Implementasi Algoritma Rabin-Karp untuk Pendeteksi Plagiarisme pada Dokumen Tugas Mahasiswa

(Rabin-Karp Algorithm Implementation to Detect Plagiarism on Student's Assignment Document)

Asvarizal Filcha[1], Mardhiya Hayaty[2]

[1,2]*Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Amikom Yogyakarta*
Jl. Ring Road Utara, Condong Catur, Sleman, Yogyakarta, (0274) 884201 - 207

[1]asvarizal.filcha@students.amikom.ac.id
[2]mardhiya_hayati@amikom.ac.id

Abstrak— Perkembangan pada dunia teknologi informasi mengakibatkan perguruan tinggi mengurangi penggunaan kertas sehingga banyak tugas mahasiswa yang dikumpulkan dalam bentuk digital. Penggunaan digital menyebabkan semakin mudahnya mahasiswa untuk melakukan plagiarisme. Sehingga diperlukan sebuah sistem untuk melakukan pemeriksaan plagiarisme pada dokumen tugas antar mahasiswa dengan cepat dan tepat. Metode yang dapat digunakan adalah menggunakan algoritma Rabin-Karp. Algoritma Rabin-Karp memiliki keunggulan pencarian *string* dengan pola yang panjang. Algoritma Rabin-karp dalam sistem ini memiliki langkah - langkah *text preprocessing* yang terdiri case folding, *tokenizing, punctuation removal, stopword removal* dan *stemming*. Hasil dari *text preprocessing* inilah yang akan di proses menggunakan algoritma Rabin-karp. Hasil dari metode ini adalah nilai kemiripan dari tugas - tugas mahasiswa yang dihitung menggunakan *dice coefficient*. Perhitungan akurasi dengan melakukan 20 perbandingan antara sistem pendeteksi plagiarisme dan *software* Plagiarisme Checker X menggunakan *confusion matrix* menghasilkan tingkat keakuratan sebesar 90%.

Kata Kunci* - Rabin-Karp, plagiarisme, *text preprocessing, dice coefficient.

Abstract— The Information and technology development causes universities reduce paper usage so that the student's assignments can be collected in digital form. The digital form usage causes students can easily plagiarism the assignments. So, it is needed a system to check the plagiarism on assignment documents among students quickly and accurate. The method that can be used is Rabin-Karp algorithm. Rabin-Karp algorithm has excellence in searching strings with long pattern. Rabin-Karp algorithm in this system has text preprocessing steps that consist of case folding, tokenizing, punctuation removal, stopword removal and stemming. The result from the text preprocessing will be processed using Rabin-Karp algorithm. The outcome of this method is the similiarity percentage of student's assignments calculated using dice coefficient. The accuracy calculation by doing 20 comparisons between plagiarism checker system and Plagiarism Checker X software using confusion matrix is 90%.

Keywords -- Rabin-Karp, plagiarism, text preprocessing, dice coefficient.

I. PENDAHULUAN

Perkembangan pada dunia teknologi informasi saat ini telah dirasakan dikalangan mahasiswa. Hal ini menyebabkan penyalahgunaan teknologi informasi yang semulanya digunakan untuk mencari referensi dengan lebih mudah namun kini digunakan sebagai alat untuk duplikasi atau plagiarisme. Faktor yang mempermudah mahasiswa melakukan plagiarisme adalah fasilitas pada komputer yang dapat melakukan menyalin dan mengubah teks antar dokumen. Dalam kenyataannya proses plagiarisme tidak hanya pada tugas essay saja tetapi pada kode program [1].

Sebelum menentukan suatu tugas mahasiswa termasuk melakukan plagiarisme atau tidak termasuk melakukan plagiarisme, maka sangatlah penting untuk mengetahui apa definisi plagiat atau plagiarisme. Plagiat adalah pengambilan karangan , pendapat, dan sebagainya dari orang lain dan menjadikannya seakan – akan karangan / pendapat sendiri, misalnya menerbitkan karya tulis orang lain atas nama dirinya sendiri. Orang yang melakukan plagiat disebut plagiator atau penjiplak. Berdasarkan pernyataan ini maka kesamaan

atau kemiripan isi dari tugas antara mahasiswa termasuk tindakan plagiarisme [1]. Saat ini yang sering terjadi di perguruan tinggi untuk melakukan pengecekan adanya tindakan plagiarisme pada dokumen tugas mahasiswa dilakukan secara manual dan membandingkan dengan dokumen tugas yang lain. Dengan begitu, pengecekan plagiarisme akan memakan waktu yang sangat lama.

