

RANCANG BANGUN APLIKASI PENDETEKSIAN KESAMAAN PADA DOKUMEN TEKS MENGUNAKAN ALGORITMA *ENHANCED CONFIX STRIPPING* DAN ALGORITMA *WINNOWING*

Muhammad Nur Khidfi^{*1}, Isnawaty², Jayanti Yusma Sari³

^{*1,2,3}Jurusan Teknik Informatika, Fakultas Teknik Universitas Halu Oleo, Kendari

^{*1}kidfi23@gmail.com, ²isna.1711@gmail.com, ³jayanti@uho.ac.id

Abstrak

Dengan semakin berkembangnya teknologi, maka semakin meningkat pula tindak plagiarisme di dunia perkuliahan. Banyak mahasiswa yang melakukan plagiarisme misalnya dalam mengerjakan tugas kuliah dan proposal tugas akhir. Oleh karena itu, dibutuhkan aplikasi yang mampu mendeteksi tingkat kesamaan (*similarity*) antara dokumen teks. Pada penelitian ini menggunakan algoritma *Enhanced Confix Stripping* (ECS) *Stemmer* untuk proses *stemming* teks dan algoritma *Winnowing* untuk menghitung tingkat kesamaan (*similarity*) antar dokumen. Dengan ditentukannya nilai *gram* dan *window* pada perhitungan algoritma *Winnowing*, memudahkan user untuk menggunakan aplikasi tanpa harus bingung untuk menentukan nilai *gram* dan *window*-nya untuk menghasilkan nilai *similarity* yang akurat. Dari hasil pengujian 5 pasang bab 1 tugas akhir mahasiswa yang berkategori sama menghasilkan nilai *similarity* sekitar 45-20%.

Kata kunci : *Enhanced Confix Stripping* (ECS) *Stemmer*, *Winnowing*, *Similarity*, *Plagiarisme*.

Abstract

With the development of technology, the increase of acts of plagiarism in the world of education. Many students who did plagiarism in doing their task and thesis as example. Therefore, an application that is able to detect the similarity between text documents is required. This research used Enhanced Confix Stripping (ECS) Stemmer algorithm for text stemming process and Winnowing algorithm to calculate similarity between documents. By specifying the gram and window values of the Winnowing algorithm, it made easier for the user to use the application without being confused to determine the value of the gram and the window to produce an accurate equality value. From the test results 5 pairs of chapters 1 thesis of students who categorized together produced a similarity score of about 45-20%.

Keywords : *Enhanced Confix Stripping* (ECS) *Stemmer*, *Winnowing*, *Similarity*, *Plagiarisme*.

1. PENDAHULUAN

Plagiarisme telah menjadi masalah yang semakin berkembang khususnya dalam dunia pendidikan. Banyak sekali karya tulis mahasiswa seperti tugas akhir, tugas kuliah dan lain-lain yang sebagian besar isinya dibuat dengan menciplak milik orang lain. Setiap karya memiliki kekhasan penulisannya masing-masing tergantung karakter dari setiap penulis. Keaslian suatu karya dapat ditunjukkan pula dengan adanya *copyright* dari pemilik tulisan itu sendiri [1].

Dengan semakin pesatnya perkembangan teknologi saat ini, juga sangat mempengaruhi tingkat plagiarisme dalam dunia pendidikan. Teknologi dimanfaatkan oleh sebagian mahasiswa dalam menyelesaikan tugas kuliah dengan melakukan *copy-paste*. Dengan tingginya tingkat plagiarisme maka dibutuhkan sebuah aplikasi yang dapat mendeteksi tingkat similaritas dari tugas-tugas mahasiswa.

Pada dasarnya sistem pendeteksian teks harus memiliki 3 unsur utama yang harus dipenuhi, yaitu : *whitespace instesity*, *noise*

suppression, dan *position independence*. Salah satu algoritma yang memenuhi unsur tersebut adalah Algoritma *Winnowing*. Dalam proses pendeteksian dibutuhkan proses *stemming* untuk mengubah suatu kata menjadi kata dasar. Salah satu algoritma *stemming* untuk kata berbahasa Indonesia adalah Algoritma *Enhanced Confix Stripping Stemmer*. Algoritma *ECS* merupakan pengembangan dari Algoritma Nazief dan Adriani. Proses *stemming* sering dilakukan dalam proses pencarian teks, aplikasi kamus, dan mesin pencari. Algoritma *winnowing* dan teknik *rolling hash* digunakan untuk membuang seluruh pemakaian karakter yang tidak relevan seperti, tanda baca, spasi, angka, dan karakter lainnya. Hanya karakter berupa huruf yang akan diproses ke tahap berikutnya. Ditahap berikutnya dokumen teks akan diberi nilai fungsi *hash*. Nilai tersebut akan dibandingkan dengan dokumen teks lainnya untuk diberi berapa persen nilai similaritas antara kedua dokumen tersebut [2].

