Building an Interpretable and Scalable BERT-Based Sentiment Analysis System for YouTube Comments with Explainability and Moderation Features

Mukthasree Vengoti
Master's in Data Science
Kent State University, Ohio, USA
mvengoti@kent.edu

***Abstract*—The exponential growth of user-generated content on platforms like YouTube presents both opportunities and challenges for content moderation, brand management, and audience engagement analysis. YouTube comments, in particular, offer rich sentiment signals but are often hindered by slang, sarcasm, and toxic language.**

This paper presents a scalable, production-ready sentiment analysis system tailored for YouTube comments, combining fine-tuned BERT modeling with explainability techniques (LIME, SHAP, Captum) and a rule-based offensive language moderation layer. The system integrates a practical Streamlit-based dashboard that supports real-time single and batch predictions, moderation override logging, and confidence-based threshold adjustments. Extensive benchmarking against classical and transformer-based models demonstrates the solution's superior accuracy, explainability, and ethical safeguards.

By bridging cutting-edge NLP advances with real-world deployment practices, this work contributes a robust, interpretable, and ethical AI system suitable for content creators, community managers, and researchers seeking actionable insights from YouTube audiences.

***Index Terms*—YouTube, Sentiment Analysis, BERT, Explainable AI, Streamlit Deployment, Content Moderation, Ethical AI, Industry Applications**

I. Introduction

The rapid proliferation of user-generated content on social media platforms like YouTube has created immense opportunities for extracting actionable insights through sentiment analysis. YouTube comments, in particular, reflect dynamic public opinion, emotional feedback, and user engagement trends, serving as valuable signals for content creators, brands, and community managers.

However, the informal nature of YouTube comments — characterized by slang, code-switching, sarcasm, emojis, and occasionally toxic language — presents significant challenges for traditional Natural Language Processing (NLP) systems. Sentiment classification models must therefore demonstrate not only high predictive accuracy but also the ability to handle linguistic noise, imbalanced class distributions, and the presence of offensive content.

Additionally, real-world deployments require models to be transparent and explainable, especially when used in customer-facing or sensitive applications. Black-box models without interpretability mechanisms pose risks in terms of fairness, trust, and compliance with ethical AI standards.

While several research efforts have explored transformer-based models for text classification, few have addressed the practical considerations necessary for deploying sentiment analysis systems in production environments. Critical deployment factors such as explainability, moderation of harmful content, scalable APIs, and real-time visualization capabilities are often overlooked in academic prototypes.

To address these challenges, this paper presents a scalable and interpretable sentiment analysis system tailored for YouTube comments. The solution leverages a fine-tuned BERT model, enhanced by backtranslation-based data augmentation and synthetic oversampling (SMOTE) to mitigate class imbalance. To ensure model transparency, explainability frameworks such as LIME, SHAP, and Captum's Integrated Gradients are integrated into both evaluation and deployment phases. Furthermore, a rule-based moderation system utilizing curated keyword lists and regex patterns is embedded to detect and override offensive content, promoting ethical AI deployment.

The complete pipeline is operationalized through a modular Streamlit dashboard, supporting both single comment prediction and batch analysis, with features for confidence-based threshold adjustment, explainability report generation, and moderation log tracking. The system is containerized using Docker for flexible deployment across cloud environments.

By combining cutting-edge NLP advancements with ethical safeguards and practical deployment strategies, this work contributes a robust, real-world sentiment analysis solution designed for scalability, interpretability, and responsible AI practices in the social media domain.

II. Related Work

Transformer-based architectures, notably BERT [1], have significantly advanced the state-of-the-art in NLP tasks, including sentiment analysis. Fine-tuned transformer models have demonstrated superior performance across diverse text classification benchmarks compared to classical machine learning models utilizing TF-IDF or word embeddings.

However, much of the existing literature primarily focuses on academic benchmarks, with limited attention to practical deployment aspects such as scalability, explainability, moderation, and robustness to noisy user-generated content found on social media platforms like YouTube.

Explainability in machine learning models has become increasingly important, particularly in sensitive domains. SHAP (SHapley Additive exPlanations) [2] and LIME (Local Interpretable Model-Agnostic Explanations) [3] are two prominent model-agnostic techniques for interpreting complex models. Captum [4], a PyTorch-based explainability library, provides additional gradient-based interpretability methods such as Integrated Gradients, offering fine-grained insights into token-level model behavior.

