

**PENGUJIAN DETEKSI TINGKAT PLAGIARISME
DENGAN MEMPERTIMBANGKAN MAKNA KATA DENGAN
MENGUNAKAN ALGORITMA
*LEVENSHTAIN DISTANCE***

TUGAS AKHIR

Oleh:

DENNY GUNAWAN

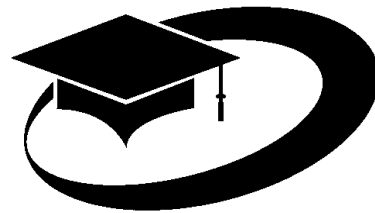
NIM. 12.111.1362

RIVALDI WARMAN

NIM. 12.111.3223

WISELY JANSEN HADI

NIM. 12.111.1044



**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
MIKROSKIL
MEDAN
2016**

**PLAGIARISM DETECTION TESTING
WITH SPECULATING THE MEANING OF THE WORD
BY USING *LEVENSHTAIN DISTANCE* ALGORITHM**

FINAL RESEARCH

By:

DENNY GUNAWAN

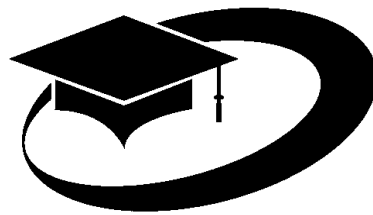
NIM. 12.111.1362

RIVALDI WARMAN

NIM. 12.111.3223

WISELY JANSEN HADI

NIM. 12.111.1044



**STUDY PROGRAM OF INFORMATICS ENGINEERING
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

MIKROSKIL

MEDAN

2016

LEMBAR PENGESAHAN
PENGUJIAN DETEKSI TINGKAT PLAGIARISME
DENGAN MEMPERTIMBANGKAN MAKNA KATA DENGAN
MENGGUNAKAN ALGORITMA
LEVENSHTEIN DISTANCE

TUGAS AKHIR

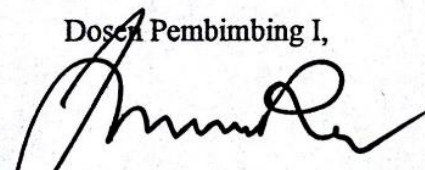
Diajukan untuk Melengkapi Persyaratan guna
Mendapatkan Gelar Sarjana Strata Satu
Program Studi Tekni Informatika

Oleh:


DENNY GUNAWAN	(12.111.1362)
RIVALDI WARMAN	(12.111.3223)
WISELY JANSEN HADI	(12.111.1044)

Disetujui Oleh:

Dosen Pembimbing I,


Dr. Pahala Sirait, S.T., M.Kom
NIP. 45970117

Dosen Pembimbing II,


Sunarto Megawan, S.Kom., M.Kom.
NIP. 45061039

Medan, 19 Agustus 2016

Diketahui dan Disahkan oleh:

Ketua Program Studi
Teknik Informatika,



ABSTRAK

Permasalahan terhadap plagiarisme telah ada selama bertahun-tahun. Namun penyebaran informasi dan teknologi komunikasi, termasuk salah satunya internet telah memberikan kontribusi untuk kemudahan menjiplak. Oleh karena itu, pendeteksi plagiarisme perlu dilakukan untuk mengurangi tingkat penjiplakan yang semakin marak.

Untuk mengurangi atau mencegah plagiarisme ini, maka dirancang sebuah sistem pengecekan plagiarisme dimana untuk mendeteksi kata per kata dilakukan dengan menggunakan algoritma *levenshtein distance* dan dalam mempertimbangkan makna kalimat dimana setiap kata pada kalimat dilakukan pencocokan dengan *databasenya* dalam dua buah dokumen yang diuji. Dokumen yang diuji adalah *file* berformat *.pdf yang berbahasa Indonesia. Terdapat 2 tipe pemrosesan dalam proses pengujian, yaitu *preprocessing* dan *non-preprocessing* yang dapat menampilkan hasil yang berbeda sesuai dengan dokumen yang akan diuji.

Uji coba yang dilakukan dalam penelitian ini akan menampilkan hasil dalam bentuk persentase. Hasil yang diberikan memiliki akurasi yang cukup tinggi dalam mendeteksi plagiat dari segi kemiripan kata yaitu 66% pada proses *preprocessing* dan makna yang bervariasi. Karena adanya informasi yang berkurang pada proses *preprocessing* tetapi pada *non-preprocessing* didapat cukup tinggi yaitu diatas 80%.

Kata kunci: plagiarisme, *levenshtein distance*, kata, kalimat, *preprocessing*

KATA PENGANTAR

Ucapan syukur kepada Tuhan Yang Maha Esa karena berkat Rahmat-Nya penulis menyelesaikan tugas akhir yang berjudul, **Pengujian Deteksi Tingkat Plagiarisme dengan Mempertimbangkan Makna Kata dengan Menggunakan Algoritma *Levenshtein distance***, sesuai yang direncanakan.

Tugas akhir ini dibuat guna untuk melengkapi persyaratan kurikulum pada Program Studi Teknik Informatika Strata Satu, STMIK Mikroskil Medan. Semoga hasil dari tugas akhir ini ada manfaatnya bagi pihak yang berkepentingan. Pada kesempatan ini, penulis ingin mengucapkan terima kasih kepada:

1. Bapak Dr. Pahala Sirait, S.T., M.Kom., selaku pembimbing I yang telah membimbing penulis selama mengerjakan tugas akhir ini.
2. Bapak Sunario Megawan, S.Kom., M.Kom., selaku pembimbing II dan Sekretaris Program Studi Teknik Informatika yang telah membimbing penulis selama mengerjakan tugas akhir ini.
3. Bapak Dr. Mimpin Ginting, M.S., selaku Ketua STMIK Mikroskil Medan.
4. Bapak Djoni, S.Kom., M.T.I., selaku Wakil Ketua STMIK Mikroskil Medan.
5. Bapak Hardy, S.Kom., M.Sc., selaku Ketua Program Studi Teknik Informatika.
6. Bapak / Ibu Dosen yang telah membimbing penulis selama pengerjaan tugas akhir ini.
7. Orang tua penulis yang telah memberikan dukungan selama pengerjaan tugas akhir ini.
8. Semua orang yang telah membantu hingga selesainya penulisan tugas akhir ini.

Dalam penulisan tugas akhir ini, penulis menyadari sepenuhnya bahwa dalam penulisan tugas akhir ini masih terdapat banyak kekurangan dan masih jauh dari sempurna. Oleh karena itu setiap saran yang bersifat membangun akan diterima dengan senang hati. Akhir kata penulis sangat mengharapkan tugas akhir ini dapat memberikan manfaat bagi para pembaca sekalian. Terima kasih

Medan, Agustus 2016

Penulis,

DAFTAR ISI

ABSTRAK	i
KATA PENGANTAR.....	ii
DAFTAR ISI	iii
DAFTAR GAMBAR	v
DAFTAR TABEL	vii
DAFTAR LAMPIRAN	viii
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan dan Manfaat	2
1.4 Batasan Masalah	2
1.5 Metodologi penelitian	3
BAB II TINJAUAN PUSTAKA.....	5
2.1 Kata	5
2.2 Kalimat.....	8
2.3 Paragraf	10
2.3.1 Jenis-jenis Paragraf	10
2.3.2 Syarat Pembentukan Paragraf	11
2.4 Makna	12
2.4.1 Jenis Makna.....	13
2.4.2 Relasi Makna.....	14
2.5 Plagiat	15
2.5.1 Jenis-jenis Teknik Plagiarisme.....	15
2.5.2 Taksonomi Plagiarisme	16
2.5.3 Metode Pendeteksian Plagiarisme.....	19
2.6 <i>Preprocessing</i>	20
2.6.1 <i>Tokenization</i>	20
2.6.2 <i>Stemming</i>	21
2.6.3 <i>Stopwords</i>	23
2.7 <i>Levenshtein distance</i>	23

BAB III METODOLOGI PENELITIAN	26
3.1 Pengumpulan Referensi	26
3.2 Analisis	26
3.2.1 Analisis Proses	26
3.2.2 Analisis Kebutuhan	41
3.3 Perancangan	45
3.3.1 Kamus data.....	45
3.3.2 Perancangan <i>User Interface</i>	46
BAB IV HASIL DAN PENGUJIAN	53
4.1 Hasil	53
4.2 Pengujian	58
BAB V KESIMPULAN DAN SARAN.....	71
5.1 Kesimpulan	71
5.2 Saran	71
DAFTAR PUSTAKA	73

DAFTAR GAMBAR

Gambar 2.1 Taksnonomi plagiarisme	18
Gambar 2.2 Metode pendeteksi plagiarisme	19
Gambar 3.2 <i>flowchart</i> proses <i>tokenization</i>	26
Gambar 3.3 <i>flowchart</i> proses <i>stopwords</i>	27
Gambar 3.4 <i>flowchart</i> proses <i>stemming</i>	28
Gambar 3.5 <i>flowchart</i> proses <i>levenshtein distance</i>	29
Gambar 3.6 hasil perhitungan <i>levenshtein</i> huruf 's' dengan 's'	31
Gambar 3.7 hasil perhitungan <i>levenshtein</i> huruf 's' dengan 'u'	31
Gambar 3.8 hasil perhitungan <i>levenshtein</i> huruf 's' dengan 'k'	32
Gambar 3.9 hasil perhitungan <i>levenshtein</i> huruf 's' dengan 'a'	32
Gambar 3.10 hasil perhitungan <i>levenshtein</i> huruf 'a' dengan 's'	32
Gambar 3.11 hasil akhir perhitungan <i>levenshtein</i> kata "saya" dengan "suka"	33
Gambar 3.12 hasil perhitungan <i>levenshtein</i> kata "saya" dengan kata contoh pada kalimat ke-2	34
Gambar 3.13 hasil perhitungan <i>levenshtein</i> kata "benci" dengan kata pada contoh kalimat ke-2	34
Gambar 3.14 hasil perhitungan <i>levenshtein</i> kata "dia" dengan kata pada contoh kalimat ke-2	35
Gambar 3.15 <i>flowchart</i> perhitungan persentase proses <i>levenshtein distance</i>	35
Gambar 3.16 <i>flowchart</i> proses sinonim	37
Gambar 3.17 Hasil <i>preprocessing</i> pada contoh proses sinonim	38
Gambar 3.18 Hasil kalimat ke-1 plagiat dengan asli	39
Gambar 3.19 Hasil kalimat ke-2 plagiat dengan asli	39
Gambar 3.20 Hasil kalimat ke-3 plagiat dengan asli	39
Gambar 3.21 Hasil kalimat ke-4 plagiat dengan asli	40
Gambar 3.22 <i>flowchart</i> proses perhitungan proses sinonim	40
Gambar 3.23 <i>Use Case diagram</i> Pengujian Deteksi Tingkat Plagiarisme dengan Mempertimbangkan Makna Kata dengan Menggunakan Algoritma <i>Levenshtein distance</i>	42

Gambar 3.25 <i>Form Input file</i>	47
Gambar 3.26 <i>Form Detil Proses (Preprocessing)</i>	48
Gambar 3.27 <i>Form Detil Proses (Non-preprocessing)</i>	49
Gambar 3.28 <i>Form Hasil</i>	51
Gambar 3.29 <i>Form About</i>	52
Gambar 4.1 Tampilan <i>input file</i>	53
Gambar 4.2 Tampilan proses <i>input file</i> dan pemilihan tipe pengecekan	54
Gambar 4.3 Tampilan Detil Proses (<i>non-preprocessing</i>)	55
Gambar 4.4 Tampilan Detil Proses (<i>preprocessing</i>).....	56
Gambar 4.5 Tampilan hasil akhir.....	57
Gambar 4.6 Tampilan <i>about</i>	57

