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

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 1, January 2024

DOI: 10.17148/IJARCCE.2024.13119

ONLINE PLAGIARISM CHECKER WITH OCR

Aryan Patil[1], Harsh Bagde[2], Prathmesh Kharwade[3], Tanmay Khobragde[4], Yash Shembe[5], Prof.M.Gotaphode[6]

Students, Department of computer Science and Engineering, Priyadarshini college of Engineering, Nagpur, India[1-5]

Guide, Department of computer Science and Engineering, Priyadarshini college of Engineering, Nagpur, India[6]

Abstract: This study focuses primarily on plagiarism, which is prevalent in schools and colleges. Many students have been found to have copied assignments from their classmates. A system could be developed for the convenience of teachers that could check the amount of plagiarism in students' assignments. This system could be mentioned as an improvement from the old manual way as it eliminates the ability to steal someone else's ideas or work and passes it off as their own. Plagiarism has been classified as a moral rights infringement in a number of countries. It has become increasingly common in today's environment of changing technology and ever-increasing Internet usage.

Keywords: Plagiarism, data mining, stop word, hash tag, cosine similarity, cleaning, and stemming.

I. INTRODUCTION

The act of using someone else's words or ideas as one's own is known as plagiarism. This grammar and plagiarism checker technology is used to assess the plagiarism data. Plagiarism harms students' education and, thus, the economic status of the nation. Synonym and verbatim overlaps, paraphrased works, and phrase conversions are all ways that plagiarism is committed, and WordNet can identify these instances. This plagiarism detector looks for plagiarism by comparing comparable pieces of content. Students' lifestyles and learning styles have changed as a result of the internet. It makes the process easier for students while also enabling them to go deeper into their learning strategy. There are several approaches to find instances of plagiarism. The most common technique for detecting plagiarism is text mining. With the help of this plagiarism detecting program, users can generate a working login ID and password and register using their basic registration details. Students can access their personal accounts by entering their login credentials. Students can then upload the assignment file, which is separated into content and reference links. This web application will go over each reference link, parse the information, and compare the original webpage's content to that one. Students may also look into the historical context of earlier publications. Pupils can also proofread the article for grammatical mistakes.

II. LITERATURE SURVEY

A. Detecting plagiarism in computer programming by extracting features from ultra-fine-trained repositories.

Overview: By monitoring all action students take in a cloud based integrated development environment (IDE) while completing assignments, this study offers a remedy to these problems. By using repository mining techniques to construct "developer profiles," this log is managed similarly to a source code repository, making it feasible to recognize to the questionable activity. Although the paper focuses on plagiarism detection, a similar method may be applied in an educational setting to identify pupils who might benefit from extra help or special attention in particular subjects. A typical issue with source repositories is poor resolution. The authors of this paper propose capturing minute changes using the IDE's "autosave" function to create an incredibly fine-grained repository that captures even individual keystrokes. "Autosave" is typically used in a manner that minimizes the effect on system performance, such that a document is stored following a certain amount of idle time. These modifications are then automatically committed to an unseen Subversion (SVN) shadow repository using file system monitoring techniques. This makes it possible for researchers to analyses and troubleshoot SVN repositories using well-known tools. Additionally, this method works with every known an integrated development environment and editor that enables autosave. The "Run program" and "Test" buttons provide output files (executables, core dumps, etc.) that may be used to analyse IDE interactions and unit test results. This paper proposes a Change Recording or Change Logging method in which the system records specific, atomic changes to the source code text. The high-resolution log is expected to reveal the semantics of the change, which can then be analysed further using machine learning. This paper's premise is that students who plagiarize assignments utilize their IDE differently than students who work alone.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 1, January 2024
DOI: 10.17148/IJARCCE.2024.13119

The solution offered here is cloudbased, thus no tools, such as keyloggers, were installed on student PCs due to ethical considerations. Students were required to sign a consent form permitting their anonymized use history to be utilized for research reasons. Those who refused were not permitted to proceed. These ultra-fine-grained repositories were created for 300 students completing programming assignments in an introductory university programming course.

