PERBANDINGAN N-GRAM *TECHNIQUE* DAN RABIN KARP PADA APLIKASI PENDETEKSI PLAGIARISME DOKUMEN TEKS BAHASA INDONESIA

Yusup Miftahuddin[1], Jasman Pardede[2], Acep Andi Andriani[3]

[1]Teknik Informatika Institut Teknologi Nasional
[2]Teknik Informatika Institut Teknologi Nasional
[3]Teknik Informatika Institut Teknologi Nasional
Jl. PH. H. Mustofa No.23, Bandung, Jawa Barat, Indonesia
[1]ymiftahuddin@gmail.com, [2]jasmanpardede78@gmail.com, [3]acepandi56@gmail.com

Abstract

Plagiarism is copying or take over a works, paper and so on from other people who seemed to be claimed as his own work. Plagiarism detection can be done by matching the test document and source document. Detection process has few steps, which is preprocessing step, TF-IDF weighting step, and document similiarity detection process. Detection process is done with N-Gram Technique and Rabin Karp method. N-gram do the sentence splitting based on defined character length, then do the character matching and counting the similiarity percentage value. While in the Rabin Karp detection do the sentence splitting based on character length, then counting the hash value, then matching it, if hash value is match, then for the next step is doing the word matching and counting the similiarity percentage value. From the test result, TF-IDF weighting can be use to plagiarism detection. From the comparison of both method, N-Gram and Rabin Karp produce the same similarity value, but different detection time. N-Gram detection process time is faster than Rabin Karp.

***Key words**: plagiarism, N-Gram Technique, Rabin Karp, Similarity, TF-IDF Weighting.*

1. Pendahuluan

1.1. Latar Belakang

Plagiarisme atau sering disebut plagiat adalah penjiplakan atau pengambilan karangan, pendapat dan sebagainya dari orang lain dan menjadikannya seolah karangan dan pendapat sendiri.[14]

Dengan adanya tindakan plagiarisme dapat dilakukan pencegahan dengan melakukan pendeteksian pada dokumen teks. Proses pendeteksian terdapat beberapa tahapan yaitu, *preprocessing*, proses pembobotan TF-IDF, dan pencocokan antara dokumen uji dengan dokumen sumber. Proses pencocokan dokumen menggunakan metode N-Gram *Technique* dan Rabin Karp. Metode N-Gram *Technique* melibatkan 2 (dua) langkah, yaitu membagi string menjadi *overlapping* N-Gram (suatu set substring dengan panjang n) dan melakukan pengecekan untuk mendapatkan substring yang memiliki struktur yang sama.[5] Metode Rabin Karp dilakukan dengan pencocokan string yang menggunakan fungsi *hash* sebagai pembanding antara string yang dicari (m) dengan substring pada teks (n). Apabila *hash value* keduanya sama maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila hasil keduanya tidak sama, maka substring akan bergeser ke kanan.[6]

Berdasarkan penelitian yang sudah dilakukan sebelumnya dengan berbagai metode dan tingkat keakuratan atau nilai persentase *sismiliarity text* yang berbeda, maka pada penelitian ini dilakukan perbandingan metode N-Gram *Technique* dan Rabin Karp untuk pendeteksi dokumen teks, serta ditambah proses pembobotan TF-IDF untuk mencari kalimat yang relevan.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka dapat dirumuskan masalah yaitu sebagai berikut :

1. Bagaimana pendeteksian plagiarisme dengan menggunakan metode pembobotan TF-IDF.
2. Bagaimana perbandingan antara N-Gram *Technique* dengan algoritma Rabin Karp untuk menghasilkan nilai *similiarity* pada pendeteksi plagiarisme dokumen teks.
3. Bagaimana implementasi kedua metode tersebut pada aplikasi pendeteksi plagiarisme dokumen teks bahasa Indonesia.

1.3. Tujuan

Tujuan dari penelitian tugas akhir ini adalah membandingkan antara metode N-Gram *Technique* dengan Rabin Karp dan membandingkan tingkat akurasinya, serta mengembangkan aplikasi pendeteksi plagiarisme dokumen teks Bahasa Indonesia.

1.4. Metode Penelitian

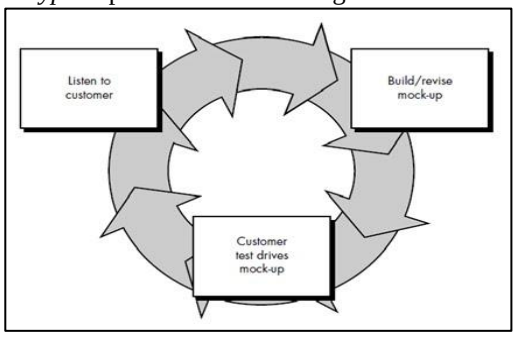
Penelitian dan penyusunan tugas akhir ini dilakukan dengan tahap – tahap sebagai berikut:

1. Studi literatur

Studi literatur dilakukan untuk mendapatkan sumber referensi dalam pengumpulan teori - teori dan informasi mengenai permasalahan yang dibahas dan mempelajari literatur - literatur dari buku, jurnal, artikel ilmiah, serta website.

