International Journal of Engineering, Science and InformationTechnology
Volume 5, No. 1 (2025) pp. 185-194
ISSN 2775-2674 (online)
Website: http://ijesty.org/index.php/ijesty
DOI: https://doi.org/10.52088/ijesty.v5i1.686
Research Paper, Short Communication, Review, Technical Paper

Plagiarism Detection Application for Computer Science Student Theses Using Cosine Similarity and Rabin-Karp

Taufik Habib Ansyari Siregar*, Dahlan Abdullah, Lidya Rosnita

Department of Informatics, Faculty of Engineering, Universitas Malikussaleh, Aceh, Indonesia

**Corresponding author Email: taufik.190170054@mhs.unimal.ac.id*

The manuscript was received on 25 June 2024, revised on 28 August 2024, and accepted on 18 December 2024, date of publication 9 January 2025

Abstract

Plagiarism detection is critical in maintaining academic integrity, particularly in higher education. This study focuses on developing a plagiarism detection application for Computer Science student theses. The application leverages the Cosine Similarity and Rabin-Karp algorithms to accurately and efficiently detect textual similarities. Developed using JavaScript, the application provides an intuitive interface and reliable performance, making it a practical tool for educational institutions. The application includes features allowing users to upload thesis documents, analyze textual content, and measure plagiarism levels by comparing them to an existing dataset. The Cosine Similarity algorithm measures the overall similarity between documents, while the Rabin-Karp algorithm focuses on identifying exact matches in phrases and sentences. The results demonstrate the efficacy of both algorithms. For titles, the Cosine Similarity algorithm achieved a 100% similarity rate for identical documents while detecting minor plagiarism with a similarity level of 5.86% for other documents. For abstracts, it achieved 100% similarity for the first document, 2.78% for the second document, and 8.37% for the third document. These findings highlight the algorithm's ability to detect exact matches and partial overlaps in textual content. The Rabin-Karp algorithm showed comparable performance, particularly in detecting phrase-level similarities. For titles, it recorded 100% similarity for identical documents, 11.42% for the second document, and 16.92% for the third document. For abstracts, the algorithm also achieved 100% similarity for the first document, 11.42% for the second document, and 16.81% for the third document. The study confirms that both algorithms complement each other in detecting different forms of plagiarism. The Cosine Similarity algorithm excels in identifying global patterns of similarity, while the Rabin-Karp algorithm is more suited for finding exact matches in specific phrases or sentences. This dual approach provides a comprehensive solution for detecting plagiarism in academic theses. The findings from this research are promising and highlight the potential of the application as a reliable tool for ensuring academic integrity. Future improvements could include expanding the dataset, enhancing the user interface, and integrating additional algorithms for cross-language plagiarism detection. This application contributes to academic honesty and is a valuable resource for educators, researchers, and students in combating plagiarism effectively.

***Keywords**: Plagiarism, Detection, Thesis, Cosine Similarity, Rabin-Karp.*

1. Introduction

Universitas Malikussaleh provides a platform for students to enhance their knowledge and skills. One of the key requirements in their academic journey is completing a thesis, which is a prerequisite for earning a bachelor's degree. Writing a thesis involves research, analysis, and creating a scientific work. However, with the rapid advancement of information technology, the risk of plagiarism has become a pressing issue, threatening the integrity of academic achievements.

Plagiarism poses a serious challenge within academic environments, negatively impacting students and educational institutions. Therefore, it is crucial to implement preventive measures and reliable detection methods to address this issue effectively. The Cosine Similarity and Rabin-Karp methods are among the most effective techniques for detecting plagiarism.

Cosine Similarity is a computational method used to evaluate the degree of similarity between two documents by analyzing the proximity of word vectors within their content. In the context of theses, this method can be utilized to compare student submissions with reference documents or other theses, enabling the identification of potential instances of plagiarism.

By employing tools such as Cosine Similarity and **Rabin-Karp**, universities can ensure a rigorous and accurate process for detecting plagiarism, safeguarding the integrity of academic works while promoting ethical research practices [1].
A related study by [2], titled "Web-Based Digital Document Checker Application for Student Assignments," concluded that the application functions effectively by displaying the percentage of plagiarism along with its status, indicating whether plagiarism is detected or not. The success of this application in presenting such information is attributed to the optimal performance and proper sequencing of the TF-IDF and Cosine Similarity algorithms implemented in the system. Based on testing student assignments, it was observed that students learn to reduce plagiarism percentages. Although some plagiarism is occasionally detected in specific tasks, the application encourages students to take greater responsibility in completing their coursework.
Plagiarism is taking someone else's writing or ideas and claiming them as one's own, such as publishing another person's work under one's name or copying without permission. Plagiarism occurs in various forms, including artistic works and scientific literature [3]. Plagiarism is copying or taking someone else's work, opinions, or ideas and presenting them as if they were one's own. This practice has created a negative impact, particularly in education, as it stifles ideas, diminishes creativity, and undermines intellectual integrity [4]. Additionally, plagiarism can involve the unauthorized use of someone else's literary work, falsely claiming it as personal property. Original works are considered the intellectual property of their authors, and others are prohibited from reproducing them without permission from the copyright holder [5].
Students must actively seek additional reference materials when they encounter topics that are not fully understood. Independence is crucial in completing assignments, especially given the limited opportunities for interaction with peers and lecturers. The difficulty increases in online learning environments for STEM-related courses, which are often challenging to comprehend, even in face-to-face classes. Therefore, developing self-reliance is essential in the learning process [6].
As an educational institution committed to academic excellence, Universitas Malikussaleh requires a system that supports the integrity and originality of students' academic work. This research aims to develop a plagiarism detection application using the Cosine Similarity and Rabin-Karp methods, specifically for the theses of students in the Computer Science program at Universitas Malikussaleh. The application is expected to contribute positively to plagiarism prevention and detection in academic settings, enhance trust in students' scientific work quality, and strengthen the university's reputation as a leading educational institution.