B. Online Assignment plagiarism checking using Data mining & NLP

Overview: The act of using someone else's words or ideas as one's own is known as plagiarism. This grammar and plagiarism checker technology is used to analyse the plagiarism data. Plagiarism harms students' education and, thus, the economic status of the nation. Synonym and verbatim overlaps, paraphrased works, and phrase conversions are all ways that plagiarism is committed, and WordNet can identify these instances. This plagiarism detector looks for plagiarism by comparing comparable pieces of content. Students' lifestyles and learning styles have changed as a result of the internet. It makes the process easier for students while also enabling them to go deeper into their learning strategy. Many methods are used to detect something. Text mining is the most widely used technique for identifying plagiarism. With this plagiarism detector program, users can generate a legitimate login ID and password by registering with their basic registration information. By providing their login credentials, students can gain access to their own accounts. A file with the assignment's content and reference links can then be uploaded by students. This webpage will examine each reference link, evaluate the information, and contrast that webpage's content with the original.

III. OBJECTIVE

- The primary goal is to use duplicate records to verify the plagiarism report.
- The goal is to verify that the assignment is free of plagiarism by comparing it to papers that are kept in a database.
- To provide a plagiarism report as a percentage of the student assignments that were turned in.
- To display similar sentences in the files that the students, if any, have supplied.

IV. SCOPE

In recent years, plagiarism detection has become a major area of research. A wide range of fields are susceptible to plagiarism, including software code, academic writings, and the arts. In the digitally advanced future, nothing will be done on paper—everything will be done online. The plagiarism detection program will therefore be very helpful in the future.

V. EXISTING SYSTEM

- For teachers, identifying copied passages in an assignment is an extremely laborious task.
- The teacher must be able to read and retain each submission, even with a small quantity of texts.
- Because the teacher's ability to recall what they have read is the basis for identifying copied passages in assignments, the results might not be comprehensive.
- Some blatant instances of copying and pasting could be easily missed. Furthermore, as the workload cannot be divided among several assistance.

VI. PROPOSED SYSTEM

A. Overview: We will create a system to identify plagiarism in academic assignments using the suggested system, which will help to deter students from plagiarizing work, enhance the standard of instruction, and help students develop their personal skills. Students will also be able to check the assignment's grammar. This system's plagiarism detector looks for plagiarism by comparing similar texts. Semantic checking with regard to assignment will also be completed. We will employ natural language processing and data mining algorithms to find instances of plagiarism.

B. Giving the input: By providing the file path where all of the papers have been stored, the input can be any type of document. All of the student submissions for assignments should be kept in that file location. The document and process will go on to the next stage as soon as we enter the file location.

C. Coinventing to text document: The following step is to convert all of the files to text format once they have been obtained. Any file type, including Word, PDF, and document (Portable Document Format), will be converted to text format. We are switching to text files since it will be simpler to perform pre-processing on them than on other types of files.