2. Pengembangan sistem

Dalam pengembangan sistem ini digunakan sistem metodologi *prototype*. Alur metodologi *prototype* dapat dilihat dalam dengan Gambar 1.



Gambar 1. Pengembangan Sistem Prototype
Sumber: (trimanfridayanto, 2014, https://murtri.wordpress.com/2014/08/25/model-model-pengembangan-perangkat-lunak-beserta-contoh-penerapannya/)

2. Metodologi

1. *Preprocessing*[8]

Preprocessing merupakan proses mempersiapkan teks menjadi data yang dapat diolah pada tahapan selanjutnya inputan awal berupa dokumen. Dalam penelitian ini ada beberapa tahapan proses yaitu, pemecahan kalimat, *case folding*, *tokenizing*, *filtering*, dan *stemming*. Sehingga dari hasil *preprocessing* akan menghasilkan kata dasar dari sebuah dokumen.

2. Pembobotan TF-IDF [3]

TF-IDF (*Term Frequency-Inverse Document Frequency)* merupakan salah satu metode yang biasa digunakan dalam pembobotan sebuah kata di dalam sistem pencarian informasi. Metode TF-IDF menghitung nilai dari masing-masing kata di dalam dokumen menggunakan frekuensi kata tersebut muncul. Kata dengan nilai TF-IDF yang tinggi, maka mempunyai hubungan yang kuat dengan dokumen di mana kata tersebut muncul, diasumsikan bahwa jika kata tersebut muncul di dalam *query* maka akan memiliki ketertarikan untuk pengguna. TF-IDF juga dapat digunakan dalam pembobotan kata untuk mencari keputusan yang relevan.

Nilai dari TF didapat menggunakan persamaan:

$$TF_{(t)} = \frac{f_{t,d}}{\sum t,d} \quad \text{..... (1)}$$

Dimana:

$f_{t,d}$ = frekuensi sebuah kata (*t*) muncul di dalam dokumen *d*,

$\sum t,d$ = total keseluruhan kata yang terdapat di dalam dokumen *d.*

Kemudian untuk menghitung nilai IDF (*Inverse Document Frequency*) dari sebuah kata di dalam kumpulan dokumen menggunakan persamaan:

$$IDF_{(t)} = log\frac{|D|}{f_{t,D}} \quad \text{.....(2)}$$

Dimana:

$|D|$ = jumlah dokumen yang ada di dalam koleksi,

$f_{t,D}$ = jumlah dokumen di mana *t* muncul di dalam *D*.

Dalam koleksi dokumen *D*, sebuah kata *t* dan dokumen individu $d \in D$, dapat dihitung nilai TF IDF menggunakan rumus:

$$TF, IDF_{(t)} = TF_{(t)} * IDF_{(t)} \quad \text{.....(3)}$$

Dimana:

$TF(t)$= *Term* Frekuensi dari sebuah kata (t),

$IDF(t)$ =*Inverse Document Frequency* dari sebuah kata (t)

3. Similarity

Similarity adalah persentase tingkat kemiripan antar dokumen. Setelah dilakukan pengujian terhadap dokumen teks dengan menerapkan metode tertentu, maka diperoleh nilai persentase kemiripannya atau nilai *similarity*nya.[6] Untuk menghitung *similarity* dapat dihitung dengan menggunakan Sorensen-Dice *Coefficient*

Sorensen-Dice *Coefficient*, atau biasa disebut Sorensen *Index* atau Dice's *Coefficient* ditemukan oleh Throvald Sorensen dan Lee Raymond Dice. Rumus yang digunakan, yaitu:

$$S = \frac{2 * |A \cap B|}{|A| + |B|} \quad \text{.....(4)}$$

Dimana:

S = nilai *similarity*.

|A| dan |B| = jumlah kata yang unik dari teks pertama dan teks kedua.

|A∩ B| = jumlah kata unik dan memiliki struktur yang sama dari masing-masing teks yang dibandingkan.[5]

Untuk menentukan jenis plagiarisme antara dokumen yang diuji ada 5 jenis penilaian persentase similarity: [7]

- 0% : Hasil uji 0% berarti kedua dokumen tersebut benar-benar berbeda baik dari segi isi dan kalimat secara keseluruhan.
- < 15%: Hasil uji 15% berarti kedua dokumen tersebut hanya mempunyai sedikit kesamaan.

- 15-50%: Hasil uji 15-50% berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang.
- >50%: Hasil uji lebih dari 50% berarti dapat dikatakan bahwa dokumen tersebut mendekati plagiarisme.
- 100%: Hasil uji 100% menandakan bahwa dokumen tersebut adalah plagiat karena dari awal sampai akhir mempunyai isi yg sama persis.