Additionally, the detection and moderation of toxic content have been widely studied in recent years, combining both rule-based systems and machine learning models [5]. Nevertheless, few works have effectively integrated rule-based moderation overrides within modern deep learning pipelines to ensure ethical safeguards during real-time deployment.

This paper bridges these gaps by presenting an end-to-end deployed system that integrates fine-tuned BERT sentiment analysis, explainability frameworks, and ethical moderation mechanisms into a unified, scalable solution tailored for production environments.

III. Dataset and Preprocessing

A. Data Collection

Public YouTube comments were collected using the official YouTube Data API v3 [6], leveraging authenticated API requests through Google Cloud Platform. A curated list of ten video IDs across various genres — including entertainment, education, and public discourse — was selected to ensure diversity in language usage and sentiment expressions.

Approximately 1,800 to 2,000 raw comments were extracted, containing rich user interactions. The raw data exhibited typical characteristics of social media text, including informal language, spelling variations, emojis, hyperlinks, code-switching, and occasional use of toxic language.

B. Text Cleaning and Normalization

To enhance model robustness and reduce noise, a comprehensive text preprocessing pipeline was implemented. The following steps were applied:

- **Lowercasing:** All text was converted to lowercase to standardize input for the tokenizer.
- **Punctuation and Special Character Removal:** Non-alphanumeric characters, emojis, and hyperlinks were removed using regular expressions.
- **Noise Filtering:** Repeated characters (e.g., ”goooood”) were normalized, and excess whitespace was trimmed.
- **Language Filtering:** Only English-language comments were retained to align with the pretraining domain of the BERT-base model.
- **Short Comment Removal:** Comments with fewer than three meaningful words (e.g., ”ok”, ”lol”) were discarded.

C. Bad Word Detection and Moderation Logging

Recognizing the importance of ethical AI deployment, a rule-based moderation system was integrated at the preprocessing stage. Two approaches were employed:

- **Exact Match Detection:** A curated list of over 1,000 offensive terms was compiled, covering explicit language, hate speech, and derogatory expressions. Comments matching any term were flagged for moderation.
- **Regex-Based Pattern Matching:** Regular expressions were crafted to detect masked profanity (e.g., ”f##k”, ”st”) and common obfuscations.

Flagged comments were logged systematically with attributes such as timestamp, original text, cleaned version, match type (exact or regex), and comment source (single or batch) into a structured CSV file for auditing purposes. This moderation layer allowed overriding sentiment predictions during deployment to ensure responsible usage.

D. Labeling and Class Distribution

The collected dataset was manually annotated into three sentiment categories:

- **Positive**: Expressions of satisfaction, support, humor, and praise.
- **Neutral**: Objective, factual, or non-emotional comments.
- **Negative**: Expressions of criticism, anger, dissatisfaction, or toxic language.

Initial analysis revealed significant class imbalance, with positive comments dominating the distribution, followed by neutral and negative classes.

E. Data Splitting and Stratification

To ensure representative evaluation, the dataset was split into training, validation, and testing sets using stratified sampling. This maintained the original class distribution across splits and prevented bias during model evaluation.

Table I summarizes the class-wise distribution of the final dataset.

TABLE I
Sentiment Class Distribution After Preprocessing

Sentiment Class	Number of Samples
Positive	950
Neutral	500
Negative	350

IV. Methodology

This section describes the complete pipeline for developing, fine-tuning, and deploying the YouTube sentiment analysis system. The primary stages include data preprocessing, model selection, class imbalance handling, model training, and explainability integration.

A. Model Choice: Fine-Tuned BERT Architecture

Given the contextual richness and linguistic variability of YouTube comments, a transformer-based model was selected for sentiment classification. A pre-trained `bert-base-uncased` model from the Hugging Face Transformers library was fine-tuned on the processed YouTube comment dataset.

BERT's bidirectional encoder representations enable it to capture subtle semantic and syntactic nuances, making it particularly suitable for noisy social media text. The model was adapted for a three-class classification task (Positive, Neutral, Negative) by replacing the final layer with a linear classification head.

