International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Plagiarism Detection and Performance Analysis

Mohit N. Chavan, Aditya S. Patil, Piyush S. Mahajan, Chinmay R. Songirkar

Department of Computer Science, Shram Sadhana Bombay Trust College of Engineering And Technology Bambhori, Jalgaon 425001, Maharashtra, India.
mohitchavan718@gmail.com, Adityapatil122002@gmail.com, mrpiyush11@outlook.com, crsongirkar2@gmail.com

ABSTRACT—

Plagiarism means using someone"'s work without any acknowledgment of that person. Copying and Pasting the work of various authors has led to Copyright Infringement. Plagiarism in free-text contexts has become distressingly commonplace, largely attributed to the vast availability of information resources. This phenomenon has driven the devel- opment of automatic plagiarism detection systems, which aim to sift through extensive repositories in search of plagiarized content. However, the task is significantly complicated by the utilization of intricate plagiarism techniques such as paraphrasing and summarization, techniques that serve to obfuscate the presence of copied content. Plagiarism detection employs an array of techniques, each with its own unique strengths. For instance, N-gram Analysis dissects text into smaller word segments, enabling crossdocument comparison to reveal hidden similarities. String Matching Algorithms seek out exact or near-identical text matches, providing a lens through which direct copying or minor content modifications can be discerned. Fingerprinting takes a different approach by generating distinctive codes for specific text sections, facilitating efficient comparison and illuminating instances of paraphrased or restructured content. Our proposed project work includes study of various plagiarism detection algo- rithms, then implementation of some of the existing algorithns and finally to carry out performance analysis of the same. [1]

Index Terms— Cosine Similarity, Algorithms, Plagiarism, textual detection.

I. INTRODUCTION

Plagiarism detection is the system of recognizing the plagiarised content thru a trustable supply or gadget. The similarity of content beyond a certain restriction between or extra documents is not applicable and consequently, recognized as plagiarism. The task calls for many steps along with accepting the input in a specific layout, com- puting the resembling words and counting the ccurrences of a unmarried word in each the files and in the end reveal a similarity score. Now-a-days, unique types of strategies are being applied to research and understand the similarity conduct in files as like in utilized in growth of the commercial enterprise [3]

With the outbreak of the COVID-19 pandemic, the whole training gadget has been hooked up to be de- pendent on era through on-line lectures, assignments, and examinations. via this concept, an much less tough spotting of plagiarism in scholar's assignments and online examinations might be completed. [2].

Plagiarism detection tool are curtai to ensure the gen- uineness of intellectual work and to provide uniqueness to their content. Plagiarism checker also allows teachers to prepare their unique study materials for students. And bloggers can also check if their content is plagiarism- free or not on our site to keep it away from plagiarism. Automated detection has gained further interest from both the commercial worlds as well as academics, with providing the ease by which contents can now be revealed, copied and rewritten [1].

II. PROBLEM STATEMENT

Nowadays, extra frequently, assignments of students are submitted in digital bureaucracy, but it leads closer to the smooth possibility of plagiarism. With the spread of records over the globe, it is very clean to duplicate the information from distinctive assets and paste it in a single work without giving any acknowledgement to the assets.those actions lead in the direction of loss of learning in students. So there may be a need for detecting plagiarism to growth and enhance the quality of studying of a scholar. manual detection of plagiarism is difficult and time-consuming.This arises the want to design an auto- mated machine to stumble on plagiarism and additionally to enhance the quality of studying of the student.

III. METHODOLOGY

Plagiarism detection is a crucial task in academia and content creation. This methodology combines various techniques, including cosine similarity, n-gram analysis, string matching, and fingerprinting, to enhance the accu- racy and robustness of the plagiarism detection system. The first step involves preprocessing the text documents by tokenizing them into words and removing stop words, punctuation, and irrelevant characters. This ensures a clean and standardized representation for further analysis.