4. N-Gram *Technique*[5]

Teknik N-Gram didasarkan pada pemisahan teks menjadi *string* dengan panjang n mulai dari posisi tertentu dalam suatu teks. Posisi n-gram berikutnya dihitung dari posisi yang sebenarnya bergeser sesuai dengan *offset* yang diberikan. N-gram untuk setiap *string* dihitung dan kemudian dibandingkan satu per satu. N-gram dapat berupa unigram (n=1), bigram (n=2), trigram (n=3), dan seterusnya.

. Dalam memperkirakan *similarity* maka teknik N-gram sering dipadukan dengan pendekatan statistika untuk memperoleh *similarity* dari 2 (dua) buah *sample*.

Sebagai contoh, bigram dari Photography dan Photographic, yaitu {Ph, ho, ot, to, og, gr, ra, ap, hy} dan {Ph, ho, ot, to, og, gr, ra, ap, hi, ic}. Dari kedua kata tersebut dapat diperoleh bigram yang memiliki struktur yang sama yaitu {Ph, ho, ot, to, og, gr, ra, ap}.

5. Algoritma Rabin Karp

Algoritma Rabin Karp adalah algoritma pencocokan string yang akan menggunakan fungsi *hash* sebagai pembanding antara string yang dicari (m) dengan *substring* pada teks (n). Apabila *hash value* keduanya sama maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila hasil keduanya tidak sama, maka *substring* akan bergeser ke kanan. Pergeseran dilakukan sebanyak (n-m) kali. Perhitungan nilai *hash* yang efisien pada saat pergeseran akan mempengaruhi performa dari algoritma ini.[6] Dalam algoritma Rabin Karp terdapat terdapat proses yaitu sebagai berikut:

a. K-Gram[7]

K-Gram merupakan sebuah metode yang diaplikasikan untuk pembangkitan kata atau karakter. Metode K-Gram ini digunakan untuk mengambil potongan-potongan karakter huruf sejumlah k dari sebuah kata yang secara kontinuitas dibaca dari teks sumber hingga akhir dari dokumen.

b. *Hashing* [2]

Hashing adalah proses mengubah karakter dari bentuk string ke dalam bilangan integer dengan panjang tertentu yang disebut nilai *hash* (*hash value*). Dalam mencari nilai *hash* Rabin-Karp memakai aturan Horner dan Operand Denumerator Modulus. Untuk sebuah kata (string) w dengan panjang m rumus *hash*(w) dituliskan pada persamaan (5):

$$Hash(w[0 \ldots m-1]) = (w[0] * b^{m-1} + w[1] * b^{m-2} + \ldots + w[m-1] * b^{0}) mod\ q \quad \ldots\ldots\ldots(5)$$

Dimana:
w[i] = nilai ASCII karakter ke-i,
b = basis,
m = banyaknya karakter dalam pola,
q = operand denumerator modulo.

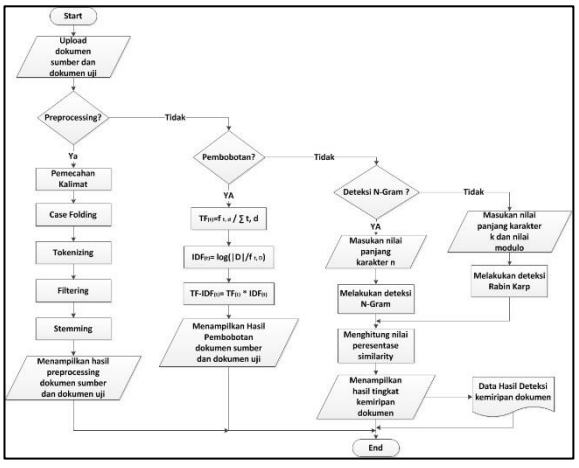
Umumnya basis dipilih 10 untuk merepresentasikan sepuluh kemungkinan angka (0 - 9). Fungsi dari modulo adalah untuk memperkecil memori yang dipakai karena nilai variable membesar secara eksponensial berbanding lurus dengan panjang pola. Dan modulo biasanya memakai bilangan prima yang cukup besar untuk memperlebar varian output sehingga mengurangi kemungkinan dua *corresponding number value* yang sama.

3. Pembahasan

1. Flowchart Aplikasi

Metode yang digunakan untuk deteksi plagiarisme yaitu N-Gram *Technique* dan algoritma Rabin Karp. Proses yang dilakukan dalam deteksi plagiarisme yaitu, *preprocessing*, proses pembobotan TF-IDF yang biasanya digunakan untuk peringkasan dokumen, pembobotan dilakukan untuk menentukan kalimat yang relevan pada suatu dokumen.

Pada `Gambar 1` merupakan *flowchart* pada pengembangan aplikasi deteksi plagiarisme dokumen teks.



Gambar 2. *Flowchart* Aplikasi Deteksi Plagiarisme

2. Studi Kasus

Pada studi kasus dapat dipaparkan tahap-tahap deteksi dengan metode N-Gram dan Rabin Karp yaitu sebagai berikut :

Dokumen sumber:

Plagiarisme adalah suatu kegiatan menjiplak karya orang lain yang melanggar hak cipta. Pelaku plagiat

disebut sebagai plagiator. Plagiator dapat dihukum berat.