DAFTAR TABEL

Tabel 2.1 Penjelasan jenis afiks (imbuhan) dan contoh.....	6
Tabel 2.2 Tahapan algoritma <i>levenshtein distance</i>	25
Tabel 3.1 Narasi <i>Use Case</i> Masukkan Dokumen.....	43
Tabel 3.2 Narasi Memilih Proses	43
Tabel 3.3 Narasi <i>Use Case</i> melihat hasil	44
Tabel 3.4 Struktur tabel kamus_tesaurus	46
Tabel 3.5 Struktur tabel <i>stopwords list</i>	46
Tabel 3.6 Struktur tabel tb_rootword	46
Tabel 3.7 Struktur tabel <i>unneeded</i>	46
Tabel 4.1 Daftar dokumen asli dan dokumen plagiat (pembanding)	58
Tabel 4.2 pengujian dokumen asli dengan dokumen plagiat	60
Tabel 4.3 Pengujian terhadap dokumen asli dan pembanding	63
Tabel 4.4 pengujian dokumen asli dengan dokumen plagiat secara acak.....	66
Tabel 4.5 Pengujian dengan sample yang di acak.....	69

DAFTAR LAMPIRAN

Lampiran 1 <i>listing program</i>	Error! Bookmark not defined.
Daftar Riwayat Hidup	107

BAB I

PENDAHULUAN

1.1 Latar Belakang

Permasalahan terhadap plagiarisme telah ada selama berabad-abad. Namun penyebaran informasi dan teknologi komunikasi, termasuk internet telah memberikan kontribusi untuk kemudahan menjiplak. Banyak layanan *online* yang ada untuk memfasilitasi pendeteksian plagiarisme, termasuk *database* esai dan alat teks "*synonymizer*" seperti *synonym.com* yang hasil keluaran dari masukan teks berupa daftar-daftar sinonim dari tiap kata. Hasil survei dari 80.000 lebih mahasiswa di Amerika Serikat dan Kanada menunjukkan bahwa sebanyak 62% sarjana dan 59% pascasarjana melakukan plagiarisme *cut and paste* dari sumber tertulis dan internet (McCabe, 2005). Di Indonesia sendiri banyak terjadi kasus plagiarisme salah satunya yaitu dosen di fakultas ekonomi dan bisnis (FEB) Universitas Gadjah Mada, Anggito Abimanyu terhadap artikelnya "Gagasan Asuransi Bencana" yang terbit di harian *Kompas*. 10 Februari 2014. Tulisan ini memiliki kesamaan dengan artikel Hotbonar Sinaga dan Munawar Kasan yang berjudul "Menggagas Asuransi Becana". (Lestari, Ade Hapsari, 25 Februari 2014)

Penelitian pada sistem untuk menganalisa dan mendeteksi plagiarisme telah dilakukan hampir dua dekade. Sistem pendeteksi plagiarisme saat ini menggunakan pertimbangan teks berbasis karakter yang canggih dan efisien. Sistem ini dapat mendeteksi kata per kata dan penyalinan teks dari sumber yang original. Akan tetapi, penataan ulang kalimat dan penggunaan makna tersirat yang kebanyakan ditemukan dalam penelitian menyebabkan perbandingan kesamaan karakter yang tidak cocok. Oleh karena itu, sistem pendeteksi plagiarisme tidak dapat mendeteksi bahwa dokumen tersebut plagiat atau tidak.

Untuk mengatasi masalah pada sistem pendeteksi kemiripan atau plagiat pada dokumen saat ini, maka digunakan algoritma *levenshtein distance*. Algoritma *Levenshtein distance* memiliki 3 proses utama dalam menentukan perbedaan antara

2 *string* (kata) yaitu proses penyisipan (*insertion*), proses penghapusan (*deletion*) dan proses penggantian (*substitution*) pada huruf yang dianggap sesuai dengan *database*. Kemudian dari hasil akhir dari proses tersebut maka akan di peroleh nilai kemiripan antar *string* (kata). (adriyani, et al, 2010)

Atas pertimbangan dari paragraf sebelumnya maka diusulkan tugas akhir dengan judul “Pengujian Deteksi Tingkat Plagiarisme dengan Mempertimbangkan Makna kata dengan Menggunakan Algoritma *Levenshtein distance*”.

1.2 Rumusan Masalah

Berdasarkan penjelasan di atas, maka masalah yang diperoleh adalah seberapa banyak kata-kata yang di plagiat yang terjadi dari dokumen orisinal dan kesamaan atau tidak pada makna kalimat antara dokumen yang orisinal dengan dokumen yang plagiat.

1.3 Tujuan dan Manfaat

Tujuan dari penulisan tugas akhir ini adalah sistem yang digunakan dapat melakukan pengujian serta mendeteksi plagiat pada sebuah dokumen dan dari makna kalimatnya.

Manfaat dari penulisan tugas akhir ini adalah:

1. Sebuah sistem yang dapat membantu masyarakat khususnya dibidang akademis seperti kampus-kampus agar dapat mengetahui mahasiswa-mahasiswa yang melakukan plagiat terhadap tugas akhir (Skripsi, Tesis, disertasi) dan tugas-tugas kuliah lainnya.
2. Mengurangi tingkat plagiarisme yang terjadi pada masa sekarang dan masa akan datang.

1.4 Batasan Masalah

Adapun batasan masalah dalam tugas akhir ini adalah:

1. Sistem akan melakukan pengecekan terhadap dokumen per kalimat.
2. Sistem memproses dokumen yang berbahasa Indonesia.
3. Sistem mendeteksi dokumen secara *offline*.

4. Dokumen yang akan diproses merupakan dokumen yang berisi teks dan bukan merupakan gambar atau hasil dari *scanning*.
5. Dokumen yang akan diproses berformat *.pdf.
6. Sistem hanya dapat menguji 2 dokumen.
7. kamus data pada kamus *thesaurus* sesuai dengan Kamus Besar Bahasa Indonesia (KBBI).
8. Pada proses makna kalimat digunakan kamus data *thesaurus* untuk mencocokkan kata pada kalimat yang ada pada kamus data tersebut.
9. Membutuhkan *Database* sebagai tempat penampung kata-kata seperti kata dasar, kata *stopwords* dan sebagai kamus untuk pengecekan kata per kata.
10. Algoritma *Levenshtein distance* hanya mampu menguji pengecekan kata per kata yaitu (*String Matching*), dan tidak mampu menguji kata yang mengandung semantik lebih dalam.

1.5 Metodologi penelitian

Langkah-langkah dalam pengerjaan tugas akhir ini, sebagai berikut:

1. Mengumpulkan dan mempelajari materi yang berhubungan dengan topik yang dibahas yaitu mengenai plagiarisme, *paraphrase* dan algoritma *levenshtein distance* serta referensi-referensi lainnya yang berhubungan dengan tugas akhir ini.
2. Pengembangan perangkat lunak dengan menggunakan model *waterfall* yang memiliki langkah kerja sebagai berikut:

- a. Analisis

Tahapan ini mencakup pada analisis proses, pemodelan sistem, identifikasi persyaratan fungsional dan non-fungsional dari sistem yang akan dirancang. Analisis proses sistem akan digambarkan dalam bentuk diagram alir untuk algoritma *levenshtein distance* dan proses sinonim. Diagram *use case* digunakan untuk memodelkan kebutuhan fungsional sistem.

- b. Perancangan

Pada tahapan ini memerjemahkan syarat kebutuhan fungsional yang sebelumnya digambarkan dengan diagram *use case* kedalam perancangan

antarmuka pengguna (*user interface*) perangkat lunak dan perancangan kamus datanya. Rancangan awal pada antarmuka pengguna akan digambarkan dengan *mockup* kemudian rancangan akhir akan dibangun dengan *windows form*. Sedangkan Perancangan *database* menggunakan Microsoft Access 2013.

c. Pengkodean

Pada tahap ini akan dilakukan implementasi dari analisis dan desain dengan menggunakan bahasa pemrograman C# dan library itextsharp untuk mempermudah pengolahan pada dokumen tipe .pdf.

d. Pengujian

Proses pengujian pada sistem pengecekan plagiarisme dilakukan sebanyak 2 kali, yaitu: pada sampel asli dengan pembanding yang plagiat dan sampel yang secara acak. Pada sampel asli dengan pembanding dilakukan proses *preprocessing* dan *non-preprocessing*. Tujuan dari pengujian ini untuk melihat apakah ada perbedaan signifikan jika dilakukan *preprocessing* ataupun tanpa melakukan *preprocessing*. hal ini juga dilakukan pada sampel acak apakah hasil yang didapatkan sesuai dengan yang diharapkan.

3. Menarik kesimpulan dari pengujian yang dilakukan.

BAB II

TINJAUAN PUSTAKA

2.1 Kata

Kata adalah morfem atau kombinasi morfem yang oleh bahasawan dianggap sebagai satuan terkecil yang dapat diujarkan sebagai bentuk yang bebas. Kata juga merupakan satuan bahasa yang dapat berdiri sendiri, terdiri dari morfem tunggal (misal batu, rumah, datang) atau gabungan morfem (misal pejuang, pancasila, mahakuasa). (KBBI, 2008:633)

Menurut Kamus Besar Bahasa Indonesia (KBBI, 1997) menyebutkan bahwa “kata” adalah (1) elemen terkecil dalam sebuah bahasa yang diucapkan atau dituliskan dan merupakan realisasi kesatuan dan pikiran yang dapat digunakan dalam berbahasa, (2) konversasi atau bahasa, (3) morfem atau kombinasi beberapa morfem yang dapat diujarkan sebagai bentuk yang bebas, (4) unit bahasa yang dapat berdiri sendiri dan terdiri dari satu morfem (mis. kata) atau beberapa morfem gabungan (mis. perkataan).

Menurut Abdul Chaer (2009: 37), secara gramatikal kata mempunyai dua status yaitu sebagai satuan terbesar dalam tataran morfologi, dan sebagai satuan terkecil dalam tataran sintaksis. Sebagai satuan terbesar dalam tataran morfologi, kata dibentuk dari bentuk dasar (yang dapat berupa morfem dasar terikat maupun bebas, atau gabungan morfem) melalui proses morfologi afiksasi, reduplikasi, atau komposisi. Sebagai satuan terkecil dalam sintaksis kata, khususnya yang termasuk kelas terbuka (nomina, verba, dan ajektifa) dapat mengisi fungsi-fungsi sintaksis.

Dalam tata Bahasa Indonesia, kata dapat digolongkan menjadi 2 bagian besar yaitu: (Chaer Abdul, 2007)

1. Klasifikasi kata berdasarkan kelas kata (bentuk kata)

Dari segi bentuk kata, pengklasifikasiannya terbagi menjadi empat bagian yaitu:

- a. Kata dasar adalah kata asli yang belum diberi imbuhan atau yang belum diberikan awalan, akhiran, sisipan dan penggabungan awalan dan akhiran.

Kata-kata seperti baik, getar, kerja, sakit, gunung disebut sebagai kata dasar karena kata-kata itu tidak berimbuhan atau belum diberi imbuhan.