Berdasarkan latar belakang tersebut, pada penelitian ini penulis akan menerapkan Algoritma *Enhanced Confix Stripping Stemmer* dan Algoritma *Winnowing* pada dokumen teks. Dengan penerapan algoritma tersebut, diharapkan dapat menentukan nilai similaritas antara 2 dokumen teks sehingga dosen dapat melihat tingkat similaritas antara tugas mahasiswa dan dapat mengurangi tingkat plagiarisme yang terjadi antara mahasiswa.

2. METODE PENELITIAN

2.1 Algoritma Stemming Bahasa Indonesia

Stemming merupakan bagian dari proses *Information Retrieval* (IR), yang mengubah beberapa kata ke bentuk kata dasarnya sebelum dilakukan pengindeksan. Contoh, kata dibaca, membaca, pembaca, akan diubah ke kata dasarnya, yaitu "baca" [3].

Pada dasarnya proses *stemming* bekerja tergantung pada bahasa yang diteliti. Khusus untuk topik berbahasa Indonesia, proses algoritma *stemming* awalnya diperkenalkan oleh Nazief dan Adriani pada tahun 1996. Algoritma ini bekerja berdasarkan struktural morfologi kalimat bahasa Indonesia, yang terdiri dari prefix (awalan), sufiks (akhiran), infiks (sisipan), dan konfiks (awalan+akhiran). Lalu Asian pada tahun 2007 mengembangkan

algoritma *stemming* tersebut dengan menambah beberapa aturan, dan diperkenalkan dengan nama algoritma *Confix Stripping Stemmer*. Penelitian selanjutnya dilakukan oleh I Putu Adhi Kerta Mahendra pada tahun 2008 dengan menambahkan kamus kata dasar dan mendukung *recording*, yakni penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih. Algoritma terbaru yang diteliti oleh Mahendra, selanjutnya dinamakan algoritma *Enhanced Confix Stripping Stemmer*.

2.2 Algoritma Enhanced Confix Stripping Stemmer

Proses *stemming* untuk bahasa Indonesia dengan performa yang paling baik adalah dengan menggunakan algoritma *Enhanced Confix Stripping (ECS) Stemmer*. Algoritma ini merupakan pengembangan dari algoritma *Confix Stripping (CS) Stemmer*, dan berhasil mereduksi jumlah *term* pada algoritma *Confix Stripping Stemmer* hingga 32.66%, sedangkan pada awalnya *Confix Stripping Stemmer* hanya mampu mereduksi 30.95% *term* [4].

Tahapan kerja algoritma *Enhanced Confix Stripping Stemmer* yaitu :

1. Aturan Dasar pada Tabel 1., jika *input* kata sesuai dengan pasangan yang ada, maka lakukan penghilangan awalan terlebih dahulu. Jika tidak ada, maka penghilangan akhiran dilakukan terlebih dahulu.
2. *Recording* (penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih) apabila diperlukan.
3. Dilakukan *loop Pengembalian Akhiran*
4. Dilakukan pengecekan apakah terdapat tanda hubung ("-") yang menandakan *input* kata tersebut adalah kata ulang. Jika benar, maka lakukan proses *stemming* pada potongan kata di sebelah kiri dan kanan tanda hubung. Apabila hasil *stemming* memberikan hasil yang sama, maka kata dasar dari kata ulang tersebut adalah hasil yang didapatkan.
5. Jika keempat proses di atas gagal, maka *input* kata yang di-*stemming* dianggap sebagai kata dasar.

Pada setiap langkah dilakukan proses pengecekan *output stemming* ke kamus data. Apabila ditemukan, maka proses berhenti.