2. Literature Review

2.1. Previous Research

Research conducted [7] entitled "Plagiarism Early Detection System Using the Levenshtein Distance Algorithm" based on the research results, the results of document testing indicate that the duration of similarity calculation is influenced by the number of strings and words contained in the document. As shown in Table 4: From Plagiarism Test 1 to Test 2, there is a time increase interval of approximately ±2 seconds, with a word count increase of 10,643. From Test 2 to Test 3, the interval remains ±2 seconds, similar to the previous interval, with a word difference of 9,446. From Test 3 to Test 4, the interval is approximately ±2 to 3 seconds, with a total word difference of 12,212. From Test 4 to Test 5, the interval is ±2 seconds, with a total word difference of 10,105.
Additionally, the document parsing process from .pdf format to .txt format also impacts the calculation time. This can be observed by comparing the system's performance before and after the script was applied. Initially, the computer could calculate quickly when the document was not in .pdf format. However, the system required a little more time after being transformed through the document parsing process. Differences: This research and the one I referred to are the methods used, the objects being studied, and the programming languages employed.
Research conducted [8] entitled "Implementation of the Rabin-Karp algorithm for Plagiarism Detection in Document Files In the Form of Web-Based Text" based on the research results, the results of tests carried out between the original document and documents tested from the test results of 10 text documents with using the Rabin-Karp algorithm, produces a high level of accuracy the largest is 47.58%. Meanwhile, the most minor level of accuracy is 19.28%. Meanwhile, the results of the analysis carried out by the author were from 10 documents, which were tested with a k-gram value of 1, and a percentage result was obtained. The most significant similarity is 57.14%, and the smallest is 28.57%. If the similarity value is 70%, the plagiarism is considerable. Difference: The differences between this research and my research fetch is the method used, then the target object as well
different, the programming language used is also different.
Research conducted [9] entitled "Similarity Detection Application Design Text Documents Using Algorithms Shingling" based on the research results, in this research, the test results show that the application created can detect the similarity of existing text documents through various manipulations, namely scaling (zoom in/out), rotation, cropping (cut part), and manipulating documents. Algorithm Shingling is implemented to detect similarities in the document text. The ease of copying information from one document to another other document is one of the effects of technological progress information. This leads to unwanted plagiarism so that the document detection process can be adequately minimized. Difference: The difference between this research and my research fetch is that the method used, target object, and programming language used are also different.
Research conducted [10] entitled "Detecting Document Similarities in the System Thesis Proposal Registration Information with Rabin-Karp Algorithm Approach" based on the research results, registration document similarity detection system. This thesis proposal can help prevent plagiarism in the academic field by comparing one document with other documents based on the similarity level of that document's contents. Proposal registration document similarity detection system This thesis can produce the similarity percentage in the thesis proposal document. Able to detect document similarities by applying the Rabin-Karp algorithm in creating a similarity detection system between these documents. Differences: Differences between this study and the research are the method used, the object, the objectives are also different, and the programming language used is also different.
Research conducted [11] entitled "Plagiarism Detection in Thesis Documents Based on Level of Similarity With Using the Longest Common Method Subsequences" based on the research results, Based on research that has been carried out, Method Longest Common Subsequence can be used for plagiarism detection by comparison of two or more comparative documents, can test more than one sentence and more than one comparison candidate, as well as results Accuracy testing carried out obtained high accuracy results. This can be an alternative for students majoring in Technology Information at Malang State Polytechnic in conducting tests on research and for the final report. The thesis committee can monitor final reports and theses. Difference: The difference between this research and my research fetch is the method used, the target object, and the programming language used is also different.

2.2. Text Mining

Text mining is exploring and analyzing unstructured textual data using software that identifies concepts, patterns, topics, keywords, and other attributes within the data. The capabilities of text mining are increasingly integrated into AI chatbots and virtual agents, which companies employ to provide automated responses to customers as part of their marketing, sales, and customer service operations [12]. A text mining system comprises several components: text preprocessing, feature selection, and data mining. The purpose of the text preprocessing component is to transform unstructured textual data, such as documents, into structured data that can be stored in a database. Feature selection identifies relevant and impactful words for the classification process. The final component applies data mining techniques to the output of the previous stages[13].

2.2.1. Text Preprocessing

Before performing text clustering, this study's initial step is the data preprocessing phase. This data preprocessing step aims to clean or remove unnecessary text within the documents. This preprocessing phase will go through five stages before the weighting process in the next phase. The five preprocessing steps are illustrated in Figure 1 below [14].

