AI-Powered Plagiarism and AI Text Detection System for Regional Languages

Ms. Asma Mulla[1], Prasad Nagnath Kshirsagar[2], Gaurav Dipak Sutar[3], Vishal Ravindra Jarande[4], Sanket Nikesh Anpat[5], Nandini Umakant Shrimangle[6]
[12346]Department of Computer Science and Engineering
[123456] Yashoda Technical Campus, Satara Faculty Of Engineering

Abstract - In today's digital world, the use of online content and artificial intelligence tools has increased rapidly. Many students and content creators use AI tools to generate text or copy information from the internet, which creates challenges in maintaining originality and academic honesty. Most existing plagiarism detection systems mainly support English and are not able to effectively detect AI-generated content or work well for regional languages like Hindi and Marathi. This paper presents a review of a multilingual AI-powered system developed using Python and Flask that can detect both plagiarism and AI-generated text. The system supports English, Hindi, and Marathi languages. It uses machine learning techniques and transformer models to analyze text and determine whether it is human-written or AI-generated. For plagiarism detection, the system splits text into sentences and searches each sentence on the internet using APIs to find matching sources and calculate plagiarism percentage. The system also supports PDF file analysis by extracting text using PyMuPDF and applying both AI detection and plagiarism checking on the extracted content. Additional features such as PDF finder and text search improve the overall functionality of the system. The results are presented in a user-friendly format showing AI probability and plagiarism percentage. This system is useful for students, teachers, researchers, and content creators to ensure originality and reduce misuse of AI tools. It provides a practical and effective solution for modern challenges in content verification.

Index Terms— Plagiarism Detection, AI-generated Text Detection, Natural Language Processing, Machine Learning, Multilingual System, Python, Flask, Transformer Models, PDF Processing, SerpAPI, Stylometric Analysis, Perplexity.

I. INTRODUCTION

In the modern digital era, the use of the internet and online content has increased very rapidly. Students, researchers, and professionals depend heavily on digital platforms to collect information and prepare assignments, reports, and articles. While this has made access to knowledge easier, it has also increased the problem of plagiarism. Plagiarism refers to copying content from other sources without giving proper credit. It reduces originality and affects academic honesty, which is a serious concern in educational institutions and professional environments [1].

Along with this, the rapid development of Artificial Intelligence (AI) tools such as text generators has created a new challenge. Tools like ChatGPT and other language models can generate high-quality, human-like text within seconds. These tools are helpful for learning and productivity, but they can also be misused by students and content writers to generate assignments or articles without actual effort. Traditional plagiarism detection tools are not able to detect such AI-generated content because it is original in structure but not written by a human. This creates a need for systems that can identify whether a piece of text is written by a human or generated by AI [2].

Another important issue is that most existing plagiarism detection systems are designed mainly for the English language. In a multilingual country like India, a large amount of content is written in regional languages such as Hindi and Marathi. These systems fail to detect plagiarism in such languages, especially when the content is translated or paraphrased. This limitation makes current tools less effective and creates a gap in maintaining academic integrity across different languages [3].

To overcome these challenges, this project proposes a multi-language AI and plagiarism detection system developed using Python and Flask. The system is designed to detect both plagiarism and AI-generated text in English, Hindi, and Marathi. It uses Natural Language Processing (NLP) and Machine Learning techniques to analyze the text. For AI detection, the system uses transformer-based models for English and Hindi, and a custom machine learning model for Marathi. It performs sentence-level analysis and provides an AI probability score to indicate whether the content is machine-generated. For plagiarism detection, the system divides the text into sentences and searches each sentence on the internet using APIs such as SerpAPI. It identifies matching sources and calculates the overall plagiarism percentage. This approach helps in detecting both exact copying and partial similarity. The system also supports PDF file processing, where text is extracted using PyMuPDF and analyzed in the same way as normal text input. In

addition, the system includes extra features such as a PDF finder, which helps in finding similar documents available online, and a text search feature that allows users to search queries using Google API. These features make the system more practical and useful in real-world applications.

Overall, the proposed system provides a complete solution for detecting plagiarism and AI-generated content in a multilingual environment. It helps students, teachers, researchers, and content creators to maintain originality and ensure fair use of information. The combination of machine learning, NLP, and real-time web search makes this system more advanced and effective compared to traditional tools [4].

