (**67.48%**), indicating that the model **fails to detect a substantial portion of actual toxic comments (false negatives)**. A low recall means that many harmful comments may go undetected.

- The F1 Score (**50.88% on validation**) suggests that the model does not effectively balance precision and recall. This score is lower than the training F1 (**64.05%**), reinforcing concerns about overfitting.
- Log Loss remains constant at **0.6931**, which is a red flag. This suggests that the model's probability estimates are **not improving**, meaning it is not confidently distinguishing between toxic and non-toxic comments.

Advantages of Logistic Regression:

- **Simple and Interpretable**: Provides clear insight into how features contribute to predictions.
- **Computationally Efficient**: Requires significantly less processing power than deep learning models.
- **Robust for Small Datasets**: Performs well even with limited labeled data.

Limitations:

- **Limited Context Awareness**: Cannot capture relationships between words as effectively as transformer models.
- **Lower Performance on Complex Data**: Struggles with nuanced and contextual language.

DistilBERT (Deep Learning Approach)

DistilBERT (Distilled BERT) is a transformer-based model designed to retain the power of **BERT** while being smaller and more computationally efficient. BERT (Bidirectional Encoder Representations from Transformers) revolutionized NLP by introducing the **self-attention mechanism**, allowing models to understand context in a bidirectional manner.

Key Features of DistilBERT:

- **Smaller and Faster**: 60% faster than BERT while retaining 97% of its accuracy.
- **Bidirectional Contextual Understanding**: Unlike traditional models, DistilBERT comprehends word meaning in context rather than relying on isolated word occurrences.
- **Transfer Learning**: The model is pre-trained on vast amounts of textual data and fine-tuned on our toxicity dataset.

Advantages of DistilBERT:

- **High Accuracy**: Outperforms traditional models in complex text classification tasks.
- **Context Awareness**: Captures semantic meaning rather than relying on simple word occurrences.
- **Efficiency**: Faster inference compared to full-sized BERT models.

Limitations:

- **Computationally Expensive**: Requires GPU acceleration for efficient training.

- **Large Training Data Requirement**: Performance is dependent on pretraining and fine-tuning on large datasets.

Implementation Steps:

1. **Pretrained Model**: We used distilbert-base-uncased, a **lighter version of BERT**, pretrained on a large corpus of text from Wikipedia and BooksCorpus. This allows the model to leverage **rich language representations** without requiring training from scratch

2. **Tokenization**: Each comment was tokenized using DistilBertTokenizer, which:
    - Converts text into **subword units** (e.g., "unbelievable" → ["un", "##believable"]) to handle unseen words.
    - **Truncates/pads** sequences to a **maximum length of 128 tokens**, ensuring consistency across inputs.
    - Uses **attention masks** to differentiate between real tokens and padded tokens, preventing unwanted attention shifts.

3. **Model Fine-Tuning**: DistilBERT was further trained using:
    - **Cross-Entropy Loss**: A standard loss function for **binary classification**, mapping model outputs into class probabilities.
    - **AdamW Optimizer**: A modified version of Adam, specifically designed for transformers to handle weight decay better.
    - **Batch Size of 8**: Chosen to balance computational efficiency while utilizing available GPU memory.
    - **Learning Rate of 2e-5**: A small learning rate to prevent catastrophic forgetting while fine-tuning.
    - **Weight Decay (0.3)**: Helps prevent overfitting by reducing the model's reliance on specific feature weights.
    - **Dropout Regularization (0.4)**: Increases robustness by randomly disabling **40% of neurons** during training, improving generalization.
    - **3 Training Epochs**: Strikes a balance between learning and overfitting, as transformer-based models typically do not require many epochs.

4. **Early Stopping**:
    - The model **monitors validation loss** after each epoch.
    - If **validation loss stops improving**, training is **automatically halted** to prevent overfitting.
    - Implemented with EarlyStoppingCallback(early_stopping_patience=1), ensuring training stops if no improvement is seen for **one consecutive epoch**.