- b. Kata turunan adalah kata yang telah mengalami penambahan atau pengimbuhan. Penambahan atau pengimbuhan disebut juga dengan afiks yang terdiri dari beberapa bentuk, yaitu:
- a) Imbuhan di awal kata (Prefiks atau awalan)
 - b) Imbuhan di tengah kata (Infiks atau sisipan)
 - c) Imbuhan di akhir kata (Sufiks atau akhiran)

Tabel 2.1 Penjelasan jenis afiks (imbuhan) dan contoh

Sumber: (Chaer Abdul, 2007)

Afiks	Keterangan	Contoh
Prefiks	Afiks yang diimbuhkan di depan bentuk kata dasar	<i>Ber-, di-, ke-, me-, meng-, mem-, meny-, pe-, pem-, peng-, peny-, per-, se-, ter</i>
Infiks	Afiks yang diimbuhkan di tengah bentuk kata dasar	<i>-el-, -er-, -em, -in-</i>
Sufiks	Afiks yang diimbuhkan di belakang bentuk kata dasar	<i>-an, -kan, -i, -nya, -pun, -lah, -kah, -wan. -wati, -asi, -isme, -in, -wi, -if, -ik, -is, -er, -al, -in, -us, -or</i>
Konfiks	Afiks yang berupa morfem yang terbagi yang berada didepan dan belakang bentuk kata dasar	<i>Ke-an, ber-an, pe-an, per-an, peng-an, peny-an, pem-an, se-an, per-i, di-kan, di-i, me-kan, me-i, ter-kan, ter-I,</i>

- c. Kata ulang adalah kata yang mengalami proses pengulangan bentuk, baik seluruh kata maupun sebagian. Semua kata ulang wajib ditulis dengan memakai tanda penghubung (-)

- d. Kata majemuk adalah gabungan beberapa kata dasar yang berbeda yang membentuk suatu arti baru.

2. Klasifikasi kata berdasarkan kelas kata

Dalam tata bahasa baku bahasa Indonesia, kelas kata terbagi menjadi tujuh bagian yaitu:

- a. Kata benda (Nomina) adalah kata yang mengacu pada benda, orang, konsep ataupun pengertian yang berfungsi sebagai objek dan subjek.
- b. Kata kerja (Verba) adalah kata yang menyatakan perbuatan atau tindakan. Kata kerja biasanya berfungsi sebagai predikat.
- c. Kata sifat (Adjektiva) adalah kata yang dipakai untuk mengungkapkan sifat atau keadaan sesuatu. Contoh: keadaan orang, binatang dan benda. Kata sifat juga berfungsi sebagai predikat.
- d. Kata keterangan (Adverbia) adalah kata yang memberi keterangan pada verba, adjektiva, nomina predikatif atau kalimat.
- e. Kata ganti (Pronomina) adalah kata yang digunakan untuk mengacu pada nomina lain. Pronomina berfungsi untuk mengganti kata benda atau nomina.
- f. Kata bilangan (Numeralia) adalah kata yang digunakan untuk menghitung banyaknya orang, binatang atau benda.
- g. Kata tugas adalah sejenis kategori kata dalam tata bahasa *formal* bahasa Indonesia yang berdasarkan peranannya dapat dibagi menjadi lima subkelompok yaitu:
 - i. Kata depan (Preposisi) adalah kata yang merangkaikan kata-kata atau bagian kalimat. Preposisi dapat berbentuk kata, contohnya kata di dan untuk, atau gabungan kata, contohnya bersama atau sampai dengan.
 - ii. Kata sambung (Konjungsi) adalah kata yang menghubungkan dua satuan bahasa; kata dengan kata, frasa dengan frasa, klausa dengan klausa dan seterusnya.
 - iii. Kata sandang (Artikula) adalah kata tugas yang membatasi makna nomina.

- iv. Kata seru (interjeksi) adalah kata yang mengungkapkan perasaan dan maksud seseorang, misalnya ah atau aduh, atau melambangkan tiruan bunyi, seperti meong.
- v. Partikel adalah kategori atau unsur yang bertugas memulai, mempertahankan atau mengukuhkan sebuah kalimat dalam komunikasi. Unsur ini digunakan dalam kalimat tanya, perintah dan pernyataan (berita).

2.2 Kalimat

Secara hierarkis, kalimat merupakan satuan klausa di bawah tataran wacana. Kalimat merupakan konstituen wacana atau pembentuk wacana. Dan disepakati bahwa kalimat sebagai satuan sintaksis terbesar setelah frasa dan klausa. (Dola, 2010:14, 76)

Satuan bahasa yang menjadi inti pembicaraan dalam sintaksis adalah kalimat. Kalimat merupakan satuan di atas klausa dan di bawah satuan wacana. Kalimat adalah satuan sintaksis yang disusun dari konstituen dasar, yang biasanya berupa klausa, dilengkapi dengan konjungsi bila diperlukan, serta disertai dengan intonasi final (Chaer Abdul, 2009: 44).

Menurut Alwi, dkk. (2003), kalimat dapat digolongkan menjadi beberapa bagian yaitu:

1. Berdasarkan jumlah klausa yang dimilikinya, kalimat dibedakan menjadi dua yaitu:
 - a. Kalimat tunggal adalah kalimat yang proposisinya satu sehingga predikatnya pun satu.
 - b. Kalimat majemuk adalah kalimat yang terdiri atas lebih dari satu proposisi sehingga mempunyai paling tidak dua predikat yang tidak dapat dijadikan satu kesatuan. Karena sifat itu maka kalimat majemuk selalu berwujud dua klausa atau lebih. Kalimat majemuk dibagi menjadi 2 yaitu:
 - i. Kalimat majemuk setara apabila kalimat itu menyatakan hubungan koordinatif (sejajar / setara).
 - ii. Kalimat majemuk bertingkat jika hubungan subordinat (bertingkat).

2. Berdasarkan bentuk sintaksis kalimat di bagi menjadi:

a. Kalimat deklaratif atau kalimat berita.

Kalimat berita merupakan kalimat yang digunakan untuk menginformasikan sesuatu. Biasanya diakhiri dengan tanda titik (.).

b. Kalimat imperatif atau kalimat perintah.

Kalimat yang bertujuan untuk mengintruksikan seseorang untuk melakukan sesuatu. Kalimat perintah biasanya diakhiri dengan tanda seru. Tapi, jika dikatakan langsung atau lisan biasanya ditandai dengan intonasi tinggi

c. Kalimat interogatif atau kalimat tanya.

Kalimat tanya mengharapkan jawaban sebagai respon atau reaksi pemberitahuan informasi yang diharapkan, biasanya diakhiri dengan tanda tanya (?). kata tanya yang digunakan bagaimana, mengapa, apa kapan, dimana dsb.

d. Kalimat ajakan.

Kalimat ajakan merupakan kalimat yang memancing minat lawan bicara. Kata yang sering digunakan adalah Ayo, Mari dsb. Biasanya ada pada iklan.

e. Kalimat pengandaian.

Kalimat pengandaian menggambarkan keinginan atau tujuan dari penulis atau pembicara yang belum atau tidak kesampaian. Contoh: Andai saja aku bisa jadi dokter bedah.

3. Berdasarkan kelengkapan unsur dibagi menjadi:

a. Kalimat lengkap atau kalimat major.

Kalimat mayor hanya memiliki subjek dan predikat. Objek, pelengkap dan keterangan boleh ditambahkan sesuka hati. Sama seperti pola dasar pertama.

b. Kalimat tak lengkap atau minor.

Kalimat yang memiliki satu inti fungsi gramatikalnya. Bentuk kalimat minor seperti kalimat tambahan, kalimat jawaban, kalimat salam, panggilan maupun judul.

4. Berdasarkan subjek predikat

a. Kalimat biasa.

Kalimat yang subjeknya mendahului predikatnya. Kalimat berpola dasar.

b. Kalimat inversi.

Kalimat ini merupakan kebalikan dari kalimat normal. Dimana predikatnya mendahului objek.

5. Berdasarkan pengucapan

a. Kalimat langsung.

Kalimat yang secara detail meniru sesuatu yang diucapkan oranglain. Tanda baca kutip tidak luput dalam jenis kalimat langsung. Kutipan dalam kalimat langsung berupa kalimat tanya, kalimat berita ataupun kalimat perintah.

b. Kalimat tak langsung.

Kalimat yang melaporkan kembali kalimat yang diucapkan orang lain. Kutipan dalam kalimatnya semuanya berbentuk berita.

2.3 Paragraf

Menurut Akhadijah (Nasucha, Rohmadi dan Wahyudi. 2009: 33) paragraf merupakan inti penuangan buah pikiran dalam sebuah pikiran. Dalam paragraf terkandung satu unit buah pikiran yang didukung oleh semua kalimat dalam paragraf tersebut, mulai dari kalimat pengenal, kalimat utama atau topik, kalimat-kalimat penjelas sampai pada kalimat penutup. Himpunan kalimat ini saling bertalian dalam suatu rangkaian untuk membentuk sebuah gagasan.

2.3.1 Jenis-jenis Paragraf

Jenis paragraf dibagi menjadi 3 yaitu:

1. Paragraf pembuka.

Rohmadi dan Nasucha (2010:39) mengemukakan Paragraf pembuka dapat disebut paragraf pendahuluan (*introduction*). Fungsinya sebagai pengantar untuk sampai kepada pokok pembicaraan dalam karangan.

2. Paragraf penghubung.

Paragraf penghubung berisi inti persoalan yang dikemukakan. Oleh karena itu, secara kuantitatif paragraf inilah yang paling panjang, dan antara paragraf dengan paragraf saling berhubungan secara logis (Nasucha, Rohmadi dan Wahyudi. 2009: 34).

3. Paragraf penutup.

Paragraf ini berisi kesimpulan dari paragraf penghubung. Paragraf penutup juga dapat berisi penegasan kembali mengenai hal-hal yang dianggap penting dalam paragraf penghubung (Nasucha dkk, 2009:35).

2.3.2 Syarat Pembentukan Paragraf

Dalam pembentukan paragraf, terdapat syarat-syaratnya yang antara lain:

1. Kesatuan.

Paragraf dianggap mempunyai kesatuan, jika kalimat– kalimat dalam paragraf itu tidak terlepas dari topiknya atau selalu relevan dengan topik (Nasucha, Rohmadi dan Wahyudi. 2009: 35). Semua kalimat berfokus pada topik dan mencegah masuknya hal-hal yang tidak relevan.

Menurut Marsa (2009: 9) kesatuan dalam sebuah paragraf hanya terbentuk apabila informasi-informasi dalam paragraf itu tetap dikendalikan oleh gagasan utama. Agar kesatuan dapat dicapai penulis senantiasa mengevaluasi kalimat-kalimat yang ditulisnya itu erat hubungannya dengan gagasan utama. Apabila tidak erat hubungannya, kalimat-kalimat itu sebaiknya dihilangkan atau disajikan secara khusus, misalnya menjadi sisipan dalam kalimat lain.

2. Kepaduan.

Satu paragraf bukanlah merupakan kumpulan atau tumpukan kalimat yang masing–masing berdiri sendiri atau terlepas, tetapi dibangun oleh kalimat-kalimat yang mempunyai hubungan timbal balik. Pembaca dengan mudah memahami dan mengikuti jalan pikiran penulis tanpa hambatan karena adanya loncatan pikiran yang membingungkan. Urutan pikiran yang teratur, akan memperlihatkan adanya kepaduan. Jadi, Kepaduan atau koherensi dititikberatkan pada hubungan antara kalimat dengan kalimat (Nasucha, Rohmadi dan Wahyudi, 2009: 37)

Kepaduan antar kalimat dalam paragraf itu meliputi dua macam, yaitu kepaduan bentuk dan kepaduan makna. Kepaduan makna adalah kepaduan informasi yang disebut koherensi dan kepaduan di bidang bentuk disebut kohesi. Paragraf yang memiliki kepaduan informasi bersifat kohesi dan kesesuaian di bidang bentuk disebut kohesif. Wacana yang baik dalam sebuah paragraf apabila memiliki dua kepaduan tersebut, yaitu kohesif dan koheren. (Rohmadi dan Nasucha, 2010: 46).

