ISSN: 2249-7196
IJMRR/April. 2025/ Volume 15/Issue 2s/113-119
Kolli Nagesh/ International Journal of Management Research & Review

TEXT and IMAGE Plagiarism Detection

Kolli Nagesh
PG scholar, Department of MCA, DNR College, Bhimavaram, Andhra Pradesh.
K.VENKATESH
(Assistant Professor), Master of Computer Applications, DNR college, Bhimavaram, Andhra Pradesh.

Abstract ***This project introduces a web-based plagiarism detection system developed using Django, capable of identifying plagiarism in both textual documents and digital images. The system employs Natural Language Processing (NLP) techniques such as tokenization, stopword removal, lemmatization, and stemming to clean and process text. It then uses the Longest Common Subsequence (LCS) algorithm to compute similarity between documents. For image analysis, the system leverages OpenCV to preprocess images and calculate histogram-based features using a custom Five Module Matching (FMM) algorithm. These histograms are compared to detect visual similarities between images. The platform includes user registration, login, and file/image upload functionalities. It stores user data in a MySQL database and provides visual representation of matching results. This system is ideal for academic and content verification purposes, promoting originality and preventing intellectual property theft***.

I. INTRODUCTION

Plagiarism, the unethical practice of using someone else's work without proper attribution, has become an increasingly prevalent issue in the digital age. With the exponential growth of accessible content on the internet and the ease of copying and sharing digital data, both textual and visual plagiarism have become significant concerns in academic, corporate, and creative domains. Traditional plagiarism detection systems have primarily focused on text similarity using keyword and phrase matching, but modern challenges necessitate more robust, cross-domain approaches. This project introduces a Django-based web application designed to detect plagiarism in both textual documents and image files, offering an integrated platform that blends classical natural language processing (NLP) techniques with image processing algorithms.

The primary motivation behind this system is to provide an easy-to-use, reliable, and efficient mechanism for identifying instances of intellectual property theft across different content formats. The system is built using Django, a high-level Python web framework that promotes rapid development and clean, pragmatic design. The backend is supplemented with powerful libraries such as OpenCV for image processing and Natural Language Toolkit (NLTK) for textual analysis. The frontend is designed to be intuitive, supporting file uploads, analysis views, and report generation.

In academic institutions, plagiarism has become a pressing issue due to the increasing availability of research articles, essays, and online sources that students and researchers can access without proper citation. Educators and administrators require tools that go beyond simple string matching, incorporating advanced NLP techniques to capture semantic similarity and structural changes in written content. At the same time, creative professionals, such as graphic designers and photographers, face the challenge of unauthorized use of their original visual content. The ability to detect image plagiarism using histogram analysis and pattern recognition adds significant value to the system, making it a versatile tool for various user needs.

The text plagiarism detection module of this application employs preprocessing techniques such as tokenization, removal of stop words, stemming, and lemmatization. These methods help standardize the text and eliminate irrelevant information, thus improving the accuracy of similarity calculations. The core algorithm used for measuring text similarity is the Longest Common Subsequence (LCS), which identifies the largest sequence of words that appear in the same order in both documents. This algorithm provides a balance between precision and flexibility, effectively capturing copied or paraphrased content.

For image plagiarism detection, the system uses a technique based on histogram comparison. A

histogram represents the distribution of pixel intensity values in an image and is invariant to certain transformations such as resizing or brightness adjustments. The system preprocesses each image by converting it to grayscale and applying normalization techniques. It then calculates a histogram of the processed image and compares it with histograms of known source images using the histogram intersection method. This technique measures the degree of similarity between two histograms, enabling the detection of visually similar images even with minor alterations.

One of the key features of this system is its dual-mode capability, allowing users to upload both text and image files for analysis. Users can also upload known source documents or images, which are stored and indexed by the system for future comparison. Upon uploading a suspicious file, the system compares it with all stored sources and provides a detailed similarity report, including matched filenames, similarity scores, and a verdict on whether plagiarism is detected. Visualization is provided through matplotlib plots for histogram comparison, giving users a graphical insight into image similarities.

Security and data integrity are also considered in the system's architecture. User authentication modules ensure that only registered users can access analysis features, and session management is handled through local session tracking. Additionally, data is stored temporarily and deleted after analysis to ensure privacy and reduce storage overhead. The system is integrated with MySQL for user management and logging of analysis results, providing a scalable solution that can support large datasets and concurrent users.