Metode yang dapat digunakan adalah metode pencocokan *string* yaitu salah satunya menggunakan Algoritma Rabin-Karp. Algoritma Rabin-Karp adalah salah satu algoritma pencarian *string* dikembangkan oleh Michael O. Rabin dan Richard M. Karp pada tahun 1987 yang menggunakan fungsi *hashing* untuk menemukan *pattern* di dalam *string* teks [2]. Algoritma Rabin-Karp merupakan algoritma untuk pencocokan *string multi pattern*. Pada pencocokan *string multi pattern* paket atau informasi yang dicari berdasarkan beberapa susunan pola *string*. Algoritma Rabin-Karp melakukan pergeseran dari kiri ke kanan, fungsi dari algoritma Rabin-Karp menghasilkan efisiensi waktu yang baik dalam pencarian *string* yang memiliki lebih dari satu pola [3].

Berdasarkan uraian diatas, peneliti akan melakukan implementasi algoritma Rabin-Karp ke dalam sistem berbasis web agar dapat mendeteksi seberapa besar persentase plagiarisme atau kemiripan antar tugas mahasiswa.

A. Plagiarisme

Plagiarisme adalah praktik penyalahgunaan hak kekayaan intelektual milik orang lain orang dan pekerjaan itu diakui tidak sah sebagai akibat dari pekerjaan pribadi. Studi empiris yang dilakukan oleh Hutton and French in Hartanto mengemukakan bahwa bahwa faktor-faktor yang menyebabkan plagiarisme adalah kemalasan mereka sendiri, karena mereka merasa stres, memiliki keyakinan bahwa perilakunya tidak akan diketahui, dan perilakunya bukanlah hal yang salah untuk dilakukan atau berbahaya [4]. Adapun jenis-jenis plagiarisme [4], yaitu:

1) Plagiarisme kata per kata, yaitu menyalin setiap kata secara langsung tanpa melakukan perubahan sama sekali.

2) Plagiarisme pengarang, yaitu mengakui karya orang lain sebagai karya sendiri.

3) Plagiarisme ide, yaitu penggunaan ulang suatu gagasan/pemikiran asli dari sebuah umber teks tanpa bergantung pada teks sumber.

plagiarisme berdasarkan persentase dibagi menjadi 3 [4], yaitu:

1) Plagiarisme ringan , jika tingkat kesamaannya dibawah 30%.

2) Plagiarisme sedang, jika tingkat kesamaannya antara 30% hingga 70%.

3) Plagiarisme berat, jika tingkat kesamaannya diatas 70%.

B. Metode Pencocokan String

Proses Pencocokan *string matching* merupakan suatu metode yang digunakan untuk menemukan keakuratan dari satu atau beberapa pola yang diberikan. Pencocokan string adalah subjek penting dalam ilmu komputer karena teks merupakan bentuk utama pertukaran informasi antara manusia, misalnya dalam literatur, makalah ilmiah, dan halaman web [4]. Pencocokan string dapat dimanfaatkan pada banyak lingkup, misalnya pencarian dokumen, pendeteksi kemiripan teks.

Algoritma pencocokan string dapat diklasifikasikan menjadi 3 jenis sesuai dengan arah pencariannya [5], yaitu:

1) Dari kiri ke kanan, ini adalah arah yang banyak digunakan algoritma pencocokan string, algoritma yang termasuk kategori ini adalah algoritma Brute Force, algoritma Knuth Morris Pratt, dan sebagainya.

2) Dari kanan ke kiri, arah yang biasanya menghasilkan hasil yang baik secra praktis. Algoritma yang termasuk dalam kategori ini adalah algoritma Boyer-Moore.

3) Urutan tertentu, dari arah yang ditentukan oleh algoritma. Salah satu contoh algoritma dalam kategori ini adalah Colossi Crochemore-Perrin.

C. Text Preprocessing

Text Preprocessing adalah proses yang sering digunakan untuk melakukan *text mining*. Tujuan dari *text preprocessing* adalah untuk mengembalikan teks menjadi bahasa yang alami [5]. Secara umum dalam tahap – tahap *text preprocessing*, yaitu:

1) Case Folding. Case Folding adalah proses untuk memanipulasi teks, semua masukan teks akan diubah menjadi huruf kecil [6].