Proses *loop Pengembalian Akhiran* bekerja seperti berikut:

1. Mengembalikan seluruh awalan yang telah dihilangkan, sehingga menghasilkan model kata seperti: $[DP+[DP+[DP]]]$ + Kata Dasar. Pemenggalan awalan dilanjutkan dengan proses pencarian di kamus.
2. Pegembalian akhiran sesuai urutan pada Tabel 2. Untuk setiap pengembalian, lakukan langkah 3) hingga 5) berikut. Khusus untuk akhiran “-kan”, pengembalian pertama dimulai dengan “k”, lalu dilanjutkan dengan “an”.
3. Dilakukan pengecekan ke kamus data. Apabila kata dasar ditemukan, proses dihentikan. Apabila gagal, maka lakukan proses pemenggalan awalan berdasarkan aturan pada Tabel 3.
4. Dilakukan *recording* jika diperlukan.
6. Apabila pengecekan di kamus tetap gagal setelah *recording*, maka awalan yang telah dihilangkan dikembalikan lagi. Aturan Dasar pada Tabel 1, jika *input* kata sesuai dengan pasangan yang ada, maka lakukan penghilangan awalan terlebih dahulu. Jika tidak ada, maka penghilangan akhiran dilakukan terlebih dahulu.
7. *Recording* (penyusunan kembali kata-kata yang mengalami proses *stemming* berlebih) apabila diperlukan.
8. Dilakukan *loop* *Pengembalian Akhiran*
9. Dilakukan pengecekan apakah terdapat tanda hubung (“-”) yang menandakan *input* kata tersebut adalah kata ulang. Jika benar, maka lakukan proses *stemming* pada potongan kata di sebelah kiri dan kanan tanda hubung. Apabila hasil *stemming* memberikan hasil yang sama, maka kata dasar dari kata ulang tersebut adalah hasil yang didapatkan.
10. Jika keempat proses di atas gagal, maka *input* kata yang di-*stemming* dianggap sebagai kata dasar.

Pada setiap langkah dilakukan proses pengecekan *output stemming* ke kamus data. Apabila ditemukan, maka proses berhenti.

Proses *loop Pengembalian Akhiran* bekerja seperti berikut:

1. Mengembalikan seluruh awalan yang telah dihilangkan, sehingga menghasilkan model kata seperti: $[DP+[DP+[DP]]]$ + Kata Dasar. Pemenggalan awalan dilanjutkan dengan proses pencarian di kamus.

2. Pegembalian akhiran sesuai urutan pada Tabel 2. Untuk setiap pengembalian, lakukan langkah 3) hingga 5) berikut. Khusus untuk akhiran “-kan”, pengembalian pertama dimulai dengan “k”, lalu dilanjutkan dengan “an”.
3. Dilakukan pengecekan ke kamus data. Apabila kata dasar ditemukan, proses dihentikan. Apabila gagal, maka lakukan proses pemenggalan awalan berdasarkan aturan pada Tabel 3.
4. Dilakukan *recording* jika diperlukan.

Apabila pengecekan di kamus tetap gagal setelah *recording*, maka awalan yang telah dihilangkan dikembalikan lagi.

Tabel 1 Aturan Dasar Awalan – Akhiran yang Berlaku

Pasangan Awalan – Akhiran yang Berlaku
Be – lah
Be – an
Me – i
Di – i
Pe – i
Te – i

Tabel 2 Urutan Pengembalian Akhiran

No	Akhiran	Tipe
1	-i, -kan, -an	Derivation Suffixes (DS)
2	-ku, -mu, -nya	Possessive Pronoun (PP)
3	-lah, -kah, -tah, - pun	Inflectional Particle (P)

Tabel 3 Aturan Pemenggalan Awalan Algoritma Stemmer Nazief dan Adriani

No	Format Kata	Pemenggalan
1	berV...	ber-V... be-r-V ...
2	berCAP...	ber-CAP... dimana C!=‘r’ & P!=‘er’
3	berCAerV...	ber-CAerV... dimana C!=‘r’
4	Belajar	bel-ajar
5	beC ₁ erC ₂ ...	be-C ₁ erC ₂ ... dimana C ₁ !={‘r’ ‘l’}
6	terV...	ter-V... te-rV
7	terCerV...	ter-CerV... dimana C!=‘r’