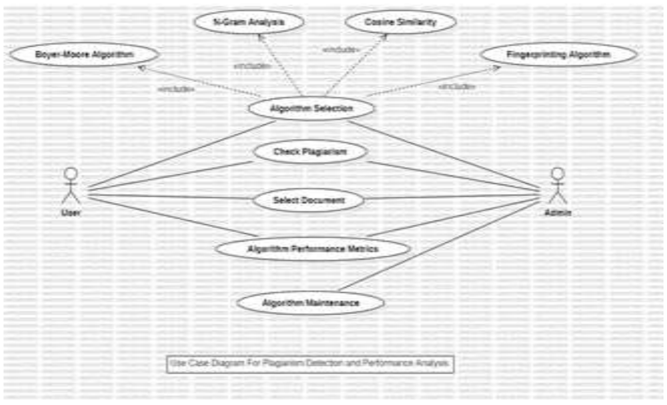
User-Centered Design (UCD): In the realm of plagiarism detection, the design process is guided by the principles of User-Centered Design, where the paramount focus is on understanding the needs and preferences of potential users, specifically those involved in developing or utilizing plagiarism detection tools.

Prototyping and Wireframing: Prototyping and wire- framing methodologies are integrated into the design process. Interactive prototypes are created to visualize the Plagiarism detection user interface, while wireframes provide a structural framework for the platform. These tools enable early user testing and feedback collection.

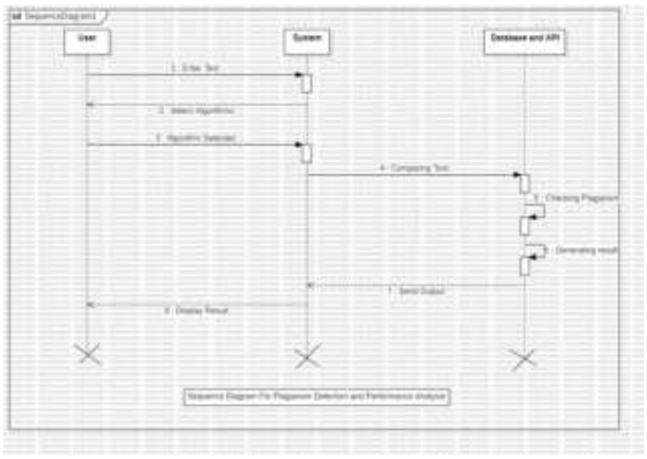
Visual Design Principles: Visual design principles are incorporated into the design process. This methodology involves selecting appropriate color schemes, typography, layouts, and visual elements to create an engaging and aesthetically pleasing user interface.

Collaborative Tools: Collaborative design tools, such as Star UML, are employed to enhance communication and teamwork within the project team. These tools assist in creating system architecture diagrams and visualizing the Plagiarism detection structure. [2].