5. **Evaluation & Model Selection**:
    - The best model is chosen based on **evaluation loss**, using load_best_model_at_end=True.
    - The classification threshold can be adjusted post-training for better recall/precision trade-offs.

This approach leverages **transfer learning**, allowing the model to detect toxic comments with **minimal labeled data and reduced training time**.

6. **Model Performance:**
    - **Training vs. Validation Loss:**
        - Training loss consistently decreases (0.4967 → 0.4545 → 0.4265), indicating that the model is learning patterns in the data.
        - Validation loss initially improves (0.4598 → 0.4558) but increases in the last epoch (0.4724), signaling possible overfitting after the second epoch.
    - **Accuracy Trend:**
        - The highest accuracy (**80.37%**) is observed in the **first epoch**, but it **drops** in later epochs (~78.25–78.37%), suggesting that training longer **does not improve generalization**.
        - The accuracy remains stable between **epochs 2 and 3**, meaning the model's decision boundary is not improving further.
    - **Precision vs. Recall Trade-off:**
        - **Epoch 1:** High **precision (0.7449)** but low **recall (0.3561)** → The model is conservative in predicting toxic comments, leading to **many false negatives**.
        - **Epoch 2:** Precision **drops (0.5633)**, but recall **significantly improves (0.6732)**, meaning the model detects **more toxic comments** at the cost of **more false positives**.
        - **Epoch 3:** Recall **slightly decreases (0.6146)**, while precision **remains stable (0.5727)**. The overall **F1 score also drops slightly**, suggesting the model is starting to overfit..
    - **Best Epoch Selection:**
        - Based on **Validation Loss & F1 Score**, **Epoch 2 (F1 = 0.6133, Validation Loss = 0.4558)** appears to be the **best-performing model** before overfitting starts.

3. Evaluations and Findings

Evaluating the performance of our models is essential to understand how well they classify toxic and non-toxic comments. We used several standard evaluation metrics to measure their effectiveness and identify areas for improvement.

3.1 Model Evaluation Metrics

We assessed the models using the following metrics:

- **Accuracy**: Measures how many predictions were correct out of all predictions made. A higher accuracy means the model is making fewer mistakes overall.
- **Precision**: The proportion of correctly predicted toxic comments out of all comments that were predicted as toxic. This is important to avoid falsely labeling non-toxic comments as toxic.
- **Recall**: The proportion of actual toxic comments that were correctly identified. High recall ensures that most toxic comments are caught by the model.
- **F1-Score**: The harmonic mean of precision and recall, balancing both metrics. A higher F1-score indicates a well-rounded model that performs well in both aspects.

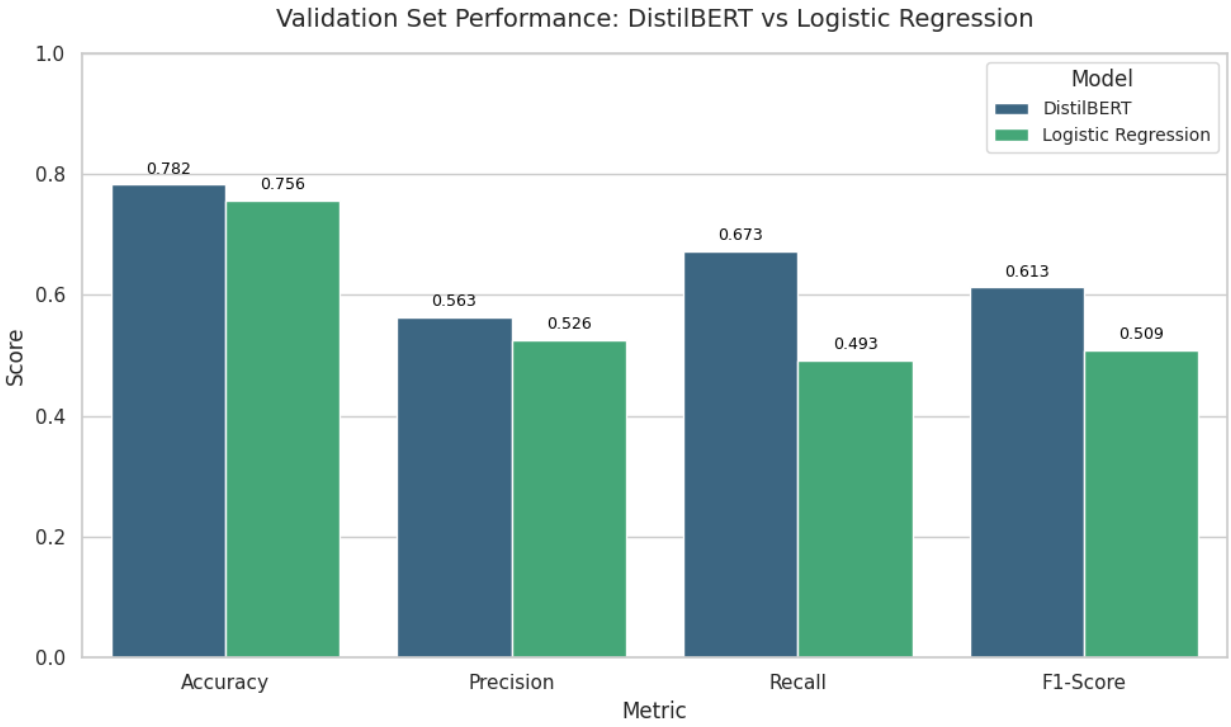
These metrics provide a comprehensive understanding of how well the models perform in classifying toxic comments while minimizing false positives and false negatives.