3. Kelengkapan.

Suatu paragraf dikatakan lengkap, jika berisi kalimat–kalimat penjelas yang cukup untuk menunjang kejelasan kalimat topik dan kalimat utama. (Nasucha, Rohmadi dan Wahyudi. 2009: 39). Sebaliknya suatu paragraf dikatakan tidak lengkap, jika tidak dikembangkan atau hanya diperluas dengan pengulangan–pengulangan.

Paragraf dapat dikatakan memiliki kelengkapan, jika kalimat topiknya dapat dikembangkan dengan pendukung yang cukup (Rohmadi dan Nasucha, 2009: 47-48). Istilah cukup adalah relatif, tetapi yang jelas lebih dari satu dan kurang dari sepuluh. Jika didukung oleh satu kalimat maka pengembangannya kering dan jika sangat banyak maka pembaca cepat bosan dan sulit menemukan keutuhan informasi.

2.4 Makna

Menurut teori yang dikembangkan Ferdinand de Saussure, makna adalah pengertian atau konsep yang dimiliki atau terdapat pada sebuah tanda linguistik. Jika tanda linguistik tersebut disamakan identitasnya dengan kata atau leksem, berarti makna adalah pengertian atau konsep yang dimiliki oleh setiap kata atau leksem. Jika disamakan dengan morfem, maka makna adalah pengertian atau konsep yang dimiliki oleh setiap morfem, baik morfem dasar maupun morfem afiks. (Chaer Abdul, 2007)

2.4.1 Jenis Makna

Dalam tata bahasa Indonesia makna dapat dikategorikan menjadi beberapa jenis yaitu: (Chaer Abdul, 2007)

1. Makna leksikal, gramatikal dan kontekstual.
 - a. Makna leksikal adalah makna yang dimiliki atau ada pada leksem meski tanpa konteks apapun.
 - b. Makna gramatikal adalah makna yang ada jika terjadi proses gramatikal seperti afiksasi, reduplikasi, komposisi atau kalimatisasi.
 - c. Makna kontekstual adalah makna sebuah leksem atau kata yang berada di dalam satu konteks.
2. Makna referensial dan non – referensial.

Sebuah kata atau leksem dikatakan bermakna referensial jika ada referensinya atau acuannya. Ada sejumlah kata yang disebut kata diektik, yang acuannya tidak menetap pada satu wujud.
3. Makna denotatif dan makna konotatif.
 - a. Makna denotatif adalah makna asli, makna asal atau makna sebenarnya yang dimiliki oleh sebuah leksem. Makna denotatif sebenarnya sama dengan makna leksikal.
 - b. Makna konotatif adalah makna lain yang ditambahkan pada makna denotatif yang berhubungan dengan nilai rasa dari orang yang menggunakan kata tersebut.
4. Makna konseptual dan makna asosiatif.
 - a. Makna konseptual sebenarnya sama dengan makna leksikal, denotatif dan makna referensial.
 - b. Makna asosiatif adalah makna yang dimiliki sebuah leksem atau kata bahasa.
5. Makna kata dan makna istilah.
 - a. Makna yang dimiliki oleh sebuah kata adalah makna leksikal, denotatif atau makna konseptual. Namun, dalam penggunaannya makna kata itu baru menjadi jelas jika kata itu sudah berada di dalam konteks kalimatnya atau konteks situasinya.

- b. Istilah mempunyai makna yang pasti, jelas, tidak meragukan, meskipun tanpa konteks kalimat. Oleh karena itu, istilah sering dikatakan bebas konteks, sedangkan kata tidak bebas konteks.
- 6. Makna idiom dan peribahasa.
 - a. Idiom adalah satuan ujaran yang maknanya tidak dapat diramalkan dari makna unsur-unsurnya, baik secara leksikal maupun secara gramatikal.
 - b. Peribahasa memiliki makna yang masih dapat ditelusuri dari makna unsurnya karena adanya “asosiasi” antara makna asli dengan maknanya sebagai peribahasa.

2.4.2 Relasi Makna

Relasi makna adalah hubungan semantik yang terdapat antara satuan bahasa yang satu dengan yang lain. (Chaer Abdul, 2007). Ada beberapa tipe relasi makna di antaranya yaitu:

1. Sinonim yaitu hubungan semantik yang menyatakan adanya kesamaan makna antara satu satuan ujaran dengan satuan ujaran lainnya. Dua buah ujaran yang bersinonim maknanya tidak akan sama persis. Ketidaksamaan itu terjadi karena faktor: waktu, tempat, keformalan, social, bidang kegiatan, factor nuansa makna
2. Antonim yaitu hubungan semantik antara dua buah satuan ujaran yang maknanya menyatakan kebalikan, pertentangan, atau kontras antara yang satu dengan yang lain.
3. Polisemi yaitu kata yang mempunyai makna lebih dari satu. Dalam kasus polisemi, biasanya makna pertama adalah makna sebenarnya, yang lain adalah makna yang dikembangkan berdasarkan salah satu komponen makna yang dimiliki kata atau satuan ujaran itu
4. Homonim yaitu dua buah kata atau satuan ujaran yang bentuknya “kebetulan” sama dan maknanya berbeda, karena masing-masing merupakan kata atau bentuk ujaran yang berlainan.

5. Hponimi yaitu hubungan semantik antara sebuah bentuk ujaran yang maknanya tercakup dalam makna bentuk ujaran yang lain. Relasi hponimi bersifat searah.
6. Ambigiutas atau ketaksaan yaitu gejala dapat terjadinya kegandaan makna akibat tafsiran gramatikal yang berbeda. Ketaksaan terjadi dalam bahasa tulis akibat perbedaan gramatikal karena ketiadaan unsur lisan, karena ketidakcermatan dalam menyusun konstruksi beranaforis.
7. Redudansi yaitu kata yang berlebih-lebihan yang menggunakan unsur segmental dalam suatu bentuk ujaran.

2.5 Plagiat

Plagiat atau plagiarisme adalah salah satu bentuk tindakan kecurangan, tetapi hal tersebut sulit untuk dijelaskan sehingga semua orang bahkan anak-anak bisa melakukannya tanpa menyadari dan memahami bahwa hal tersebut sebenarnya tidak dibenarkan. (Steven, Dowshen, 2011)

Plagiat dapat dianggap sebagai tindakan melanggar hukum karena dikategorikan sebagai tindakan mencuri hak cipta seseorang. Di dunia pendidikan, para pelaku plagiat akan mendapatkan hukuman yang cukup berat bila pelaku plagiat adalah seorang mahasiswa atau seorang pendidik (Dosen, Guru) maka pelaku akan dikeluarkan dari universitas / sekolah tempat pelaku memperoleh ilmu atau yang berprofesi sebagai tenaga pendidik. (Steven, Dowshen, 2011)

2.5.1 Jenis-jenis Teknik Plagiarisme

Berdasarkan pola, modus dan teknik plagiat dapat dikategorikan menjadi 4 jenis plagiarisme (Andreas Lako, 2012) yaitu:

- a) Plagiarisme total adalah tindakan plagiat yang dilakukan oleh seorang penulis dengan cara melakukan penjiplakan hasil karya orang lain secara keseluruhan dan kemudian mengklaim bahwa hasil plagiat tersebut sebagai karyanya sendiri.
- b) Plagiarisme parsial adalah tindakan plagiat yang dilakukan oleh seorang penulis dengan cara melakukan penjiplakan hanya sebagian dari hasil karya

orang lain, biasanya dalam melakukan plagiat seorang penulis hanya mengambil beberapa pernyataan, landasan teori, pembahasan sampai kesimpulan dan tanpa menyebutkan sumber aslinya.

- c) Auto-Plagiasi (Self-plagiarisme) adalah tindakan plagiat yang dilakukan oleh seorang penulis terhadap hasil karyanya sendiri, baik hanya sebagian besar dari hasil karyanya atau secara keseluruhan.
- d) Plagiarisme antarbahasa adalah tindakan plagiat yang dilakukan oleh seorang penulis dengan cara menerjemahkan suatu karya tulis yang berasal dari negara lain atau karya tulis yang berbahasa asing kemudian diterjemahkan ke dalam bahasa Indonesia.

Selain keempat jenis tersebut plagiarisme juga dapat dikelompokkan atau diklasifikasi lebih detail berdasarkan proporsi atau presentase kata, kalimat atau paragraf yang dibajak (Sastroasmoro Sudigdo , 2007) antara lain:

- 1. Plagiarisme ringan dengan berkisar antara: < 30%
- 2. Plagiarisme sedang dengan berkisar antara: 30-70%
- 3. Plagiarisme berat atau total berkisar antara: > 70%

2.5.2 Taksonomi Plagiarisme

Taksonomi / pengelompokkan plagiarism dibagi menjadi 2 bagian besar yaitu: plagiarisme secara literal (*literal Plagiarism*) dan plagiarisme secara cerdas (*intelligent plagiarism*). (Alzahrani,Salha M. et al 2012)

1. *Literal Plagiarism.*

Literal plagiarism / plagiarisme secara literal adalah suatu tindakan yang paling umum dan paling sering dilakukan dimana seorang plagiat tidak perlu menghabiskan waktu untuk menyembunyikan kejahatan akademik yang telah dilakukan (Alzahrani,Salha M. et al ,2012)

Beberapa teknik *Literal Plagiarism* yaitu:

- a) *Extract copy* ; teknik ini melakukan pengambilan / mengcopy secara keseluruhan dari suatu dokumen atau beberapa bagian penting dari dokumen original.

- b) *Near copy* ; teknik ini melakukan penambahan, penghapusan, substitusi, pengabungan atau pemisahan suatu kalimat pada dokumen original.
- c) *Modified copy / Restructing* ; teknik ini melakukan modifikasi dari segi struktur kalimat dari dokumen original dengan penataan kembali frase pada kalimat tersebut.

2. *Intelligent plagiarism.*

Intelligent plagiarism / plagiarisme secara pintar adalah suatu tindakan ketidakjujuran yang merupakan perhatian serius akademik dimana seorang plagiat mencoba untuk menipu pembaca dengan mengganti kontribusi orang lain dan menjadi hasil mereka. Seorang plagiat akan mencoba untuk menyembunyikan, mengaburkan, dan mengubah karya asli dalam berbagai cara yang cerdas, termasuk manipulasi teks, terjemahan dan adopsi ide (Alzhrani, Sallha M. , et al , 2012)

Beberapa jenis teknik dari intelligent plagiarism yaitu:

a) *Text manipulation.*

Plagiat dapat mengaburkan makna dengan memanipulasi teks dan mengubah semua dari penampilan teks dari dokumen.

Text manipulation juga dibagi 2 yaitu

i. *Paraphrasing.*

Paraphrasing / Paraphrase adalah strategi pemahaman makna suatu bentuk karya sastra dengan cara mengungkapkan kembali karya pengarang tertentu dengan menggunakan kata-kata yang berbeda dengan kata-kata yang digunakan pengarang. Tujuannya adalah menyederhanakan pemakaian kata atau kalimat (Aminuddin,2010:41).