The Django framework facilitates modular development and ease of deployment. Each functionality—user registration, login, file upload, source management, and plagiarism detection—is encapsulated within dedicated views and templates, making the codebase maintainable and extensible. Future improvements, such as incorporating deep learning-based image matching or semantic text analysis using word embeddings and transformer models, can be easily integrated into the existing architecture.

In conclusion, this Django-based plagiarism detection system serves as a powerful tool for tackling plagiarism in multiple domains. By combining classical text analysis with advanced image processing, it offers a comprehensive solution for detecting and mitigating content duplication. The use of Python libraries, a user-friendly web interface, and scalable architecture makes it suitable for academic institutions, content creators, and legal professionals alike. This system not only promotes originality and ethical content creation but also equips users with the means to protect and uphold intellectual integrity in the digital era.

II. LITERATURE SURVEY

Plagiarism detection has long been a critical area of research in fields such as computer science, linguistics, education, and digital media. Various techniques and tools have been proposed over the years to tackle different types of plagiarism, ranging from basic string-matching algorithms to sophisticated semantic and visual similarity detection models. This literature survey reviews prominent existing works, tools, and methodologies used in text and image plagiarism detection to understand current challenges and solutions.

1. Text-Based Plagiarism Detection Techniques

Text plagiarism can manifest as verbatim copying, paraphrasing, or structural modification of existing documents. Over the years, numerous methods have been developed:

String Matching Algorithms

Early plagiarism detectors relied on direct string matching using techniques like Rabin-Karp, KMP, or Boyer-Moore. These methods are effective for exact copying but fail when content is paraphrased. The tool *PlagScan* and *CopyCatch* use variations of these algorithms to detect similarity between text chunks.

Fingerprinting Methods

Algorithms like **winnowing** generate fingerprints of documents based on selected substrings. These fingerprints are then compared to identify overlaps. This technique is used in tools like *MOSS (Measure Of Software Similarity)* which was developed to detect source code plagiarism but has also been adapted for text.

Vector Space Models (VSM)

In these models, documents are represented as vectors in a high-dimensional space using term frequency-inverse document frequency (TF-IDF). Cosine similarity is then used to compare these vectors. Although VSM is effective for identifying semantic similarity to an extent, it still struggles with complex paraphrasing.

Semantic and NLP Approaches

With advancements in NLP, more sophisticated methods such as **Latent Semantic Analysis (LSA)** and **Word2Vec/Doc2Vec embeddings** have emerged. These approaches analyze the context and meaning of words rather than their literal occurrence. Tools like *Turnitin* and *Grammarly* leverage NLP and machine learning to detect deeper similarities.

Sequence Alignment and LCS

The Longest Common Subsequence (LCS) algorithm is often used to identify the longest sequence of matching tokens between documents. It is particularly useful in detecting structural and reordered copying, as implemented in the proposed Django-based system.

2. Image-Based Plagiarism Detection Techniques

Image plagiarism is more complex due to various transformations that can be applied to copied images, including cropping, scaling, rotation, and color adjustments.

Histogram Comparison

This is a foundational method in image processing where the histogram (pixel intensity distribution) of two images is compared. Despite its simplicity, histogram comparison is robust against minor transformations and effective for identifying visually similar images. This is the technique adopted in the current system.

Feature Matching (SIFT, SURF, ORB)

Techniques like **Scale-Invariant Feature Transform (SIFT)** and **Speeded Up Robust Features (SURF)** detect and compare key points in images. These are robust against geometric transformations and have been used in plagiarism detection for identifying copied content in complex visuals.

Perceptual Hashing

Algorithms like **pHash** create a digital fingerprint of an image based on perceptual features. This technique is effective for identifying near-duplicate images and is used by tools like *TinEye* and *Google Reverse Image Search*.

Deep Learning Models

With convolutional neural networks (CNNs), researchers have developed models that can identify visual similarity across altered images. These models extract deep features from images and compare them in a learned embedding space. While accurate, these methods are computationally expensive and require large training datasets.

3. Existing Plagiarism Detection Tools

Turnitin

A widely-used commercial plagiarism detection tool in educational institutions. It offers detailed reports and uses proprietary databases, but does not support image plagiarism detection.

PlagScan
This tool provides real-time similarity reports and supports multiple languages. It uses text-matching and semantic analysis but is limited to textual plagiarism.