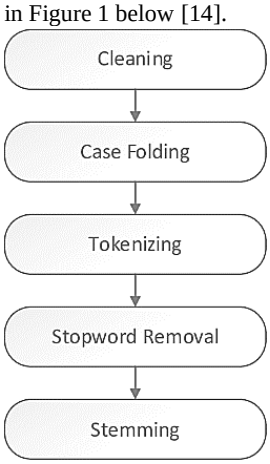

Fig 1. Flowchart of Preprocessing

a. Text Cleaning
   Text cleaning is the process aimed at removing non-alphanumeric symbols or characters from the text.
b. Case Folding
   Case folding is a process that converts all letters in a document from uppercase to lowercase, where only letters from a to z are accepted. The goal is to standardize or normalize the word forms in the text, ensuring that there are no discrepancies in capitalization that could affect consistency and uniformity in subsequent analysis or processing [15].
c. Tokenizing
   Tokenizing or sentence segmentation breaks down a text string in a document, splitting a long text into individual words. By doing this, the document text is transformed into separate words in the form of an array [14]
d. Stopword removal
   The stopword filtering step is the process of removing words that have no significant meaning. Stopwords frequently appear in documents but are not descriptive and often related to a specific theme. In the Indonesian language, stopwords are conjunctions or function words that can be categorized as unimportant [14].
e. Stemming
   The stemming step is a text transformation process that converts words into their root forms and reduces the dimensionality of words within the document. Stemming can also be described as finding the root word of each phrase token by stripping away affixes and returning the word to its base form (stem) [14].

2.2.2. Text Processing

This process involves editing and combining selected sentences from the previous steps to create a more structured and concise summary. The goal is to identify patterns or knowledge from the entire text. The technique applied involves term weighting derived from the preprocessing steps. Each term is assigned a weight based on the chosen weighting scheme. Many applications use weighting methods, such as the multiplication of Term Frequency-Inverse Document Frequency (TF-IDF), commonly referred to as TF-IDF [16].

2.3. Cosine Similarity

Cosine Similarity is a metric used to measure the similarity between two vectors in a multi-dimensional space based on the cosine of the angle between them. This comparison is performed by calculating the cosine of the angle between two vectors, where the cosine of a 0-degree angle is 1, and it decreases for angles greater than 0. In information retrieval, each word or term is considered a separate dimension, and a document is represented as a vector, where the value of each dimension corresponds to the frequency of occurrence of that term within the document [14]. The following is the formula for Cosine Similarity:

$$Cos\ a = \frac{A\ .B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i\ x\ B}{\sqrt{\sum_{i=1}^{n}(A_i)^2\ x\ \sum_{i=1}^{n}(B_i)^2}} \quad \text{..... (1)}$$

Description:
A = Vector A, which will be compared for similarity
B = Vector B, which will be compared for similarity
A B = Dot product between vector A and vector B

|A| = Length (magnitude) of vector A
|B| = Length (magnitude) of vector B
|A||B| = Cross product between |A| and |B|

2.4. Rabin-Karp

The Rabin-Karp algorithm is a string-matching method developed by Michael O. Rabin and Richard M. Karp. This approach works by comparing the hash values of each section of text to find a specific pattern. For a text of length n and a search pattern of length m, the best-case time complexity can be achieved as O(n), while in the worst-case scenario, the time complexity is O(mn). The Rabin-Karp algorithm has several characteristics, including using K-Gram and Hashing. The application of the Rabin-Karp algorithm is carried out after the preprocessing phase. Below are the steps involved in the Rabin-Karp algorithm.

2.4.1. K-Gram

K-Gram is used in text processing and string analysis to refer to a subsequence or substring of length k taken from a string or text. K-Gram is commonly used in various applications such as text search, DNA analysis, pattern recognition, and plagiarism detection. For example, if we have a string "ABCDEFG" and use k = 3, the resulting k-grams would be "ABC", "BCD", "CDE", "DEF", and "EFG". K-gram helps break down text into smaller segments for further analysis [17].

2.4.2. Hashing

Hashing transforms input data of varying sizes into a fixed-size output, known as the hash value or hash code, typically represented as a number or string. The function that performs this transformation is called a hash function. Hashing is used in various computational applications, such as fast searching, data encryption, and data integrity. Below is the hashing formula [18]. In the Rabin-Karp calculation process, each character is first transformed into a decimal number.

$$H(c_1 \dots c_k) = \left(c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \cdots + c_{(k-1)} * b^k + c_k\right) mod\ q \quad (2)$$

Description:
H = Substring
C = ASCII value per word
B = A constant prime number
K = number of characters
Q = A prime modulus number

2.5. TF-IDF

The Term Frequency-Inverse Document Frequency (TF-IDF) method is used to assign weights to the relationship between words (terms) and documents. In the information retrieval architecture, there is a process of term weighting, both locally and globally. Local weighting is based on the term frequency within a document without considering the frequency of the term across other documents. The most common approach in local weighting is term frequency (tf). However, schemes such as binary weighting, augmented normalized tf, logarithmic tf, and alternatives can also be used [19]. The TF-IDF value can be calculated using the following equation:

1. Raw TF (Term Frequency): Raw TF represents the term frequency value based on the number of term occurrences in a document. Term Frequency (TF) refers to the number of times a term appears in a document. It is calculated by counting how often a specific term appears within that document. The higher the term's frequency in the document, the more relevant it is considered for that document.
2. Inverse Document Frequency (IDF). Inverse Document Frequency (IDF) is a calculation that measures how widely a term is distributed across a collection of documents. IDF indicates the availability of a term across all documents. The fewer documents that contain a particular term, the higher the IDF value for that term. This helps to emphasize the importance of terms that appear less frequently across the document collection, as they are likely to be more specific and relevant to a particular document.