II. LITERATURE REVIEW

Plagiarism detection has been widely studied over the years due to the increasing use of digital content. In the early stages, plagiarism detection systems mainly relied on simple techniques such as string matching and keyword comparison. These methods were useful for detecting exact copying of text but were not effective in identifying paraphrased or modified content. As a result, many copied texts with slight changes could not be detected properly, which reduced the reliability of these systems [5]. To improve the performance of plagiarism detection, researchers introduced more advanced methods such as TF-IDF (Term Frequency–Inverse Document Frequency) and n-gram models. TF-IDF helps in identifying important words in a document, while n-grams analyze sequences of words to detect partial matches. These techniques improved detection accuracy to some extent, especially for partially copied text. However, they still had limitations in understanding the actual meaning of the text, which is important for detecting paraphrased or translated content [6].

With the advancement of Natural Language Processing (NLP) and deep learning, transformer-based models such as BERT were developed. These models are capable of understanding the semantic meaning of sentences rather than just matching words. This made it possible to detect similarity between texts even when the wording is different. For multilingual applications, models like IndicBERT and LaBSE were introduced, which support multiple languages including Indian languages. These models allow cross-language comparison and improve plagiarism detection for translated content [7]. In addition to plagiarism detection, recent research has focused on detecting AI-generated text. With the growth of AI tools, it has become important to identify whether content is written by humans or machines. One commonly used method is perplexity, which measures how predictable a text is. AI-generated text usually has lower perplexity because it follows patterns learned during training. Another important method is stylometric analysis, which studies writing style features such as sentence length, vocabulary usage, and punctuation. These features help in distinguishing between human and AI-generated writing [8].

Many recent studies suggest that using a hybrid approach gives better results. Combining lexical techniques (like TF-IDF), semantic methods (like embeddings), and stylistic features (like stylometry) improves overall system performance. Such systems are more effective in handling complex cases like paraphrased text, translated content, and AI-generated text. This approach forms the foundation for modern plagiarism and AI detection systems and is used in the proposed project to achieve better accuracy and reliability.

III. PROPOSED SYSTEM

The proposed system is a multi-language AI and plagiarism detection system developed using Python and Flask. It is designed to detect whether a given text is copied from other sources or generated using AI tools. The system supports English, Hindi, and Marathi, making it suitable for a multilingual environment.

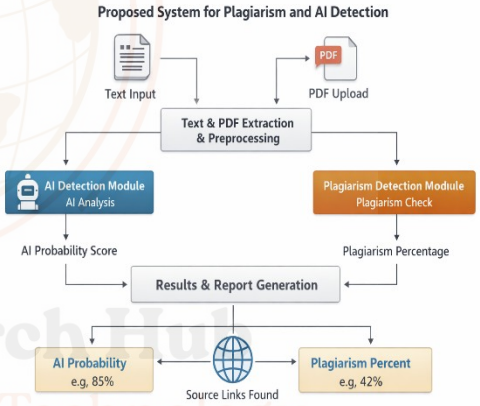

Figure 1. Proposed System Architecture

The major components and working of the proposed system are explained below:

A. Overall System Design

The system follows a modular design where each component performs a specific task. The main goal is to process user input, analyze it using different techniques, and generate a final report showing plagiarism percentage and AI probability. The modular approach makes the system easy to manage, update, and extend in the future [9].

B. User Input Module

The system allows users to:

a. Enter text manually

b. Upload PDF files

This module acts as the starting point of the system. The input provided by the user is sent to the backend server for further processing. The system is designed to handle both small and large text inputs efficiently.

C. PDF Processing Module

When a user uploads a PDF file:

a. Text is extracted using PyMuPDF

b. Extracted text is cleaned and prepared

This module ensures that the system can analyze not only plain text but also document files, making it more practical for real-world applications.

D. Text Preprocessing Module

Before analysis, the input text is processed to improve accuracy:

1. Removal of special characters
2. Conversion to lowercase
3. Sentence splitting
4. Tokenization

This step ensures that the data is clean and structured properly for further analysis [10].

E. AI Detection Module

This module checks whether the text is AI-generated or human-written. It uses different techniques for different languages:

1. **English & Hindi:**
   Uses transformer-based models to understand text patterns and context
2. **Marathi:**
   Uses a custom machine learning model trained on stylometric features

The module performs sentence-level analysis and calculates an AI probability score. This helps in identifying machine-generated content effectively [11].

