IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511137

Machine Learning and NLP-Based Integrated Text and Image Plagiarism Detection System

Pradnya Chhotu Patil[1], Prerna Motilal Patil[2], Mihir Vikas Wagh[3], Rohit Dilip Solunke[4], Dr. Y. R. Pathak[5]

Department of Computer Science and Engineering (Data Science), R. C. Patel Institute of Technology, Shirpur, Maharashtra, India[1-4]

Project Guide, Department of Computer Science and Engineering (Data Science), R. C. Patel Institute of Technology, Shirpur, Maharashtra, India[5]

Abstract: Plagiarism detection is a fundamental requirement in academic writing and in the management of digital content. It's getting harder and harder to detect plagiarism as content can be copied directly, paraphrased, scanned, or even be in the form of an image containing text. Traditional text matching approaches can fail to identify similarities between sentences if the sentences have been reworded or restructured. To deal with this, we present PlagiarismGuard, a web-based system that performs a range of analyses to determine textual similarity. The system proposed combines TF-IDF, N-gram Jaccard, and Sentence-BERT (SBERT) to look at the lexical, phrase, and semantic relationships between text passages, respectively. It also has Tesseract OCR to pull readable text from the supported image formats. The app is built on React for the front end and Python Flask for the back end. When all three analysis techniques are available, the overall similarity score is calculated with SBERT at 50%, TF-IDF at 30%, and N-gram at 20%. If Sentence-Transformers is not available, then the alternative calculation is used (TF-IDF at 60%, N-gram at 40%). PlagiarismGuard is designed to assist with the identification of potentially similar content and to help users with their own review. A proper assessment of its detection capability will require testing against a properly labelled dataset prior to any reporting of quantitative results.

Keywords: Plagiarism Detection, Machine Learning, Natural Language Processing (NLP), Term Frequency–Inverse Document Frequency (TF-IDF), N-gram Similarity, Sentence-BERT (SBERT), Optical Character Recognition (OCR).

I. INTRODUCTION AND PROBLEM STATEMENT

The proliferation of digital information has made it easier to access and re-use academic material. That's a good thing in so far as it promotes the sharing of knowledge, but it also makes the prospect of plagiarism all the more likely and the issue of originality all the more difficult to manage. Plagiarism is the act of using someone else's ideas or work without proper attribution. It can be done by direct copying, by paraphrasing, by using altered sentences, or by taking text from scanned documents and images. That's why plagiarism detection isn't simply a matter of matching words. Traditional text matching can fail to pick up on similar content if different words are used to convey the same meaning. Semantic analysis is the way to go in such cases. OCR is used to extract readable text from the images that we support. Proposed PlagiarismGuard uses TF-IDF, N-gram Jaccard, and Sentence-BERT (SBERT) for the analysis. Tesseract OCR is used for the extraction of text from images. The app is built on React and Python Flask. The similarity score will indicate the potential for overlap and can be used to flag content for further investigation. The actual determination of plagiarism will still require a human to look at it and check the sources.

II. LITERATURE REVIEW

Foltýnek, Meuschke and Gipp [1] looked at academic plagiarism detection methods and pointed to the value of using a combination of techniques to detect similar content.

Salton and Buckley [2] looked at term weighting techniques which form the basis of TF-IDF. That, in conjunction with cosine similarity, is what's used to measure the lexical similarity between text passages.

Broder [5] looked at document similarity and containment in terms of shared content. N-gram Jaccard similarity works by comparing common n-grams to identify common phrases and patterns.

Reimers and Gurevych [3] introduced Sentence-BERT, which produces sentence embeddings to identify semantically similar text even when the words are different.

Smith [4] described the Tesseract OCR engine, which extracts machine-readable text from images and scanned

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511137

documents for further analysis.

Based on these studies, PlagiarismGuard uses a combination of TF-IDF, N-gram Jaccard similarity, Sentence-BERT, and Tesseract OCR in a web app to analyse text similarity and extract readable text from text in images.

III. PROPOSED METHODOLOGY

PlagiarismGuard runs through a few steps: collecting input, pulling the text, preprocessing it, checking for similarity, combining scores, and generating results. Mixing different methods looks at how words, phrases, and meaning line up between texts.