8	TerCP	ter-CP... dimana C!= 'r' dan P!= 'er'
9	teC ₁ erC ₂	te-C ₁ erC ₂ ... dimana C ₁ != 'r'
10	me{l r w y}V...	me-{l r w y}V...
11	mem{b f v}...	mem-{b f v}...
12	mempe{r l}	mem-pe...
13	mem{rV V}...	me-m{rV V}... me-p{rV V}...
14	men{c d j z}...	men-{c d j z}...
15	menV...	me-nV... me-tV...
16	meng{g h q}...	meng-{g h q}...
17	mengV...	meng-V... meng-kV...
18	menyV...	meny-sV...
19	mempV...	mem-pV... dimana V!= 'e'
20	pe{w y}V...	pe-{w y}V...
21	perV...	per-V... pe-rV...
22	perCAP...	per-CAP... dimana C!= 'r' dan P!= 'er'
23	perCAerV...	per-CAerV... dimana C!= 'r'
24	pem{b f V}...	pem-{b f V}...
25	pem{rV V}...	pe-m{rV V}... pe-p{rV V}...
26	pen{c d j z}...	pen-{c d j z}...
27	penV...	pe-nV... pe-tV...
28	peng{g h q}...	peng-{g h q}...
29	pengV...	peng-V... peng-kV...
30	penyV...	peny-sV...
31	pelV...	pe-lV... kecuali "pelajar" yang menghasilkan "ajar"
32	peCerV...	per-erV... dimana C!= {r w y l m n}...
33	peCP...	pe-CP... dimana C!= {r w y l m n} dan P!= 'er'

Pada Tabel 3 dan Tabel 4, simbol C merupakan konsonan, simbol V merupakan vokal, simbol A merupakan vokal atau konsonan, dan simbol P merupakan partikel atau fragmen dari suatu kata, misalnya "er". Dari kedua tabel dapat dilihat beberapa perbedaan. Awalan yang diikuti huruf awal pada setiap kata dasar telah dikelompokkan menjadi kumpulan konsonan, vokal, atau partikel. Seperti, aturan no.29 pada awalan algoritma *Stemmer* Nazief dan Adriani, pemenggalan awalan "peng-{g|h|q}" telah

dikelompokkan menjadi "peng-C" pada awalan algoritma *Enhanced Confix Stripping Stemmer*.

Tabel 4 Aturan Pemenggalan Awalan Algoritma *Enhanced Confix Stripping Stemmer*

No	Format Kata	Pemenggalan
1	berV...	ber-V... be-r-V ...
2	berCAP...	ber-CAP... dimana C!= 'r' & P!= 'er'
3	berCAerV...	ber-CAerV... dimana C!= 'r'
4	Belajar	bel-ajar
5	beC ₁ erC ₂ ...	be-C ₁ erC ₂ ... dimana C ₁ != { 'r' 'l' }
6	terV...	ter-V... te-rV
7	terCerV...	ter-CerV... dimana C!= 'r'
8	TerCP	ter-CP... dimana C!= 'r' dan P!= 'er'
9	teC ₁ erC ₂	te-C ₁ erC ₂ ... dimana C ₁ != 'r'
10	me{l r w y}V...	me-{l r w y}V...
11	mem{b f v}...	mem-{b f v}...
12	mempe	mem-pe...
13	mem{rV V}...	me-m{rV V}... me-p{rV V}...
14	men{c d j s z}...	men-{c d j s z}...
15	menV...	me-nV... me-tV...
16	meng{g h q k}...	meng-{g h q k}...
17	mengV...	meng-V... meng-kV... (meng-V... jika V='e')
18	menyV...	meny-sV...
19	mempA...	mem-pA... dimana A!= 'e'
20	pe{w y}V...	pe-{w y}V...
21	perV...	per-V... pe-rV...
22	perCAP...	per-CAP... dimana C!= 'r' dan P!= 'er'
23	perCAerV...	per-CAerV... dimana C!= 'r'
24	pem{b f V}...	pem-{b f V}...
25	pem{rV V}...	pe-m{rV V}... pe-p{rV V}...
26	pen{c d j z}...	pen-{c d j z}...
27	penV...	pe-nV... pe-tV...
28	PengC	peng-C...
29	pengV...	peng-V... peng-kV... (pengV-... jika V='e')
30	penyV...	peny-sV...

31	peIV...	pe-IV... kecuali "pelajar" yang menghasilkan "ajar"
32	peCerV...	per-erV... dimana $C! = \{r w y l m n\}...$
33	peCP...	pe-CP... dimana $C! = \{r w y l m n\}$ dan $P! = "er"$
34	terC ₁ erC ₂ ...	ter-C ₁ erC ₂ ... dimana $C_1! = "r"$
35	peC ₁ erC ₂	pe-C ₁ erC ₂ ... dimana $C_1 = \{r w y l m n\}$

2.3 Algoritma *Winnowing*

Salah satu algoritma yang digunakan untuk mendeteksi bentuk kesamaan pada dokumen teks adalah algoritma *Winnowing*. Pada dasarnya sistem pendeteksian haruslah memiliki 3 unsur utama yang harus dipenuhi, seperti [5] :