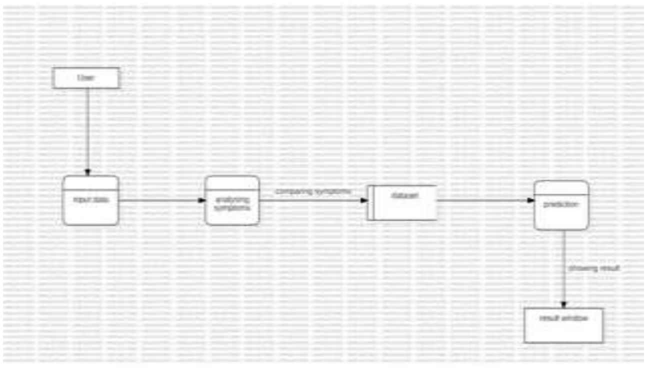
IV. DESIGN PROCESS



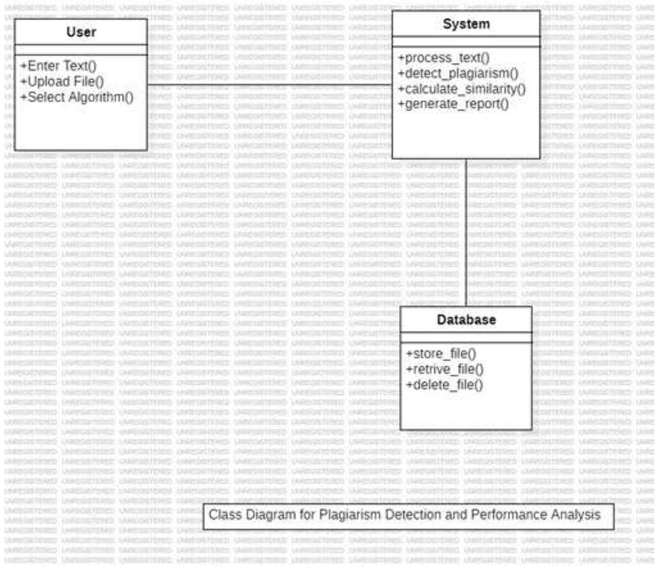
UseCase Diagram



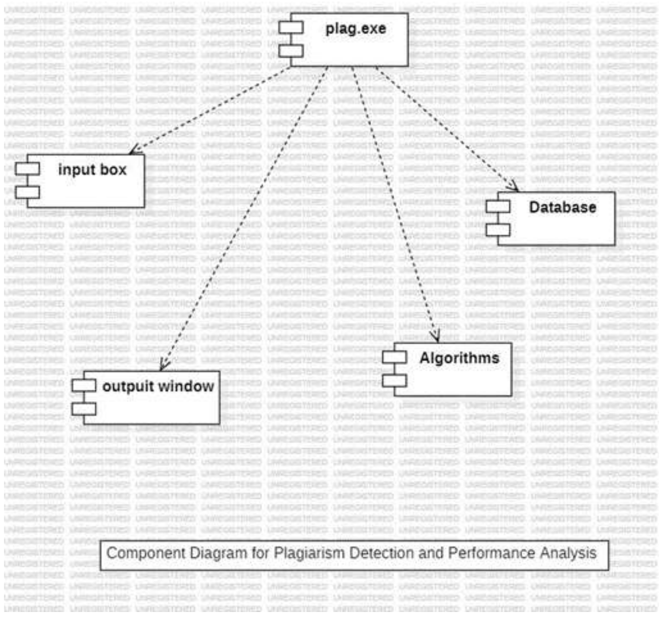
Sequence Diagram



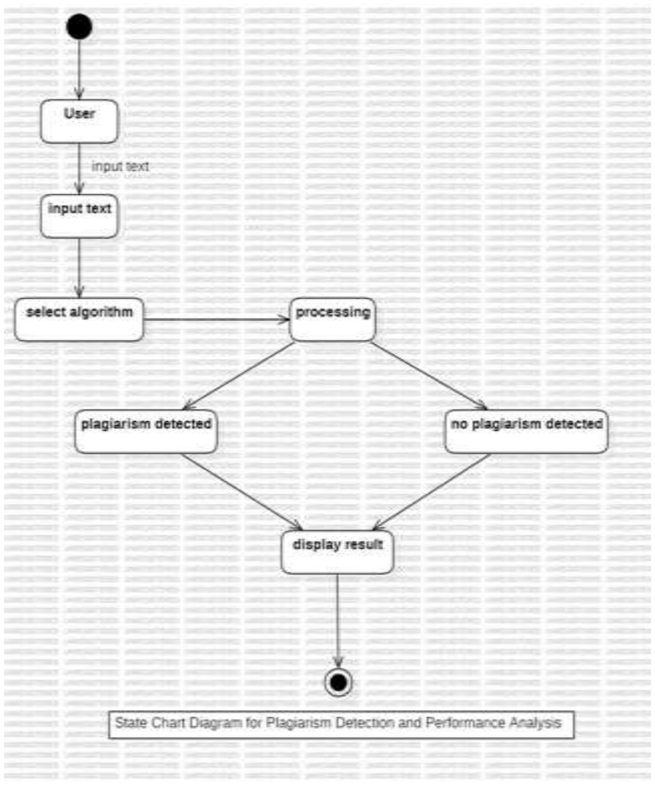
DFD-0 Diagram



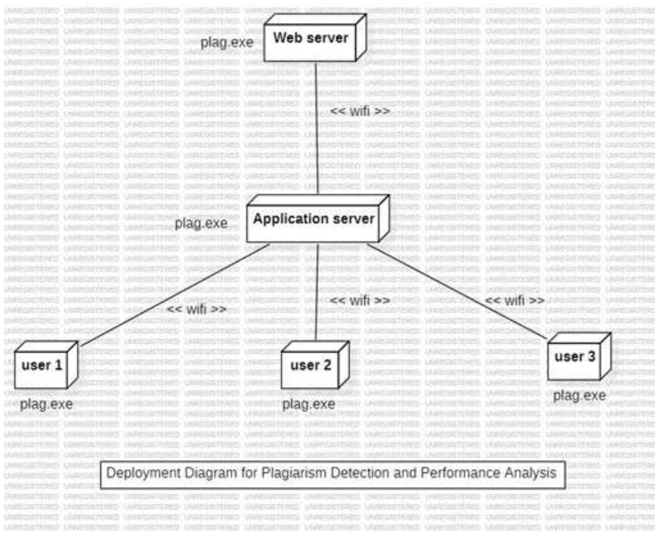
Class Diagram


plag.exe
input box
Database
Algorithms
outpuit window
Component Diagram for Plagiarism Detection and Performance Analysis

State transition Diagram



Sate-Chart Diagram

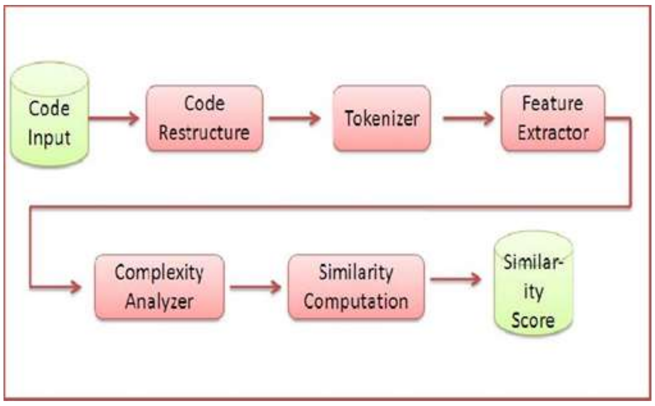


Deployment Diagram

V. SYSTEM DESIGN

Detection of plagiarism data using API is designed to check the quality of data at the source and in the suspicious data repository. In this section, we discuss the process outlined in Figure 1. We use the Thai thesaurus

[50] to extract keywords (or common words) from the data. The process of extracting words is represented by the characteristics of the main content, and the database provides the product of the legal content. The important points will then be processed into a lattice concept to store the relevant information in the plagiarism detection document. Similarly, data is expected to be used through integration and recovery of the lattice concept.



System Design

VI. ALGORITHM IMPLEMENTATION

Algorithms are the building blocks for finding textual similarities between documents. These algorithms pro- vide a deep understanding of programming concepts and problem-solving techniques that are essential for design- ing efficient detection tools. By carefully processing input documents through tokenization, stemming and feature vector construction these algorithms allow the compar- ison of text content using different similarity metrics such as cosine or jaccard similarity. They then evaluate pairwise similarities using predefined thresholds to dif- ferentiate originality from possible plagiarism cases. Post- processing and performance evaluations provide detailed reports highlighting the similarities found and ensuring the robustness and reliability of the detection process.

