IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.18, No.4, October 2024, pp. 441~452
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.100308

The Application of the Rabin-Karp Algorithm with the Synonym Recognition Approach to Detect Plagiarism in Student Assignments

Irma Handayani [1*], Anita Fira Waluyo[2]
[1,2] University of Technology Yogyakarta, Yogyakarta, Indonesia
e-mail: [*1]**irma.handayani@staff.uty.ac.id**, [2] anitafira@uty.ac.id

Abstrak

Kemajuan teknologi yang pesat telah mempermudah segala hal, termasuk dalam bidang pendidikan. Namun, kecanggihan ini juga mengakibatkan penyalahgunaan teknologi, khususnya dalam hal duplikasi atau plagiarisme. Masalah ini tidak hanya terjadi pada tugas esai tetapi juga kode program. Untuk mengatasi hal ini, telah dilakukan penelitian untuk mendeteksi plagiarisme pada tugas mahasiswa dengan menggunakan metode Rabin-Karp dan pendekatan Synonym Recognition. Penelitian ini menemukan bahwa tingkat kemiripan terkecil adalah 20%, sedangkan yang terbesar adalah 76%. Penelitian ini bertujuan untuk memberikan solusi yang cepat dan akurat untuk mencegah maraknya aktivitas plagiarisme di bidang akademik.

Kata kunci— *Algoritma rabin karp, plagiarisme, kesamaan, tugas mahasiswa*

Abstract

Rapid technological advances have made everything easier, including in the field of education. However, this sophistication has also resulted in misuse of technology, especially in terms of duplication or plagiarism. This problem does not only occur in essay assignments but also in program code. To overcome this, research has been conducted to detect plagiarism in student assignments using the Rabin-Karp method and the Synonym Recognition approach. This study found that the smallest similarity level was 20%, while the largest was 76%. This study aims to provide a fast and accurate solution to prevent the rampant activity of plagiarism in the academic field.

Keywords— *Rabin karp algorithm, plagiarism, similarity, student assignments*

1. INTRODUCTION

Along with the very rapid development of technology today, especially in the fields of technology and the internet, many positive impacts have been reaped from technological progress. This progress has led to swift development in the digital realm [1]. However, it has also ushered in several inevitable negative impacts, one of the most significant being plagiarism, a major issue in the academic world [2]. In today's digital era, easy access to information via the internet has accelerated the exchange of knowledge in various fields, including education. Yet, this ease also presents significant challenges, such as an increase in plagiarism cases among students, not only in essay assignments but also in program code [3]. Plagiarism is an unethical practice involving the use of someone else’s work as one’s own without appropriate credit, often facilitated by the ease of duplicating text and materials provided by sophisticated technology [4]. Previous research

Received September 30th,2024; Revised October 30th, 2024; Accepted October 31th, 2023

has identified and developed various methods for detecting plagiarism, from manual to algorithmic approaches [5]. Algorithms such as Winnowing and Jaro-Winkler have been widely used to detect text similarities, but these methods often fall short in recognizing more complex forms of plagiarism that involve synonyms or substantially changed text structures [6]. Despite advances in plagiarism detection technology, there remains a significant gap in the ability to identify plagiarism involving synonyms and text restructuring [6]. This study aims to address this gap by using a modified Rabin-Karp algorithm with a synonym recognition approach, which has not been widely explored in previous literature [7]. The main contribution of this study is the development of a method that can efficiently improve the accuracy of plagiarism detection, speed up the checking process, and ultimately improve academic integrity [8]. The purpose of this study is to develop and validate a plagiarism detection system that can accurately identify text similarities, including the use of synonyms, in students' academic documents [9]. This system is expected to be used by educational institutions to proactively reduce plagiarism incidents and support educational efforts in teaching good academic ethics to students [10]. The modified Rabin Karp algorithm has been shown to select the best K-Gram values, showing highest performance with k = 3 achieving interpretations of 1-14% (Little degree of similarity) and 15-50% (Medium level of similarity) which significantly enhances the method's efficacy [9]. Moreover, when combined with the Jaro-Winkler algorithm in tests, the accuracy of text similarity identification has been greatly increased, as evidenced by a study on Indonesian text [10]. Further tests have shown that the system's average document similarity test result was 24.13, with an accuracy rate of 94.7% [11]. After analyzing ten documents with a k-gram 1 value, the Rabin-Karp algorithm was implemented to detect plagiarism in web-based text document files, finding the greatest percentage of similarity at 57.14%, while the lowest was 28.57% [12]. The Indonesian text document similarity detection system uses both confix-stripping and Rabin-Karp algorithms [13]. Word similarity detection using the Rabin Karp algorithm, based on the findings of the tests that have been performed using 10 abstract document data in the thesis, it produces an accuracy rate of 95.08% and the time to process the tested documents is an average of 11.8 seconds [14]. One benefit of the Rabin-Karp algorithm is its ability to search long pattern strings. The text preprocessing stages of the Rabin-Karp algorithm in this system include case folding, tokenizing, filtering or stopword removal, and stemming. Meanwhile, Synonym Recognition is the detection of plagiarism through a synonym approach [15].