F. Stylometric and Perplexity Analysis

The system uses writing style analysis to improve AI detection:

1. Sentence length variation
2. Word usage patterns
3. Punctuation usage
4. Perplexity score

AI-generated text tends to be more structured and predictable, while human writing shows more variation. These features help in accurate classification.

G. Plagiarism Detection Module

This module checks whether the text is copied from other sources:

a. The text is split into sentences

b. Each sentence is searched on the internet using SerpAPI

c. Matching sources are identified

The system calculates:

a. Similarity score

b. Plagiarism percentage

This approach helps in detecting both exact copying and partial similarity [12].

H. API Integration Module

The system uses external APIs such as:

- SerpAPI (Google Search API)

This allows the system to:

1. Search content in real-time
2. Find matching sources from the internet

This makes the plagiarism detection process more dynamic and accurate.

I. PDF Finder Module

An additional feature of the system is the PDF finder:

1. Extracts important lines from uploaded PDF
2. Searches for similar PDFs online

This helps users find related documents and verify sources.

J. Text Search Module

The system provides a text search feature:

1. Users can enter any query
2. Results are fetched using Google API

This improves usability and makes the system more interactive.

K. Report Generation Module

After processing, the system generates a final report that includes:

a. AI detection percentage

b. Plagiarism percentage

c. Matching sources

The report is displayed in a simple and user-friendly format, making it easy to understand.

L. Overall Working of the System

a. User enters text or uploads PDF

b. Text is extracted and preprocessed

c. AI detection is performed

d. Plagiarism detection is performed

e. API searches are executed

f. Results are combined

g. Final report is displayed

M. Advantages of Proposed System

a. Supports multiple languages

b. Detects both plagiarism and AI-generated text

c. Provides detailed analysis

d. Works with PDF files
e. Uses real-time internet data

IV. METHODOLOGY

The methodology of the proposed system explains the step-by-step process used to detect plagiarism and AI-generated text. The system follows a structured pipeline where the input text is processed, analyzed, and evaluated using different techniques from Natural Language Processing (NLP) and Machine Learning. The complete methodology is described below:

A. Data Input

The system accepts input in two forms:

1. Direct text input entered by the user
2. PDF file uploaded by the user

If a PDF file is uploaded, the system extracts text using PyMuPDF. The extracted text is then passed to the next stage for processing. This flexibility allows the system to handle both simple and document-based inputs effectively.

B. Text Extraction and Cleaning

After receiving the input, the system performs text cleaning to remove unnecessary data. This includes:

a. Removing special characters and symbols
b. Removing extra spaces
c. Converting text to lowercase

This step ensures that the text is in a standard format, which improves the accuracy of further analysis [13].

C. Sentence Segmentation

The cleaned text is divided into smaller units called sentences. Sentence-level processing helps the system to:

a. Analyze each part of the text separately
b. Perform more accurate plagiarism detection
c. Identify specific portions of AI-generated content

This step is very important for detailed and precise analysis.

D. Tokenization and Preprocessing

Each sentence is further processed using tokenization:

a. Breaking sentences into words (tokens)
b. Removing stopwords (common words like “is”, “the”, “and”)
c. Normalizing words

This step prepares the text for feature extraction and improves the efficiency of machine learning models [14].

E. Feature Extraction

In this stage, important features are extracted from the text. These features are used for both plagiarism detection and AI detection. The system uses:

1. **TF-IDF:** Measures importance of words
2. **N-grams:** Detects word sequences
3. **Embeddings:** Captures semantic meaning
4. **Stylometric features:** Analyzes writing style

These features help the system understand both the structure and meaning of the text.

F. AI Detection Process

The system checks whether the text is AI-generated or human-written using the following approach:

1. For **English and Hindi:**
   Transformer-based models are used to analyze context and language patterns
2. For **Marathi:**
   A custom machine learning model is used

The system also uses:

a. **Perplexity:** Measures predictability of text
b. **Stylometry:** Studies writing patterns

Based on these features, the system calculates an AI probability score [15].

G. Plagiarism Detection Process

The plagiarism detection is performed as follows:

1. The text is split into sentences
2. Each sentence is searched using SerpAPI
3. Matching content is identified from online sources
4. Similarity scores are calculated

The system then computes the overall plagiarism percentage based on matched sentences. This method helps in detecting both exact copying and partial similarity.

H. API-Based Web Search

The system uses external APIs (such as SerpAPI) to perform real-time web searches. This allows:

a. Access to large amounts of online data
b. Detection of plagiarism from internet sources
c. More accurate and updated results

This step makes the system more dynamic and practical.