A. Cosine Similarity Algorithm :

The cosine similarity algorithm is used in plagiarism detection. it is nothing but a similarity and variations of vectors among or amongst many documents. It unearths the space and angle of separation between two vectors. If the space between two vectors increases, the attitude between the vectors will also increase, making similarity between the vectors decrease and vice versa. It also used in recommendation systems and extensively utilized to categorize the documents without difficulty. each time whilst a code is submitted within the coding platform, it is saved on a library. it's miles then checked with all other saved codes for plagiarism take a look at it. To implement a cosine similarity set of rules, we must first vectorize the target documents after completing all levels of pre-processing.

After vectorization, we find the similarities using the components. The result of the formulation degrees from zero to one. The accuracy of the end result is more whilst cosine similarity is done with stemming characteristic than with out doing stemming function whilst the similarity index is more than 0.7, it is able to be stated as copied and the individual that copied may be rejected from the test. but the cosine similarity isn't sufficient set of rules because it has less accuracy. So we moved directly to n-gram algorithm for better and plenty accuracy.

The formula for cosine similarity is given by:

$$\text{Cosine Similarity} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

where Ai and Bi represent the components of vectors A and B, respectively, and n is the dimensionality of the vectors.

B. N-gram Algorithm

N-gram is a contiguous sequence of N items, which can be words, sentences, letters, syllables, or phrases extracted from the given text. The parameter N represents the number of string pairs.