Teknik *paraphrase* meliputi pengalihan bentuk dari hasil karya orang lain tapi tetap menyertakan sumber karya ilmiah. (1) Perubahan kata / frasa kunci dengan kata lain yang memiliki makna yang mirip atau sama. Proses ini menyangkut pemilihan kata yang memiliki persamaan arti (sinonim). (2) Perubahan bentuk kalimat asal dengan kalimat dengan susunan atau pola berbeda tanpa mengubah maknanya. (3) Perubahan

bentuk wacana menjadi uraian yang lebih pendek berupa ringkasan atau rangkuman.

ii. *Summarizing.*

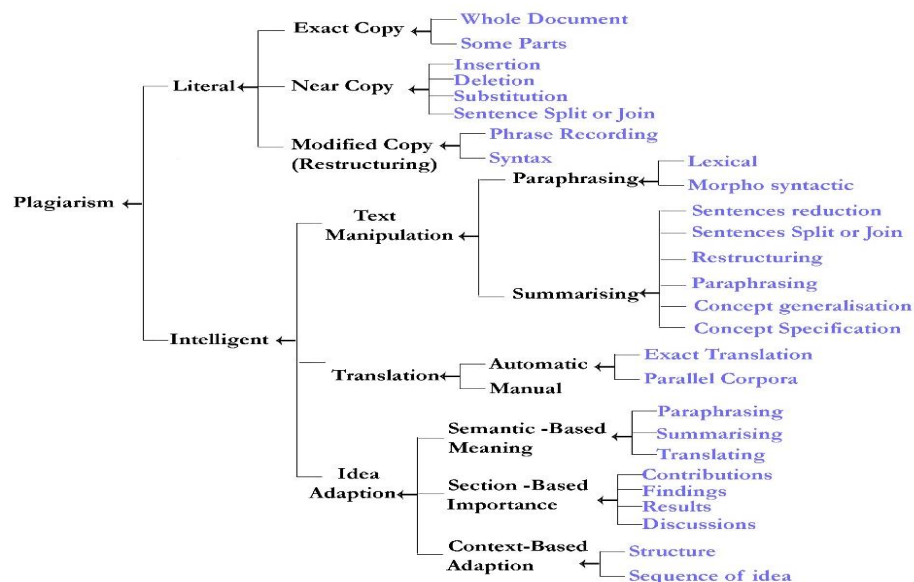
Summary / summarizing / ringkasan adalah strategi yang melibatkan ide utama yang akan digunakan untuk plagiat. Ide akan diringkas dari sumber aslinya. Ringkasan secara signifikan lebih pendek dari pada aslinya. (Driscoll, dana lynn & Allen Brizee, 2010)

b) *Translation.*

Translation / menerjemahkan adalah proses mengalihkan satu kebudayaan ke kebudayaan lain atau suatu pengertian dari kebudayaan yang satu ke pengertian kebudayaan yang lain. (mayantara, 23 oktober 2012)

c) *Idea Adoption.*

Idea Adoption adalah tindakan plagiarisme yang paling serius mengacu pada penggunaan ide, hasil, konstribusi, temuan, dan kesimpulan tanpa mengutip sumber asli dari ide (Alzhrani,Sallha M., et al, 2012)



Gambar 2.1 Taksnonomi plagiarisme

Sumber: (<http://www.plagiarsmchecker.net/plagiarism-detection.php>)

2.5.3 Metode Pendeteksian Plagiarisme

Metode pendeteksi plagiarisme dapat dibagi menjadi 3 bagian yaitu metode perbandingan teks lengkap, metode dokumen *fingerprinting* dan metode kesamaan kata kunci. (George R.S Weir,2004)



Gambar 2.2 Metode pendeteksi plagiarisme

Sumber: (Stein,2006).

Berikut ini adalah penjelasan dari masing-masing metode dan algoritma pendeteksi plagiarisme (Stein, B., 2006):

1. Perbandingan teks lengkap.

Metode ini diterapkan dengan membandingkan semua isi dokumen. Dapat diterapkan untuk dokumen yang besar. Pendekatan ini membutuhkan waktu yang lama tetapi cukup efektif, karena kumpulan dokumen yang diperbandingkan adalah dokumen yang disimpan pada penyimpanan lokal.

Metode perbandingan teks lengkap tidak dapat diterapkan untuk kumpulan dokumen yang tidak terdapat pada dokumen lokal. Algoritma yang digunakan pada metode ini adalah algoritma *Brute-Force*, algoritma *Edit Distance*, algoritma *Boyer-Moore* dan algoritma *Levenshtein distance*.

2. Dokumen *fingerprinting*.

Dokumen *fingerprinting* merupakan metode yang digunakan untuk mendeteksi keakuratan salinan antar dokumen atau hanya sebagian teks saja. Prinsip kerja dari metode dokumen *fingerprinting* ini adalah dengan menggunakan teknik *hashing*. Teknik *hashing* adalah sebuah fungsi yang mengkonversi setiap *string* menjadi bilangan. Misalnya *Rabin-Karp*, *Winnowing* dan *Manber*.

3. Kesamaan kata kunci.

Prinsip dari metode ini adalah mengekstrak kata kunci dari dokumen dan kemudian dibandingkan dengan kata kunci pada dokumen yang lain. Pendekatan yang digunakan pada metode ini adalah teknik dot.

2.6 *Preprocessing*

Preprocessing adalah proses pembersihan dan persiapan teks untuk tahap selanjutnya. Teks *online* biasanya banyak noise dan bagian informasi tidak berguna seperti *HTML tags*, *scripts* dan iklan. Sebaliknya, pada tingkat kata, banyak kata pada teks tidak berpengaruh pada orientasi umum. Menjaga kata itu akan menyebabkan dimensi masalah tinggi dan kesulitan klasifikasi karena setiap kata dalam teks diperlakukan sebagai satu dimensi. Hipotesis jika dilakukan *preprocessing*: mengurangi noise di teks yang membantu meningkatkan *performa* dan kecepatan pada klasifikasi. (Haddi, emma, ed al, 2013).

2.6.1 *Tokenization*

Tokenization adalah proses memecahkan setiap susunan kalimat dalam dokumen menjadi beberapa bagian kata yang disebut *token*, dalam waktu yang sama membuang beberapa karakter tertentu yang tidak diperlukan seperti tanda baca dan mengubah semua huruf besar menjadi huruf kecil berikut contoh dari *tokenization* (Manning, Christopher Et al, 2009).

Input: Friends, Romans, Countrymen, Lend me your ears;

Output:

Friends	Romans	Countrymen	lend	Me	your	Ears
---------	--------	------------	------	----	------	------

Tokenization berguna baik dalam linguistik (di mana merupakan bentuk dari segmentasi teks), dan dalam dunia komputer *tokenization* merupakan bentuk dari analisis leksikal. Beberapa masalah yang muncul dalam proses *tokenization* adalah seperti penghapusan tanda baca, karakter lain seperti tanda kurung, tanda penghubung dan lain sebagainya yang juga membutuhkan pemrosesan juga. Penggunaan utama *tokenization* adalah mengidentifikasi kata kunci yang memiliki makna. (Gurusamy, Vairaprakash, 2014)

2.6.2 Stemming

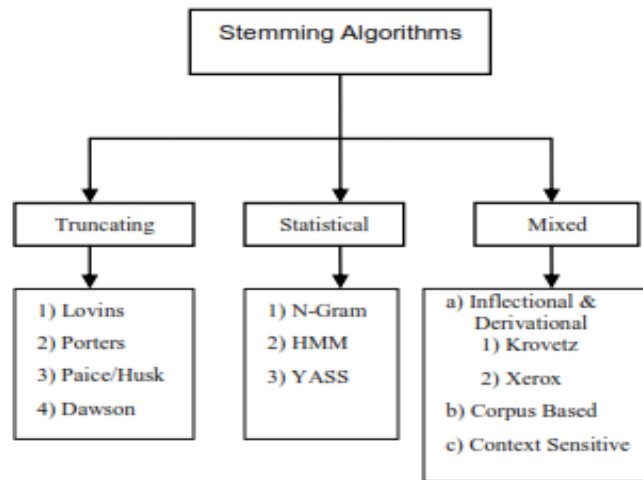
Stemming adalah sebuah proses dimana akhiran kata atau imbuhan lain dihilangkan atau diubah agar bentuk kata yang berbeda dalam cara-cara non-relevan dapat mungkin digabung dan diperlakukan setara. Program komputer yang melakukan transformasi tersebut disebut *stemmer* atau *stemming algorithm*. Hasil keluaran dari *stemming algorithm* dikenal sebagai *stem*. (paice, 2009)

Stemming mengacu pada penghapusan dari setiap imbuhan akhiran (kadang-kadang pada imbuhan lainnya) dari kata yang dimasukkan untuk menghasilkan *stem*. Dua kata akan dianggap setara jika mereka berasal identik (contohnya “memakan” dan “dimakan” berasal dari makan). Metode ini banyak disukai karena cepat. Dalam hal ini, semua kata diubah menjadi *stem* untuk *input* ke sistem dan selanjutnya dilakukan *string matching* sederhana. (paice, tamer, 2009)

Terdapat 2 jenis kesalahan pada *stemming*, antara lain: (jivani, A. G., 2011)

1. *Understemming*, ketika dua kata yang seharusnya berasal dari kata dasar yang sama tetapi tidak. Hal ini dikenal sebagai *false negative*.
2. *Overstemming*, ketika dua kata yang berbeda *stem* tetapi berasal dari kata dasar yang sama. Hal ini dikenal sebagai *false positive*.

Secara luas, algoritma *stemming* dapat diklasifikasikan menjadi 3 grup yaitu: metode *truncating*, metode *statistikal*, dan metode *mixed*. Setiap dari grup ini mempunyai khas sendiri dalam menemukan stem pada varian kata.



Gambar 2.3 Klasifikasi algoritma *stemming*

Sumber: (jivani, A. G., 2011).

1. *Truncating*

Seperti namanya yang dijelaskan menunjukan metode ini dengan menghapus akhiran atau awalan (imbuhan) dari sebuah kata. *Stemming* yang paling dasar yaitu dengan menjaga N huruf dan menghapus sisanya.

Pada metode ini, kata yang lebih pendek dari N akan tetap simpan seperti itu. Peluang *overstemming* akan meningkat jika panjang kata kecil. Contoh algoritmanya yaitu *lovins stemmer*, *porter stemmer*, *paice / husk stemmer*, *dawson stemmer*, dan lain-lainnya.

2. *Statistical*

Stemming ini berdasarkan analisa dan teknik *statistical*. Kebanyakan metode ini menghapus imbuhan awalan tetapi setelah mengimpelmentasikan beberapa prosedur statistik. Contoh yang menggunakan metode ini adalah N-gram, HMM (*Hidden Markov Model*), YASS (*Yet Another Suffix Striper*), dan lain-lainya.