1. *Whitespace insensitivity*, sistem pencocokan teks seharusnya tidak terpengaruh pada spasi, adanya huruf kapital, berbagai tanda baca, dan sebagainya;
2. *Noise surpression*, sistem haruslah menghindari pencocokan kata yang terlalu pendek;
3. *Position independence*, sistem seharusnya tidak bergantung pada posisi kata yang dicari sehingga apabila ditemukan kata yang terindeksi sama dengan posisi berbeda masih dapat dikenali;

Algoritma *Winnowing* dipilih karena algoritma ini sudah memenuhi unsur untuk proses pendeteksian. Implementasi dari algoritma *Winnowing* membutuhkan masukan berupa *file* teks dan menghasilkan keluaran berupa nilai *hash* yang disebut *finger print* [2]. Setiap kata yang terkandung dalam *file* teks diubah terlebih dahulu menjadi sebuah kumpulan nilai *hash* dengan teknik *rolling hash*. Nilai *hash* merupakan nilai numerik dari perhitungan ASCII untuk setiap karakter. Lalu kumpulan nilai *hash* yang disebut *finger print* tersebut digunakan untuk mendeteksi kemiripan antar dokumen [6].

1) Rolling Hash

Teknik *Rolling Hash* pada awalnya digunakan pada algoritma *Rabin-Karp*. Setiap karakter di dalam dokumen teks diubah (*encode*) menjadi nilai *array* bilangan bulat, sehingga nilai masukan yang awalnya berupa

karakter menjadi fungsi *hash* berupa angka. Untuk membandingkan dua *string* yang dianggap sama, maka setiap $A[i] = B[i]$ dan membutuhkan waktu sebesar $O(n)$. Panjang waktu yang dibutuhkan tergantung pada panjang iterasi elemen *string* yang dibandingkan [7].

Metode dasar untuk mencari perbandingan antara kedua *string* dokumen A dan B adalah:

- a. Asumsikan dokumen A memiliki panjang elemen *string* p , dan dokumen B memiliki panjang q .
- b. Lakukan *hashing* pada dokumen A untuk mendapatkan $h(A)$ dengan waktu sebesar $O(p)$.
- c. Lakukan iterasi pada dokumen B dengan panjang elemen *string* p , dan bandingkan $h(A)$ dengan waktu sebesar $O(qp)$.
- d. Jika nilai *hash substring* tidak cocok dengan $h(A)$, bandingkan *substring* yang ada dengan A. Jika cocok, berhenti, jika tidak, lakukan kembali hingga ditemukan waktu sebesar $O(p)$.

Untuk mengurangi waktu komputasi, dapat dilakukan teknik *rolling hash* dengan mengambil waktu sebesar $O(p)$ sehingga didapatkan banyak kecocokan. Contoh, lakukan *hashing 5 substring* pada kata "komputer". Hash I: "kompu", hash II: "omput", dan seterusnya. Dengan teknik *rolling hash*, maka didapatkan bahwa kedua *hash* yang saling dibandingkan akan menghasilkan *substring* yang sama, yaitu: "ompu" dan berlaku untuk perbandingan hasil *hash* berikutnya.

Digunakannya perhitungan operasi modulo agar tidak mempersulit system menghitung dalam jumlah banyak, selama nilai modulo yang digunakan tidak terlalu besar pula [8].

Persamaan (1) menunjukkan teknik *rolling hash* [7].

$$h(k) = (k[0]b^{L-1} + k[1]b^{L-2} + k[2]b^{L-3} + k[3]b^{L-4}) \quad (1)$$

Untuk menghitung *hash* lanjutan digunakan Persamaan (2).

$$h(S_{i+1}) = b(h(S_i) - b^{L-1}S[i]) + S[i+L] \quad (2)$$

Dimana:

b : Nilai bilangan basis (10)

k : Nilai ASCII karakter

$h(k)$: Nilai *hash*

m : Nilai bilangan prima (10007)

L : Banyaknya karakter yang di-*hashing*

$S(i)$: Nilai *hash* awal

$S(i+1)$: Nilai *hash* berikutnya

2) Tahapan Penerapan Algoritma *Winnowing*
Beberapa tahapan dalam penerapan algoritma *Winnowing* adalah sebagai berikut [2] :

1. Tahap Pertama : Membuang karakter yang tidak relevan seperti tanda baca, spasi, dan simbol-simbol lainnya.
2. Tahap Kedua : Membentuk rangkaian *gram*.
3. Tahap Ketiga : Melakukan proses *rolling hash* untuk mencari nilai *hash* dari setiap *gram*.
4. Tahap Keempat : Membentuk *window* yang terdiri dari nilai *hash* yang dihasilkan.
5. Tahap Kelima : Membentuk nilai *finger print* yang unik, dengan memilih nilai terendah dari setiap baris di dalam *window*.