$$IDFj = \log(D/dfj) \quad (3)$$

Description:
IDF = Inverse Document Frequency
D = The total Number of documents in the system
dfj = The Number of documents in the collection that contain the term

3. Term Weightening TF-IDF. Term Weighting TF-IDF is the combination of the raw TF (Term Frequency) formula and the IDF (Inverse Document Frequency) formula by multiplying the TF value with the IDF value [19].

$$W_{ij} = tf_{ij}\, x\ log\left(\frac{D}{df_j}\right) \quad (4)$$

Description:

W_{ii} = Weight of the term concerning the document
tf_{ii} = number of occurrences of the term in the document
D = Total Number of documents retrieved in the system
dfj = number of documents in the collection containing the term

2.6. N-Gram

N-gram is a simple model that assigns probabilities to sentences and sequences of words. N-gram is a sequence consisting of N words that make up the structure of a text [20].

a. 1-gram (Unigram): A sequence of words consisting of a single word.
b. 2-gram (Bigram): A sequence of words consisting of two words.
c. 3-gram (Trigram): A sequence of words consisting of three words.
d. N-gram (N-gram): A sequence of words consisting of N words.

The total number of word sequences that can be formed from a sentence can be seen in the following equation:

$$N = Lenght(s) - 1 \quad (5)$$

Description:
N = total number of possible sequences (n-grams)
Length(s) = number of words in the sentence.

3. Research Methods

At this stage, the author designs the system to provide a clear overview and a comprehensive design to the system users.

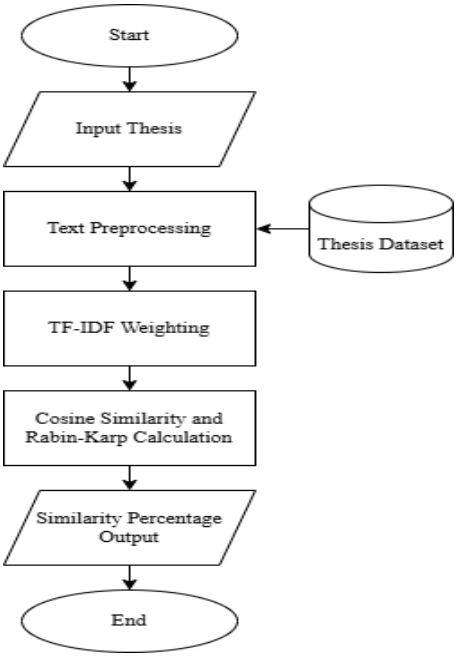

Fig 2. Flowchart System

Explanation:

1. The input thesis stage is the initial step in the thesis detection process. In this system design, the process involves entering the thesis that will be checked for similarity against a repository of documents provided by the system. The system will then read the file that has been uploaded into the detection system.
2. In the text preprocessing stage of this research, the content of the thesis will be processed to detect similarity by converting it into tokens and terms for use in the sentence weighting process. An illustration of the text preprocessing process applied to the thesis can be seen in the image.

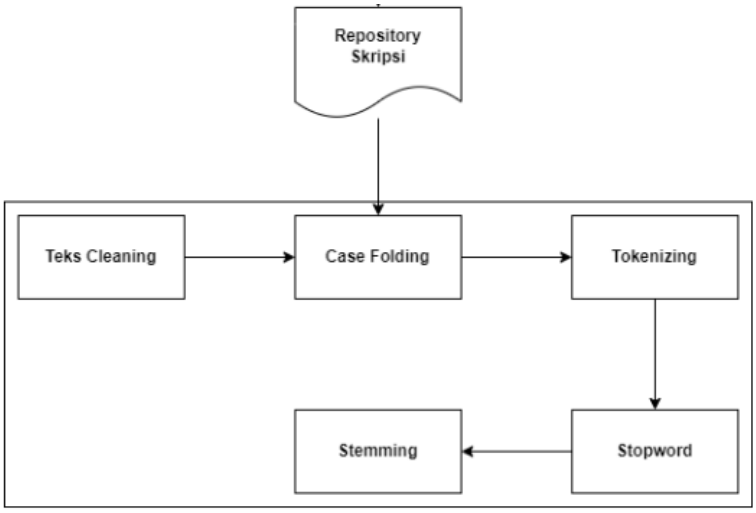

Fig 3. Text Preprocessing Design

3. At this stage, the system will perform word weighting using the TF-IDF method and then calculate the similarity value to obtain the similarity percentage between the input thesis and the repository. This will be achieved using the Cosine Similarity and Rabin-Karp methods.
4. The final stage is the output, which displays the similarity percentage of the thesis that the system has entered.

4. Result and Discussions

4.1. Tools and Dataset

The application developed by the author has a file input mechanism. The allowed file types for input are .docx and .pdf files. Therefore, the tools or libraries used to assist in the parsing and text extraction process, as well as applying the methods in this application, are as follows:

1. Sastrawy.js: This tool helps with the text filtering process for the Indonesian language and stemming, which converts words into their base forms.
2. Officeparser.js: This tool is used to parse and extract content from .docx files.
3. Pdfparse: This library parses and extracts content from PDF files.