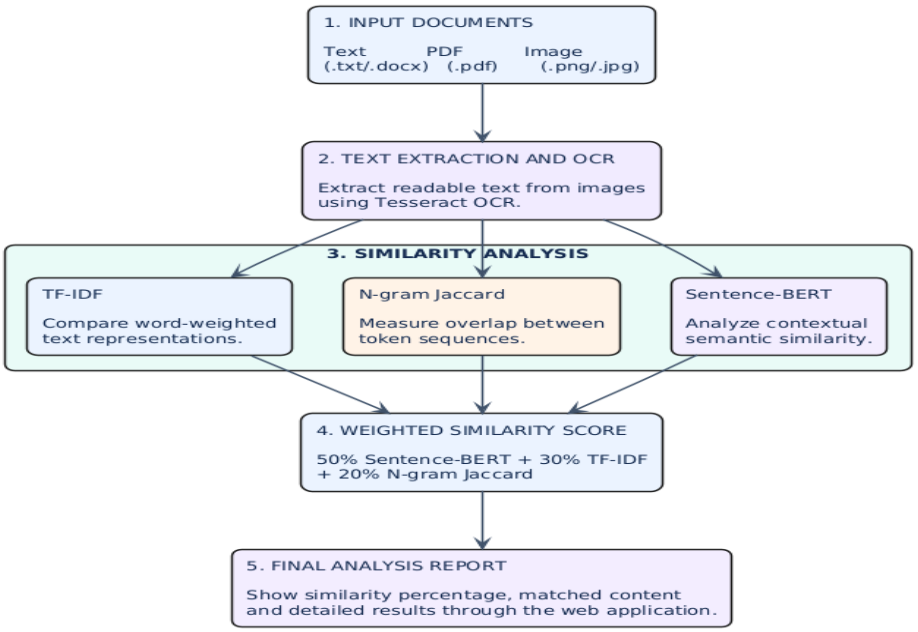


Fig. 1. System Architecture of Proposed Plagiarism System

A. System Input and Text Preprocessing

The app can take text for plagiarism checks and pull text from images using Tesseract OCR. The text gets turned into a machine-readable format and sent straight to the similarity pipeline. How well the images turn out really comes down to how clear and readable the starting stuff is. The system can also pull text from PDFs and send it off for preprocessing and similarity checks.

B. Text Pre-processing

The extracted material is then made ready for similarity testing. The pre-processing step ensures that the material is normalized and transformed into a common form. This process can involve stripping away extra spaces and the normalization of text. The same pre-processing procedure should be followed on both the input and the reference material.

C. TF-IDF Similarity

Term Frequency-Inverse Document Frequency (TF-IDF) is the weight of a term based on its frequency in a document and its frequency in the corpus. It converts text into numeric vectors and uses cosine similarity to calculate similarities between those vectors. It helps in finding common terms and important terms in texts. The downside of this approach is that it relies mostly on lexical similarity and does not detect paraphrases at all.

D. N-gram Jaccard Similarity

An n-gram is a sequence of n tokens that appear consecutively in a text document. N-gram Jaccard similarity measure: The similarity between the token sequences of the text passages is assessed by applying the formula,

$$J(A, B) = |A \cap B| / |A \cup B|$$

where A and B are sets of N-grams extracted from the two text documents. The score is between 0 and 1. The closer to 1, the more similar the sets of N-grams are.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511137

E. Sentence-BERT Semantic Similarity

Sentence-BERT produces dense vector representations that capture the context-dependent meaning of sentences. You can compare embeddings of two sentences or two chunks of text with cosine similarity. This technique will be able to detect even sentences that have semantically similar meaning but different words. In the proposed system, the semantic similarity feature is provided by Sentence-BERT.

F. Weighted Similarity Calculation

If all three components are present, then they are combined into an overall similarity measure via certain weights. The weights indicate the contribution of each component to the final measure. The overall similarity measure will look like this:

Overall Score = (0.50 * SBERT) + (0.30 * TF-IDF) + (0.20 * N-gram Jaccard)

These are the current weights, and they are not ideal in any sense whatsoever. They are not ideal and have not been experimentally validated. If Sentence-BERT is not available, then a 60/40 split of TF-IDF and N-gram Jaccard similarity is used.

G. Report Generation

Similarity analysis results are then processed and prepared by the backend and presented via the React app. The output will show the similarity indicator and the relevant content if any. It's worth noting that similarity is an automatic indicator and does not in itself imply plagiarism.

IV. EXPERIMENTAL RESULTS

The proposed Plagiarismguard system looks at the text content and the text that can be extracted from the images via Tesseract OCR. It uses TF-IDF, N-gram Jaccard and Sentence-BERT to compare text passages for similarity. Fig 2 is the main application interface, Fig 3 is the frontend and backend execution, Fig 4 is the image based text extraction via OCR and Fig 5 is the plagiarism analysis with similarity scores.