3) Pengukuran dan Persentase *Similarity*

Perhitungan similaritas antar dua dokumen diambil dari pemilihan nilai *finger print hash* terunik, ditunjukkan oleh Persamaan (3) [9].

$$S = \frac{Nt}{(Nx + Ny) - Nt} \quad (3)$$

Keterangan:

S : Similaritas

Nt : Total hash yang sama

Nx : Total substring pembandingan

Ny : Total substring uji

Penilaian persentase similaritas antar dua dokumen yang dibandingkan adalah sebagai berikut:

1. Kategori Nihil (0%)
Kedua dokumen tidak terindikasi plagiat karena benar-benar berbeda baik dari segi isi dan kalimat secara keseluruhan.
2. Kategori Sedikit Kesamaan (<15%)
Kedua dokumen hanya mempunyai sedikit kesamaan.
3. Kategori Plagiat Sedang (15-50%)
Kedua dokumen terindikasi plagiat tingkat sedang.
4. Kategori Mendekati Plagiarisme (>50%)
Hasil uji menunjukkan lebih dari 50%, dapat dikatakan bahwa dokumen yang diuji mendekati tingkat plagiarisme.

5. Kategori Plagiarisme (100%)

Dokumen uji dapat dipastikan murni plagiat karena dari awal dan sampai akhir isi dokumen adalah sama.

3. HASIL DAN PEMBAHASAN

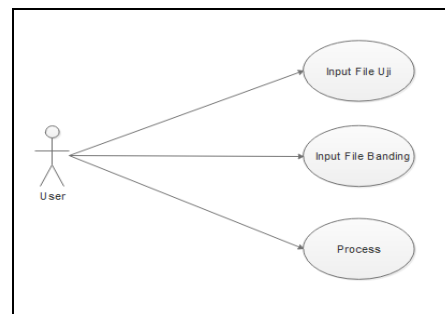
3.1 Gambaran Umum

Gambaran umum aplikasi pendeteksi kesamaan pada dokumen teks menggunakan algoritma *Enhanced Confix Stripping* (ECS) *Stremmer* dan algoritma *Winnowing* untuk menemukan berapa nilai *similarity* antara 2 dokumen teks.

3.2 Rancangan Sistem

Rancangan Sistem untuk aplikasi pendeteksi kesamaan pada dokumen teks menggunakan algoritma *Enhanced Confix Stripping* (ECS) *Stremmer* dan algoritma *Winnowing*, menggunakan *Unified Modelling Language* (UML). Dalam penelitian ini, penulis menyajikan rancangan sistem menggunakan 2 diagram yaitu Diagram *Use Case* dan Diagram *Activity*.

1. *Use Case* diagram digunakan untuk memodelkan dan menyatakan fungsi yang disediakan oleh sistem. Diagram *Use Case* ditunjukkan pada Gambar 1.



Gambar 1 Diagram *Use Case*

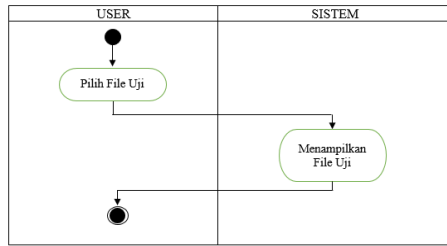
2. Diagram *Activity*

Diagram *Activity* memodelkan proses-proses yang terjadi pada sistem. Diagram *Activity* ditunjukkan pada Gambar 2, Gambar 3 dan Gambar 4.

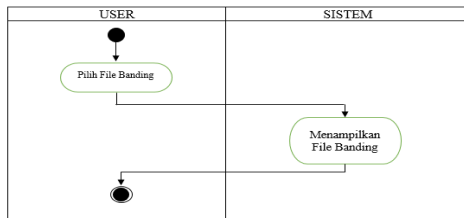
3.3 Implementasi Sistem

Pada tahap ini akan diperlihatkan tampilan antarmuka (*interface*) dan sistem yang berjalan dibelakang *interface* dari aplikasi pendeteksi kesamaan pada dokumen teks menggunakan *enhanced confix stripping stremmer* dan *winnowing*. Aplikasi