4.2. Reference Data

Table 1. Reference data

Judul	Abstrak
Implementation of the Bidirectional Search Algorithm for Finding the Shortest Route Between Dayah Institutions in North Aceh Based on a Web Platform	A Dayah (Islamic boarding school) has strict discipline regarding time; every given moment must be utilized as efficiently as possible, and there is no tolerance for even a slight delay. This is an important issue that needs to be addressed and resolved. One of the problems in the Dayah environment is the limited visiting time, which forces the students' parents to hurry to meet their children. Therefore, implementing a Geographic Information System (GIS) to find the shortest route between Dayah institutions in North Aceh is crucial to help parents arrive on time. This application uses the Bidirectional Search algorithm to find the fastest route to Dayah locations in the Aceh province, specifically in North Aceh. In the search process, the Bidirectional Search algorithm requires the distance data between cities before starting the search. The results from the algorithm show that the route generated by the Bidirectional Search algorithm is more consistent, both in terms of distance and travel time, than the route generated by Google Maps. Keywords: Dayah, Bidirectional Search algorithm, shortest path, GIS (Geographic Information System).
Utilizing WebGIS for Landslide Hazard Mapping in Central Aceh with Scoring and Weighting Method	The frequent occurrence of landslides in Central Aceh has raised concerns among residents and tourists visiting the area, as they feel unsafe due to the lack of information regarding landslide-prone zones. To avoid unforeseen events, a solution is needed in the form of a landslide hazard mapping application. The method used to determine landslide-prone areas involves scoring, which assigns values or scores to each parameter, and weighting, which applies Weight to each scored parameter for each sub-district in Central Aceh. The aim is to categorize the areas into four levels of landslide vulnerability: very high, high, moderate, and low. The results of this study for the 11 sub-districts in Central Aceh show that 18.18% of the areas are classified as very high risk, 27.27% as high risk, 18.18% as moderate risk, and 36.37% as low risk. Utilising WebGIS, the mapping results produce an informative system for the local community and tourists. Keywords: WebGIS, scoring, weighting, landslides.
Face Expression Detection and Recognition using Haralick and Haar Features for Real-time Face Detection	Face expression detection and recognition is a challenging task. Real-time face tracking is complicated due to the limited nature and location of where it occurs. Face recognition is a key step in face recognition systems. Face detection refers to processing a specific image to

	determine the presence of a human face, its position, and size, with the accuracy of the position directly affecting the face detection results. Face recognition methods primarily rely on geometric feature-based, skin colour-based, and statistical theory-based methods. Due to the rapid advancement in digital image technology, the development of artificial intelligence for real-time face expression detection and recognition is necessary. In this context, the researcher is interested in exploring the extraction of Haralick and Haar features for real-time face expression detection and recognition using the Haar Cascade modelling for classification. The implementation results from testing with Haralick features show that for the happy expression, the percentage is 94.429%; for the sad expression, the percentage is 38.777%; and for the angry expression, the percentage is 49.3777%. Meanwhile, testing with Haar features resulted in a happy expression percentage of 78.329%, a sad expression percentage of 36.292%, and an angry expression percentage of 39.517%. Keywords: Haralick features, Haar features, Haarcascade, face detection, face expression.

4.3. Test Text Data

Table 2. Test Text Data

Judul	Abstrak
Implementation of the Bidirectional Search Algorithm for Finding the Shortest Route Between Dayahs in North Aceh Using a Web-Based System	Dayah is disciplined in managing time; every allocated moment must be utilized optimally, as slight delays are not tolerated. This is a significant issue that needs to be addressed and resolved. One of the problems in dayah areas is the very short visiting hours, which forces students' parents to rush to meet their children. Therefore, implementing a Geographic Information System (GIS) to find the shortest route between dayahs in North Aceh is necessary to help guardians arrive on time. This application applies a search method using the Bidirectional Search Algorithm to determine the fastest route to dayah locations in the Aceh province, specifically in North Aceh. In the search process, the Bidirectional Search Algorithm requires distance data between each city before starting the route search process. The results of the algorithm's process show that the routes produced by the Bidirectional Search Algorithm are more consistent in terms of both distance and travel time compared to those generated by Google Maps. Keywords: dayah, bidirectional search algorithm, shortest route, GIS (Geographic Information System).

4.4. Preprocessing Stage

The dataset is sourced from the Universitas Malikussaleh student repository, filtered to include only published undergraduate theses from Informatics Engineering students between 2017 and 2022. A total of 116 theses meet this filtering criterion; however, to explain the methods and algorithm mechanisms, 1 sample will be used for testing, along with three reference datasets to prevent excessively lengthy calculations.

4.5. TF-IDF Process

In the Rabin-Karp algorithm, n-gram detects potential matches between the search text (pattern) and the main text. This occurs during the pattern-matching stage of the algorithm.

4.6. N-Gram Formation Process

In the Rabin-Karp algorithm, n-gram detects potential matches between the search text (pattern) and the main text. This occurs during the pattern-matching stage of the algorithm.