2. METHODS

2.1. System Description

This study faces several important limitations to consider. The dataset used is limited to only assignment documents from informatics students at Universitas Teknologi Yogyakarta, which may affect the generalizability of the results due to the lack of variety of academic documents from other disciplines or institutions. This study relies heavily on the Rabin-Karp algorithm to detect text similarity, but this algorithm may not be completely effective in identifying all types of plagiarism, especially those involving heavily altered or rearranged text. In addition, implementing the Rabin-Karp algorithm with a synonym recognition approach can add complexity and require longer processing times, especially for very long documents or large datasets. The variability in the performance of the Rabin-Karp algorithm, which can change depending on the k-gram value used, suggests that there is a trade-off between speed and accuracy that must be carefully managed, with smaller k-gram values tending to produce better accuracy. The system is designed to detect plagiarism in student assignments using the Rabin Karp algorithm with the Synonym Recognition approach. The process applied to the system is divided into 4 stages, namely Preprocessing, Synonym Recognition, Rabin Karp modeling dan Dice Similarity Coefficient. Figure 1 below illustrates the system flow in further detail. [16].

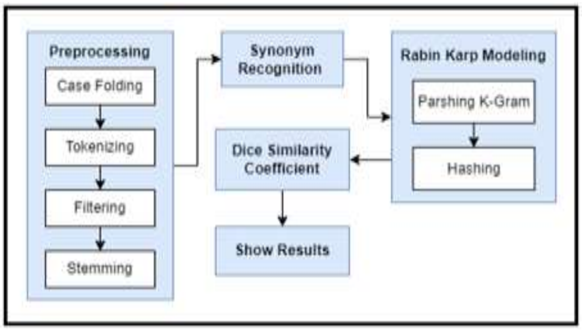


Figure 1. System Flow

2.2. Data Collection

This stage is the first step to starting research. Data obtained by collecting primary data, namely data on assignment documents from Informatics students, UTY. The dataset used is text data and not a file that can be uploaded.

2.3. Data Processing

The dataset used is an assignment document for students majoring in Informatics, UTY. Examples of data used in this research are 10 student assignment documents and 1 document is used as a dataset which is then used to compare and look for words that contain similarities, after which the value or level of similarity is calculated.

2.4. Preprocessing

The first step in the text mining process is preprocessing. It is utilized to convert unstructured textual data into organized textual data [17]. Preprocessing is the process of taking a document's unnecessary text and removing it from the document so that it won't cause noise during the next step. Three separate steps comprise preprocessing: tokenizing, filtering, and case folding [18]. Case Folding is the stage of changing capital letters to lower case [19]. Only letters a to z are accepted. Characters other than letters are omitted and are considered delimiters [20]. Tokenizing is the process of breaking down a document into groups of words [21]. Furthermore, tokenization eliminated all numbers, symbols, and punctuation because they had no distinct score and had nothing to do with the string that needed to be processed [22]. Words that have no meaning are eliminated during the filtering process. Stopwords are a common term for the meaningless words. Stopwords include phrases like "juga," "dan," "untuk," and "adalah." [23]. These stopwords must be removed because they significantly reduce the text similarity percentage and affect the text similarity method's accuracy when conjunctions are used frequently in sentences [24]. In a text document, the stemming process helps to eliminate affixes from words so that the word taken is the root word. It was carried out in order to make the next procedure easier. Affixes include things like "mem," "kan," "ber," "pun," and "mem-an" [25]. The acquired root words were employed as tokens in every text passage to improve syntactic matching precision and efficiency. For example, the words "belajar" and "mengajar" were found in document 1 and document 2, respectively. The word "belajar dan mengajar" became "ajar" following the stemming process since "ajar" is the root of both "belajar" and "mengajar." [22].

2.5. Synonym Recognition

One technique for identifying instances of text plagiarism using the synonym approach is synonym recognition. [15]. To say that the degree of similarity is more accurate in this instance, words that contain synonyms are found when comparing two documents. The Synonym Recognition stage is carried out during the preprocessing stage. Figure 2 below shows the process of recognizing synonyms.