2) Tokenizing. Tokenizing adalah proses pemisahan kata berdasarkan susunan kata. Hasil dari pemisahan kata disebut token [6].

3) Punctuation Removal. Punctuation Removal adalah proses menghapus karakter – karakter unik seperti karakter tanda seru, tanda tanya, tanda koma dan sebagainya [7].

4) Stopword Removal. Stopword Removal adalah kata-kata yang tidak deskriptif yang dapat dibuang.

Contoh *stopword* adalah "yang", "dan", "di", "dari" dan sebagainya. *Stopword Removal* adalah proses penghapusan kata yang tidak relevan dalam teks [6].

5) Stemming. Stemming adalah proses untuk merubah kata menjadi kata dasar [6].

D. Algoritma Rabin-Karp

Algoritma Rabin-Karp adalah salah satu algoritma pencarian *string* dikembangkan oleh Michael O. Rabin dan Richard M. Karp pada tahun 1987 yang menggunakan fungsi *hashing* untuk menemukan *pattern* di dalam *string* teks.

Algoritma Rabin-Karp memiliki beberapa karakteristik yaitu menggunakan K-Gram dan *hashing*. Penerapan algoritma Rabin-Karp dilkakukan setelah melewati tahapan *preprocessing* [8]. Berikut tahapan algoritma Rabin-Karp.

1) K-Gram. K-gram adalah rangkain token yang panjang dengan panjang k. Metode K-Gram ini mengambil potongan - potongan karakter huruf sejumlah nilai k dari sebuah teks yang secara kontiunitas dibaca dari awal teks sumber hingga akhir teks sumber [6]. Contoh K-Gram dengan nilai k = 4 dapat dilihat pada Tabel I.

TABEL I
CONTOH K-GRAM

Kalimat	:	komputer adalah perangkat elektronik
Preprocessing	:	komputerperangkatelektronik
K-Gram{4}	:	{komp} {ompu} {mput} {pute} {uter} {terp} {erpe} {rper} {pera} {eran} {rang} {angk} {ngka} {gkat} {kate} {atel} {tele} {elek} {lekt} {ektr} {ktro} {tron} {roni} {onik}

2) Hashing. Hashing merupakan salah satu cara untuk mengubah karakter *string* menjadi *integer* yang disebut nilai *hash*. Proses pengubahan menjadi nilai *hash* menggunakan fungsi *rolling hash* [6]. Persamaan *rolling hash* dapat dilihat pada persamaan 1.

$$H(c_1 \cdots c_k) = (c_1 * b^{(k-1)} + c_2 * b^{(k-2)} + \cdots + c_{(k-1)} * b^k + c_k) \bmod q \quad \text{..... (1)}$$

Keterangan :
H : substring
c : nilai ascii per-karakter
b : konstan bilangan prima
k : banyak karakter
q: modulo bilangan prima

Berikut contoh perhitungan *rolling hash* terhadap *substring* makan dengan nilai K-Gram 4 dapat dilihat pada Tabel II.

TABEL II
CONTOH *ROLLING HASH*

Atribut	Nilai Array
Rolling hash pertama	[0] => maka 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 kedua	[1] => akan 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

E. Dice Coefficient Similarity

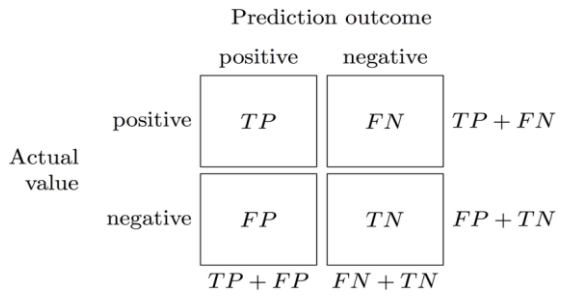
Dice Coefficient Similarity merupakan metode pengukuran yang paling umum digunakan untuk menghitung nilai kesamaan dengan pendekatan K-Gram [9].

$$S = \frac{2*C}{A+B} \text{ ..... (2)}$$

Pada persamaan 2, S adalah nilai similaritas, A dan B adalah jumlah dari *fingerprint hash* pada teks 1 dan *fingerprint hash* pada teks 2. C adalah jumlah dari *fingerprint hash* gabungan A dan B. *Fingerprint hash* merupakan *hash* yang unik dan tidak terduplikasi.