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

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 1, January 2024

DOI: 10.17148/IJARCCE.2024.13119

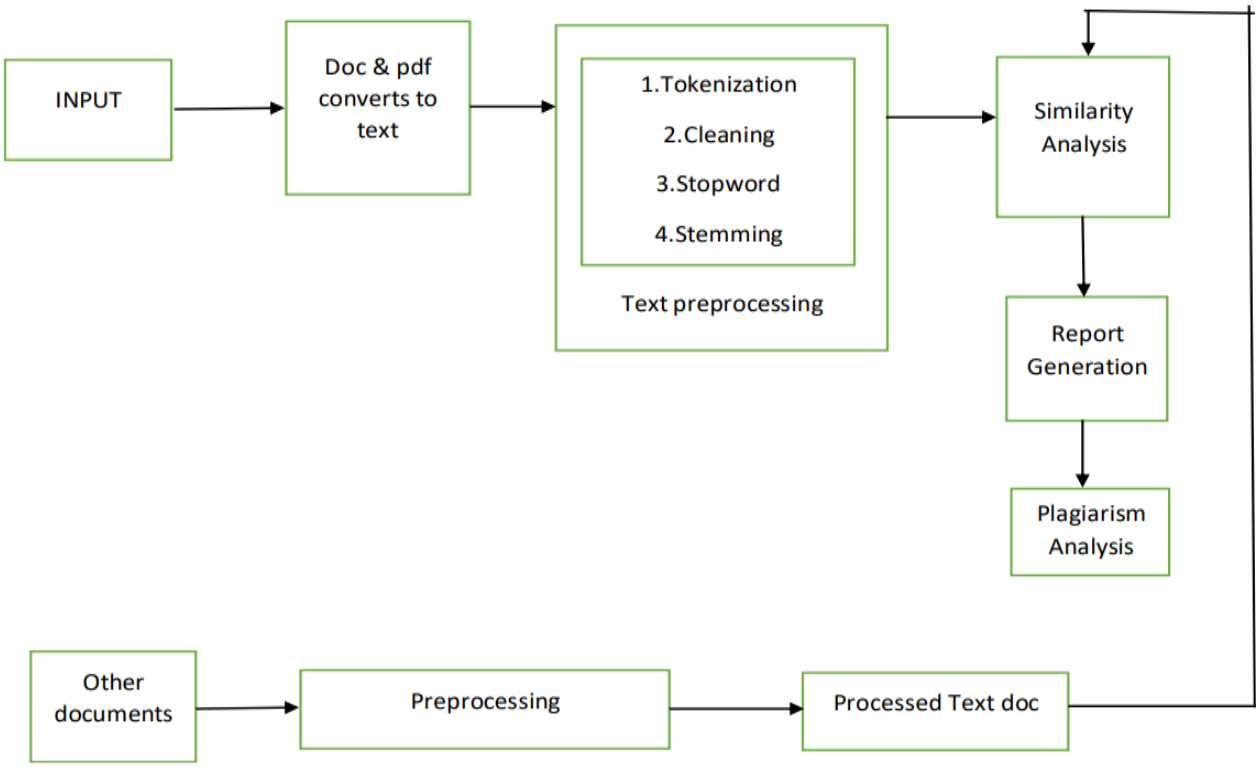

Fig. 1 Block Diagram

D. Pre-Processing: The text files will undergo pre-processing, which entails the subsequent steps, after which they are prepared for comparison analysis.

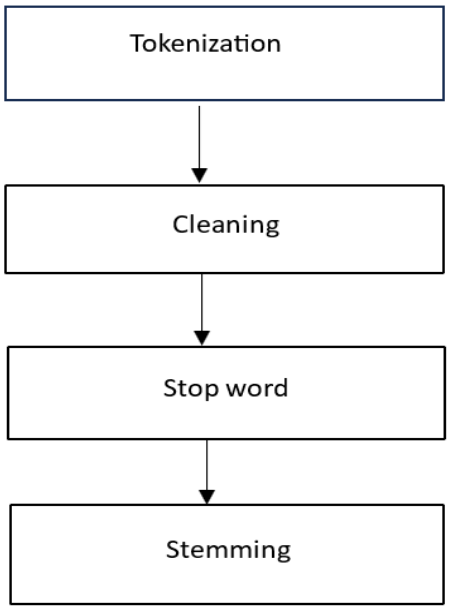

Fig. 2 Pre-Processing

E. Tokenization: Tokenization is the act of dividing a text string into a list of tokens, or tokenizing the text. Tokens can be conceptualized as parts; for instance, a word in a sentence is a token, and a sentence in a paragraph is a token.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 1, January 2024
DOI: 10.17148/IJARCCE.2024.13119

F. Cleaning: Data cleaning involves repairing or eliminating erroneous, corrupted, improperly formatted, duplicate, or incomplete data from a dataset. Combining data from many sources might lead to duplicates or inaccurate classifications. Inaccurate data can render algorithms and results untrustworthy, even if they are correct. Data cleaning methods differ between datasets, making it difficult to identify specific stages. G. Final Word: A commonly used word (such "the," "a," "an," or "in") that a search engine is programmed to ignore while indexing and retrieving entries in response to a search query is known as a stop word.

G. Steaming: Stemming is the process of creating morphological variations of a root or base word. Stemming programs are referred to as stemmers or stemming algorithms. Words like "chocolates," "chocolatey," and "Choco" reduce to the root word "chocolate," while words like "retrieval," "retrieved," and "retrieves" decrease to the stem "retrieve. The algorithm designed to find similarities between text documents first removes stop words and stems phrases to determine content similarity between given texts. The similarity between document samples (assignments) is quantified using a variety of metrics, including cosine similarity. A written report is arranged with facts for a specific readership and goal. While report summaries may be given verbally, full reports are normally provided in written form. This will show the percentage of plagiarism in the files that the student submitted. The percentage will be displayed for documents where plagiarism has been found, and 0 for documents where plagiarism has not been found.

H. Plagiarism Analysis: The act of claiming to be the creator of something that someone else actually wrote is known as plagiarism. This paper's concentration is on text documents, which is related to this definition. Of course, one of the most crucial questions is how much work like this can be automated. This will display the lines that are similar between two of the given papers.