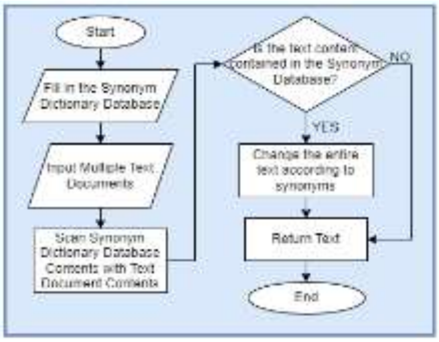

Start
Fill in the Synonym Dictionary Database
Input Multiple Text Documents
Scan Synonym Dictionary Database Contents with Text Document Contents
Is the text content contained in the Synonym Database?
NO
YES
Change the entire text according to synonyms
Return Text
End

Figure 2.Synonym Recognition Process

2.6. Rabin Karp Modeling

This algorithm looks for possible patterns in the input text using the hash function. The average and best-case running times for the text length n and pattern p of mutual length m are O (n+m) in space O (p), and the worst-case time is O (nm) in space O(m) [26]. The Rabin-Karp algorithm employs hashing and K-Gram, among other features. The preprocessing phase is completed before implementing the Rabin-Karp algorithm [11]. The pseudocode for the Rabin-Karp algorithm in Table 1 is as follows.

Table 1. Pseudocode Rabin-Karp Algorithm

```
Function
Rabin-Karp (input s: s [1..m], text: string [1..n] boolean
{Searching string s in text strings with the Rabin-Karp
algorithm}
Declaration
i : integer
found = boolean
Algorithm
found ← false
hs ← hash (s[1..m])
hsub ← hash (text [1..i+m-1])
  for i ← 1 to n do
        if hsub = hs then
        if text [i..i+m-1] = s then
        found ← true
     else
        hsub hash (text [i+1..i+m])
end for
return found
```

The phases of the Rabin-Karp algorithm are as follows:

a) K-Gram

A k-gram is a long sequence of tokens of length k. This K-Gram method takes pieces of letter characters with k values from a text which are continuously read from the beginning of the source text to the end of the source text [11]. An example of a K-Gram with a value of k = 3 can be seen in Table 2.

Table 2. K-Gram Example

Sentence	Rabin Karp Algorithm
Preprocessing	rabinkarpalgorithm
K-Gram	{rab} {ink} {arp}{alg} {ori} {thm}

b) Hash

Hashing is a way to convert string characters into integers called hash values. The process of converting it into a hash value uses the rolling hash function. The rolling hash equation can be seen in equation 1 [27].

$$H(C_1 \ldots . C_k) = (C_1 * b^{k-1}) + \ (C_2 * b^{k-2}) + \cdots + \left(C_{(k-1)} * \ b^k + C_k\right) \, mod \, q \qquad (1)$$

Information:

h: substring
c: ASCII value per character
b: constant prime number
k: many characters
q: modulo prime number

The following is an example of a rolling hash number for a substring with a K-Gram value of 4 seen in Table 3.

Table 3. Example of Hash Calculation

Attribute	**Array value**

Rolling Hash 1	[0] => so
	m = 109, a = 97, k = 107, a=97, basis=11, mod = 10007
	H=c_m*b^(k-1)+c_a*b^(k-2)+c_k*b^(k-3)+c_a*b^(k-4)
	H=109*11^3+97*11^2+107*11^1+97*11^0
	H=145079+11737+1177+97
	H=158090 Mod 10007
	H= 7985

Rolling Hash 2	[1] => will
	a = 97, k = 107, a = 97, n = 110, basis = 11, mod = 10007
	H=c_a*b^(k-1)+c_k*b^(k-2)+c_a*b^(k-3)+c_n*b^(k-4)
	H=97*11^3+107*11^2+97*11^1+110*11^0
	H=129107+12947+1067+110
	H=143231 Mod 10007
	H= 3133

c) Dice's Coefficient Similarity

Dice's Similarity Coefficient is an algorithm used to calculate the level of similarity between two objects by multiplying by 2 the number of intersection values between the document and the query, then dividing it by the number of document values and the query value [28]. The application of Dice's Similarity Coefficient in calculating similarity values using the k-gram approach is in equation 2 as follows.

$$S = \frac{2 \ x \ C}{(A+B)} \text{ x } 100 \qquad (2)$$

Information:

S: similarity value
A dan B: the sum of the sets of kgrams in text 1 and text 2
C: the number of similar k-grams from the texts being compared

To determine the type of plagiarism between the documents tested, there are 5 types of percentage assessment, as follows:

1) 0%: A test result of 0% indicates that the content and overall sentence structure of the two documents are entirely different.
2) < 15%: A test result of 15% indicates a low degree of similarity between the two documents.
3) 15–50%: A score of 15–50% indicates that there is moderate plagiarism in the document.
4) >50%: A test result of more than 50% indicates that the document may be on the verge of being plagiarized.
5) 100%: Because the content is exactly the same throughout, a test result of 100% indicates that the document is plagiarized.