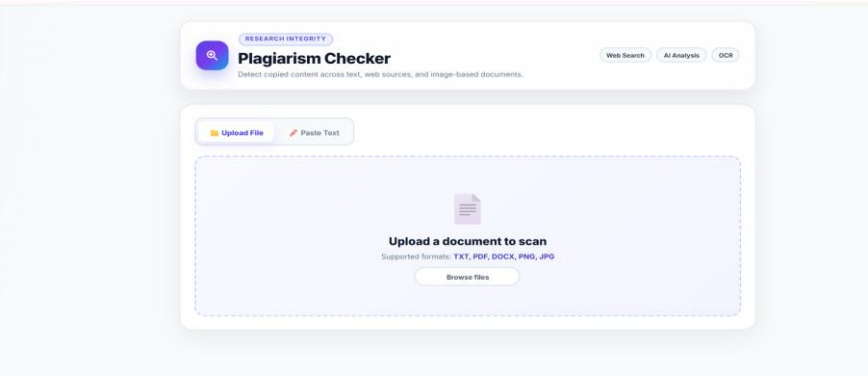
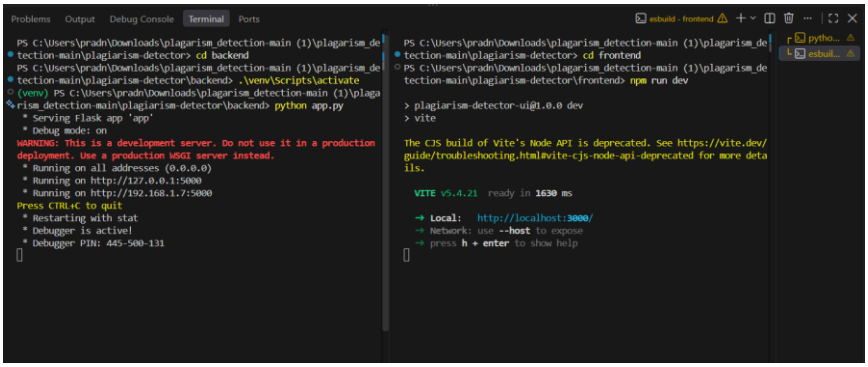


Fig. 2. Main Interface of the Proposed Plagiarism Detection System



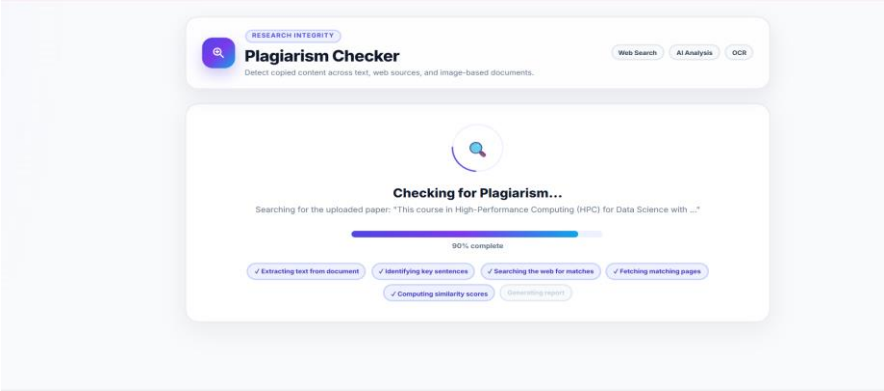
(a)