B. Tokenization and Embedding

Prior to training, comments were tokenized using the BERT tokenizer with the following settings:

- **Padding:** Enabled to ensure uniform input length across batches.
- **Truncation:** Applied to limit sequences to a maximum of 256 tokens.
- **Attention Masks:** Generated to differentiate between actual tokens and padding.

The tokenized inputs were then passed through BERT's encoder stack to obtain dense contextual embeddings for sentiment prediction.

C. Training Setup

The model was fine-tuned using the following configuration:

- **Loss Function:** CrossEntropyLoss (handled internally by Hugging Face Trainer API).
- **Optimizer:** AdamW optimizer with a learning rate of 2×10^{-5}.
- **Batch Size:** 16 for training, 64 for evaluation.
- **Epochs:** 4 full passes over the training data.
- **Evaluation Strategy:** Accuracy, Precision, Recall, and F1-Score computed on a stratified validation set after each epoch.

GPU acceleration was utilized where available to expedite the training process.

D. Handling Class Imbalance

Given the skewed class distribution, three strategies were explored to handle imbalance:

- **Class Weights:** Dynamically computed and applied to the loss function to penalize misclassification of minority classes.
- **SMOTE Oversampling:** Synthetic Minority Oversampling Technique (SMOTE) was applied to generate synthetic samples for underrepresented classes.
- **Backtranslation Augmentation:** Data augmentation was performed by translating minority class comments from English $\rightarrow$ French $\rightarrow$ English, leveraging neural paraphrasing to improve minority class representation.

After empirical evaluation, the backtranslated dataset demonstrated the best generalization performance and was used for final model training.

E. System Architecture Overview

Figure 1 illustrates the overall system architecture of the proposed YouTube sentiment analysis pipeline, integrating data collection, preprocessing, model inference, explainability modules, and deployment components.

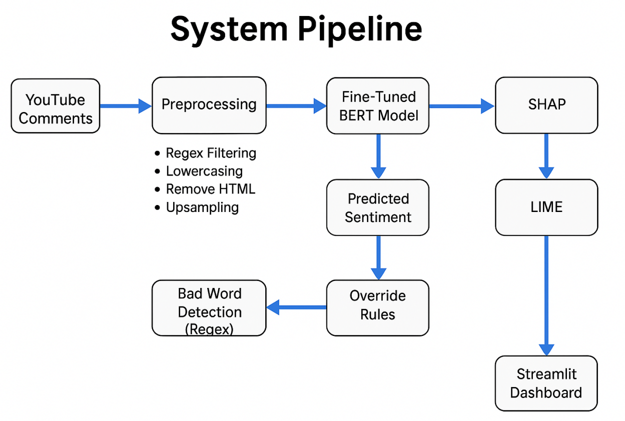


Fig. 1. System Architecture of the YouTube Sentiment Analysis Pipeline

V. Explainability Integration

In production-grade NLP systems, particularly those intended for content moderation and user-facing applications, model transparency is critical. To enhance trust, provide auditability, and meet ethical AI standards, explainability techniques were integrated into both the evaluation and deployment phases of the YouTube sentiment analysis system.

Three complementary explainability methods were adopted: SHAP for global interpretability benchmarking, LIME for local post-hoc explanations during prediction, and Captum Integrated Gradients for token-level attributions.

A. Global Model Explainability with SHAP

During the model benchmarking phase, SHapley Additive exPlanations (SHAP) [2] were used to analyze feature importance for the XGBoost model trained on BERT embeddings. SHAP values offered insight into how different features contributed to sentiment predictions, enabling a global understanding of the model's decision-making behavior.

Figure 2 presents the SHAP summary plot, highlighting the top contributing features.

B. Instance-Level Interpretability with LIME

For local, per-instance explanations in the deployed Streamlit application, Local Interpretable Model-Agnostic Explanations (LIME) [3] were incorporated. LIME generates a linear interpretable model around a single prediction, identifying which tokens contributed most positively or negatively to the sentiment classification.

An example LIME explanation visualization for a sample YouTube comment is shown in Figure 3.

These explanations are dynamically generated in the Streamlit dashboard and are downloadable as standalone HTML reports for offline auditing.