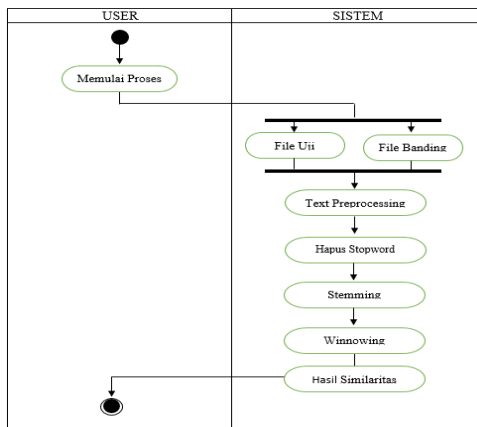
ini memiliki 1 *form* dimana semua proses terjadi *form* tersebut. *Interface* dari aplikasi pendeteksi kesamaan pada dokumen teks ditunjukkan pada Gambar 5.



Gambar 2 Diagram Activity File Uji



Gambar 3 Diagram Activity File Banding



Gambar 4 Diagram Activity Process



Gambar 5 Tampilan Interface

3.4 Pengujian Kinerja Sistem

Pengujian sistem dilakukan untuk mengetahui apakah sistem berjalan sesuai dengan rancangan yang telah dibuat sebelumnya. Proses pengujian akan dilakukan dengan membandingkan beberapa dokumen bab 1 dari tugas akhir mahasiswa Teknik Informatika UHO, dimana data dokumen uji yang akan dibandingkan dengan data dokumen banding merupakan dari kategori yang sama. Dilakukan 1 pengujian yang menggunakan dokumen uji dan dokumen banding yang sama. Dokumen uji, dokumen banding dan hasil perbandingan ditunjukkan pada Tabel 5, Tabel 6 dan Tabel 7.

Tabel 5 Dokumen uji

No	Judul Tugas Akhir	Kategori	Size	Eks.
1	Enkripsi video MPEG-1 dengan Algoritma Video Encryption Algorithm (VEA)	Enkripsi	75 KB	*.doc
2	Implementasi Analytical Network Process Dalam Sistem Pendukung Keputusan Penerimaan Karyawan Pada PT. Media Kita Sejahtera	SPK	21 KB	*.docx
3	Sistem Peramalan Persediaan Obat dengan Metode Weight Moving Average dan Reorder Point	JST	28 KB	*.docx
4	Perancangan Game Edukasi Ular Tangga Dengan Metode Algoritma Fisher - Yates Shuffle	Game	289 KB	*.pdf
5	Analisis Quality of Service (QoS) Jaringan Internet Berbasis Wireless LAN Pada Layanan IndiHome	jaringan	272 KB	*.pdf

Tabel 6 Dokumen Banding

No	Judul Tugas Akhir	Kategori	Size	Eks.
1	Enkripsi video MPEG-1 dengan Algoritma Video Encryption Algorithm (VEA) - copy	Enkripsi	75 KB	*.doc
2	Implementasi Algoritma Serpent pada Enkripsi Video MPEG	Enkripsi	58 KB	*.doc
3	Sistem Pendukung Keputusan Penilaian Calon Debitur Penerima Kredit Modal Kerja Bank Sultra Menggunakan Credit Scoring	SPK	163 KB	*.pdf
4	Perancangan Aplikasi Data Warehouse Menggunakan Metode FP-GROWTH Untuk Memprediksi Penjualan Alat-Alat Kesehatan (Studi Kasus Apotek Kimia Farma Korem)	JST	25 KB	*.docx
5	Rancang Bangun Game Edukasi Puzzle Memperkenalkan Kebudayaan Sulawesi Tenggara Dengan	Game	38 KB	*.pdf

	Algoritma Fisher-Yates Shuffle			
6	Analisis Quality Of Service (QoS) Kinerja Sistem Hotspot Pada Routerboard Mikrotik 951ui-2hnd Pada Jaringan Teknik Informatika	Jaringan	288 KB	*.pdf