Dokumen uji:

Plagiarisme adalah penjiplakan yang melanggar hak cipta. Pelaku plagiat disebut sebagai plagiator. Plagiarisme banyak dilakuakan pada kegiatan akademik.

1. *Preprocessing*

Pada proses *preprocessing* terdapat beberapa tahap yaitu sebagai berikut :

a. Pemecahan Kalimat: Pada tahap ini dokumen sumber dan dokumen uji dipecah menjadi potongan kalimat berdasakan tanda titik, tanda tanya, tanda seru.
b. *Case Folding*: Pada tahap ini dilakukan perubahan huruf kapital menjadi huruf kecil.
c. *Tokenizing*: Pada tah ini dilakukan pemecahan kata berdasarakan spasi pada kalimat hasil *case folding*.
d. *Filtering*: Pada tahap *filtering* dilakukan proses untuk menghilangkan kata yang tidak penting dengan cara mencocokan kata dengan *stopwordlist* yang ada.
e. *Stemming*: Dari hasil *filtering* dilakukan pencarian kata dasar dengan menghilangkan imbuhan.

Hasil dari *preprocessing* ditunjukan pada Tabel 1.

Tabel 1. Hasil *Preprocessing*

Dokumen Sumber		Dokumen Uji	
Kata	**Hasil Stemming**	**Kata**	**Hasil Stemming**
plagiarisme	plagiarisme	plagiarisme	plagiarisme
kegiatan	giat	plagiarisme	plagiarisme
penjiplakan	jiplak	penjiplakan	jiplak
karya	karya	melanggar	langgar
melanggar	langgar	kegiatan	giat
hak	hak	hak	hak
cipta	cipta	akademik	akademik
pelaku	laku	cipta	cipta
plagiat	plagiat	pelaku	laku
plagiator	plagiator	plagiat	plagiat
plagiator	plagiator	plagiator	plagiator
dihukum	hukum		
berat	berat		

2. Pembobotan TF-IDF

Pada tahap ini dilakukan proses pembobotan TF-IDF untuk mendapatkan kalimat yang relevan. Pada Tabel 2 untuk dokumen sumber dan pada Tabel 3 untuk dokumen uji menunjukan hasil dari proses pembobotan TF-IDF.

Tabel 2. Pembobotan TF-IDF Dokumen Sumber

Kata	Dokumen Sumber							
	DS1	DS2	DS3	df	IDF	TF*IDF		
						DS1	DS2	DS3
plagiarisme	1			1	0.4771	0.4771		
giat	1			1	0.4771	0.4771		
jiplak	1			1	0.4771	0.4771		
karya	1			1	0.4771	0.4771		
langgar	1			1	0.4771	0.4771		
hak	1			1	0.4771	0.4771		
cipta	1			1	0.4771	0.4771		
pelaku		1		1	0.4771		0.4771	
pagiat		1		1	0.4771		0.4771	
plagiator		1	1	2	0.1761		0.1761	0.1761
hukum			1	1	0.4771			0.4771
berat			1	1	0.4771			0.4771
					Jumlah	3.3398	1.1303	1.1303
					Urutan	1	2	3

Tabel 3. Pembobotan TF-IDF Dokumen Uji

Kata	Dokumen Uji							
	DU1	DU2	DU3	df	IDF	TF*IDF		
						DU1	DU2	DU3
plagiarisme	1		1	2	0.1761	0.1761		0.1761
jiplak	1			1	0.4771	0.4771		
langgar	1			1	0.4771	0.4771		
giat			1	1	0.4771			0.4771
hak	1			1	0.4771	0.4771		
akademik			1	1	0.4771			0.4771
cipta	1			1	0.4771	0.4771		
laku		1		1	0.4771		0.4771	
plagiat		1		1	0.4771		0.4771	
plagiator		1		1	0.4771		0.4771	
					Jumlah	2.0846	1.4314	1.1303
					Urutan	1	2	3

Hasil perhitungan pembobotan TF-IDF diambil 40% dari jumlah kalimat (D) dan diambil nilai terbesar dari jumlah bobot kalimat.[3] Sehingga didapat hasil pembobotan TF-IDF sebagai berikut:

Dokumen Sumber:

Plagiarisme adalah suatu kegiatan menjiplak karya orang lain yang melanggar hak cipta

Dokumen Uji:

Plagiarisme adalah penjiplakan yang melanggar hak cipta.