From the assessment percentage above, the tolerance level for pelagiarism is 50% or means indicating that the document includes moderate level plagiarism.

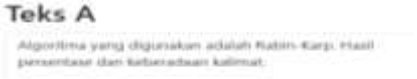
3. RESULTS AND DISCUSSION

3.1 Process of Compiling Student Assignment Data

The data used to detect plagiarism is 10 student assignment data with k-gram values = 4 and 5 and prime numbers = 7.

3.2 Calculation Example

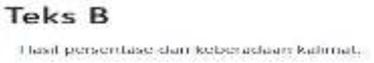
There are two documents entered into the system, namely training data and test data or Text A and Text B, as seen in Figure 3 and Figure 4 below.



Teks A

Algoritma yang digunakan adalah Rabin-Karp. Hasil persentase dan keberadaan kalimat.

Figure 3 Example of Text Data A



Teks B

Hasil persentase dan keberadaan kalimat.

Figure 4 Example of Text Data B

3.3 Preprocessing Stage

After entering the Text A and Text B documents, the next step is preprocessing, which consists of case folding or changing uppercase letters to lowercase, tokenizing or the process of separating words based on word order, and the filtering process. The results of the preprocessing stage are shown in table 4 below.

Table 4. Hash Calculation Example

Document Name	Document Contents
Doc Text A	algoritmarabinkarphasilpersentasekeberadaankalimat
Doc Text B	hasilpersentasekeberadaankalimat

3.4 Parsing K-Gram

In parsing k-grams, use the example of K-gram 5 in each text A document as shown in table 5 and text B document in table 6 below. This means that each sentence will take pieces of 5 or 6 letter characters from a text.

Table 5. Parsing K-gram 5 Document Text A

No	Hasil	No	Hasil	No	Hasil
1	algor	17	rphas	33	ekebe
2	lgori	18	phasi	34	keber
3	gorit	19	hasil	35	ebara
4	oritm	20	asilp	36	berad
5	ritma	21	silpe	37	erada
6	itmar	22	ilper	38	radaa
7	tmara	23	lpers	39	adaan
8	marab	24	perse	40	daank
9	arabi	25	ersen	41	aanka
10	rabin	26	rsent	42	ankal
11	abink	27	senta	43	nkali
12	binka	28	entas	44	kalim
13	inkar	29	ntase	45	alima
14	nkarp	30	tasek	46	limat
15	karph	31	aseke		
16	arpha	32	sekeb		

Table 6. Parsing K-gram 5 Document Text B

No	Hasil	No	Hasil	No	Hasil
1	hasil	13	ersen	25	erada
2	asilb	14	rsent	26	radaa
3	silbe	15	senta	27	adaan
4	ilber	16	entas	28	daank
5	lberu	17	ntase	29	aanka
6	berup	18	tasek	30	ankal
7	erupa	19	aseke	31	nkali
8	rupap	20	sekeb	32	kalim
9	upape	21	ekebe	33	alima
10	paper	22	keber	34	limat
11	apers	23	ebara		
12	perse	24	berad		

3.5 String Matching Stages

a) Rolling Hash

Examples of words that will be used are the words “algor” and “lgori”. The ASCII graded decimal results are shown in table 7 below, with the specified base value being 10.

Table 7. ASCII Value

Char	Dec	Char	Dec
A	97	l	108
L	108	g	103
G	103	o	111
O	111	r	114
R	114	i	105

Hashing calculations using equation 1 are as follows.

$$H_{(algor)} = ascii(a) * 10^{(4)} + ascii(l) * 10^{(3)} + ascii(g) * 10^{(2)} + ascii(o) * 10^{(1)} + ascii(r) * 10^{(0)}$$

$$H_{(algor)} = 97 * 10000 + 108 * 1000 + 103 * 100 + 111 * 10 + 144 * 1 = 1089542$$

The value of hashing in the substring “algor” is 1089524.

$$H_{(lgori)} = \left(H_{(algor)} - ascii(a) * 10^{(4)}\right) * 10 + ascii(i)$$

$$H_{(lgori)} = (1089524 - 97 * 10000) * 10 + 105 = 1195345$$

The value of rolling hashing in the substring “lgori” is 1195345.

b) String Matching with the Rabin-Karp Algorithm

In this process, it will match the hash of text A document and the hash of text B document which are the same. The results look like in table 8 for document Text A and table 9 for document text B.