I. Result Aggregation

After both AI detection and plagiarism detection are completed:

a. Results are combined
b. Scores are calculated

The system ensures that both outputs are clearly separated and understandable.

J. Report Generation

The final output is presented in a user-friendly format. The report includes:

1. AI-generated probability (%)
2. Plagiarism percentage (%)

3. Matching sources and links

This helps users easily understand the results and take necessary action.

K. Overall Workflow Summary

The complete methodology follows these steps:

1. User inputs text or uploads PDF
2. Text is extracted and cleaned
3. Text is divided into sentences
4. Preprocessing and tokenization are performed
5. Features are extracted
6. AI detection is performed
7. Plagiarism detection is performed
8. API search is executed
9. Results are combined
10. Final report is generated

L. Advantages of Methodology

1. Accurate detection using multiple techniques
2. Supports multilingual processing
3. Combines AI detection and plagiarism detection
4. Uses real-time web data
5. Provides detailed and understandable results

M. Discussion

The methodology shows that a hybrid approach using NLP, machine learning, and APIs is effective in solving modern challenges of content verification. By combining multiple techniques, the system achieves better accuracy and reliability compared to traditional methods [16].

V. SYSTEM OVERVIEW

The proposed system is a multi-language AI and plagiarism detection system designed to analyze text and determine its originality. It is developed using Python and Flask and integrates various technologies such as Natural Language Processing (NLP), Machine Learning, and web APIs. The system is capable of processing text in English, Hindi, and Marathi and provides accurate results for both plagiarism and AI-generated content detection. The overall working and features of the system are explained below.

A. General Description of the System

The system is designed to provide a complete solution for content verification. It accepts input in the form of text or PDF files, processes the content, and generates a detailed report. The system focuses on two main tasks:

a. Detecting plagiarism
b. Detecting AI-generated text

By combining these two features, the system becomes more advanced than traditional tools that focus only on plagiarism detection [17].

B. Multilingual Support

One of the key features of the system is its ability to support multiple languages. It works with:

a. English
b. Hindi
c. Marathi

This makes the system useful in the Indian context, where content is created in different languages. The system uses different models and techniques to handle each language effectively.

C. AI Text Detection Feature

The AI detection module plays a major role in the system. It determines whether the input text is written by a human or generated using AI tools.

a. Transformer-based models are used for English and Hindi
b. A custom machine learning model is used for Marathi
c. Sentence-level analysis is performed
d. AI probability score is generated

This feature helps in identifying misuse of AI tools in academic and professional work [18].

D. Plagiarism Detection Feature

The plagiarism detection module checks whether the content is copied from other sources.

a. Text is divided into sentences
b. Each sentence is searched using Google API (SerpAPI)
c. Matching sources are identified
d. Plagiarism percentage is calculated

This method helps in detecting both exact copying and partial similarity. It also provides links to matched sources for verification.

E. PDF Processing Capability

The system supports PDF file uploads, which makes it more practical for real-world use.

a. Text is extracted from PDF using PyMuPDF
b. Extracted content is processed like normal text
c. Both AI detection and plagiarism detection are applied

This feature is especially useful for analyzing assignments, reports, and research papers [19].

F. PDF Finder Feature

An additional feature of the system is the PDF finder.

a. Extracts important lines from uploaded documents
b. Searches for similar PDFs available online

c. Helps users find related documents and references

This improves the usability of the system and provides extra functionality.

G. Text Search Functionality

The system also provides a text search feature.

a. Users can enter any query
b. Results are fetched using Google API
c. Helps users explore related information

This feature makes the system more interactive and useful for research purposes.

H. Integration of External APIs

The system uses external APIs such as SerpAPI for web search. This allows:

a. Real-time data retrieval
b. Access to large online content
c. Better plagiarism detection accuracy

API integration plays an important role in making the system dynamic and efficient [20].

I. Output and Report Generation

After processing, the system generates a detailed report that includes:

a. AI-generated probability (%)
b. Plagiarism percentage (%)
c. Matching sources and links

The results are displayed in a simple and user-friendly format, making it easy for users to understand.