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

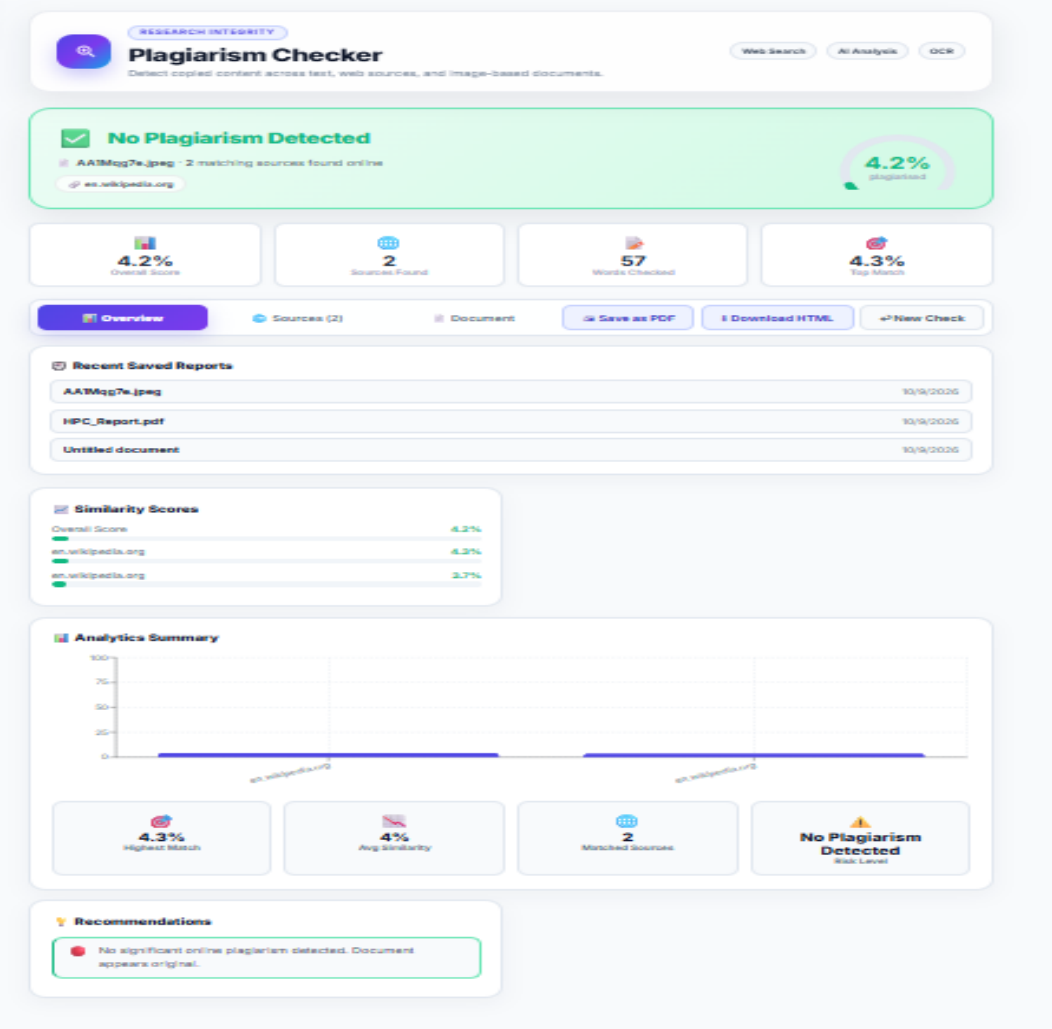
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511137



(b)

Fig. 3. Execution of Frontend and Backend Modules and Plagiarism Analysis Process



 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

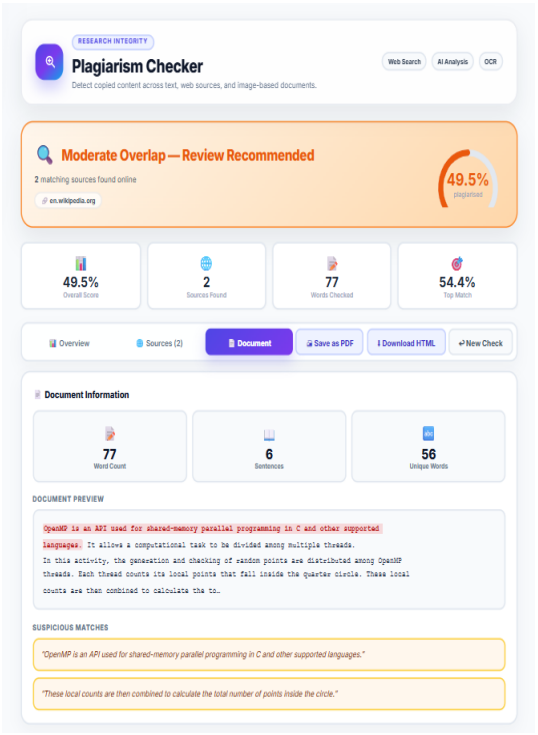
International Journal of Advanced Research in Computer and Communication Engineering