Table 8. Rolling Hash Result Text A

No	Term	Hash	No	Term	Hash
1	algor	1089524	24	perse	1233651
2	lgori	1195345	25	ersen	1136620
3	gorit	1153566	26	rsent	1266316
4	oritm	1235769	27	senta	1263257
5	ritma	1257787	28	entas	1132685
6	itmar	1177984	29	ntase	1226951
7	tmara	1279937	30	tasek	1269617
8	marab	1199468	31	aseke	1096271
9	arabi	1094785	32	sekeb	1262808
10	rabin	1247960	33	ekebe	1128181
11	abink	1079707	34	keber	1181924
12	binka	1097167	35	ebara	1119337
13	inkar	1171784	36	berad	1093470
14	nkarp	1217952	37	erada	1134797
15	karph	1179624	38	radaa	1248067
16	arpha	1096337	39	adaan	1080780
17	rphas	1263485	40	daank	1107907
18	phasi	1234955	41	aanka	1079167
19	hasil	1149658	42	ankal	1091778
20	asilp	1096692	43	nkali	1217885
21	silpe	1267021	44	kalim	1178959
22	ilper	1170324	45	alima	1089687
23	lpers	1203355	46	limat	1196986

Table 9. Rolling Hash Result Text B

No	Term	Hash	No	Term	Hash
1	hasil	1149658	18	tasek	1269617
2	asilb	1096678	19	aseke	1096271
3	silbe	1266881	20	sekeb	1262808
4	ilber	1168924	21	ekebe	1128181
5	lberu	1189257	22	keber	1181924
6	berup	1093682	23	ebara	1119337
7	erupa	1136917	24	berad	1093470
8	rupap	1269282	25	erada	1134797
9	upape	1292921	26	radaa	1248067
10	paper	1229324	27	adaan	1080780
11	apers	1093355	28	daank	1107907
12	perse	1233651	29	aanka	1099167
13	ersen	1136620	30	ankal	1091778
14	rsent	1266316	31	nkali	1217885
15	senta	1263257	32	kalim	1178959
16	entas	1132685	33	alima	1089687
17	ntase	1226951	34	limat	1196986

It can be seen that the string matching results between documents Text A and Text B contain the same number of hashes of 24.

c) Calculating Similarity

The hash count of the test document $(n_{(uji)})$ = 34

Same hash count (n_f) = 24

$$S = \frac{(n_f)}{n_{(uji)}} X\ 100 = \frac{24}{34} X\ 100 = 70{,}58\%$$

So, the calculation result of the similarity between the test document and the training document is 70.58%.

d) Same Sentence Matching Process

This process takes data from documents Text A and Text B by cutting paragraphs into sentences. Then match the sentences. The results of the process of displaying the same sentences in matching two documents can be seen in table 10 below.

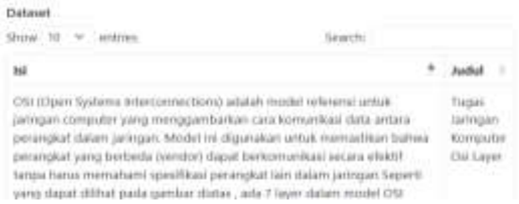
Table 10. Similarity Matching Results

Document Name	Document Contents
Doc Text A	The algorithm used is Rabin-Karp. Results of percentage and presence of sentences
Doc Text B	Percentage results and presence of sentences.

3.6 *Implementation*

a) Input Dataset

The first step in the plagiarism detection system is to enter the dataset that will be used to check plagiarism in student assignments, an example dataset is as shown in Figure 3 below.