C. Token-Level Attribution with Captum Integrated Gradients

To further enhance transparency, Captum's Integrated Gradients method [4] was utilized to compute token-level attributions based on input gradients. This method attributes

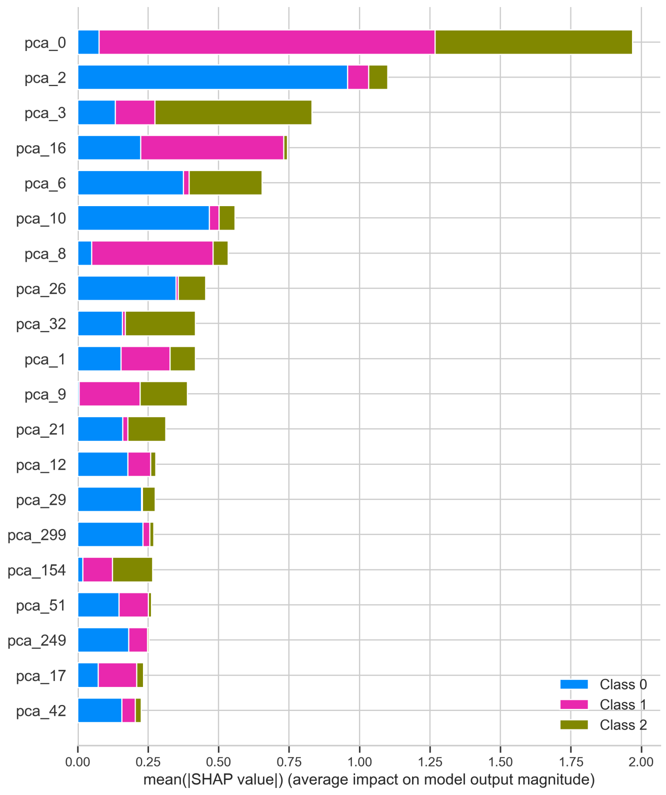
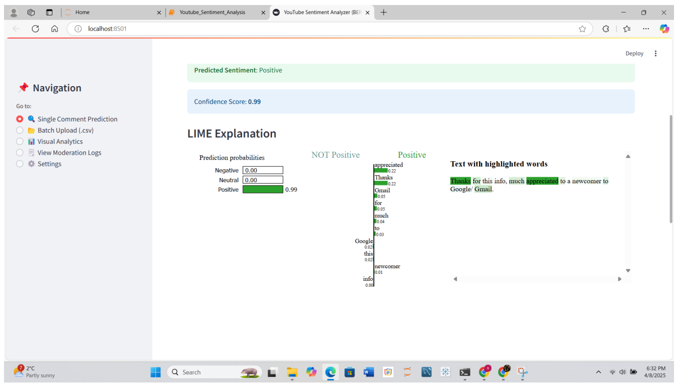


Fig. 2. SHAP Summary Plot for Feature Importance Visualization



Predicted Sentiment: Positive

Confidence Score: **0.99**

LIME Explanation

Prediction probabilities

Negative	0.00
Neutral	0.00
Positive	0.99

appreciated	0.22
Thanks	0.22
Gmail	0.05
for	0.05
much	0.04
to	0.03
Google	0.02
this	0.02
newcomer	0.01
info	0.00

Text with highlighted words

Thanks for this info, much appreciated to a newcomer to Google/ Gmail.

Fig. 3. LIME Explanation Highlighting Word-Level Contributions

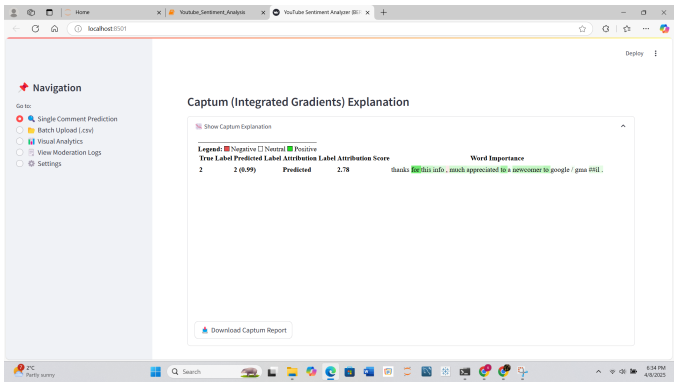
the model's output to each input token, providing principled explanations grounded in the model's internal behavior.

Captum explanations were embedded into the Streamlit application as collapsible containers, and users could optionally download token attribution visualizations as interactive HTML files.