F. Confusion Matrix

Confusion Matrix adalah sebuah tabel yang menyatakan jumlah data uji yang benar diklasifikasikan dan jumlah data uji yang salah diklasifikasikan. Contoh *confusion matrix* dapat dilihat di gambar 1.



Gambar 1. *Confusion Matrix*

- *True Positive* (TP) adalah jumlah prediksi benar pada data dengan label benar.
- *True Negative* (TN) adalah jumlah prediksi benar pada data dengan label salah.
- *False Positive* (FP) adalah jumlah prediksi salah pada data dengan label benar.
- *False Negative* (FN) adalah jumlah prediksi salah pada data dengan label salah.

Perhitungan akurasi dinyatakan dalam persamaan 3 [10]:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \dots\dots\dots\dots (3)$$

II. METODE

Pada bagian ini menjelaskan langkah – langkah yang dilakukan untuk mendeteksi plagiarisme dimulai dari dokumen teks hingga hasil persentase kemiripan. Dokumen teks dapat dilihat pada Tabel III.

TABEL III
DOKUMEN TEKS

Dokumen Pertama	Dokumen Kedua
Dalam strategi pemasaran, langkah pertama yang harus dilakukan adalah identifikasi pasar.	Dalam memulai strategi pemasaran, langkah yang harus dilakukan yaitu identifikasi pasar.

Selanjutnya melakukan *text preprocessing* dari data dokumen teks. Berikut proses *text preprocessing* pada Tabel IV.

TABEL IV
TEXT PREPROCESSING

Dokumen	Case Folding	Tokenizing	Punctuation Removal	Stopword Removal	Stemming
Dokumen Pertama	dalam strategi pemasaran, langkah pertama yang harus dilakukan adalah identifikasi pasar.	dalam strategi pemasaran, langkah pertama yang harus dilakukan adalah identifikasi pasar.	dalam strategi pemasaran langkah pertama yang harus dilakukan adalah identifikasi pasar	strategi pemasaran langkah pertama dilakukan identifikasi pasar	strategi pasar langkah pertama lakui identifikasi pasar
Dokumen Kedua	Dalam memulai strategi pemasaran, langkah yang harus dilakukan yaitu identifikasi pasar.	dalam memulai strategi pemasaran, langkah yang harus dilakukan yaitu identifikasi pasar.	dalam memulai strategi pemasaran langkah yang harus dilakukan yaitu identifikasi pasar	memulai strategi pemasaran langkah dilakukan identifikasi pasar	mulai strategi pasar langkah lakui identifikasi pasar

Setelah melalui proses *text preprocessing* maka dapat dilihat hasil *text preprocessing* pada Tabel V.

TABEL V
HASIL *TEXT PREPROCESSING*

Teks Pertama	:	strategipasarlangkahpertamalakuidentifikasipasar
Teks Kedua	:	mulaistrategipasarlangkahlakuidentifikasipasar

Penerapan algoritma Rabin-Karp dilkakukan setelah melewati tahapan *preprocessing* , Berikut langkah - langkah algoritma Rabin-Karp pada Tabel VI.