For instance, considering the text "There are many people in the world":

1) Unigram (n=1) = (There), (are), (many), (people), (in), (the), (world)

2) Bigram (n=2) = (There, are), (are, many), (many, people), (people, in), (in, the), (the, world)

3) Trigram (n=3) = (There, are, many), (are, many, people), (many, people, in), (people, in, the), (in, the, world)

The n-gram algorithm captures the frequency of oc- currence and sequence of these N-grams in the text. It's commonly used in natural language processing tasks such as language modeling, text generation, and information retrieval. The formula for calculating the n-gram proba- bility is: to skip comparisons of certain characters and perform advantageous shifts during the search process, leading to significant performance improvements compared to naive string searching methods.

The Boyer-Moore algorithm significantly reduces the number of character comparisons required to locate a pattern within the text, making it particularly suitable for large-scale plagiarism detection tasks. Its effectiveness lies in its ability to quickly eliminate portions of the text that cannot possibly match the pattern, thus focusing the search on potential matches.

The efficiency of the Boyer-Moore algorithm is typically quantified by its time complexity, which is expressed as follows:

Time Complexity $=O(n+m)$

where n is the length of the text and m is the length of the pattern.

This time complexity indicates that in the average case, the algorithm's performance grows linearly with the size of the text and the pattern. However, in the worst case, where the pattern occurs at every position in the text, the time complexity can degrade to $O(n \cdot m)$, which is still more efficient than many other string-searching algorithms.

VII. RESULT AND OUTPUT

P (wn|w1, w2,..., wn

) = count (w1, w2, ..., wn)

− count (w1, w2,..., wn−1)

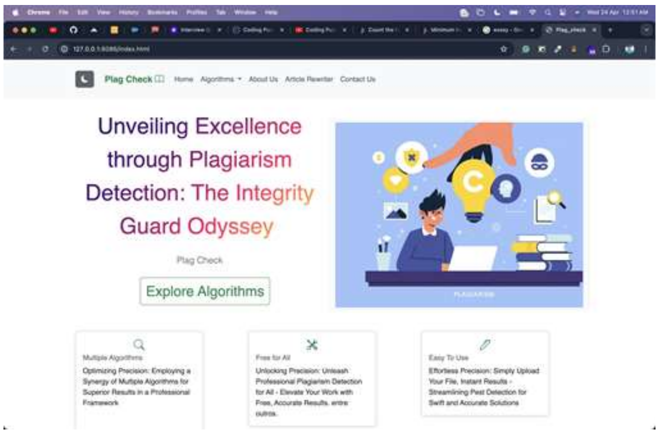
where w1, w2,..., wn represents the sequence of N items, and count (w1, w2,..., wn) denotes the frequency of occur- rence of the N-gram sequence in the text.

C. Fingerprinting Algorithm

Fingerprinting is a technique used in plagiarism detection to generate unique identifiers for specific text sections, enabling efficient comparison and identification of similarities. The fingerprinting algorithm operates by analyzing the structural characteristics and content patterns within the text to create distinctive codes or signatures. These signatures serve as compact representations of the text segments, facilitating rapid matching and detection of plagiarized content.

The formula for fingerprinting is given by:

$F(T) = \text{hash}(T)$



Unveiling Excellence through Plagiarism Detection: The Integrity Guard Odyssey

Plag Check

Multiple Algorithms

Optimizing Precision: Employing a Synergy of Multiple Algorithms for Superior Results in a Professional Framework

Free for All

Unlocking Precision: Unleash Professional Plagiarism Detection for All - Elevate Your Work with Free, Accurate Results. entre outros.

Easy To Use

Effortless Precision: Simply Upload Your File, Instant Results - Streamlining Pest Detection for Swift and Accurate Solutions

Fig 1

- This picture shows the home page featuring the introduc- tion to plagiarism checking and exploring algorithms. the unique requirements and patterns present in textual content.

Furthermore, we have To enhance efficiency, with Ma- chine learning algorithms and provide user data reports. The designing phase has provided us with a robust foun- dation for the development and implementation of the plagiarism detection tool and report generator. Moving forward, it is imperative to proceed with a well-structured development phase, ensuring smooth interactions for users.

