Technical Summary — AI Generated Text Detection & Paraphrase System

Team: Tech Optimizers
College: Vishwakarma Institute of Information Technology, Pune

Tech Stack Used

- Python 3.x
- Streamlit (web interface)
- HuggingFace Transformers (BERT, mT5)
- PyTorch deep learning backend
- langdetect, numpy, torch.nn.functional
- Local / HuggingFace model loading with caching
- Device-adaptive execution (CPU/GPU)

Unique Algorithms / Approaches Implemented

- Multilingual text classification using BERT-based model.
- Confidence-based paraphrase triggering.
- Length-preserving paraphrase generation using mT5.
- Optimized model loading using Streamlit caching.
- Device-aware model execution (auto-select CPU/GPU).
- Optional fully offline mode via local model directory.

Project Summary

The system is designed as an end-to-end intelligent text-processing pipeline that can evaluate, classify, and transform written content in real time. It addresses the rapidly growing requirement to differentiate between human-written and AI-generated text, especially across educational, corporate, and digital communication spaces. With generative AI becoming common, maintaining authenticity and ensuring responsible content usage has become crucial. This solution provides an automated, multilingual, and accessible mechanism for detecting AI-generated text with strong reliability.

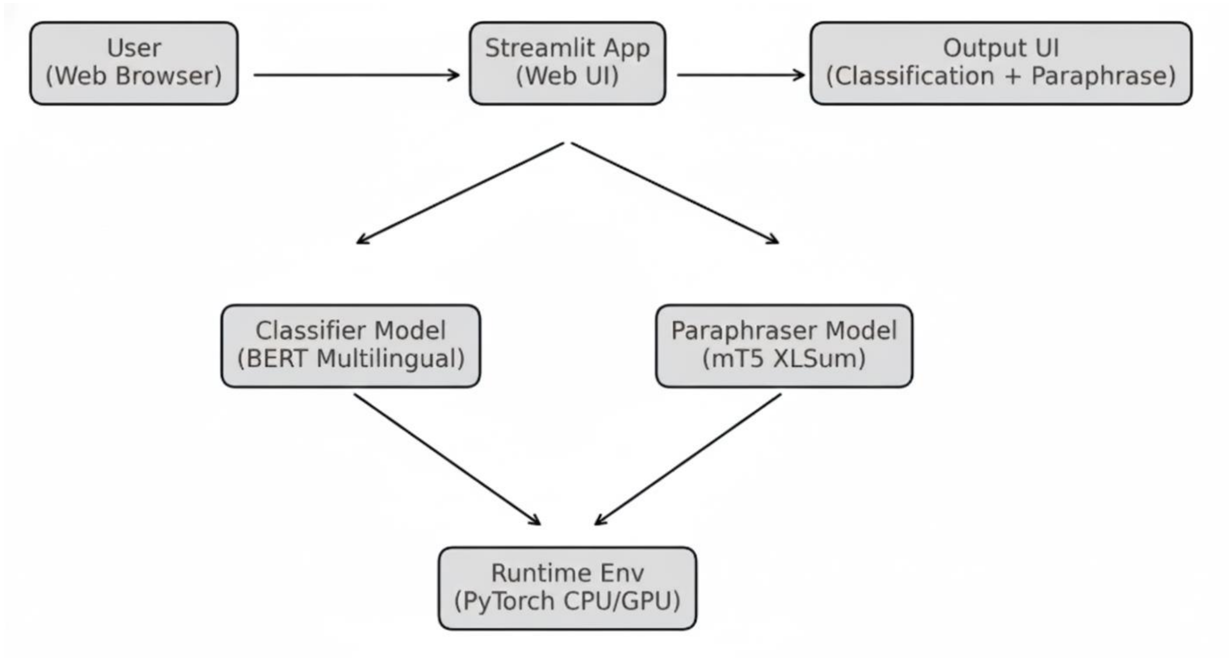
The platform is powered by a multilingual BERT-based classifier capable of analyzing text from multiple languages, making it suitable for diverse global users. It supports both local model loading and dynamic downloading from the HuggingFace Model Hub, ensuring maximum flexibility during offline or low-connectivity scenarios. Using a probability-based evaluation mechanism, the classifier computes softmax-based confidence scores to determine whether a given text resembles AI-generated or human-written patterns.

A key feature of the system is its conditional paraphrasing capability. When the classifier detects text as AI-generated above a configurable confidence threshold, the system activates an mT5-based multilingual paraphrasing model. This model rewrites the text while preserving meaning, tone, and approximate length. Paraphrases are generated using a combination of beam search and probabilistic sampling techniques such as top-p filtering and temperature scaling, ensuring the rewritten content appears natural, coherent, and human-like.

The entire workflow is delivered through an intuitive and responsive Streamlit web interface. Users can paste text or upload files, view classification results, inspect confidence scores, and download paraphrased outputs—all from a single browser window. Streamlit's caching system optimizes performance by storing model instances in memory, significantly reducing load times and making the system well-suited for demonstrations, evaluations, and hackathon environments.

From a system-engineering standpoint, the architecture is modular, scalable, and future-ready. Core components such as text classification, language detection, paraphrasing, preprocessing, and UI rendering operate independently, enabling easy upgrades and feature expansion. The use of PyTorch ensures compatibility with both CPU and GPU devices, improving performance without requiring code changes. Overall, the project demonstrates a practical deployment of modern NLP technologies, offering a portable, multilingual, and highly effective solution for identifying AI-generated content.

System Architecture Diagram


User (Web Browser)
Streamlit App (Web UI)
Output UI (Classification + Paraphrase)
Classifier Model (BERT Multilingual)
Paraphraser Model (mT5 XLSum)
Runtime Env (PyTorch CPU/GPU)