3. Deteksi N-Gram

Teks hasil pembobotan TF-IDF selanjutnya dilakukan pencocokan dokumen uji dengan dokumen sumber. Proses yang dilakukan yaitu pemecahan kalimat menjadi kata berdasarkan panjang karakter n-gram, dan kemudian dicocokan antar kata. Pada kasus ini panjang n-gram yang digunakan, yaitu 4. Berikut ini merupakan hasil dari pemecahan kalimat menjadi kata dan hasil dari pencocokan dokumen uji dengan dokumen sumber:

Dokumen Uji

plag lagi agia giar iari aris rism isme smej meji ejip jipl ipla plak lakm akme kmel mela elan lang angg ngga ggar garh arha rhak hakc akci kcip cipt ipta

Dokumen Sumber

plag lagi agia giar iari aris rism isme smeg megi egia giat iatj atji tjip jipl ipla plak lakk akka kkar kary arya ryal yala alan lang angg ngga ggar garh arha rhak hakc akci kcip cipt ipta

Selanjutnya dihitung jumlah kata pada masing - masing dokumen dan dihitung jumlah kata sama antara dokumen uji dan dokumen sumber.

Jumlah N-Gram pada dokumen sumber = 38 kata

Jumlah N-Gram dokumen uji = 29 kata

Jumlah kata sama = 23 kata.

Setelah didapat hasil kata yang sama selanjutnya dilakukan perhitungan persentase *sismilarity* menggunakan persamaan (4).

$$S = \frac{2 * 23}{38 + 29} = \frac{46}{67} = 0.6866$$

Jadi, nilai *similarity* = 0.6866 dan persentase *similarity* = 0.6866 * 100 = 68,66%. Dari hasil yang didapat bahwa persentase *sismilarity* > 50%, maka dokumen uji memiliki tingkat kemiripan yaitu, "mendekati plagiarisme".

4. Deteksi Rabin Karp

Deteksi plagiarisme dengan menggunakan metode rabin karp dapat dilakukan dengan tahapan sebagai berikut:

a. K-Gram

Pada tahap ini dilakukan proses pemecahan kalimat menjadi kata berdasarkan panjang karakter k-gram. Pada kasus ini digunakan panjang karakter = 4. Hasil dari pemecahan yaitu, sebagai berikut:

Dokumen Uji

plag lagi agia giar iari aris rism isme smej meji ejip jipl ipla plak lakm akme kmel mela elan lang angg ngga ggar garh arha rhak hakc akci kcip cipt ipta

Dokumen Sumber

plag lagi agia giar iari aris rism isme smeg megi egia giat iatj atji tjip jipl ipla plak lakk akka kkar kary arya ryal yala alan lang angg ngga ggar garh arha rhak hakc akci kcip cipt ipta

b. *Hashing*

Pada tahap ini dilakukan perhitungan nilai *hash* pada masing-masing kata pada dokumen uji dan dokumen sumber menggunakan persamaan (4). Berikut contoh perhitungan nilai *hash*: Kata "plag"

$$\text{Hashing} = \big((112 * 1000) + (108 * 100) + (97 * 10) + (103 * 1)\big) \bmod 809$$

$$\text{Hashing} = \big((112000) + (10800) + (970) + (103)\big) \bmod 809$$

$$\text{Hashing} = (123873) \bmod 809$$

$$\text{Hashing} = 96$$

Tabel 4. Hasil Perhitungan Hash

Dokumen Sumber				Dokumen Uji			
Kata	Hash	Kata	Hash	Kata	Hash	Kata	Hash
plag	**96**	akka	461	**plag**	**96**	**ngga**	**77**
lagi	**721**	kkar	670	**lagi**	**721**	**ggar**	**315**
agia	**41**	kary	656	**agia**	**41**	**garh**	**684**
giar	**515**	arya	492	**giar**	**515**	**arha**	**322**
iari	**258**	ryal	165	**iari**	**258**	**rhak**	**82**
aris	**350**	yala	10	**aris**	**350**	**hakc**	**800**
rism	**364**	alan	474	**rism**	**364**	**akci**	**389**
isme	**386**	**lang**	**789**	**isme**	**386**	**kcip**	**757**
smeg	0	**angg**	**727**	smej	3	**cipt**	**712**
megi	503	**ngga**	**77**	meji	533	**ipta**	**152**
egia	805	**ggar**	**315**	ejip	311		
giat	517	**garh**	**684**	**jipl**	**423**		
iatj	279	**arha**	**322**	**ipla**	**72**		
atji	550	**rhak**	**82**	**plak**	**100**		
tjip	749	**hakc**	**800**	lakl	764		
jipl	**423**	**akci**	**389**	akla	471		
ipla	**72**	**kcip**	**757**	klan	766		
plak	**100**	**cipt**	**712**	**lang**	**789**		
lakk	763	**ipta**	**152**	**angg**	**727**		
Jumlah Kata	38			29			
Jumlah Kata Sama	23						

Hasil dari perhitungan *hash* selanjutnya dilakukan pencocokan nilai *hash*, jika nilai *hash* sama maka dilakukan pencocokan kata. Pada Tabel 4 merupakan hasil pencocokan hash dan pencocokan kata. Sehingga didapat jumlah kata sama = 23 kata.