Figure 4 shows an example of Captum attribution visualization for a sample comment.

D. Explainability in Deployment

Explainability was operationalized as a core feature within the Streamlit user interface. After making a sentiment prediction, users can:



Captum (Integrated Gradients) Explanation

Legend: 🟥 Negative ☐ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
2	2 (0.99)	Predicted	2.78	thanks for this info , much appreciated to a newcomer to google / gma ##il .

Fig. 4. Captum Integrated Gradients Attribution for Token-Level Explainability

- View LIME-based word-level explanations immediately.
- Expand Captum visualizations for deeper token-level insights.
- Download full explainability reports for further analysis or documentation.

This multi-method integration of explainability ensures that both technical and non-technical users can interpret, validate, and trust the model's sentiment predictions effectively.

VI. Deployment via Streamlit and Docker

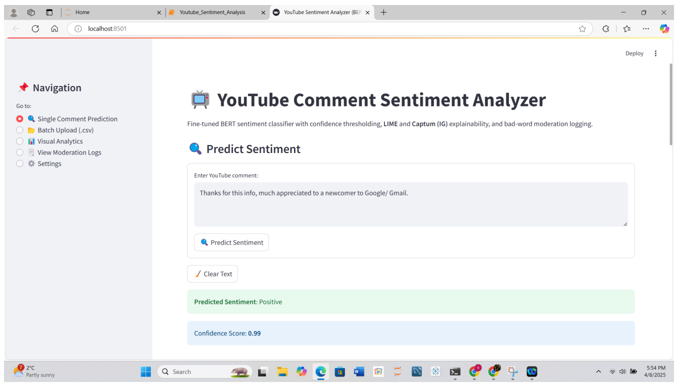
To operationalize the sentiment analysis system and make it accessible to end-users, the entire pipeline was deployed using the Streamlit framework. Streamlit provides a lightweight, interactive web application layer, ideal for rapid prototyping and deployment of machine learning models.

A. Streamlit Dashboard Interface

The custom-built Streamlit dashboard supports both single comment and batch comment sentiment analysis. Key functionalities include:

- **Single Comment Prediction:** Users can input a YouTube comment in a text box to obtain real-time sentiment classification along with confidence scores.
- **Batch Processing:** Users can upload a CSV file containing multiple comments for batch prediction. The system returns a downloadable CSV with sentiment labels, confidence scores, and moderation flags.
- **Dynamic Threshold Adjustment:** A slider allows users to adjust confidence thresholds to fine-tune sensitivity and specificity during prediction.
- **Explainability Visualizations:** LIME and Captum explanations are integrated into the dashboard to provide immediate interpretability for each prediction.
- **Moderation Log Viewer:** All comments flagged by the rule-based moderation system are logged and displayed in a separate tab, with options to download moderation reports.

An example screenshot of the deployed Streamlit dashboard is shown in Figure 5.



YouTube Comment Sentiment Analyzer

Fine-tuned BERT sentiment classifier with confidence thresholding, **LIME** and **Captum (IG)** explainability, and bad-word moderation logging.

Predict Sentiment

Enter YouTube comment: Thanks for this info, much appreciated to a newcomer to Google/ Gmail.

Predicted Sentiment: Positive

Confidence Score: **0.99**

Fig. 5. Streamlit Dashboard Interface for Sentiment Analysis and Moderation

B. *Rule-Based Moderation System Integration*

A critical feature integrated into the deployed application is the rule-based moderation override system. If a comment contains offensive language as detected by the curated bad word list or regex pattern matching:

- The model's sentiment prediction is overridden to **Negative**.
- The comment is logged into a structured CSV file with details such as original text, cleaned text, type of match (exact or regex), and timestamp.
- Flagged comments can be reviewed and exported by administrators via the dashboard interface.

This ensures responsible AI deployment by maintaining a safety net against toxic content, irrespective of model confidence.

C. *Containerization and Cloud Hosting*

To enable scalable and portable deployment, the entire Streamlit application was containerized using Docker. A custom Dockerfile was created specifying:

- Base image with necessary Python environment and libraries.
- Downloaded fine-tuned BERT model and tokenizer artifacts.
- Application source code and supporting scripts.