3. *Mixed*

Pada metode mixed terbagi 3 pengelompokkan yaitu:

- a. Metode *inflectional* dan *derivational*.
- b. *Corpus based*.
- c. *Context sensitive*

2.6.3 *Stopwords*

Stopwords adalah kata-kata yang sering muncul dan / atau bertindak sebagai lem untuk membentuk kalimat yang merdu tetapi tidak memiliki kontribusi apapun untuk arti sebuah dokumen (teks). *Stopwords* biasanya kata ganti, kata depan, dan sebagainya. Menghapus *stopwords* menurunkan *overhead* pengindeksan tanpa mengurangi akurasi pencarian teks (atau sistem serupa – seperti *summarization*, klasifikasi, dan lain-lainnya). Pada kebanyakan kasus, *stopwords* merupakan daftar kata-kata biasa yang dalam bentuk *inflect* yang dihapus pada saat tahap *preprocessing*. (galiotou, eleni , et al, 2013)

2.7 *Levenshtein distance*

Levenshtein distance dibuat oleh Vladimir *Levenshtein* pada tahun 1965. Perhitungan *edit distance* didapatkan dari matriks yang digunakan untuk menghitung jumlah perbedaan *string* antara dua *string*. Perhitungan jarak antara dua *string* ini ditentukan dari jumlah minimum operasi perubahan untuk membuat suatu *string* A menjadi *string* B (Adriyani,et al, 2012)

Algoritma ini digunakan secara luas dalam berbagai bidang, misalnya mesin pencari, pengecek ejaan (*spell checking*), pengenalan pembicaraan (*speech recognition*), pengucapan dialek, analisis DNA, pendeteksi pemalsuan, dan lain-lain. (adiwidya B.M.D, 2009)

Pada dasarnya, algoritma ini menghitung jumlah minimum pentransformasian suatu *string* menjadi *string* lain yang meliputi penggantian, penghapusan, dan penyisipan. Algoritma ini digunakan untuk mengoptimalkan pencarian tersebut karena sangat tidak efisien jika dilakukan pencarian setiap kombinasi operasi-operasi *string* tersebut. Oleh karena itu, algoritma ini tergolong program dinamis dalam pencarian nilai minimal tersebut. (Priyana, et al , 2013)

Ada 3 macam operasi utama yang dapat dilakukan oleh algoritma ini yaitu: (Adriyani NM, et al, 2012)

1. Operasi pengubahan karakter. Operasi pengubahan karakter merupakan operasi menukar sebuah karakter dengan karakter lain yang dianggap salah atau tidak sesuai.
2. Operasi penambahan karakter. Operasi penambahan karakter berarti menambahkan karakter ke dalam suatu *string*. Penambahan karakter tidak hanya dilakukan diakhir kata, namun bias ditambahkan diawal maupun disisipkan ditengah *string*.
3. Operasi penghapusan karakter. Operasi penghapusan karakter dilakukan untuk menghilangkan karakter dari suatu *string* yang dianggap tidak perlu atau tidak sesuai.

Metode *levenshtein distance* dapat digunakan untuk melakukan perhitungan beda jarak antara dua *string*. Jarak atau *distance* adalah jumlah minimum dari operasi hapus, *insert* atau substitusi yang dibutuhkan untuk merubah *string* asal (*s*) menjadi *string target* (*t*).Sebagai contoh:

1. Jika *s* adalah “coba” dan *t* adalah “coba”, maka $LD(s,t) = 0$, dikarenakan tidak ada transformasi yang dibutuhkan. Kedua *string* adalah identik.
2. Jika *s* adalah “coba” dan *t* adalah “coka”, maka $LD(s,t) = 1$, dikarenakan satu substitusi (merubah “b” ke “k”) dicukupkan untuk mentransformasi *s* menjadi *t*.

Perhitungan harga peng-*edit*-an pada setiap operasi yang dilakukan adalah sesuai dengan aturan berikut:

- a) $d(a,\epsilon) = 1$ harga untuk menghapus *substring* *a*
- b) $d(\epsilon,a) = 1$ harga untuk penyisipan *substring* *a*
- c) $d(a,b) = 1$ harga untuk substitusi *substring* *a* ke *substring* *b*
- d) $d(a,a) = 0$

Semakin besar angka yang dihasilkan oleh operasi *Levenshtein distance* maka semakin besar perbedaan di antara kedua *string* tersebut. (Gilleland,M , people.cs.pitt.edu)

Berikut adalah tahapan-tahapan dari algoritma *Levenshtein distance* (Gilleland M ,et al, people.cs.pitt.edu)

Tabel 2.2 Tahapan algoritma *levenshtein distance*

Sumber: (Gilleland, 2009).

Step	Description
1	Set n to be the length of s. Set m to be the length of t. If n = 0, return m and exit. If m = 0, return n and exit. Construct a matrix containing 0..m rows and 0..n columns.
2	Initialize the first row to 0..n. Initialize the first column to 0..m.
3	Examine each character of s (i from 1 to n).
4	Examine each character of t (j from 1 to m).
5	If s[i] equals t[j], the cost is 0. If s[i] doesn't equal t[j], the cost is 1.
6	Set cell d[i,j] of the matrix equal to the minimum of: a. The cell immediately above plus 1: d[i-1,j] + 1. (Deletion) b. The cell immediately to the left plus 1: d[i,j-1] + 1. (Insertion) c. The cell diagonally above and to the left plus the cost: d[i-1,j-1] + cost. (Substitution)
7	After the iteration steps (3, 4, 5, 6) are complete, the distance is found in cell d[n,m].

BAB III

METODOLOGI PENELITIAN

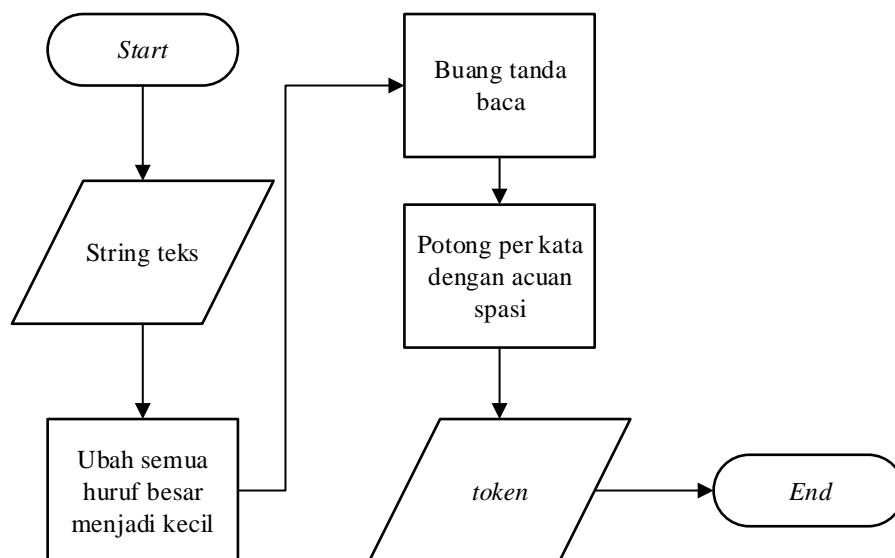
3.1 Pengumpulan Referensi

Referensi yang telah dikumpulkan adalah referensi-referensi mengenai permasalahan plagiarisme dan referensi mengenai penelitian-penelitian yang dilakukan dalam mendeteksi plagiarisme. Dari pengumpulan referensi maka didapat awal digunakannya algoritma *levenshtein distance* dalam mendeteksi plagiarisme dengan mempertimbangkan makna kata (sinonim).

3.2 Analisis

3.2.1 Analisis Proses

Tahapan awal dari sistem pengecekan plagiarisme dimulai dari *load file* dokumen asli dan dokumen plagiat. Sebelum memasukkan proses *levenshtein distance*, dilakukan pemilihan untuk melakukan proses *preprocessing* ataupun tidak. Pada proses *preprocessing* terdapat beberapa proses didalam yaitu *tokenization*, *stopwords* dan *stemming*. Tahapan pertama pada proses *preprocessing* yaitu *tokenization* yang dapat dilihat pada gambar 3.2.

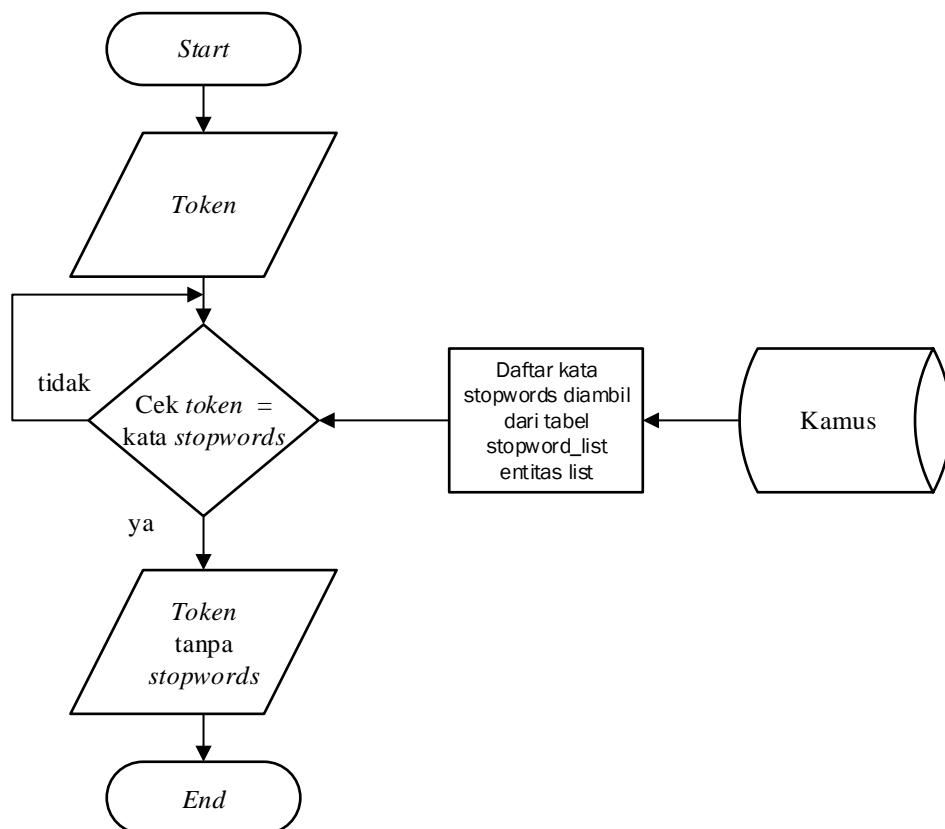


Gambar 3.1 *flowchart* proses *tokenization*

Flowchart untuk tahapan proses *tokenization* dapat dijelaskan sebagai berikut:

1. Teks dari dokumen yang sebelumnya di-*input* oleh pengguna baik itu dokumen asli maupun dokumen plagiat akan diproses oleh sistem. Proses pertama yang itu mengubah semua huruf besar menjadi huruf kecil.
2. Selanjutnya dibuang tanda baca seperti tanda titik, tanda seru, tanda tanya, tanda koma, dan lain-lainya.
3. Langkah berikutnya memotong teks tersebut menjadi per kata dengan acuan pemotongannya yaitu spasi sehingga didapatkan per katanya yang sering disebut sebagai *token*.

Tahapan berikut dari proses *preprocessing* setelah *tokenization* yaitu *stopwords* yang dapat dilihat pada gambar 3.3.