Tabel 7 Hasil Uji

Dokumen Uji	Dokumen Banding	Similarity (%)	Lama Proses (s)
Enkripsi video MPEG-1 dengan Algoritma Video Encryption Algorithm (VEA)	Enkripsi video MPEG-1 dengan Algoritma Video Encryption Algorithm (VEA) – copy	100	54.23
Enkripsi video MPEG-1 dengan Algoritma Video Encryption Algorithm (VEA)	Implementasi Algoritma Serpent pada Enkripsi Video MPEG	35,93	56.65
Implementasi Analytical Network Process Dalam Sistem Pendukung Keputusan Penerimaan Karyawan Pada PT. Media Kita Sejahtera	Sistem Pendukung Keputusan Calon Debitur Penerima Kredit Modal Kerja Bank Sultra Menggunakan Credit Scoring	22,6	39.9
Sistem Peramalan Persediaan Obat dengan Metode Weight Moving Average dan Reorder Point	Perancangan Aplikasi Data Warehouse Menggunakan Metode FP-GROWTH Untuk Memprediksi Penjualan Alat-Alat Kesehatan (Studi Kasus Apotek Kimia Farma Korem)	21,23	44.02
Perancangan Game Edukasi Ular Tangga Dengan Metode Algoritma Fisher - Yates Shuffle	Rancang Bangun Game Edukasi Puzzle Memperkenalkan Kebudayaan Sulawesi Tenggara Dengan Algoritma Fisher-Yates Shuffle	24.95	42.02
Analisis Quality of Service (QoS) Jaringan Internet Berbasis Wireless LAN Pada Layanan IndiHome	Analisis Quality Of Service (QoS) Kinerja Sistem Hotspot Pada Routerboard Mikrotik 951ui-2hnd Pada Jaringan Teknik Informatika	43,39	36.64

4. KESIMPULAN

Dari penelitian dan pembahasan Rancang Bangun Aplikasi Pendeteksi Kesamaan pada Dokumen Teks Menggunakan Algoritma *Enhanced Confix Stripping* (ECS) *Stremmer* dan algoritma *Winnowing*, dapat ditarik kesimpulan sebagai berikut :

1. Sistem dapat melakukan pendekteksian pada dokumen teks dengan format .doc, .docx, dan .pdf (*nonsecured*).
2. Lama proses pendekteksian tidak terpengaruh oleh ukuran dan ekstensi dokumen, melainkan tergantung oleh banyaknya jumlah kata yang terdapat pada dokumen.

5. SARAN

Untuk penelitian selanjutnya, saran yang dapat diberikan yaitu :

1. Sistem dapat dilakukan untuk beberapa opsi dokumen teks lainnya dengan metode *parsing* yang lebih baik.
2. Penelitian selanjutnya dilakukan dengan metode lain yang memungkinkan penggunaan waktu yang lebih minim dan dapat menampilkan *highlight* kalimat/paragraf yang sama.

DAFTAR PUSTAKA

- [1] Sonneborn, L. 2011. *Frequently Asked Questions About Plagiarism*. The Rosen Publishing Group, Inc.: New York (google-books)
- [2] Purwitasari, D. Kusmawan, P.Y. dan Yuhana, U.L. 2011. *Deteksi Keberadaan Kalimat Sama Sebagai Indikasi Penjiplakan Dengan Algoritma Hashing Berbasis NGram*. Jurnal Kursor Menuju Solusi Teknologi Informasi 6(1): 37-44.
- [3] Peng, F., Ahmed, N., Li, X. dan Lu, Y. 2007. *Context Sensitive Stemming for Web Search. Domain Specific NLP*. Sunnyvale, California.
- [4] Mahendra, I P. A. K., Arifin, A. Z. dan Ciptaningtyas, H. T. 2008. *Enhanced Confix Stripping Stemmer And Ants Algorithm For Classifying News Document In Indonesian Language*. International Conference on Information & Communication Technology and System (ICTS) ISSN 2085-1944:149-157.
- [5] Schleimer, S., Wilkerson, D., dan Aiken, A. 2003. *Winnowing: Local Algorithms for Document Printing*. Proceedings of

- the ACM SIGMOD International Conference on Management of Data, pp. 76-85.
- [6] Aziz, I. W., Hermawan dan Cahyani, A. D. 2012. *Pengembangan Mesin Pencarian Antiplagiasi Pada SIM Jurnal Mahasiswa Menggunakan Algoritma Winnowing Fuzzy K-Means*. Jurnal Sarjana Teknik Informatika. 1(1): 1-10.
- [7] Cormen, T. H., Leiserson, C. E., Rivest, R. L. dan Stein, C. 2009. *Introduction to Algorithms 3rd Ed*. The MIT Press: United States of America.
- [8] Ellard, D. 1997. The Rabin-Karp Algorithm. From <http://ellard.org/dan/www/Q-97/HTML/root/node43.html>. 1 Desember 2017.
- [9] Taufik, D. A. 2012. *Sistem Pengukuran Tingkat Similaritas Dokumen*. Skripsi. Universitas Komputer Indonesia.
-