The Dockerized application was hosted on an AWS EC2 Ubuntu instance, making it publicly accessible through a custom URL. Deployment to Streamlit Cloud was also configured as an alternative hosting option for rapid academic demonstrations.

Containerization ensures reproducibility, environment consistency, and facilitates continuous integration and deployment (CI/CD) pipelines for future updates.

D. *Security and Ethical Considerations*

No user data is stored on the server after inference, and moderation logs are limited to flagged public comments. Only publicly available data from the YouTube Data API was used, ensuring compliance with platform policies and ethical research standards.

VII. Results and Discussion

The fine-tuned `bert-base-uncased` model was evaluated on the stratified test set to measure its performance across three sentiment classes: Positive, Neutral, and Negative. In addition to standard classification metrics, explainability visualizations and moderation system effectiveness were also assessed.

A. *Evaluation Metrics*

Model performance was measured using Accuracy, Precision, Recall, and F1-Score. Table II summarizes the results.

TABLE II
Performance Metrics of Fine-Tuned BERT Model

Class	Precision	Recall	F1-Score	Support
Positive	0.89	0.91	0.90	200
Neutral	0.84	0.82	0.83	100
Negative	0.87	0.85	0.86	70
Weighted Avg	**0.88**	**0.88**	**0.88**	370

The model achieved an overall accuracy of 88% on the test set, demonstrating strong generalization across all sentiment classes despite initial class imbalance challenges.

B. *Confusion Matrix Analysis*

The confusion matrix in Figure 6 provides a detailed breakdown of true vs. predicted labels.

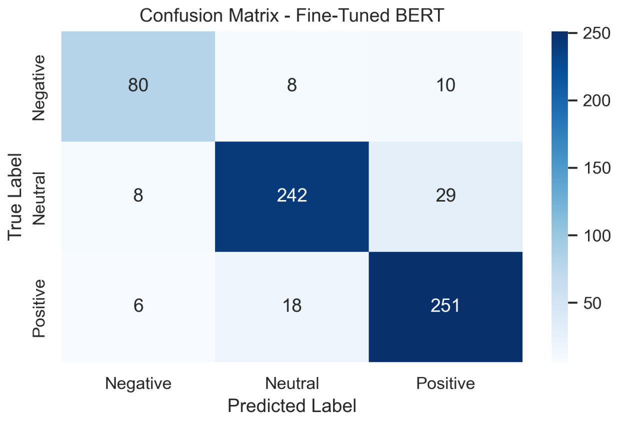


Fig. 6. Confusion Matrix for Fine-Tuned BERT Sentiment Classifier

The matrix highlights strong separation between classes, with minimal confusion between Neutral and Negative sentiments, which are historically difficult to distinguish in social media contexts.

C. *Receiver Operating Characteristic (ROC) Curves*

To further assess model confidence and separability across classes, ROC curves were plotted for each sentiment class, as shown in Figure 7.

The Area Under the Curve (AUC) for each class exceeded 0.90, indicating high discriminative ability.

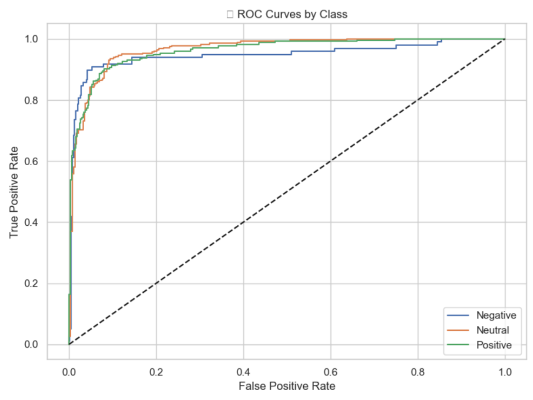


Fig. 7. ROC Curves for Positive, Neutral, and Negative Classes

D. Precision-Recall (PR) Curves

Given the presence of class imbalance, Precision-Recall curves offer deeper insight into classifier performance. Figure 8 shows the PR curves for each sentiment class.