4.7. Application of the Rabin-Karp Method

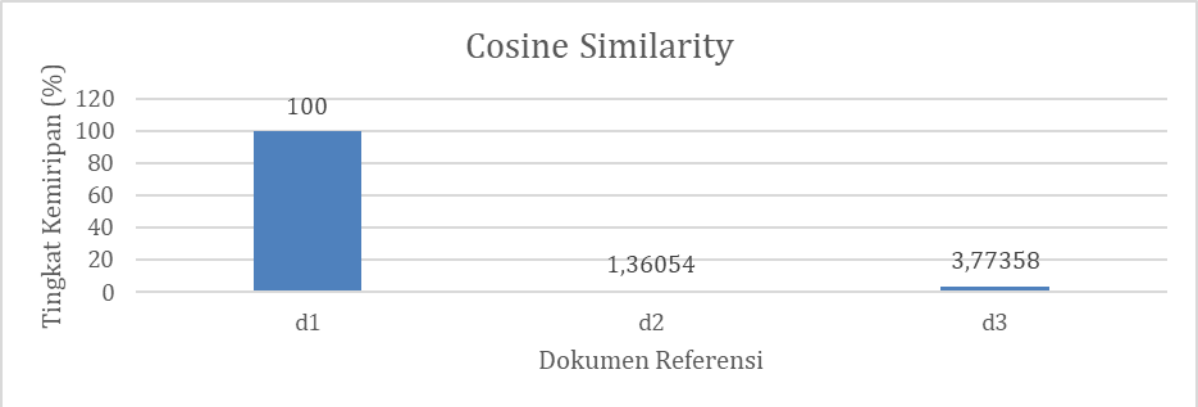

Fig 4. Title similarity results with Cosine Similarity

The graph above shows the results of applying the Cosine Similarity algorithm to the input document titles compared to the titles of 3 reference documents in the dataset. There is a significant difference between the three documents. Document 1 (D1) is very similar to the input document, while the other two do not.

The Cosine Similarity measures the cosine angle between two vectors in vector space. If two vectors point in the same direction, the cosine of the angle is 1, meaning the vectors are very similar. If two vectors are orthogonal to each other, the cosine of the angle is 0, meaning there is no similarity. In this application, the vectors used result from the TF-IDF calculation. The Cosine Similarity calculation to find the similarity between titles is as follows:

$$\text{Cosine D1} = \frac{Q * D1}{|Q| * |D1|} = \frac{0{,}0958}{|0{,}30952| * |0{,}30952|} = \frac{0{,}0958}{0{,}0958} * 100 = 100\%$$

$$\text{Cosine D2} = \frac{Q * D2}{|Q| * |D2|} = \frac{0{,}30952}{|0{,}30952| * |0{,}42646|} = \frac{0}{0{,}132} * 100 = 0\%$$

$$\text{Cosine D3} = \frac{Q * D3}{|Q| * |D3|} = \frac{0{,}007}{|0{,}30952| * |0{,}38492|} = \frac{0{,}007}{0{,}11914} * 100 = 5{,}867\%$$

Description:
D1, D2, …, Dn = Represent the reference texts
Q = Represents the term or word in the test text

The TF-IDF calculation followed by the application of the Cosine Similarity method above shows that the similarity level between the testing document and the first training document (D1) is 100%, which means they are detected as identical documents or plagiarized. Then, the similarity between the testing document and the second training document (D2) is 0%, indicating no similarity between them, or it is not classified as plagiarism. Finally, the similarity with the third document (D3) is 5.86%, indicating a low plagiarism level between the testing and training documents. The results of applying the Cosine Similarity algorithm to the input document for the abstract can be seen in the image below.

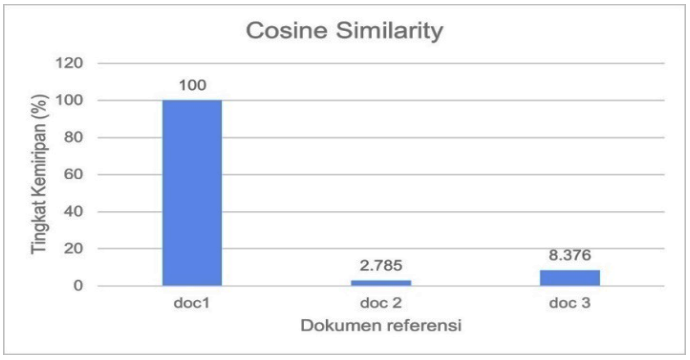

Fig 5. Abstract similarity results with Cosine Similarity

The graph above illustrates the results of applying Cosine Similarity to the input document for the abstract. It shows that the first document is significantly more similar than the other two papers.

4.8. Application of the Cosine Similarity Method

Data underwent preprocessing after the training and testing, and n-grams were formed. The next step was the hash process using the polynomial rolling hash technique on both datasets, which generates decimal values. These decimal values are then used as instruments to perform matching to determine document similarity. Once each n-gram has been hashed, the document similarity is calculated by comparing the hash values of each n-gram using the Dice Coefficient Similarity method. Below are the results of applying the Rabin-Karp algorithm to the title input document.