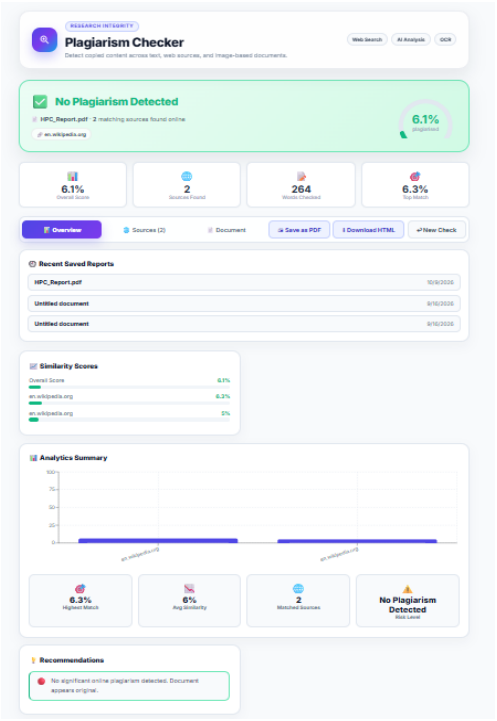
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511137

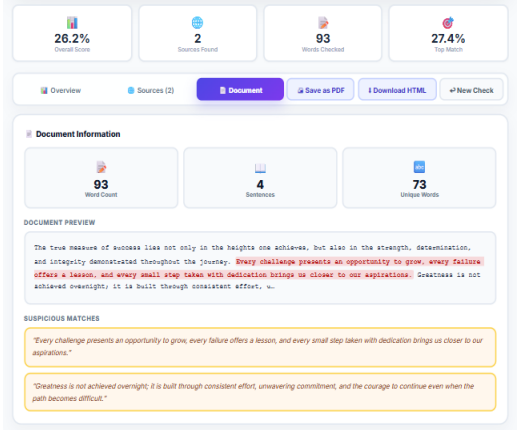


Fig. 4. Plagiarism Detection in Image-Based Documents Using OCR

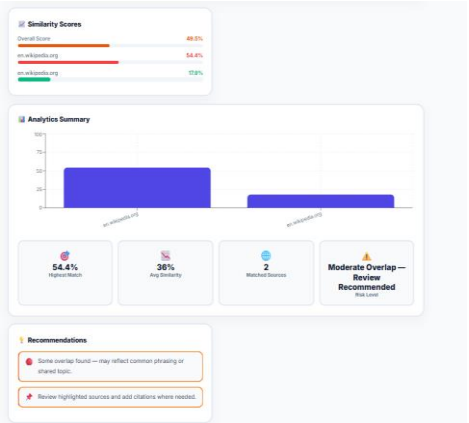




(b) PDF-Based Plagiarism Detection



(a)Text-Based Plagiarism Detection



(c) Similarity Scores and Analytics

Fig. 5. Text and PDF Plagiarism Detection Results with Similarity Score Analysis.

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511137

V. CONCLUSION

PlagiarismGuard uses Machine Learning and Natural Language Processing to identify potential similarities in text. The various models used are TF-IDF, N-gram Jaccard Similarity and Sentence-BERT for lexical, phrase and semantic matching respectively. Tesseract OCR is also used to extract readable text from images.

These are implemented via the weighted similarity model through a web app based on React and Python Flask. The aim of the project is to assist users in investigating potential similarities in content, with the caveat that any actual decision must be made by a human. Potential future work would be to try out larger labelled data sets, to measure the performance of the application using appropriate metrics, to improve the efficiency and to extend the functionality to detect visual similarities as well as the text extracted from images.

REFERENCES

[1] T. Foltýnek, N. Meuschke and B. Gipp, “Academic Plagiarism Detection: A Systematic Literature Review", ACM Comput. Surv., vol. 52, no. 6, Article 112, 2019. doi: 10.1145/3345317

[2] G. Salton and C. Buckley, “Term-Weighting Approaches in Automatic Text Retrieval", Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988. doi: 10.1016/0306-4573(88)90021-0.

[3] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks", in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3982–3992, 2019. doi: 10.18653/v1/D19-1410.

[4] R. Smith, “An Overview of the Tesseract OCR Engine”, in Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR), pp. 629–633, 2007. doi: 10.1109/ICDAR.2007.4376991.

[5] A.Z. Broder, “On the Resemblance and Containment of Documents”, in Proceedings of the Compression and Complexity of Sequences 1997, pp. 21–29, 1997. doi:10.1109/SEQUEN.1997.666900.

.

 This work is licensed under a Creative Commons Attribution 4.0 International License