VII. FUTURE SCOPE

The future of online plagiarism checkers using OCR seems hopeful, as these systems grow and improve. Here are some possible advancements that we might anticipate in the following years:

Improved AI Detection: Online plagiarism checkers using OCR will improve their detection of AI-generated text and paraphrase as technology progresses. This will assist to confirm the authenticity of written material and avoid unintended plagiarism.

Integration of Learning Management Systems: Online plagiarism checkers using OCR may be connected with learning management systems to give a consistent experience for students and teachers. This will provide automated plagiarism detection and real-time feedback to students, allowing them to improve their writing abilities.

Multilingual Support: As the globe grows increasingly linked, online plagiarism checkers using OCR must handle different languages in order to reach a worldwide audience. This will necessitate the advancement of OCR technology capable of effectively recognizing and analysing text in several languages.

Improved User Experience: Online plagiarism checkers using OCR will continue to improve the user experience, making them simpler and easier to use. This will allow users to traverse the platform more quickly and deliver a more consistent experience. Finally, online plagiarism checkers with OCR are effective tools for confirming the authenticity of written content. Finally, online plagiarism checkers with OCR are effective tools for confirming the authenticity of written content. As technology advances, we can anticipate these tools to become more complex and user-friendly, offering a consistent experience for students, instructors, and professionals.

VIII. CONCLUSION

Online plagiarism checkers with OCR (Optical Character Recognition) have become indispensable tools in academic and professional writing for ensuring content originality. These technologies employ OCR to scan and compare text to a large database of existing web information, therefore detecting instances of plagiarism. The use of OCR technology enables these checkers to analyse text within photos and PDFs, considerably increasing their efficacy in detecting plagiarism. The usage of plagiarism detection software is becoming increasingly relevant in academic and professional settings. With the rise of digital content, correctly assessing the originality of written work has become increasingly important. Plagiarism detection software, along with OCR, provides a comprehensive solution to this problem, allowing users to conduct extensive checks for plagiarized text across a variety of document formats. The combination of OCR with plagiarism detection software tackles the changing nature of plagiarism, such as the usage of pictures with embedded text and the alteration of textual material inside them. These programs use OCR technology to extract and evaluate text from photos, guaranteeing that no potential source of plagiarism is neglected. Finally, using OCR technology into online

DOI: 10.17148/IJARCCE.2024.13119

plagiarism detectors marks a big step forward in the fight against academic and professional plagiarism. These technologies give a more complete and effective method of detecting copied content by allowing for text analysis within pictures and PDFs. As the digital world evolves, OCR-augmented plagiarism checkers will play an important role in ensuring originality and integrity in writing and research.