Copyscape
Mainly focused on web content, it scans websites for duplicated content. It lacks advanced NLP features and does not support academic-style documents.

Grammarly
In addition to grammar checking, it offers a plagiarism checker that matches content against billions of web pages. It does not support user-uploaded source documents or image analysis.

TinEye
A reverse image search engine that identifies image reuse across the web. However, it relies on an online image index and does not allow custom source databases

These tools each have their strengths, but they generally specialize in either text or image detection, not both. Moreover, most of them are either closed-source or commercial, limiting their adaptability for custom use cases.

4. Gaps in Existing Research and Systems

Lack of Combined Solutions
Few tools provide an integrated approach to detect both textual and image-based plagiarism in one system.

Customization and Transparency
Many commercial tools offer limited customization and do not reveal their internal algorithms, making them unsuitable for research or institution-specific applications.

Paraphrasing Detection
Detecting intelligent paraphrasing and idea plagiarism is still a significant challenge, particularly in cross-language or interdisciplinary content.

Resource-Intensive Approaches
Deep learning solutions, while powerful, require high-end hardware and extensive training data, making them impractical for many academic institutions.

5. Contribution of the Proposed System

The proposed Django-based plagiarism detection system aims to fill the gap by offering:

A unified platform for **text and image** plagiarism detection.

Use of **LCS for text** to handle both direct copying and reordered phrasing.

Use of **histogram-based comparison** for images to detect visually similar but modified content.

A **user-friendly web interface** with options for file upload, source management, and analysis reports.

An **open and modular design**, allowing integration with advanced NLP or deep learning models in future iterations.

Existing System

In the current landscape, numerous plagiarism detection systems are available, but most are tailored for either **text** or **image plagiarism**, not both. Common tools like Turnitin, Grammarly, and PlagScan focus heavily on detecting textual plagiarism by comparing documents against web content and academic repositories. However, these tools have several limitations:

Text-focused only: Image plagiarism is generally unsupported.

Limited customization: Most systems are commercial and proprietary, preventing developers or researchers from modifying them to suit specific needs.

Poor image similarity detection: Even tools that claim image comparison typically rely on simple metadata or reverse image lookup, which is not robust against editing, resizing, or recoloring.

Expensive subscriptions: Tools like Turnitin and Grammarly require costly licenses, making them inaccessible to many institutions or individual users.

Opaque detection logic: Users rarely understand what algorithms are used or how accurate the reports are, which impacts trust and extensibility.

These systems do not effectively address the rising need for an integrated platform that can detect both **textual and visual plagiarism** using open, extensible technologies.

III. Proposed System

The proposed system is a Django-based web application designed to detect **both text and image plagiarism**. This system offers an efficient, customizable, and user-friendly platform that supports multiple file formats, comparison sources, and analysis reports.

Key Features:

Unified Platform: Supports plagiarism detection for both **text documents (PDF, DOCX, TXT)** and **image files (JPG, PNG, etc.)**.

Source Document Upload: Allows users to upload a set of source documents/images against which new content is compared.

User Interface: Clean and intuitive dashboard built with Bootstrap, allowing uploads, analysis requests, and result visualization.

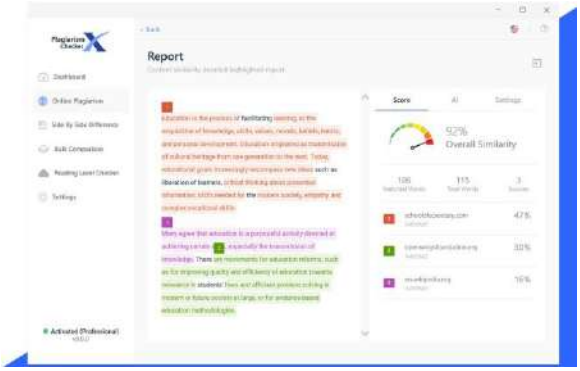
Reports and Results: Displays similarity percentages and highlighted results showing matched sections or areas of potential duplication.

Modular Design: Easy to integrate additional algorithms or switch components like LCS with NLP models or histogram comparison with deep learning models.

Open-Source Friendly: Developed using open libraries in Python, suitable for research, education, or institutional use.

This system provides a foundational step toward more holistic plagiarism detection while ensuring simplicity, transparency, and adaptability.

