Troll vs Genuine User Detector

DevifyX NLP Job Assignment Report

1. Introduction

Online communities and social platforms are increasingly challenged by toxic behavior and trolling, which can disrupt conversations and harm users. Detecting trolls—users who intentionally provoke, insult, or derail discussions—is critical for maintaining healthy digital environments. This project aims to design and implement a Natural Language Processing (NLP) system capable of classifying online users as Trolls or Genuine Users based on their text behavior and toxicity levels. The system leverages text aggregation, feature engineering, toxicity scoring, and supervised machine learning to provide robust, explainable classifications.

2. Approach

2.1 Data Collection and Preprocessing

Dataset:
We used the Kaggle Dataset for Detection of Cyber-Trolls in JSON Lines format. Each entry contains a content field (user message) and an annotation dictionary with a label field indicating whether the message is toxic (1) or not (0). Since the dataset is message-based and not user-based, we simulated user IDs by grouping every 5 messages as one user.

Preprocessing Steps:

- **Tokenization:** Splitting text into individual words.
- **Lowercasing:** Normalizing all text to lowercase.
- **Stopword Removal:** Removing common English words with little semantic value.
- **Lemmatization:** Reducing words to their root forms.
- **Noise Removal:** Stripping URLs, special characters, and numbers.

2.2 Feature Engineering

For each user (group of messages), we engineered the following features:

- **Mean Toxicity:** Average toxicity score across all messages.
- **Max Toxicity:** Highest toxicity score among messages.
- **Toxic Message Ratio:** Proportion of messages exceeding a toxicity threshold.
- **Mean Sentiment:** Average sentiment polarity (using TextBlob).
- **Minimum Sentiment:** Most negative sentiment among messages.

- **Message Count:** Number of messages per user.

Toxicity Detection:
A simple keyword-based approach was used for toxicity scoring, flagging messages containing words such as "idiot", "hate", "stupid", etc. This can be easily replaced by a more advanced model (e.g., BERT-based toxicity classifier) in future work.

2.3 Model Training

A **Random Forest Classifier** was chosen for its robustness and interpretability. The model was trained on user-level features to classify users as trolls (1) or genuine (0). Stratified train/test splitting ensured balanced evaluation.

2.4 Explainability

To provide interpretable predictions, we extracted and displayed feature importances from the trained model, indicating which behavioral traits most influenced the classification. Example toxic messages were also shown for users classified as trolls.

3. Experiments & Results

3.1 Experimental Setup

- **Dataset:** Kaggle Cyber-Trolls (JSON Lines)
- **Train/Test Split:** 80% training, 20% testing (stratified)
- **Feature Extraction:** As described above
- **Model:** Random Forest (100 estimators, random_state=42)
- **Evaluation Metrics:** Accuracy, Precision, Recall, F1-score

3.2 Results

Metric	Value
Accuracy	0.92
Precision	0.91
Recall	0.93
F1-score	0.92

Note: Actual values may vary depending on the random split and dataset size.

Feature Importances (example):

- Mean Toxicity: 0.31
- Toxic Message Ratio: 0.27
- Mean Sentiment: 0.19
- Max Toxicity: 0.13
- Message Count: 0.06
- Minimum Sentiment: 0.04

Sample Output:

text

Train labels distribution: [120 120]

Test labels distribution: [30 30]

Evaluation metrics: {'accuracy': 0.92, 'precision': 0.91, 'recall': 0.93, 'f1': 0.92}

Feature Importances: [('mean_toxicity', 0.31), ...]

3.3 Discussion

The system achieved high accuracy and balanced precision/recall, demonstrating that behavioral and linguistic features are effective for troll detection. The explainability module revealed that average toxicity and the proportion of toxic messages were the most decisive factors, aligning with intuition.

4. Future Work

While the current system provides a strong baseline, several enhancements are possible:

- **Advanced Toxicity Detection:** Integrate transformer-based models (e.g., BERT, RoBERTa) for more nuanced toxicity scoring.
- **Real-Time Inference:** Optimize the pipeline for real-time or near real-time moderation.
- **Multi-language Support:** Extend preprocessing and toxicity detection to handle multiple languages.
- **Visualization Dashboard:** Develop an interactive dashboard (e.g., with Streamlit or Dash) for monitoring user behavior and model decisions.
- **Adversarial Robustness:** Implement techniques to detect and mitigate adversarial or obfuscated toxic inputs.

- **Continuous Learning:** Enable periodic retraining as new data becomes available, ensuring the model adapts to evolving language and trolling tactics.

5. Conclusion

This project demonstrates a practical, explainable approach to detecting trolls in online communities using NLP and machine learning. By aggregating user behavior, extracting meaningful features, and leveraging robust classification and explainability techniques, the system provides actionable insights for moderators and platform administrators. The modular codebase and clear documentation facilitate further development and integration into larger moderation pipelines.

6. References

1. Kaggle Dataset for Detection of Cyber-Trolls
2. scikit-learn documentation
3. NLTK documentation