Dataset

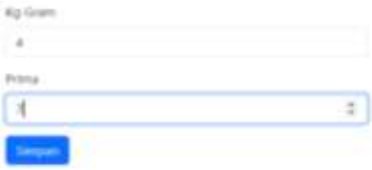
Show 10 entries Search:

Isi	Judul
OSI (Open Systems Interconnections) adalah model referensi untuk jaringan computer yang menggambarkan cara komunikasi data antara perangkat dalam jaringan. Model ini digunakan untuk memastikan bahwa perangkat yang berbeda (vendor) dapat berkomunikasi secara efektif tanpa harus memahami spesifikasi perangkat lain dalam jaringan Seperti yang dapat dilihat pada gambar diatas , ada 7 layer dalam model OSI	Tugas Jaringan Komputer OSI Layer

Figure 5 Example Dataset

b) Determining K-Gram Value

After entering the data set, the second step is to determine the k-gram values and prime numbers, as shown in Figure 4 below.



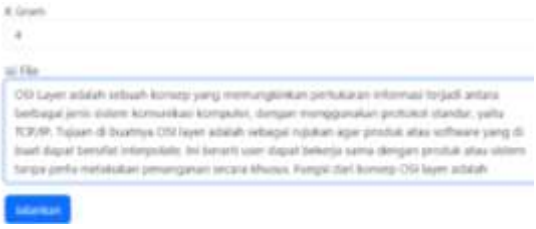
Kg Gram: 4

Prima: 1

Figure 6 K-Gram and Prime Number Settings

c) Student Assignment Data Input

Then in the third step, enter the student assignment data which will be checked for the level of similarity, as shown in Figure 5 below.



K Gram: 4

Isi File: OSI Layer adalah sebuah konsep yang memungkinkan pertukaran informasi terjadi antara berbagai jenis sistem komunikasi komputer, dengan menggunakan protokol standar, yaitu TCP/IP. Tujuan di buatnya OSI layer adalah sebagai rujukan agar produk atau software yang di buat dapat bersifat interoperable. Ini berarti user dapat bekerja sama dengan produk atau sistem tanpa perlu melakukan penanganan secara khusus. Fungsi dari konsep OSI layer adalah

Figure 7 Student Assignment Data Input

d) Similarity Result

The following are the results of the similarity level of the plagiarism detection process using the Rabin Karp Algorithm, as seen in Figure 6.

Hasil Similarity

[illegible]

Nilai Similarity 20%

Figure 8 Similarity Result

The results of the similarity of student assignments from 10 data using k-gram value = 4 can be seen in table 11 below:

Table 11. Similarity Result K-Gram = 4

Document Name	K-Gram	Prime	Similarity Result
Doc. Text 1	4	7	38%
Doc. Text 2	4	7	58%
Doc. Text 3	4	7	49%
Doc. Text 4	4	7	39%
Doc. Text 5	4	7	25%
Doc. Text 6	4	7	20%
Doc. Text 7	4	7	27%
Doc. Text 8	4	7	76%
Doc. Text 9	4	7	56%
Doc. Text 10	4	7	31%

It can be seen from the similarity results above that the lowest similarity value is in text document 6, which is 20% and the highest similarity value is in text document 8, which is 76%.
Furthermore, the results of the similarity of student assignments from 10 data using k-gram value = 5, can be seen in table 12 below.

Tabel 12. Similarity Results K-Gram = 5

Document Name	K-Gram	Prima	Similarity Result
Doc. Text 1	5	7	28%
Doc. Text 2	5	7	39%
Doc. Text 3	5	7	37%
Doc. Text 4	5	7	29%
Doc. Text 5	5	7	19%
Doc. Text 6	5	7	15%
Doc. Text 7	5	7	21%
Doc. Text 8	5	7	72%
Doc. Text 9	5	7	42%
Doc. Text 10	5	7	24%

It can be seen from the similarity results above that the lowest similarity value is in text document 6, which is 15% and the highest similarity value is in text document 8, which is 72%.

4. CONCLUSIONS

The effective implementation of the Rabin-Karp algorithm with the Synonym Recognition approach in detecting plagiarism in student assignments at Universitas Teknologi Yogyakarta. Through the application of this method, plagiarism checking on student documents can be done quickly and accurately, which shows a significant improvement compared to traditional manual checking methods. Key findings include the ability of the system to determine similarity in student assignments with varying degrees of similarity efficiently. For example, the use of a k-gram value of 4 produces similarity levels ranging from 20% to 76%, while a k-gram value of 5 produces similarity levels from 15% to 72%. Therefore, it can be concluded that a smaller k-gram value will produce better similarity value accuracy compared to a larger K-gram value. These results underline the importance of the choice of k-gram value in influencing the accuracy of plagiarism detection. Future improvements may include refining the Synonym Recognition feature to capture a wider range of synonymous relationships and exploring the use of more sophisticated algorithms to further reduce processing time and improve system scalability. In conclusion, this study not only reaffirms the capability of the Rabin-Karp algorithm in detecting text similarity, but also paves the way for its further improvement and application in educational settings to enforce academic standards and prevent unethical practices such as plagiarism.

