Transformers beat traditional NLP approaches in Multi-Class Toxic Comment Classification

Sunesh Praveen Raja Sundarasami, Aaron Cuthinho, Prof. Dr. Jörn Hees, MSc Tim Metzler

Abstract

Online platforms increasingly rely on automated systems to moderate content and ensure respectful discourse. Identifying toxic comments is crucial for maintaining community standards and protecting users from harmful interactions. We are using the dataset from the "Toxic Comment Classification Challenge" [1] on Kaggle that provides a benchmark dataset [2] for developing models that can classify online comments into multiple toxicity categories, such as threats, obscenity, or hate speech. We explored different NLP techniques for multi-class classification and investigated how different architectures and embeddings influence classification performance. We found that Transformers (BERT, T5, GPT-1, etc) outperform traditional NLP approaches such as SVM, XGBoost, etc. Our dataset was an unbalanced dataset, the distribution of which you can find in the plot below.

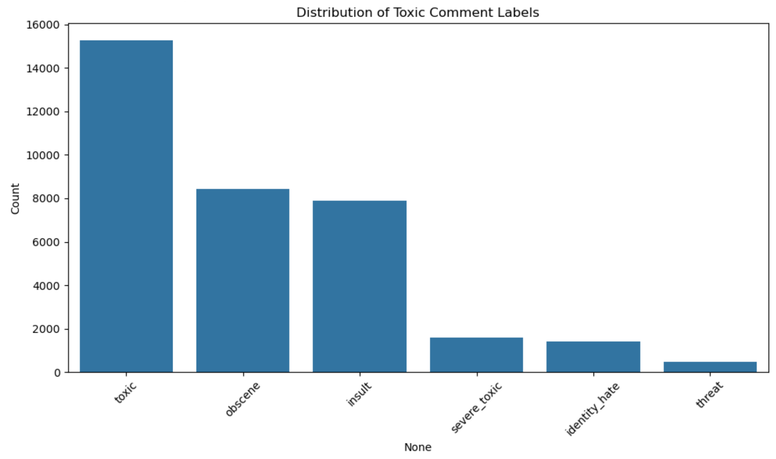

Figure 1: Distribution of the dataset used for toxic comment classification

Methodology

Our approach involved the following steps:

1. **Data Acquisition and Cleaning:** Loading and inspecting the dataset from Kaggle [2]. Applying preprocessing steps such as lowercasing, punctuation removal, and tokenization. Handling missing data and analyzing class imbalance.
2. **Embeddings:** Integrating various word embeddings (e.g., pretrained GloVe, FastText) and evaluating their effect on model performance.
3. **Model Training:** Implementing and comparing several neural network architectures:
   - Simple RNN, LSTM, GRU, BiLSTM.
   - Transformer encoder (e.g., DistilBERT or BERT fine-tuning).
   - Finetuning LLMs like GPT-1 and Llama 3.2 using LoRA.
4. **Threshold Optimization:** Use validation data to tune thresholds for each label instead of defaulting to 0.5, which may not be optimal due to class imbalance.
5. **Evaluation:** Evaluate using standard metrics (e.g., ROC-AUC, F1-score) and analyze per-class performance.

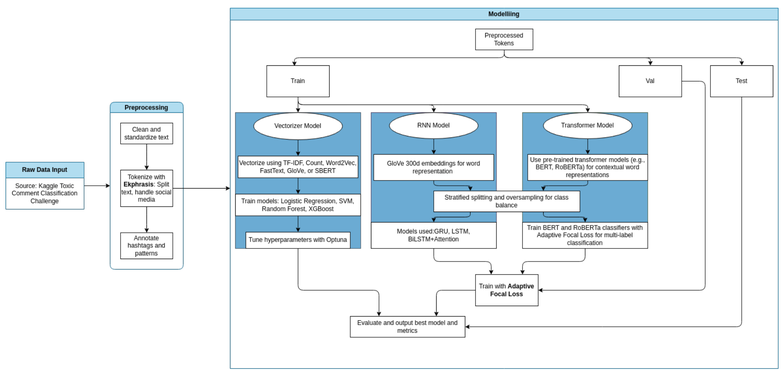

Figure 2: Overview of the model pipeline for toxic comment classification

Results

Your content here

Link to Code and Usage Guidelines

Please scan the QR code to the side or visit the link mentioned below to access the code and usage guidelines.
`https://github.com/SuneshSundarasami/Multi_Label_Toxic_Comment_Classifier/`

References

[1] cjadams, Jeffrey Sorensen, Julia Elliott, Lucas Dixon, Mark McDonald, nithum, and Will Cukierski. Toxic comment classification challenge. `https://kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge`, 2017. Kaggle.

[2] Jigsaw Wikipedia and Google. Wikipedia comments dataset. `https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data`, 2017. Kaggle.

Acknowledgement

We would like to thank Prof. Dr. Jörn Hees for giving us this opportunity and providing valuable guidance during this project.

Contact

Sunesh Praveen Raja Sundarasami
Email: sunesh.sundarasami@smail.inf.h-brs.de
Aaron Cuthinho
Email: aaron.cuthinho@smail.inf.h-brs.de
Hochschule Bonn-Rhein-Sieg