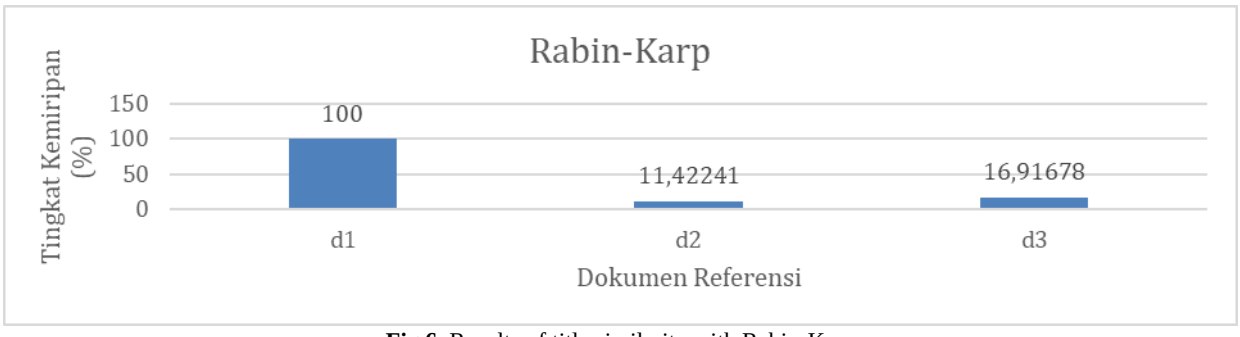

Fig 6. Results of title similarity with Rabin-Karp

The graph above illustrates the results of applying the Rabin-Karp algorithm to the input document title compared to the titles of three reference documents in the dataset. A significant difference is evident among the three papers. Document one (D1) demonstrates a very high similarity to the input document, while the other two documents show no such similarity. The results of applying the Rabin-Karp algorithm to the input document for abstracts can be seen in the graph below.

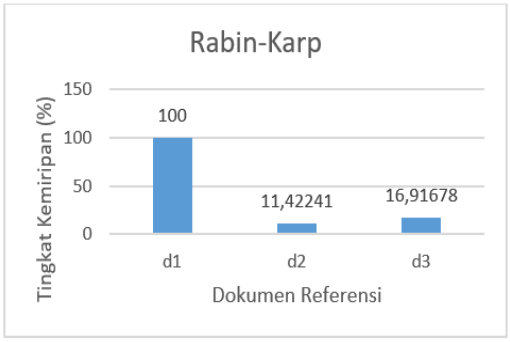

Fig 7. Abstract similarity results with Rabin-Karp

The graph above illustrates the results of applying the Rabin-Karp algorithm to the input document for abstracts. Document one (D1) is significantly more similar than the other two papers.

4.9. Application Display Results

Implementation of student thesis plagiarism detection application interface. This application only has one page with a menu for uploading abstract documents, which can be changed to PDF and MS Word-type files. Then, there is a results section, a tab panel model for displaying plagiarism results from applying the Cosine Similarity and Rabin-Karp methods.

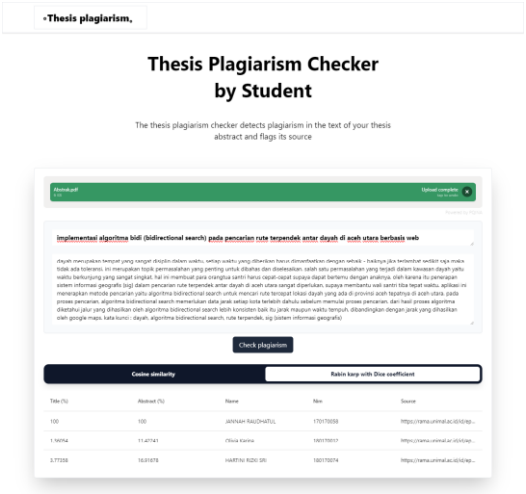

Fig 8. Application Interface

The system testing process used in this research is using the black box testing method. The results of system testing can be seen in the following steps:

1. Upload the document to be detected in PDF and docx format. Then, the system will display the abstract and title of the input document.
2. Check plagiarism by pressing the check plagiarism button on the display page after init. The system will display the plagiarism results with Cosine Similarity and Rabin-Karp.

5. Conclusion

The conclusions obtained in this research are as follows:

1. Developing a thesis plagiarism detection application using the Cosine Similarity and Rabin-Karp algorithms with the JavaScript programming language has increased the accuracy and efficiency of detection. This application allows users to upload documents, evaluate text before detection, and measure the level of plagiarism against existing thesis datasets.
2. Research shows that the Cosine Similarity method effectively detects overall title similarity, with a similarity level reaching 100% for identical documents and 5.86% in detecting light plagiarism. The abstract similarity for the first document is 100%, the second is 2.78%, and the third is 8.37%. On the other hand, the Rabin-Karp algorithm also recorded 100% similarity for identical documents and 11.42% and 16.92% for the other two papers. The abstract similarity for the first document is 100%, the second is 11.42%, and the third is 16.81%, indicating its ability to detect similarities in specific phrases or sentences.

References

[1] T. A. Prismadana, “Aplikasi Ruang Tugas Dengan Deteksi Kemiripan Teks Pada Dokumen Tugas Menggunakan Cosine Similarity,” vol. 15, no. 1, pp. 31–37, 2023, doi: https://doi.org/10.33795/jtim.v15i1.4405.

[2] L. Hermawan and M. B. Ismiati, “Aplikasi Pengecekan Dokumen Digital Tugas Mahasiswa Berbasis Website,” *J. Buana Inform.*, vol. 11, no. 2, pp. 94–103, 2020, doi: 10.24002/jbi.v11i2.3706.