TABEL VI
PROSES ALGORITMA RABIN-KARP

Langkah - Langkah Algoritma Rabin-Karp	Teks pertama	Teks Kedua
K-Gram	{stra} {trat} {rate} {ateg} {tegi} {egip} {gipa} {ipas} {pasa} {asar} {sarl} {arla} {rlan} {lang} {angk} {ngka} {gkah} {kahp} {ahpe} {hper} {pert} {erta} {rtam} {tama} {amal} {mala} {alak} {laku} {akui} {kuid} {uide} {iden} {dent} {enti} {ntif} {tifi} {ifik} {fika} {ikas} {kasi} {asip} {sipa} {ipas} {pasa} {asar}	{mula} {ulai} {lais} {aist} {istr} {stra} {trat} {rate} {ateg} {tegi} {egip} {gipa} {ipas} {pasa} {asar} {sarl} {arla} {rlan} {lang} {angk} {ngka} {gkah} {kahl} {ahla} {hlak} {laku} {akui} {kuid} {uide} {iden} {dent} {enti} {ntif} {tifi} {ifik} {fika} {ikas} {kasi} {asip} {sipa} {ipas} {pasa} {asar}
Rolling Hash	8340 9261 4736 4259 7743 8063 1022 4384 2059 4105 6052 4088 5867 6693 3559 35 1106 5305 2926 3096 2551 9500 6834 7317 3373 7996 3251 6674 3348 7724 9521 2971 6549 9024 1591 8216 3254 9643 3779 5419 4191 6987 4384 2059 4105	409 9855 6650 3095 4955 8340 9261 4736 4259 7743 8063 1022 4384 2059 4105 6052 4088 5867 6693 3559 35 1106 5301 2878 2561 6674 3348 7724 9521 2971 6549 9024 1591 8216 3254 9643 3779 5419 4191 6987 4384 2059 4105
Fingerprint	8340 9261 4736 4259 7743 8063 1022 4384 2059 4105 6052 4088 5867 6693 3559 35 1106 5305 2926 3096 2551 9500 6834 7317 3373 7996 3251 6674 3348 7724 9521 2971 6549 9024 1591 8216 3254 9643 3779 5419 4191 6987	409 9855 6650 3095 4955 8340 9261 4736 4259 7743 8063 1022 4384 2059 4105 6052 4088 5867 6693 3559 35 1106 5301 2878 2561 6674 3348 7724 9521 2971 6549 9024 1591 8216 3254 9643 3779 5419 4191 6987
	8340 9261 4736 4259 7743 8063 1022 4384 2059 4105 6052 4088 5867 6693 3559 35 1106 6674 3348 7724 9521 2971 6549 9024 1591 8216 3254 9643 3779 5419 4191 6987	

Berikut perhitungan *similarity* menggunakan persamaan 2.

S = ((2*C)/(A+B)) * 100
C = Jumlah fingerprint yang sama dari A dan B
A = Jumlah fingerprint pada dokumen A
B = Jumlah fingerprint pada dokumen B
S = ((2*32)/(42+40)) * 100
S = 78.05%

Berdasarkan perhitungan diatas, dapat diketahui persentase *similarity* antara dokumen pertama dan kedua adalah 78.05%.

III. HASIL DAN PEMBAHASAN

A. Pengujian Akurasi

Pada penelitian ini melakukan pengujian akurasi berdasarkan tingkatan plagiarisme yaitu plagiarisme berat atau tidak berat. Perhitungan akurasi dilakukan dengan menggunakan hasil *similarity* yang diperoleh pada sistem pendeteksi plagiarisme tugas mahasiswa yang dibuat oleh peneliti dan hasil *similarity* yang diperoleh dari *software* Plagiarism Checker X. Plagiarism Checker X adalah aplikasi *desktop* yang dapat digunakan untuk mendeteksi kemiripan dokumen secara *offline*. Pada proses perhitungan akurasi peneliti menggunakan perhitungan dari tabel *confusion matrix*. Hasil tabel *confusion matrix* dapat diketahui dari tabel klasifikasi data uji. Tabel klasifikasi data uji dapat dilihat pada TABEL VII.

TABEL VII
KLASIFIKASI DATA UJI

Dokumen 1	Dokumen 2	Sistem Peneliti		Plagiarsm Checker X		Klasifikasi
		Berat	Tidak Berat	Berat	Tidak Berat	
15.11.9292 Review Game.docx	15.11.9324 Review Game.docx		✓		✓	TN
15.11.9292 Review Game.docx	15.11.9330 Review Game.docx		✓		✓	TN
15.11.9292 Review Game.docx	15.11.9333 Review Game.docx	✓		✓		TP
15.11.9292 Review Game.docx	15.11.9334 Review Game.docx		✓		✓	TN
15.11.9324 Review Game.docx	15.11.9330 Review Game.docx	✓			✓	FN
15.11.9324 Review Game.docx	15.11.9333 Review Game.docx		✓		✓	TN
15.11.9324 Review Game.docx	15.11.9334 Review Game.docx		✓		✓	TN
15.11.9330 Review Game.docx	15.11.9333 Review Game.docx		✓		✓	TN
15.11.9330 Review Game.docx	15.11.9334 Review Game.docx		✓		✓	TN
15.11.9333 Review Game.docx	15.11.9334 Review Game.docx		✓		✓	TN
15.11.9288 Review Game.docx	15.11.9295 Review Game.docx	✓		✓		TP
15.11.9288 Review Game.docx	15.11.9304 Review Game.pdf		✓		✓	TN
15.11.9288 Review Game.docx	15.11.9331 Review Game.docx		✓		✓	TN
15.11.9288 Review Game.docx	15.11.9336 Review Game.docx		✓		✓	TN
15.11.9295 Review Game.docx	15.11.9304 Review Game.pdf		✓		✓	TN
15.11.9295 Review Game.docx	15.11.9331 Review Game.docx		✓		✓	TN
15.11.9295 Review Game.docx	15.11.9336 Review Game.docx		✓		✓	TN
15.11.9304 Review Game.pdf	15.11.9331 Review Game.docx		✓		✓	TN
15.11.9304 Review Game.pdf	15.11.9336 Review Game.docx		✓		✓	TN
15.11.9331 Review Game.docx	15.11.9336 Review Game.docx		✓	✓		FP