ACKNOWLEDGMENT

Thank you to the Directorate of Research, Technology and Community Service, Directorate General of Higher Education, Research and Technology, Ministry of Education, Culture, Research and Technology for funding the implementation of this research through the Research Grant for the “Penelitian Dosen Pemula” Scheme for Fiscal Year 2023.

REFERENCES

[1] S. Inan Zahida and B. Santoso, “Perlindungan Hak Cipta Terhadap Gambar Yang Telah Diunggah Pada Media Sosial Instagram,” *Jurnal Pembangunan Hukum Indonesia*, vol. 5, no.1, pp. 186-203, 2023.

[2] M. A. Shadiqi, “Memahami dan Mencegah Perilaku Plagiarisme dalam Menulis Karya Ilmiah,” *Buletin Psikologi*, vol. 27, no. 1, pp. 30–42, Jun. 2019, doi: 10.22146/buletinpsikologi.43058.

[3] M. Jiffriya, M. A. Jahan, and R. G. Ragel, “Plagiarism detection tools and techniques: A comprehensive survey,” *Journal of Science-FAS-SEUSL*, vol. 02, no. 02, pp. 47-64, 2021.

[4] Riki, Edy, and Maryanto, “Plagiarism Detection Application Uses Winnowing Algorithm with Synonym Recognition for Indonesian Text Documents,” *Selangor Science & Technology Review (SeSTeR)*, vol. 3, no.1, pp.18-31, 2019.

[5] M. Francisca Abad-García, “Plagiarism and predatory journals: A threat to scientific integrity,” *An Pediatr (Barc)*, 90:57, e1-57.e8, 2019.

[6] A. Filcha and M. Hayaty, “Rabin-Karp Algorithm Implementation to Detect Plagiarism on Student’s Assignment Document,” *JUITA*, vol.VII, no. 1, May 2019.

[7] N. Bansal, “An Elementary Algorithm for Pattern Matching,” *IJCSEC*, vol. 6, no. 1, pp. 1780-1787, 2018.

[8] S. Wulandari, “Perancangan Aplikasi Game Acak Kata Bahasa Inggris Tingkat SD Menggunakan Algoritma Rabin Karp Berbasis Android,” *Jurnal Ilmu Komputer, Teknologi Dan Informasi*, vol. 1, no. 1, pp. 21–26, 2023.

[9] W. Hidayat, E. Utami, and A. Sunyoto, “Selection of the Best K-Gram Value on Modified Rabin-Karp Algorithm,” *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 16, no. 1, pp. 11–22, Jan. 2022, doi: 10.22146/ijccs.63686.

[10] M. A. Yulianto and N. Nurhasanah, “The Hybrid of Jaro-Winkler and Rabin-Karp Algorithm in Detecting Indonesian Text Similarity,” *Jurnal Online Informatika*, vol. 6, no. 1, pp. 88–95, Jun. 2021, doi: 10.15575/join.v6i1.640.

[11] A. Santoso and A. Solichin, “Document Similarity Analysis of Student Thesis using Rabin-Karp Algorithm and Dice Coefficient Similarity,” *Techno.COM*, vol. 22, No. 1, pp. 12-27, Feb. 2023.

[12] Aldian and Mubarak, “Implementasi Algoritma Rabin-Karp Untuk Pendeteksian Plagiarisme Pada File Dokumen Berupa Text Berbasis Web,” *JOSH*, vol. 3, no. 3, pp. 150–154, 2022.

[13] D. D. Sinaga and S. Hansun, “Indonesian text document similarity detection system using rabin-karp and confix-stripping algorithms,” *International Journal of Innovative Computing, Information and Control*, vol. 14, no. 5, pp. 1893–1903, Oct. 2018, doi: 10.24507/ijicic.14.05.1893.

[14] S L B Ginting, Y R Ginting, Sutono, and W A Sirait, “Aplikasi Deteksi Kemiripan Kata Menggunakan Algoritma Rabin-Karp,” *JATI*, vol. 12, no. 2, pp. 162–175, 2022.

[15] N. Prima Putra and S. Sularno, “Penerapan Algoritma Rabin-Karp Dengan Pendekatan Synonym Recognition Sebagai Antisipasi Plagiarisme Pada Penulisan Skripsi,” *Jurnal Teknologi Dan Sistem Informasi Bisnis*, vol. 1, no. 2, pp. 48–58, 2019, doi: 10.47233/jteksis.v1i2.52.