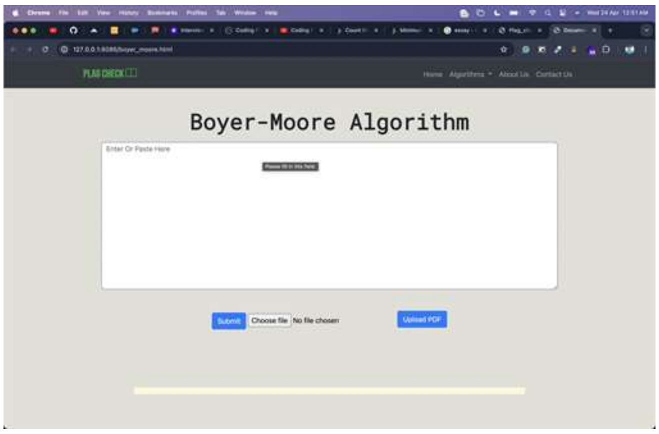


Fig 2

- This picture shows input textbox and the selected algorithm i.e Boyer-Moore Algorithm

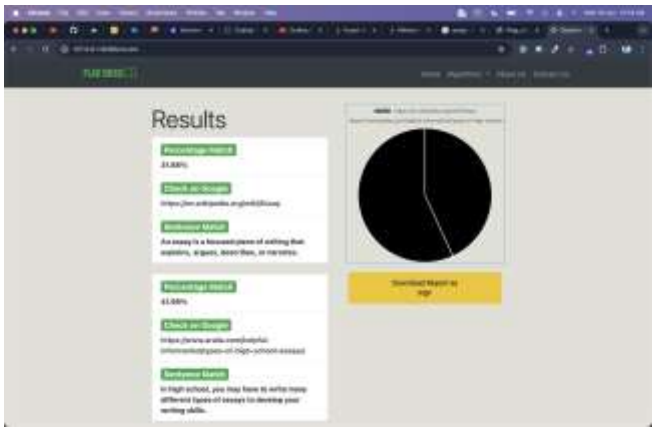


Fig 3

-This picture shows the results generated by the algorithm.

The algorithm successfully detected plagiarism in the provided text, providing accurate results on time.

VIII. CONCLUSIONS

In conclusion, the implementation phase of our re- search paper on Plagiarism detection has illuminated sev- eral crucial insights and considerations. We have outlined the key aspects of the design, emphasizing the importance of a robust architecture that incorporates machine learn- ing algorithms.

Our research underscores the importance of meticulous data collection and training procedures in the domain of plagiarism detection. Emphasizing the necessity for a comprehensive and varied dataset, the aim is to augment the system's language comprehension and analytical ca- pabilities. The insights garnered during the design phase shed light on the pivotal roles of intent recognition, context management, and personalization. These factors are identified as critical components in the creation of a plagiarism detection system that can adeptly adapt to

REFERENCES

[1] Sri, M.B., Bhavsar, R. and Narooka, P., 2018. String matching algorithms. Int. J. Eng. Comput. Sci, 7(03), pp.23769-23772.

[2] Siddharth Tata, Suguri Charan Kumar, P.Dec 2019 , Extrinsic Pla- giarism Detection Using Fingerprinting Int. Dept. of ECE, GITAM University, Hyderabad, India.

[3] Efstathios Stamatatos Intrinsic Plagiarism Detection Using Charac- ter n-gram Profiles University of the Aegean p . 2019 , 83200 - Karlovassi, Samos, Greece.

[4] Chitra, A. and Rajkumar, A. (2016) Plagiarism Detection Using Ma- chine Learning-Based Paraphrase Recognizer. Journal of Intelligent Systems, Vol. 25 (Issue 3), pp. 351-359. https://doi.org/10.1515/jisys- 2014-0146

[5] Farhan Ullah, “Software Plagiarism detection in multiprogramming language using machine learning approach”, Concurrency and Computation: Practice and Experience 2018.

[6] Zakiy Firdaus Alfikri, Ayu Purwarianti, “Detailed Analysis of Ex- trinsic Plagiarism Detection System Using Machine Learning Ap- proach”, 2014 TELKOMNIKA Indonesian Journal of Electrical Engi- neering

[7] Cynthia Kustanto, Inggriani Liem, “Automatic Source Code plagia- rism Detection”, IEEE Xplore 2009