Dari hasil tabel klasifikasi data uji maka diketahui tabel *confusion matrix* sebagai berikut pada TABEL VIII.

TABEL VIII
HASIL *CONFUSION MATRIX*

		Predicted (Sistem Peneliti)	
		Berat	**Tidak Berat**
Actual (Plagiarisme Checker X)	**Berat**	(TP) 2	(FN) 1
	Tidak Berat	(FP) 1	(TN) 16

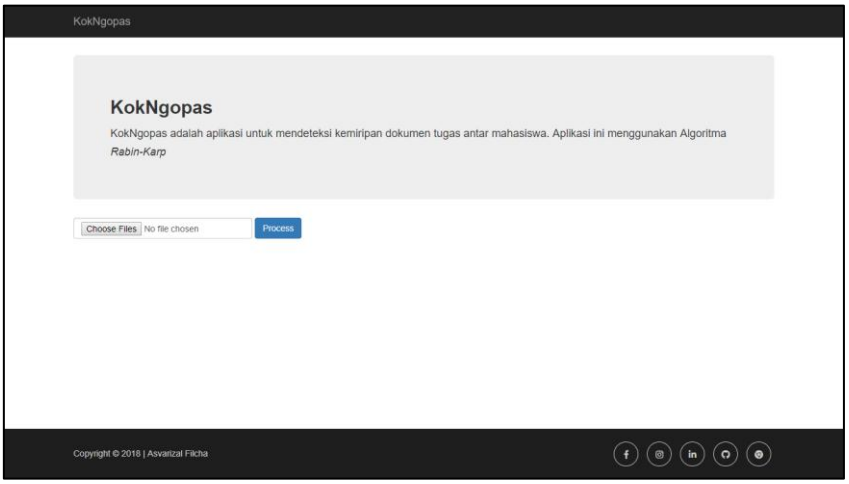
Setelah mengetahui tabel *confusion matrix* maka peneliti melakukan perhitungan akurasi menggunakan persamaan 3. Berikut perhitungannya akurasi dari tabel *confusion matrix*.

Akurasi = (TP+TN) / (TP+TN+FP+FN)
Akurasi = (2+16) / (2+16+1+1)
Akurasi = 0.9 * 100
Akurasi = 90%

Berdasarkan perhitungan dari tabel *confusion matrix* maka nilai akurasinya adalah 90%.

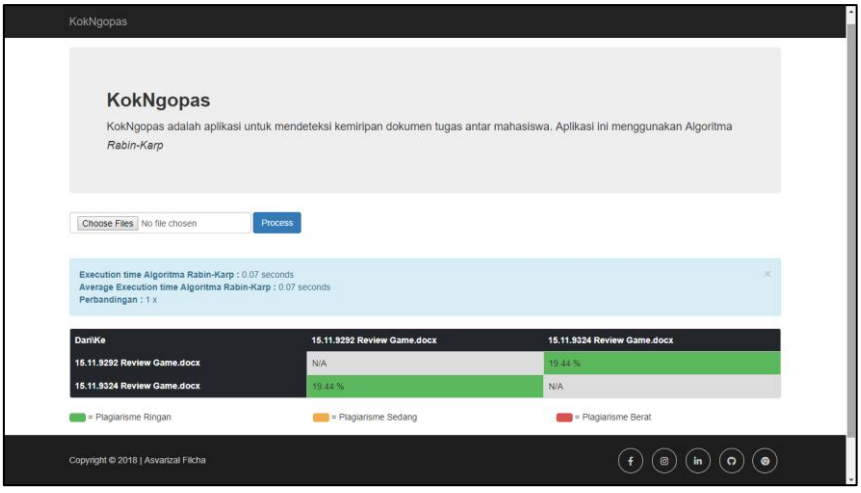
B. Tampilan Sistem

1) Halaman Unggah Berkas. Halaman unggah berkas merupakan halaman utama untuk pengguna mengunggah berkas tugas mahasiswa. Selanjutnya pengguna menekan tombol *process* untuk mengecek persentase plagiarisme tugas mahasiswa (Gambar 2).