[3] M. Ayunda and N. Aisya, "Fenomena plagiarisme akademik di era digital The phenomenon of academic plagiarism in the digital age," vol. 1, no. 2, pp. 16–33, 2021, doi: https://doi.org/10.48078/publetters.v1i2.23.
[4] N. Nurdin, R. Rizal, and R. Rizwan, "Pendeteksian Dokumen Plagiarisme dengan Menggunakan Metode Weight Tree," *Telematika*, vol. 12, no. 1, p. 31, 2019, doi: 10.35671/telematika.v12i1.775.
[5] R. A. Salim, M. R. D. Septian, S. Suhartini, D. Anggraini, and Q. Qomariyah, "Aplikasi Pendeteksi Kesamaan Dokumen Dengan Menggunakan Algoritma Jarak Jaro Winkler Dan Levenshtein," *Sebatik*, vol. 25, no. 1, pp. 35–41, 2021, doi: 10.46984/sebatik.v25i1.1309.
[6] I. Muhammad, "Pengaruh Perkuliahan Daring Terhadap Kemandirian Belajar Mahasiswa Prodi Pendidikan Matematika Universitas Malikussaleh," *J. Ilm. Pendidik. Mat. Al Qalasadi*, vol. 4, no. 1, pp. 24–30, 2020, doi: 10.32505/qalasadi.v4i1.1567.
[7] M. H. Febiawan, A. Setiawan, and A. Primadewi, "Sistem Pendeteksi Dini Plagiarisme Menggunakan Algoritma Levenshtein Distance," *J. Komtika (Komputasi dan Inform.*, vol. 3, no. 1, pp. 18–27, 2020, doi: 10.31603/komtika.v3i1.3464.
[8] A. Aldian and M. Mubarak, "Implementasi Algoritma Rabin-Karp Untuk Pendeteksian Plagiarisme Pada File Dokumen Berupa Text Berbasis Web," *J. Inf. Syst. Res.*, vol. 3, no. 3, pp. 150–154, 2022, doi: 10.47065/josh.v3i3.1404.
[9] I. S. Simanullang, "Perancangan Aplikasi Deteksi Kemiripan Dokumen Teks Menggunakan Algoritma Shingling," *J. Sist. Komput. dan Inform. Hal*, vol. 2, no. 1, pp. 36–41, 2020, doi: 10.30865/json.v2i1.2451.
[10] P. P. Putra, Afriansyah, and M. Syaifullah, "Pendeteksi Kesamaan Dokumen Pada Sistem Informasi Pendaftaran Proposal Skripsi Dengan Pendekatan Algoritma Rabin-Karp," ペインクリニック学会治療指針２, vol. 2, no. 2, pp. 1–13, 2019.
[11] A. E. Naiman, E. Farber, and Y. Stein, "Cyclic longest common subsequence," *Discret. Math. Algorithms Appl.*, vol. 15, no. 4, pp. 217–226, 2023, doi: 10.1142/S1793830922501038.
[12] Runimeirati, Abdul Muis, and Figur Muhammad, "Pelatihan Text Mining Menggunakan Bahasa Pemrograman Python," *Abdimas Langkanae*, vol. 3, no. 1, pp. 36–46, 2023, doi: 10.53769/abdimas.3.1.2023.83.
[13] R. Tjut Adek, R. Kesuma Dinata, and A. Ditha, "Online Newspaper Clustering in Aceh using the Agglomerative Hierarchical Clustering Method," *Int. J. Eng. Sci. Inf. Technol.*, vol. 2, no. 1, pp. 70–75, 2021, doi: 10.52088/ijesty.v2i1.206.
[14] I. Mawanta, T. S. Gunawan, and W. Wanayumini, "Uji Kemiripan Kalimat Judul Tugas Akhir dengan Metode Cosine Similarity dan Pembobotan TF-IDF," *J. Media Inform. Budidarma*, vol. 5, no. 2, p. 726, 2021, doi: 10.30865/mib.v5i2.2935.
[15] S. Pawestri and Y. Suyanto, "Analisis Perbandingan Metode Similarity untuk Kemiripan Dokumen Bahasa Indonesia pada Deteksi Kemiripan Teks Bahasa Indonesia," *J. Media Inform. Budidarma*, vol. 8, no. 3, p. 1440, 2024, doi: 10.30865/mib.v8i3.7648.
[16] R. P. Pratama, M. Faisal, and A. Hanani, "Deteksi Plagiarisme pada Dokumen Jurnal Menggunakan Metode Cosine Similarity," *SMARTICS J.*, vol. 5, no. 1, pp. 22–26, 2019, doi: 10.21067/smartics.v5i1.2848.
[17] S. L. B. Ginting, Y. R. Ginting, S. Sutono, and W. A. Sirait, "Aplikasi Deteksi Kemiripan Kata Menggunakan Algoritma Rabin-Karp," *J. Teknol. dan Inf.*, vol. 12, no. 2, pp. 162–175, 2022, doi: 10.34010/jati.v12i2.6947.
[18] A. Filcha and M. Hayaty, "Implementasi Algoritma Rabin-Karp untuk Pendeteksi Plagiarisme pada Dokumen Tugas Mahasiswa," *JUITA J. Inform.*, vol. 7, no. 1, p. 25, 2019, doi: 10.30595/juita.v7i1.4063.
[19] R. Rismanto, Y. Yunhasnawa, and R. A. Bhakti, "Penerapan Metode Cosine Similarity Dalam Aplikasi Chatbot Layanan Wisata Di Wilayah Malang," 2019. [Online]. Available: www.malangkota.go.id,
[20] F. Pramono, Didi Rosiyadi, and Windu Gata, "Integrasi N-gram, Information Gain, Particle Swarm Optimation di Naïve Bayes untuk Optimasi Sentimen Google Classroom," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 3, no. 3, pp. 383–388, 2019, doi: 10.29207/resti.v3i3.1119.