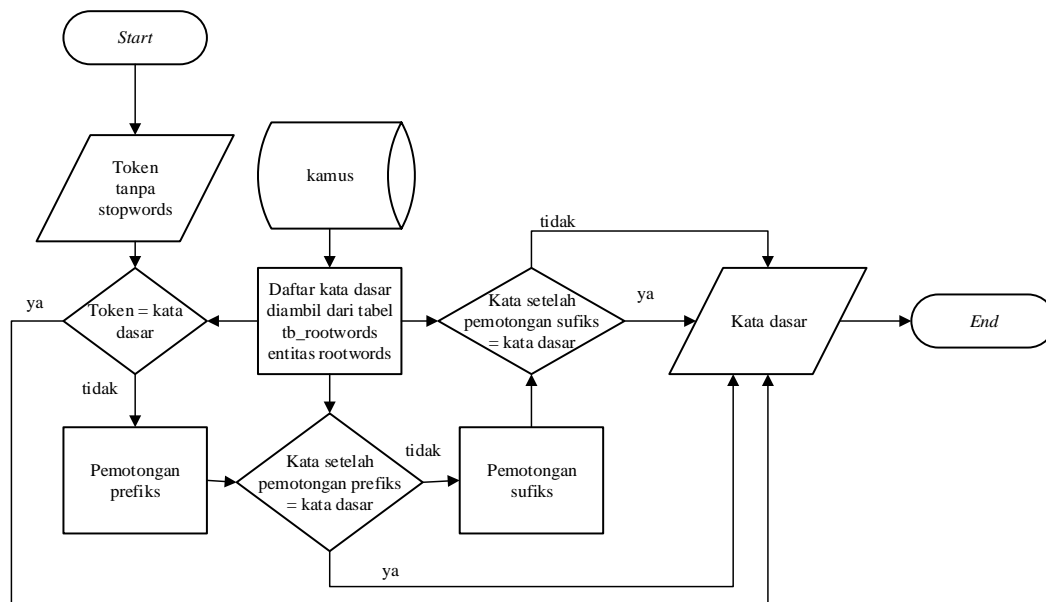


Gambar 3.2 *flowchart* proses *stopwords*

Flowchart untuk tahapan pada proses *stopwords* dapat dijelaskan sebagai berikut:

Setiap *token* hasil dari proses *tokenization* selanjutnya dicek pada kamus data. Pengecekan tersebut dilakukan dengan menyimpan daftar *stopwords* yang ada di kamus data disimpan didalam *memory* yang selanjutnya dibandingkan setiap kata di daftar *stopwords* dengan setiap *token*nya untuk dihapus jika terdapat *token* yang sama dengan kata di daftar *stopwords*nya.

Setelah didapatkan *token* tanpa *stopwords*nya, dilakukan tahapan terakhir dari proses *preprocessing* yaitu *stemming* yang dapat dilihat pada gambar 3.4.



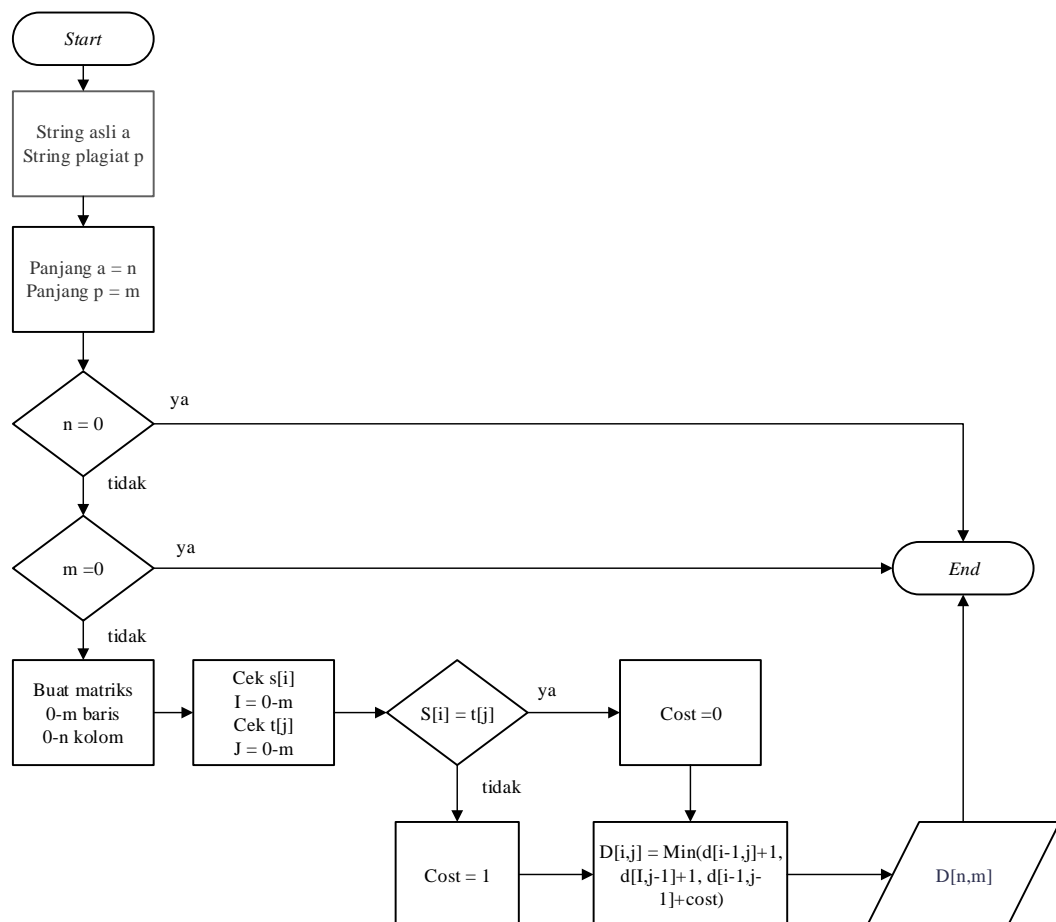
Gambar 3.3 *flowchart* proses *stemming*

Flowchart untuk tahapan pada proses *stemming* dapat dijelaskan sebagai berikut:

Setiap kata yang tidak terhapus dalam proses *stopwords* akan melewati proses *stemming* (penghapusan kata imbuhan). Jika kata yang dicek sama dengan kata dasar di *database* maka proses *stemming* berhenti dan tampilkan hasil. Jika kata yang dicek tidak sama dengan daftar kata dasar yang diambil dari *database* maka akan dilakukan proses penghapusan imbuhan. Proses penghapusan imbuhan prefiks (*Ber-*, *di-*, *ke-*, *me-*, *meng-*, *mem-*, *meny-*, *pe-*, *pem-*, *peng-*, *peny-*, *per-*, *se-*, *ter*) yang merupakan imbuhan yang berada di depan kata dasar (*rootword*). Setelah dilakukan proses penghapusan prefiks maka dilakukan proses pengecekan kata

yang telah dihapus imbuhan prefiksnya ke *database*, jika kata tersebut sama dengan kata dasar yang ada di *database* maka proses akan berhenti dan tampilkan hasil. Jika tidak maka akan dilakukan proses penghapusan imbuhan sufiks(-an, -kan, -i, -nya, -pun, -lah, -kah, -wan, -wati, -asi, -isme, -in, -wi, -if, -ik, -is, -er, -al, -in, -us, -or) yang merupakan imbuhan yang berada di belakang kata dasar. Setelah dilakukan proses penghapusan sufiks maka dilakukan proses pengecekan kata yang telah dihapus imbuhan sufiksnya ke *database*, jika kata tersebut sama dengan kata dasar yang ada di *database* maka proses berhenti dan tampilkan hasilnya. Jika tidak maka akan proses berhenti dan tampilkan hasil.

Setelah tahapan proses *preprocessing* selesai, hasil dari tahapan proses *preprocessing* selanjutnya diolah pada proses *levenshtein distance* dapat dilihat pada gambar berikut:



Gambar 3.4 flowchart proses levenshtein distance

Proses *levenshtein* digambar dengan sebuah matriks yang melibatkan koordinat dari masing masing huruf. Untuk penjelasannya dapat dilihat contoh 2 kalimat dibawah akan ditunjukkan cara perhitungan *levenshtein* tersebut. 2 kalimat tersebut akan diproses sedemikian rupa dengan satu kata per kata.

Contoh kalimat 1: Saya benci dia

Contoh kalimat 2: Saya tidak suka dia

Dari contoh kalimat diatas, “Saya benci dia” dan “Saya tidak suka dia”, kata “Saya” di kalimat pertama sebagai acuan kemudian dibandingkan dengan kata “Saya” di kalimat kedua, kemudian dengan kata “Tidak”, kata “Suka” dan kata “Dia”. Setelah semua selesai kata “Saya” di kalimat pertama, maka dilanjutkan acuan berikutnya yaitu kata “Benci”. Kemudian dibandingkan dengan kata “Saya” di kalimat kedua dan seterusnya sampai semua kata telah diproses.

Dalam *tabel* tersebut, angka di sudut kanan bawah merupakan hasil angka akhir perbedaan dari dua kata tersebut. Contoh seperti tabel “Saya” dan “Saya” dibawah ini dimana angka sudut kanan bawah hasilnya nol yang berarti tidak ada perbedaan yang harus di-*edit* karena semua huruf persis sama. Contoh lain adalah *tabel* “Saya” dan “Suka” dimana hasil angka akhir adalah dua, yang menjelaskan bahwa ada dua huruf yang tidak sama yaitu huruf “a, y” dan “u, k”. Proses ini akan dijelaskan dengan contoh *tabel* “Saya” dan “Suka”.

Berdasarkan *tabel* dibawah, baris pertama selalu berisi huruf (*horizontal*) dan kolom pertama selalu berisi huruf (*vertical*). Baris kedua dan kolom kedua akan selalu berisi urutan angka dari 0 sampai panjang kolom atau baris. Inisial angka 1 dari kolom dan baris adalah huruf awal dari setiap kata kemudian dilanjutkan sampai akhir. Untuk menentukan angka selanjutnya, rumus tersebut digunakan.

Berikut proses perhitungan *levenshtein*:

		S	A	Y	A
	0	1	2	3	4
S	1	0			
U	2				
K	3				
A	4				

Gambar 3.5 hasil perhitungan *levenshtein* huruf 's' dengan 's'

Angka 0 yang di kotak merah didapatkan dari $1+1$ (koordinat Y dari huruf S di kiri), $1+1$ (koordinat X dari huruf S di atas), 0 (angka diagonal) + 0 (cost dimana kasus ini huruf S dan S itu sama maka nilainya 0). Hasilnya adalah 2, 2, 0. Ada 3 hasil angka yang didapat dan yang diambil adalah angka terkecil yaitu 0. Maka didapatkan 0 di kotak merah tersebut.

		S	A	Y	A
	0	1	2	3	4
S	1	0			
U	2	1			
K	3				
A	4				

Gambar 3.6 hasil perhitungan *levenshtein* huruf 's' dengan 'u'

Angka 1 yang di kotak merah didapatkan dari $2+1$ (koordinat Y dari huruf U di kiri), $0+1$ (koordinat X dari huruf S di atas), 1 (angka diagonal) + 1 (cost dimana kasus ini U dan S itu tidak sama maka nilainya 1). Hasilnya adalah 3, 1, 2. Ada 3 hasil angka yang didapat dan yang diambil adalah angka terkecil yaitu 1. Maka didapatkan 1 di kotak merah tersebut.

		S	A	Y	A
	0	1	2	3	4
S	1	0			
U	2	1			
K	3	2			
A	4				

Gambar 3.7 hasil perhitungan *levenshtein* huruf 's' dengan 'k'

Angka 2 yang di kotak merah didapatkan dari $3+1$ (koordinat Y dari huruf K di kiri), $1+1$ (koordinat X dari huruf S di atas), 2 (angka diagonal) + 1 (cost dimana kasus ini K dan S itu tidak sama maka nilainya 1). Hasilnya adalah 4, 2, 3. Ada 3 hasil angka yang didapat dan yang diambil adalah angka terkecil yaitu 2. Maka didapatkan 2 di kotak merah tersebut.