REFERENCES

[1]. L. Prechelt and G. Malpohl, ‘‘Finding plagiarisms among a set of programs with JPlag,’’J. Universal Comput. Sci., vol. 8, no. 11, pp. 1016– 1038, Mar. 2003.
[2]. D. Grune and M. Huntjens, ‘‘Detecting copied submissions in computer science workshops,’’ Inf. Faculteit Wiskunde Informatica, Vrije Universiteit, Amsterdam, The Netherlands, Tech. Rep., 1989. [Online].
Available:http://www.dickgrune.com/Programs/sim ilarity_tester/Paper.ps
[3]. S. Schleimer, D. S. Wilkerson, and A. Aiken, ‘‘Winnowing: Local algorithms for document fifingerprinting,’’in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2003, pp. 76–85.
[4]. Q. D. Soetens, R. Robbes, and S. Demeyer, ‘‘Changes as fifirst-class citizens: A research perspective on modern software tooling,’’ACM Comput. Surveys, vol. 50, no. 2, pp. 1–38, Jun. 2017, doi: 10.1145/3038926.
[5]. J. Schneider, A. Bernstein, J. V. Brocke, K. Damevski, and D. C. Shepherd, ‘‘Detecting plagiarism based on the creation process,’’ IEEE Trans. Learn. Technol., vol. 11, no. 3, pp. 348–361, Jul. 2018.
[6]. K. Mierle, K. Laven, S. Roweis, and G. Wilson, ‘‘Mining student CVS repositories for performance indicators,’’ ACM SIGSOFT Softw. Eng. Notes, vol. 30, no. 4, pp. 1–5, Jul. 2005.
[7]. S. Neara, M. Vakilian, N. Chen, R. E. Johnson, and D. Dig, ‘‘Is it dangerous to use version control histories to study source code evolution?’’ in Proc. 26th Eur. Conf. Object-Oriented Program. (ECOOP), 2012, pp. 79–103
[8]. E. Giger, M. Pinzger, and H. C. Gall, ‘‘Comparing fifine-grained source code changes and code churn for bug prediction,’’in Proc. 8th Work. Conf. Mining Softw Repositories, 2011, pp. 83–92.
[9]. S. Negara, M. Codoban, D. Dig, and R. E. Johnson, ‘‘Mining fifine-grained code changes to detect unknown change patterns,’’ in Proc. 36th Int. Conf. Softw. Eng. (ICSE), 2014, pp. 803–813.
[10]. W. Maalej, T. Fritz, and R. Robbes, ‘‘Collecting and processing interaction data for recommendation systems,’’ in Recommendation Systems in Software Engineering. Berlin, Germany: Springer, 2014, pp. 173–197.
[11]. R. Robbes, D. Pollet, and M. Lanza, ‘‘Replaying IDE interactions to evaluate and improve change prediction approaches,’’ in Proc. 7th IEEE Work. Conf. Mining Softw. Repositories (MSR), May 2010, pp. 161–170.
[12]. Y. Yoon and B. A. Myers, ‘‘Capturing and analyzing low-level events from the code editor,’’ in Proc. 3rd ACM SIGPLAN Workshop Eval. Usability Program. Lang. Tools, 2011, pp. 25–30.
[13]. J. Hage, P. Rademaker, and N. van Vugt, ‘‘A comparison of plagiarism detection tools,’’ Utrecht Univ., Utrecht, The Netherlands, Tech. Rep. UU-CS-2010-015, 2010.
[14]. M. Mozgovoy, ‘‘Enhancing computer-aided plagiarism detection,’’ Ph.D. dissertation, Univ. Joensuu, Joensuu, Kuopio, 2007.
[15]. S. Burrows, ‘‘Source code authorship attribution,’’ Ph.D. dissertation, RMIT Univ., Melbourne, VIC, Australia, 2010.
[16]. V. T. Martins, D. Fonte, P. R. Henriques, and D. da Cruz, ‘‘Plagiarism detection: A tool survey and comparison,’’ in Proc. 3rd Symp. Lang., Appl. Technol. (SLATE), vol. 38, Braganáa, Portugal, 2014, pp. 143–158.
[17]. M. Agrawal and D. K. Sharma, ‘‘A state of art on source code plagiarism detection,’’ in Proc. 2nd Int. Conf. Next Gener. Comput.Technol. (NGCT), Oct. 2016, pp. 236–241.
[18]. D. Ganguly, G. J. F. Jones, A. Ramírez-delaCruz, G. Ramírez-de-la-Rosa, and E. VillatoroTello, ‘‘Retrieving and classifying instances of source code plagiarism,’’ Inf. Retr. J., vol. 21, no. 1, pp. 1–23, Feb. 2018.
[19]. X. Chen, B. Francia, M. Li, B. McKinnon, and A. Seker, ‘‘Shared information and program plagiarism detection,’’ IEEE Trans. Inf. Theory, vol. 50, no. 7, pp. 1545–1551, Jul. 2004.
[20]. J. A. W. Faidhi and S. K. Robinson, ‘‘An empirical approach for detecting program similarity and plagiarism within a university programming environment,’’ Comput. Edu., vol. 11, no. 1, pp. 11–19, Jan. 1987.
[21]. S. Engels, V. Lakshmanan, and M. Craig, ‘‘Plagiarism detection using feature-based neural networks,’’ ACM SIGCSE Bull., vol. 39, no. 1, pp. 34–38, Mar. 2007.
[22]. D. Gitchell and N. Tran, ‘‘Sim: A utility for detecting similarity in computer programs,’’ ACM SIGCSE Bull., vol. 31, no. 1, pp. 266– 270, Mar. 1999.
[23]. M. El Bachir Menai and N. S. Al-Hassoun, ‘‘Similarity detection in java programming assignments,’’ in Proc. 5th Int. Conf. Comput. Sci. Edu., Aug. 2010, pp. 356–361.
[24]. O. Karnalim and Simon, ‘‘Syntax trees and information retrieval to improve code similarity detection,’’ in Proc. 32nd Australas. Comput. Edu. Conf., Feb. 2020, pp. 48–55.
[25]. A. Budiman and O. Karnalim, ‘‘Automated hints generation for investigating source code plagiarism and identifying the culprits on in class individual programming assessment,’’ Computers, vol. 8, no. 1, p. 11, 2019