The models were assessed using:

- **Accuracy**: Percentage of correct classifications.
- **Precision**: Ratio of correctly predicted toxic comments to total predicted toxic comments.
- **Recall**: Ability to correctly identify actual toxic comments.
- **F1-Score**: Harmonic mean of precision and recall, balancing false positives and false negatives.

3.2 Model Performance Comparison

After evaluating the models on the validation dataset, we obtained the following results:



Findings:

- DistilBERT outperformed Logistic Regression across all key metrics, including accuracy, precision, recall, and F1-score.
- DistilBERT achieved a higher **accuracy (78.2%)** compared to Logistic Regression (75.6%), indicating that it made more correct predictions overall. The most significant difference was in **recall**, where DistilBERT scored **67.3%**, significantly higher than Logistic Regression's **49.3%**. This suggests that DistilBERT was more effective in detecting toxic comments and reducing false negatives, ensuring fewer toxic comments went undetected.
- Precision was also slightly higher for DistilBERT at **56.3%** compared to Logistic Regression's **52.6%**, meaning it was slightly better at avoiding false positives, though the difference was not as pronounced as recall. The F1-score, which balances precision and recall, was **61.3% for DistilBERT** compared to **50.9% for Logistic Regression**, confirming that DistilBERT provided a better trade-off between capturing toxic comments and minimizing false positives.
- Logistic Regression struggled with capturing **nuanced language, implicit toxicity, and contextual cues**, leading to lower recall and F1-score. Being a transformer-based model, **DistilBERT leveraged contextual understanding**, making it better at detecting toxicity in more complex linguistic structures, such as sarcasm and indirect insults.

- While Logistic Regression is **computationally efficient**, its lower recall suggests that it **failed to detect a significant number of toxic comments**. DistilBERT, although more resource-intensive, demonstrated superior performance in recognizing context-dependent toxicity, making it a **more reliable choice for toxic comment detection**.

Based on these findings, **DistilBERT was chosen as the final model** to generate predictions for the test dataset due to its superior ability to detect toxic comments accurately while maintaining an acceptable balance between false positives and false negatives.

4. Test Dataset Predictions

Predictions on the **test dataset** were generated using the **fine-tuned DistilBERT model**. Each comment was classified as **true (toxic) or false (non-toxic)** and stored in distilbert_predictions.csv

5. Conclusion

In this project, we implemented and compared **DistilBERT** and **Logistic Regression** for toxic comment classification. DistilBERT demonstrated superior performance and was selected for test dataset predictions. Future improvements could include **data augmentation**, **fine-tuning on a larger toxic comment dataset**, and **experimenting with ensemble models for enhanced robustness**.