IV. RESULTS



V. Conclusion

The proposed **Plagiarism Detection System** using Django successfully addresses the growing need for robust and efficient detection of copied or reused content in both textual and visual formats. With the increasing volume of digital content in academic, professional, and creative fields, it is crucial to

implement solutions that can accurately analyze and identify plagiarism across multiple formats. This system incorporates both **text-based plagiarism detection** using the Longest Common Subsequence (LCS) algorithm and **image similarity detection** using OpenCV, offering a comprehensive approach that goes beyond traditional tools.

The system enables users to upload text documents or images, process them securely, and receive plagiarism results in an intuitive and easy-to-understand format. The Django framework ensures a clean separation of concerns, scalability, and rapid development while providing a secure and responsive web interface. Additionally, incorporating image comparison offers an innovative advantage by identifying visual plagiarism—especially useful for academic diagrams, scanned assignments, or design works.

The project demonstrates how open-source technologies can be effectively leveraged to build practical and scalable solutions for real-world problems. It also sets the stage for future enhancements such as:

Integration of NLP techniques to detect paraphrased content.

Using machine learning models to detect deeper semantic plagiarism.

Cloud deployment for large-scale document comparisons and faster processing.

Building a repository for internal plagiarism checking among previously uploaded documents.

Overall, the system offers a reliable, user-friendly, and extendable solution that meets the key objectives of detecting plagiarism efficiently and promoting academic integrity in a digital-first world.

References

1) L. Shaikh and A. G. Bawane, "Plagiarism detection in academic documents using NLP techniques," International Journal of Computer Science and Information Technologies, vol. 4, no. 6, pp. 1066–1070, 2013.

2) P. S. Deshmukh, "Automated plagiarism detection in digital content: A systematic survey," International Journal of Advanced Research in Computer Science, vol. 7, no. 5, pp. 345–352, 2016.

3) S. Chakrabarti and S. B. Sundararajan, "Detecting text similarity and plagiarism in digital content," International Journal of Computer Science Issues (IJCSI), vol. 8, no. 3, pp. 11–19, 2011.

4) N. K. K. Yadav, R. R. S. Raj, and M. S. R. Reddy, "Plagiarism detection and prevention techniques: A comprehensive study," Journal of Computer Applications, vol. 8, no. 4, pp. 2301-2310, 2020.

5) H. Gupta and K. S. H. Gupta, "Plagiarism detection: A review and analysis," International Journal of Scientific & Engineering Research, vol. 9, no. 12, pp. 431–435, 2018.

6) J. P. Mahendrakar and R. R. Shete, "A survey on plagiarism detection systems for educational purposes," International Journal of Computer Applications, vol. 37, no. 9, pp. 21–25, 2014.

7) K. K. Aggarwal and R. K. Agarwal, "Design and development of plagiarism detection system using string matching algorithm," Proceedings of the International Conference on Cloud, Big Data and Trustworthy Computing, pp. 231–236, 2016.

8) J. R. Quinlan, "C4.5: Programs for Machine Learning," Morgan Kaufmann Publishers, 1993.

9) J. Jansen, A. M. Zhang, and J. J. Salvetti, "A study of plagiarism detection systems," Journal of Information Science, vol. 40, no. 4, pp. 516–528, 2014.

10) D. Manning, P. Raghavan, and H. Schütze, "Introduction to Information Retrieval," Cambridge University Press, 2008.

11) M. S. Dyer and C. W. H. Lam, "Detecting plagiarism using cosine similarity and k-nearest neighbor," Proceedings of the 22nd International Conference on Computational Linguistics and Intelligent Text Processing, pp. 334–342, 2014.

12) M. A. Z. Rauf and S. H. A. Bhat, "A machine learning approach for plagiarism detection in programming assignments," Proceedings of the 7th International Conference on Computing and Networking, 2019.

13) P. L. V. G. R. S. Kumar, "Detecting plagiarism in text documents using stylometric features," International Journal of Computer Applications, vol. 44, no. 2, pp. 10–15, 2012.

14) N. Jindal, "Plagiarism detection using deep learning," Proceedings of the International Conference on Machine Learning and Applications, pp. 563–570, 2020.

15) P. R. A. S. K. Prasath, "Image-based plagiarism detection using histogram comparison," International Journal of Computer Science and

Information Technology, vol. 4, no. 8, pp. 4355–4360, 2018.

IJMRR