Gambar 2. Halaman Unggah Berkas

2) Halaman Hasil Similarity Tugas Mahasiswa. Halaman hasil *similarity* tugas mahasiswa merupakan halaman yang menampilkan waktu eksekusi, rata – rata waktu eksekusi dan tabel kemiripan tugas mahasiswa (Gambar 3).



Gambar 3. Halaman Hasil Similarity Tugas

IV. PENUTUP

A. Simpulan

Berdasarkan hasil dan pembahasan maka peneliti dapat mengambil kesimpulan diantaranya Algoritma Rabin-Karp berhasil diimplementasikan pada sistem pendeteksi plagarisme dokumen tugas mahasiswa. Sistem ini berhasil menampilkan persentase kemiripan dokumen tugas antar mahasiswa. Hasil perhitungan akurasi dengan *confusion matrix* pada sistem pendeteksi plagiarisme dokumen tugas mahasiswa adalah 90% yang diperoleh dari 20 perbandingan dokumen tugas mahasiswa. Algoritma yang digunakan pada sistem pendeteksi plagiarisme dokumen tugas mahasiswa tidak memiliki perbedaan persentase saat urutan perbandingan diubah.

B. Saran

Beberapa saran untuk pengembangan lebih lanjut terhadap penelitian ini yaitu sistem ini dapat dikembangkan lebih lanjut, yakni dapat membedakan persentase saat urutan perbandingan diubah serta sistem ini dapat dikembangkan lebih lanjut, yakni menggunakan algoritma atau metode lain yang dapat mengetahui kalimat – kalimat yang mengandung unsur plagiarisme. Perlu adanya pengembangan metode *stemming* untuk mencari kata dasar dari kata - kata slang seperti sotoy, baper, dan sebagainya.

DAFTAR PUSTAKA

[1] P. L. Yanuarista, H. W. Dwi, and D. Wulandari, "Analisis Plagiarisme Dalam Penulisan Skripsi Mahasiswa Program Studi S1 Pendidikan Ekonomi Pembangunan Tahun 2010 -2014 Universitas Negeri Malang," *J. Pendidik. Ekon.*, vol. 8, no. 1, pp. 36–47, 2015.

[2] N. Bansal, "An Elementary Algorithm for Pattern Matching," *Int. J. Comput. Sci. Eng. Commun.*, vol. 6, no. 1, pp. 1780–1787, 2018.

[3] D. D. Sinaga and S. Hansun, "Detection System Using Rabin-Karp," *Int. J. Innov. Comput. Inf. Control*, vol. 14, no. 5, pp. 1893–1903, 2018.

[4] B. Leonardo and S. Hansun, "Text documents plagiarism detection using Rabin-Karp and Jaro-Winkler distance algorithms," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 5, no. 2, pp. 462–471, 2017.

[5] N. P. Katariya and M. S. Chaudhari, "Text Preprocessing for Text Mining Using Side Information," *Int. J. Comput. Sci. Mob. Appl.*, vol. 3, pp. 3–7, 2015.

[6] R. K. Wibowo and K. Hastuti, "Penerapan Algoritma Winnowing Untuk Mendeteksi Kemiripan Teks pada Tugas Akhir Manusia," *Techno.COM*, vol. 15, no. 4, pp. 303–311, 2016.

[7] A. Squicciarini, A. Tapia, and S. Stehle, "Sentiment analysis during Hurricane Sandy in emergency response," *Int. J. Disaster Risk Reduct.*, vol. 21, no. December 2016, pp. 213–222, 2017.

[8] A. Putera, U. Siahaan, S. Aryza, and E. Hariyanto, "Combination of levenshtein distance and rabin-karp to improve the accuracy of document equivalence level," vol. 7, pp. 17–21, 2018.

[9] T. Mardiana, T. B. Adji, and I. Hidayah, "The Comparation of Distan ce-based Similarity Measure to Detection of Plagiarism in Indonesian Text," *Commun. Comput. Inf. Sci.*, vol. 516, no. March, 2015.

[10] S. Visa, B. Ramsay, A. Ralescu, and E. Van Der Knaap, "Confusion matrix-based feature selection," *CEUR Workshop Proc.*, vol. 710, pp. 120–127, 2011.