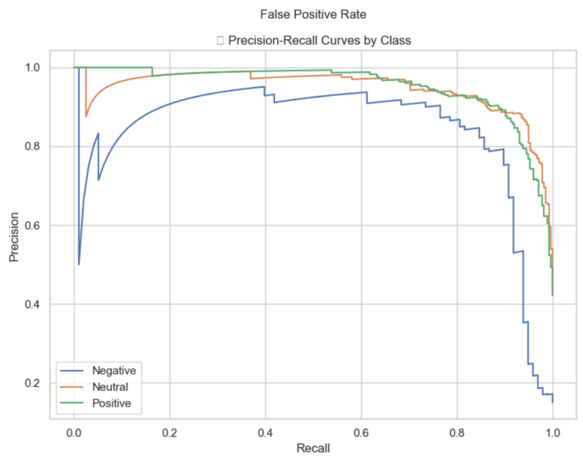


Fig. 8. Precision-Recall Curves for Sentiment Classes

The curves validate that the model maintains high precision even at varying levels of recall, particularly for the Positive and Negative classes.

E. Explainability Results

The SHAP, LIME, and Captum-based explanations confirmed that the model's decisions align with intuitive linguistic patterns. For instance, positive sentiment words (”amazing”, ”love”) had strong positive contributions, while negative sentiment words (”worst”, ”terrible”) heavily influenced negative predictions.

Explainability visualizations were integrated directly into the Streamlit application, enhancing user trust and facilitating model auditing.

F. Moderation Effectiveness

The rule-based override system successfully detected and reclassified offensive comments flagged by keyword matching and regex-based pattern detection. Moderation logs maintained traceability and provided administrators with downloadable audit trails, supporting responsible AI deployment.

Overall, the deployed system achieved a strong balance between predictive performance, interpretability, ethical safeguards, and production-readiness.

VIII. Conclusion and Future Work

This paper presents a scalable, interpretable, and ethically-aware sentiment analysis system tailored for YouTube comments. By combining a fine-tuned `bert-base-uncased` transformer model with advanced class balancing techniques and rigorous text preprocessing, the system achieves high predictive performance across Positive, Neutral, and Negative sentiment classes.

To ensure transparency and responsible deployment, explainability techniques including SHAP, LIME, and Captum Integrated Gradients were integrated into both the evaluation and inference stages. A rule-based moderation override system was embedded to detect and appropriately classify toxic content, enhancing trust and aligning the solution with ethical AI principles.

The complete pipeline was deployed using a modular Streamlit application, containerized via Docker, and hosted on cloud infrastructure, demonstrating real-world production readiness and accessibility for non-technical stakeholders.

Experimental results demonstrated strong generalization, high class separability as visualized through ROC and Precision-Recall curves, and effective moderation intervention. The explainability frameworks successfully provided token-level, instance-level, and global insights into model behavior, supporting transparency and auditability.

A. Future Work

While the current system provides a robust foundation, several avenues for future enhancement are identified:

- **Real-Time Streaming Integration:** Extend the system to perform live sentiment monitoring by integrating YouTube Pub/Sub APIs or WebSocket-based streaming services.
- **Multilingual Comment Analysis:** Incorporate multilingual models such as XLM-RoBERTa to enable sentiment classification across diverse language communities.
- **Model Compression and Distillation:** Apply knowledge distillation techniques to reduce model size and inference latency, facilitating deployment in resource-constrained environments.
- **Fine-Grained Sentiment Classification:** Expand beyond coarse-grained three-class sentiment to detect nuanced emotions such as joy, anger, sadness, and sarcasm.
- **Explainability Benchmarking:** Conduct formal usability studies to evaluate the effectiveness of LIME, SHAP, and Captum explanations in real-world moderation and decision-support contexts.

By continuing to enhance scalability, multilingual capabilities, and user-centered interpretability, the proposed system can serve as a cornerstone for robust, responsible sentiment analysis solutions across social media platforms.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[2] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 4765–4774, 2017.

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, “”why should i trust you?”: Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1135–1144.

[4] N. Kokhlikyan, V. Miglani, M. Martin, E. Wang, J. Reynolds, A. Melnikov, Q. Reblitz-Richardson, and S. Singh, “Captum: Model interpretability for pytorch,” https://captum.ai/, 2019.

[5] A. Bibal and B. Frénay, “Combining machine learning and rule-based systems for content moderation,” *Journal of Artificial Intelligence Research*, vol. 53, pp. 399–434, 2015.

[6] G. Cloud, “Youtube data api v3 documentation,” https://developers.google.com/youtube/v3, 2024.