c. *Similarity*

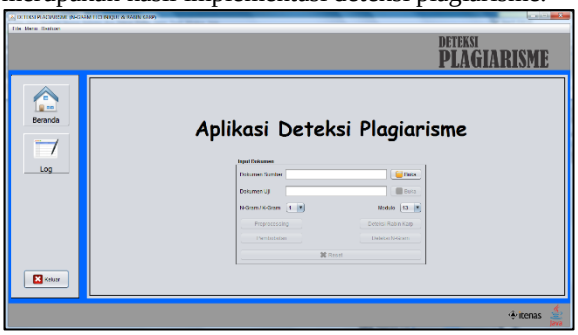
Pada tahap ini dilakukan perhitungan nilai *similarity* atau persentase *similarity*. Berikut ini merupakan perhitungan peresentase *similarity* menggunakan persamaan (4).

$$S = \frac{2 * 23}{38 + 29} = \frac{46}{67} = 0.6866$$

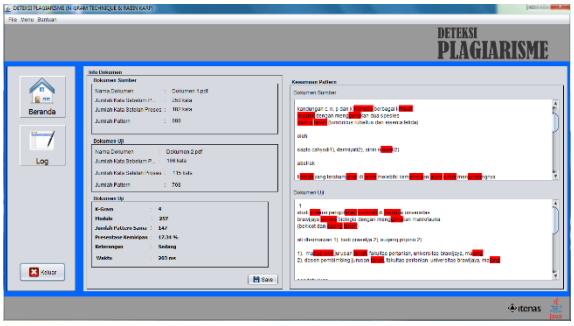
Jadi nilai *similarity* = 0.6866, dan persentase *similarity* = $0.6866 \times 100 = 68.66\%$. Sehingga tingkat kemiripan dokumen persentase *similarity* > 50%, sehingga dokumen sumber dan dokumen sumber memiliki tingkat kemiripan "mendekati plagiarisme".

Hasil Implementasi

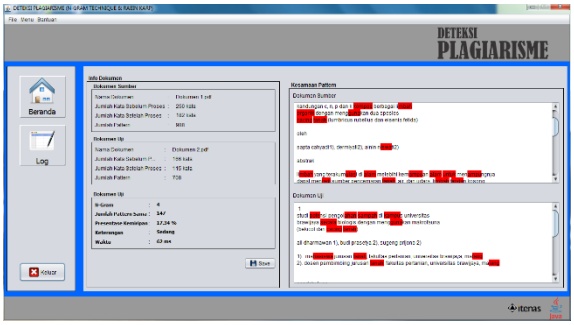
Pada Gambar 3, Gambar 4, dan Gambar 5 merupakan hasil implementasi deteksi plagiarisme.



Gambar 3. Halaman Utama



Gambar 4. Hasil Deteksi N-Gram



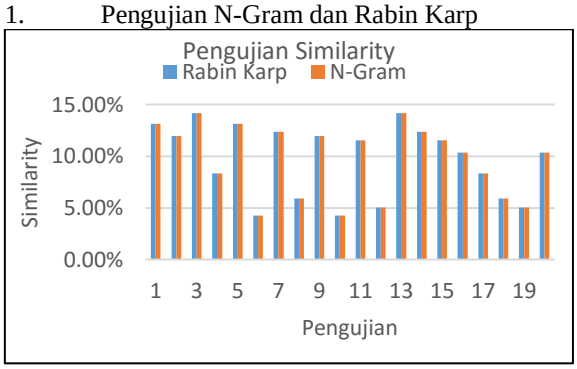
Gambar 5. Hasil Deteksi Rabin Karp

Pengujian

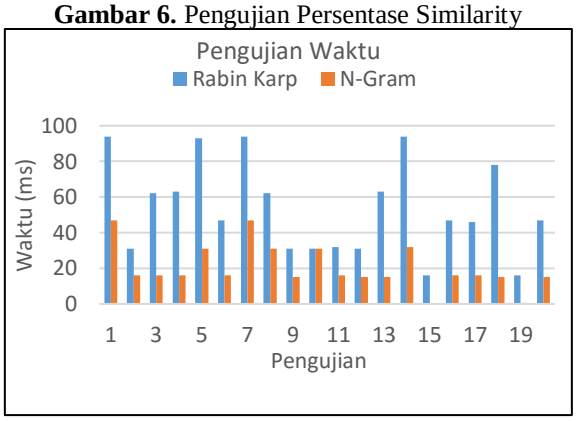
Pengujian yang dilakukan yaitu membandingkan metode N-Gram dan Rabin Karp, membandingkan menggunakan panjang karakter n-

gram dan k-gram, pengujian jumlah kata pada dokumen uji, dan pengujian nilai modulo yang digunakan untuk perhitungan *hash* pada metode Rabin Karp. Berikut ini merupakan hasil pengujian:

1. Pengujian N-Gram dan Rabin Karp



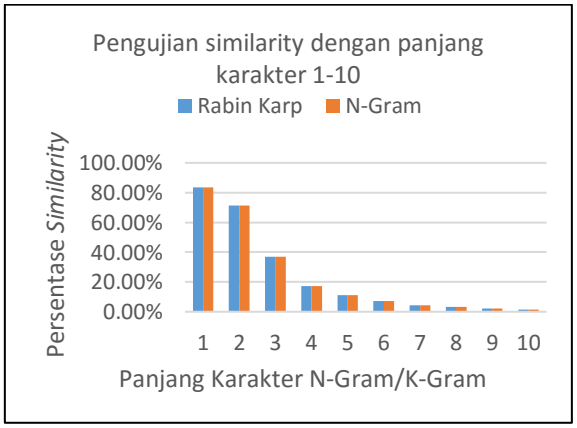
Gambar 6. Pengujian Persentase Similarity



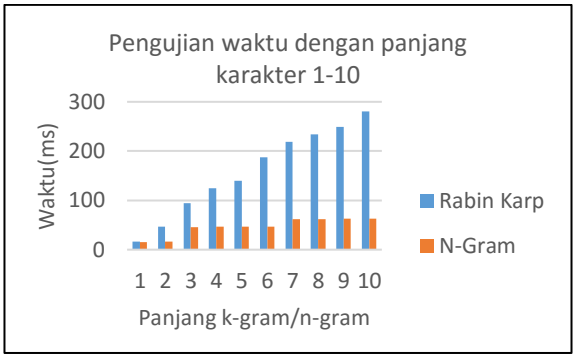
Gambar 7. Pengujian Waktu

Pada Gambar 6 dan Gambar 7 merupakan pengujian N-Gram dan Rabin Karp. Hasil pengujian memiliki nilai *similarity* yang relatif sama, tetapi waktu pendeteksian berbeda. N-Gram memiliki waktu yang relative lebih cepat dari Rabin Karp.

2. Pengujian berdasakan panjang K-Gram dan N-Gram



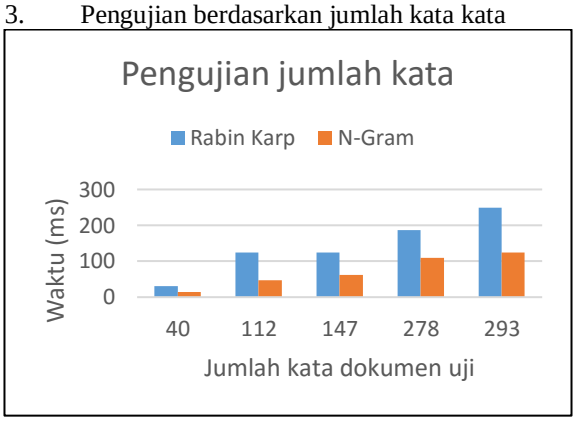
Gambar 8. Pengujian Persentase Similarity Berdasarkan Panjang Karakter N-Gram dan K-Gram



Gambar 9. Pengujian Waktu Berdasarkan Panjang N-Gram dan K-Gram

Pada Gambar 8 dan Gambar 9 merupakan pengujian panjang karakter k-gram dan n-gram yaitu 1-10. Hasil pengujian yang dilakukan bahwa nilai *similarity* lebih kecil panjang karakter, maka nilai *similarity* yang dihasilkan semakin besar dan waktu yang dilakukan semakin bersar.

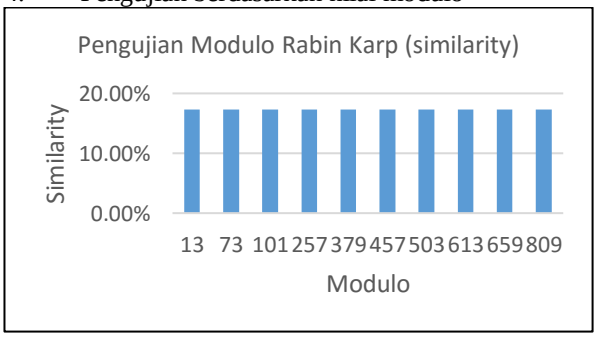
3. Pengujian berdasarkan jumlah kata kata



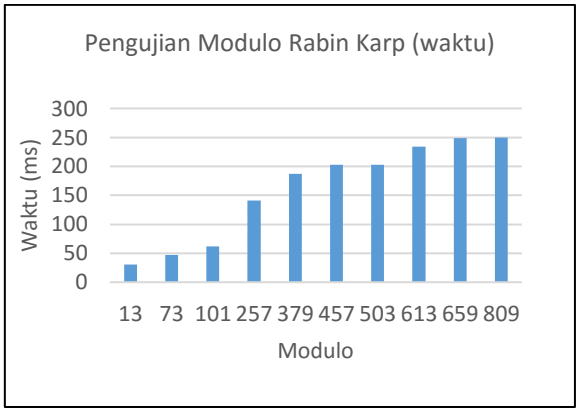
Gambar 10. Pengujian Jumlah Kata

Pada Gambar 10 merupakan pengujian dengan jumlah kata yang berbeda pada dokumen uji. Waktu yang diperlukan dalam pendeteksian semakin banyak kata pada dokumen uji maka semakin lama waktu pendeteksian.