[16] I. Handayani and Ikrimach, “Accuracy Analysis of K-Nearest Neighbor and Naïve Bayes Algorithm in the Diagnosis of Breast Cancer,” *JURNAL INFOTEL*, vol. 12, no. 4, pp. 151–159, Nov. 2020, doi: 10.20895/infotel.v12i4.547.

[17] M. R. Faisal, D. Kartini, and T. H. Saragih, “Belajar Data Science: Text Mining Untuk Pemula I,” *Scripta Cendekia, Banjarbaru, Kalimantan* Selatan, Nov. 2022. [Online]. Available: https://www.researchgate.net/publication/359619425

[18] H. A. Rouf, A. Wijayanto, and A. Aziz, “Deteksi Plagiarisme Skripsi Mahasiswa dengan Metode Single-link Clustering dan Jaro-Winkler Distance,” *PILAR TEKNOLOGI*, vol. 5, no. 1, pp. 26-31, Mar. 2020. [Online]. Available: http://pilar.unmermadiun.ac.id/index.php/pilarteknologi

[19] I. Mawanta, T. S. Gunawan, and W. Wanayumini, “Uji Kemiripan Kalimat Judul Tugas Akhir dengan Metode Cosine Similarity dan Pembobotan TF-IDF,” *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 5, no. 2, p. 726, Apr. 2021, doi: 10.30865/mib.v5i2.2935.

[20] A. Sanjaya and S. D. Sasongko, “Sentences Similarity Test Using Countable Function On Pre-Processing And Cosine In Indonesian,” *Jurnal Ilmiah NERO,* vol. 7, no. 2, pp. 95-104, 2022.

[21] A. Sanjaya, A. Bagus Setiawan, U. Mahdiyah, I. Nur Farida, A. Risky Prasetyo, and U. Nusantara PGRI Kediri, “PENGUKURAN KEMIRIPAN MAKNA MENGGUNAKAN COSINE SIMILARITY DAN BASIS DATA SINONIM KATA,” *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 10, no. 4, pp. 747–752, 2023, doi: 10.25126/jtiik.2023106864.

[22] S. Purwaningrum, A. Susanto, and A. Kristiningsih, “Synonym Recognition Influence in Text Similarity Detection Using Winnowing and Cosine Similarity,” *Jurnal Nasional Teknik Elektro dan Teknologi Informasi,* vol. 12, no. 3, pp. 219–226, 2023.

[23] N. C. Haryanto, L. D. Krisnawati, and A. R. Chrismanto, “Retrieval of source documents in a text reuse system,” *Jurnal Teknologi dan Sistem Komputer*, vol. 8, no. 2, pp. 140–149, Apr. 2020, doi: 10.14710/jtsiskom.8.2.2020.140-149.

[24] I. M. S. Putra, Putu Jhonarendra, and Ni Kadek Dwi Rusjayanthi, “Deteksi Kesamaan Teks Jawaban pada Sistem Test Essay Online dengan Pendekatan Neural Network,” *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 6, pp. 1070–1082, Dec. 2021, doi: 10.29207/resti.v5i6.3544.

[25] N. Luh Wiwik Sri Rahayu Ginantra and N. Wayan Wardani, “IMPLEMENTASI METODA NAÏVE BAYES DAN VECTOR SPACE MODEL DALAM DETEKSI KESAMAAN ARTIKEL JURNAL BERBAHASA INDONESIA,” *Jurnal Infomedia,* vol. 4, no. 2, pp. 94-100, 2019.

[26] I. Obeidat and M. Alzubi, “DEVELOPING A FASTER PATTERN MATCHING ALGORITHMS FOR INTRUSION DETECTION SYSTEM,” *International of Computing,* vol. 18, no. 3, pp. 278-284, Sep. 2019. [Online]. Available: www.computingonline.net

[27] M. Misbah Musthofa and A. Yaqin, “Implementation of Rabin Karp algorithm for essay writing test system on organization XYZ,” in *2019 International Conference on Information and Communications Technology, ICOIACT 2019,* Institute of Electrical and

Electronics Engineers Inc., Jul. 2019, pp. 502–507. doi: 10.1109/ICOIACT46704.2019.8938562.

[28] A. Nur Khusna, L. Rizkawati, and A. Dahlan Jl Ahmad Yani, "Information Retrieval pada Pencarian Menu Balita Menggunakan Dice Coefficient," *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, vol. 6, no. 1, pp. 7-12, Apr. 2020, [Online]. Available: www.ayahbunda.co.id.