J. Overall Working Summary

The system works in the following steps:

1. User inputs text or uploads a PDF
2. Text is extracted and cleaned
3. Text is analyzed for AI detection
4. Text is checked for plagiarism
5. API searches are performed
6. Results are combined
7. Final report is displayed

K. System Benefits

a. Supports multiple languages
b. Detects AI-generated content
c. Provides accurate plagiarism results
d. Works with PDFs
e. Uses real-time internet data
f. Easy to use interface

L. Discussion

The system overview shows that integrating multiple technologies such as NLP, machine learning, and APIs provides a powerful solution for content verification. The system is capable of handling modern challenges like AI-generated text and multilingual plagiarism effectively, making it suitable for academic and professional use.

VI. CONCLUSION

In this project, a multi-language AI and plagiarism detection system has been successfully designed and developed using Python and Flask. The system is capable of detecting both copied content and AI-generated text, which are major challenges in today's digital world. It supports multiple languages such as English, Hindi, and Marathi, making it useful in a multilingual environment like India. The system uses a combination of Natural Language Processing (NLP), Machine Learning, and transformer-based models to analyze text effectively. For AI detection, it uses advanced models for English and Hindi, along with a custom machine learning model for Marathi. It also uses stylometric features and perplexity to identify writing patterns. For plagiarism detection, the system divides the text into sentences and searches each sentence on the internet using APIs to find matching sources and calculate similarity.

One of the important features of the system is its ability to process PDF files. It extracts text using PyMuPDF and performs both AI detection and plagiarism checking on the extracted content. Additional features such as PDF finder and text search improve the overall functionality and make the system more practical for real-world use. The results of the system show that the hybrid approach used in this project is effective in detecting both plagiarism and AI-generated content. It provides clear and user-friendly reports, which help users easily understand the originality of their content. This system is very useful for students, teachers, researchers, and content creators to maintain academic honesty and avoid misuse of AI tools. Although the system has some limitations such as dependency on internet APIs and higher computational requirements, it still provides a strong and practical solution for modern content verification problems. With further improvements like adding more languages, improving datasets, and optimizing performance, the system can become more accurate and widely used.

Overall, this project successfully addresses the problem of plagiarism and AI-generated text detection in a multilingual setting and provides a complete and effective solution for ensuring content originality.

VII. REFERENCES

[1] S. Gehrmann, H. Strobelt, and A. M. Rush, “GLTR: Statistical Detection and Visualization of Generated Text,” *Proc. ACL*, 2019.

[2] OpenAI, “GPT-4 Technical Report,” 2023.

[3] T. Brown et al., “Language Models are Few-Shot Learners,” *Proc. NeurIPS*, 2020.

[4] J. Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *Proc. NAACL-HLT*, 2019.

[5] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT Networks,” *Proc. EMNLP*, 2019.

[6] D. Kakwani et al., “IndicNLPSuite: Monolingual Corpora and Pre-trained Models for Indian Languages,” *Proc. EMNLP*, 2020.

[7] Google AI, “Language-agnostic BERT Sentence Embedding (LaBSE),” 2020.

[8] X. Zhu et al., “Detecting AI-Generated Text using Deep Learning Techniques,” *IEEE Access*, 2023.

[9] S. Ippolito et al., “Automatic Detection of Generated Text is Easiest when Humans are Fooled,” 2020.

[10] E. Stamatatos, “A Survey of Modern Authorship Attribution Methods,” *J. Assoc. Inf. Sci. Technol.*, 2018.

[11] M. Potthast et al., “Cross-language Plagiarism Detection,” *Proc. LREC*, 2011.

[12] M. Alzahrani, N. Salim, and A. Abraham, “Understanding Plagiarism Linguistic Patterns,” *IEEE Trans. Systems, Man, and Cybernetics*, 2012.

[13] S. Chowdhury and M. Rahman, “Multilingual Plagiarism Detection using NLP Techniques,” *IEEE Access*, 2021.

[14] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” *JMLR*, 2011.

[15] A. Joulin et al., “FastText.zip: Compressing Text Classification Models,” 2016.

[16] H. Maurer, F. Kappe, and B. Zaka, “Plagiarism – A Survey,” *J. Universal Computer Science*, 2006.

[17] Turnitin, “Plagiarism Detection Technologies: A White Paper,” 2021.

[18] K. Clark et al., “ELECTRA: Pre-training Text Encoders as Discriminators,” 2020.

[19] P. Keskar et al., “CTRL: A Conditional Transformer Language Model for Controllable Generation,” 2019.

[20] S. Welleck et al., “Neural Text Generation with Unlikelihood Training,” 2020.