4. Pengujian berdasarkan nilai modulo



Gambar 11. Pengujian Persentase *Similarity* Berdasarkan Nilai Modulo



Gambar 12. Pengujian Waktu Berdasarkan Nilai Modulo

Pada Gambar 11 dan Gambar 12 merupakan pengujian nilai modulo yang digunakan dalam perhitungan *hash*. Nilai modulo yang berbeda tidak berpengaruh terhadap nilai *similariy* tetapi berpengaruh terhadap waktu pendeteksian. Semakin besar nilai modulo maka waktu pendeteksian semakin lama.

4. Kesimpulan

Berdasarkan penelitian yang telah dilakukan, pembobotan TF-IDF dapat dimanfaatkan dan diimplementasikan untuk pendeteksian plagiarisme pada kalimat yang dianggap relevan. Dari hasil pengujian metode N-Gram dan Rabin Karp memiliki nilai *similarity* yang sama berdasarkan grafik Gambar 6, tetapi waktu pendeteksian kedua metode tersebut berbeda. N-Gram memiliki proses pendeteksian yang lebih cepat dibandingkan dengan Rabin Karp berdasarkan grafik Gambar 7. Panjang nilai karakter N-Gram dan K-Gram semakin kecil maka nilai *similarity* semakin besar berdasakan grafik Gambar 8 dan waktu pendeteksian semakin cepat berdasarkan grafik Gambar 9. Jika kata pada dokumen uji semakin banyak waktu pendeteksian semakin lama berdasarkan grafik Gambar 10. Pada metode Rabin Karp nilai modulo tidak berpengaruh terhadap nilai *similarity* tetapi berpengaruh terhadap waktu pendeteksian berdasarkan grafik Gambar 11 dan Gambar 12.

Daftar Pustaka

[1] Astuti, Budi, 2012, *Identifikasi Perilaku Plagiat pada Mahasiswa Fakultas Ilmu Pendidikan, Universitas Negeri Yogyakarta*, Artikel Penelitian, Yogyakarta : Universitas Negeri Yogyakarta.

[2] Dewanto, Sandy, Indriati, Cholissodin, Imam, *Deteksi Plagiarisme Dokumen Teks Menggunakan Algoritma Rabin-Karp dengan Synonym Recognition*. Malang : Universitas Brawijaya Malang

[3] Evan, Fabianus Hendy, P., Y. Sigit Purnomo W., Pranowo, 2014, *Pembangunan Perangkat Lunak Peringkas Dokumen dari Banyak Sumber Menggunakan Sentence Scoring dengan Metode TF-IDF*. Yogyakarta : Universitas Atma Jaya.

[4] Firdaus, Hari Bagus, 2008, *Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin-Karp*. Bandung : Institut Teknologi Bandung

[5] Lisangan, Erick Alfons, 2013, *Implementasi N-Gram Technique dalam Deteksi Plagiarisme Pada Tugas Mahasiswa*. Universitas Atma Jaya Makassar.

[6] Mujahidin, Zainal, 2013, *Implementasi Metode Rabin Karp Untuk Mendeteksi Tingkat Kesamaan Dua Dokumen*, Pekanbaru Riau : Universitas Islam Negeri Sultan Syarif Kasim.

[7] Nugroho, Eko, 2011, *Perancangan Sistem Deteksi Plagiarisme Dokumen Teks dengan Menggunakan Algoritma Rabin-Karp*. Malang : Universitas Brawijaya.

[8] Pardede, Jasman, Alvian, Leo, 2015, *Rancang Bangun Aplikasi Pendeteksi Plagiarisme Menggunakan Algoritma Sherlock*. Bandung : ITENAS

[9] Pardede, Jasman, Tonianto, 2016, *Implementasi Metode Non-Negative Matrix Factorization pada Aplikasi Peringkas Dokumen Bahasa Indonesia*. Bandung : ITENAS

[10] Purwitasari, Diana, Kusmawan, Putu Yuwono, Yuhana, Umi Laili, *Deteksi Keberadaan Kalimat Sama sebagai Indikasi Penjiplakan dengan Algoritma Hashing Berbasis N-Gram*. ITS.

[11] Putra, Gifny Dwi, N., Youllia Indrawaty, 2011, *Pembangunan Aplikasi Pengukuran Tingkat Similaritas Antar Dokumen Berbasis Teks Menggunakan Metode Document Fingerprinting*. Bandung: ITENAS.

[12] Widiantoro, Agustinus, 2014, *Peringkasan Teks Otomatis pada Dokumen Berbahasa Jawa Menggunakan Metode TF-IDF*. Yogyakarta : Universitas Sanata Dharma.

[13] Yoga, Kadek Versi Yana, 2012, *Pengembangan Aplikasi Pendeteksi Plagiarisme Pada Dokumen Teks Menggunakan Aalgoritma Rabin-Karp*. Universitas Pendidikan Ganesha.

[14] ___, 2015, *Plagiarisme*, (online), (https://id.wikipedia.org/wiki/Plagiarisme, diakses pada tanggal 13 Maret 2016).