		S	A	Y	A
	0	1	2	3	4
S	1	0			
U	2	1			
K	3	2			
A	4	3			

Gambar 3.8 hasil perhitungan *levenshtein* huruf 's' dengan 'a'

Angka 1 yang di kotak merah didapatkan dari $4+1$ (koordinat Y dari huruf A di kiri), $2+1$ (koordinat X dari huruf S di atas), 3 (angka diagonal) + 1 (cost dimana kasus ini U dan S itu tidak sama maka nilainya 1). Hasilnya adalah 5, 3, 4. Ada 3 hasil angka yang didapat dan yang diambil adalah angka terkecil yaitu 3. Maka didapatkan 3 di kotak merah tersebut.

		S	A	Y	A
	0	1	2	3	4
S	1	0	1		
U	2	1			
K	3	2			
A	4	3			

Gambar 3.9 hasil perhitungan *levenshtein* huruf 'a' dengan 's'

Angka 1 yang di kotak merah didapatkan dari $0+1$ (koordinat Y dari huruf S di kiri), $2+1$ (koordinat X dari huruf A di atas), 1 (angka diagonal) + 1 (cost dimana kasus ini S dan A itu tidak sama maka nilainya 1). Hasilnya adalah 1, 3, 2. Ada 3 hasil angka yang didapat dan yang diambil adalah angka terkecil yaitu 1. Maka didapatkan 1 di kotak merah tersebut. Proses ini berlangsung sampai dapat hasil akhir yaitu:

		S	A	Y	A
	0	1	2	3	4
S	1	0	1	2	3
U	2	1	1	2	3
K	3	2	2	2	3
A	4	3	2	3	2

Gambar 3.10 hasil akhir perhitungan *levenshtein* kata "saya" dengan "suka"

Setelah selesai mengisi semua isi matriks, hasil akhir untuk kata "saya" dengan "suka" diambil nilai pada koordinat x paling terakhir dan y paling terakhir yaitu: 2. Angka 2 ini menyatakan bahwa ada dua huruf yang berbeda pada *string matching* kata "saya" dengan "suka".

Demikian hasil perhitungan dari 2 kalimat tersebut berada dibawah ini:

		S	A	Y	A
	0	1	2	3	4
S	1	0	1	2	3
A	2	1	0	1	2
Y	3	1	1	0	1
A	4	2	1	1	0

		S	A	Y	A
	0	1	2	3	4
T	1	1	2	3	4
I	2	2	2	3	4
D	3	3	3	3	4
A	4	4	3	4	3
K	5	5	4	4	4

		S	A	Y	A
	0	1	2	3	4
S	1	0	1	2	3
U	2	1	1	2	3
K	3	2	2	2	3
A	4	3	2	3	2

		S	A	Y	A
	0	1	2	3	4
D	1	1	2	3	4
I	2	2	2	3	4
A	3	3	2	3	3

Gambar 3.11 hasil perhitungan *levenshtein* kata "saya" dengan kata contoh pada kalimat ke-2

		B	E	N	C	I
	0	1	2	3	4	5
S	1	1	2	3	4	5
A	2	2	2	3	4	5
Y	3	3	3	3	4	5
A	4	4	4	4	4	5

		B	E	N	C	I
	0	1	2	3	4	5
T	1	1	2	3	4	5
I	2	2	2	3	4	4
D	3	3	3	3	4	5
A	4	4	4	4	4	5
K	5	5	5	5	5	5

		B	E	N	C	I
	0	1	2	3	4	5
S	1	1	2	3	4	5
U	2	2	2	3	4	5
K	3	3	3	3	4	5
A	4	4	4	4	4	5

		B	E	N	C	I
	0	1	2	3	4	5
D	1	1	2	3	4	5
I	2	2	2	3	4	4
A	3	3	3	3	4	5

Gambar 3.12 hasil perhitungan *levenshtein* kata "benci" dengan kata pada contoh kalimat ke-2

		D	I	A
	0	1	2	3
S	1	1	2	3
A	2	2	2	2
Y	3	3	3	3
A	4	4	4	3

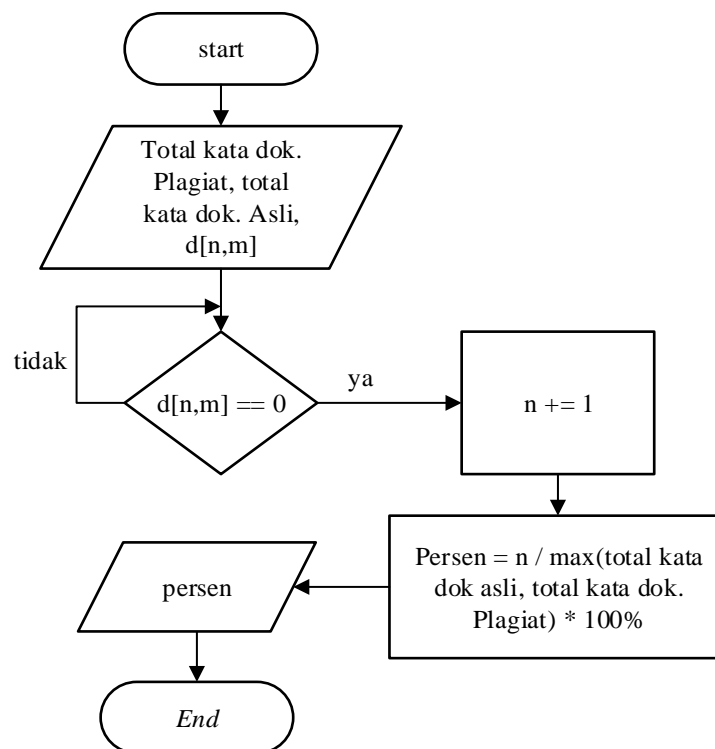
		D	I	A
	0	1	2	3
T	1	1	2	3
I	2	2	1	2
D	3	2	2	2
A	4	3	3	2
K	5	4	4	3

		D	I	A
	0	1	2	3
S	1	1	2	3
U	2	2	2	3
K	3	3	3	3
A	4	4	4	3

		D	I	A
	0	1	2	3
D	1	0	1	2
I	2	1	0	1
A	3	2	1	0

Gambar 3.13 hasil perhitungan *levenshtein* kata "dia" dengan kata pada contoh kalimat ke-2

Proses perhitungan persentase kemiripan menggunakan *levenshtein distance* dapat dilihat pada pemodelan berikut:



Gambar 3.14 *flowchart* perhitungan persentase proses *levenshtein distance*

Flowchart tahapan perhitungan persentase proses *levenshtein distance* dapat dijelaskan sebagai berikut:

Proses perhitungan dimulai dari menghitung total jumlah kata dari dokumen yang asli dan plagiat serta menghitung nilai beda dari kata yang dicek (nilai *levenshtein*). Nilai beda dari proses *levenshtein* yang ditampung di *variable* $d[n,m]$, cek nilai beda dari proses *levenshtein* jika nilainya sama dengan 0 maka *variable* n akan ditambahkan nilai 1, jika tidak cek kembali nilai di *variable* $d[n,m]$. Kemudian dilakukan proses perhitungan dengan persamaan:

$$persen = \frac{n}{\max(\text{total kata dokumen asli}, \text{total kata dokumen plagiat})} \times 100\%$$

Sebagai contoh Proses perhitungan nilai persentase *levenshtein* dapat dilihat dari contoh kalimat proses *levenshtein* yang berupa kalimat “saya tidak suka dia” dan “saya benci dia” didapatkan nilai:

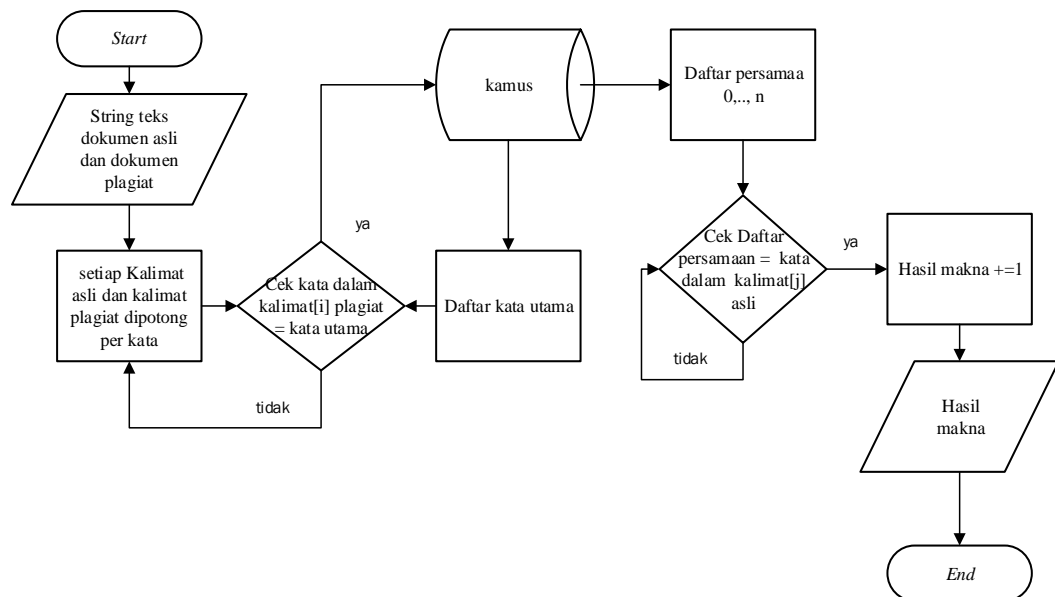
$n = 2$, total kata dari kalimat “saya tidak suka dia” = 4 dan total kata dari kalimat “saya benci dia” = 3

maka perhitungan persen:

$$persen = \frac{2}{\max(3, 4)} \times 100\%$$

$$persen = 50 \%$$

Proses pengecekan kata atau kalimat dari segi sinonimnya masih dilakukan dengan cara *brute force* dengan mencocokkan *string* atau kalimat dengan *database* kamus *thesaurus* atau sinonim bahasa Indonesia. tahapan proses *preprocessing* selanjutnya diolah pada proses *levenshtein distance* dapat dilihat pada gambar berikut:



Gambar 3.15 *flowchart* proses sinonim

Penjelasan dari *flowchart* pada gambar 3.14 dapat dilihat pada 2 contoh paragraf dari 2 *file* atau dokumen yang akan diuji dari segi sinonim atau kesamaan maknanya. Berikut adalah kalimat dari kedua *file* atau dokumen yang belum mengalami proses *stopwords* maupun *stemming*.

Contoh kalimat dari *file* dokumen asli:

“Memang perkembangan zaman yang modern ini mendekatkan yang jauh tapi menjauhkan yang dekat. Kita semakin jarang untuk berinteraksi dengan sekitar yang nyata dan yang terlihat oleh mata. Suatu saat nanti perkembangan teknologi yang modern bakal membunuh manusia sebagai penciptanya. Itu terbukti kita seakan mati jika kita kehilangan perangkat yang maju tersebut dibandingkan ditinggalkan oleh orang yang kita sayangi.”

Contoh kalimat dari *file* dokumen plagiat:

“Kemajuan di abad yang maju ini mengakrabkan yang jauh tapi menjauhkan yang akrab. Kita semakin kurang berhubungan dengan yang ada sekeliling yang jelas tampak oleh mata. Satu saat manusia akan binasa dan kemajuan teknologi yang maju yang akan melenyapkannya. Itu benar karena orang seakan binasa karena kehilangan alat yang modern dibandingkan ditinggalkan oleh orang yang kita cintai .”

Proses selanjutnya adalah yang akan dilakukan adalah proses *tokenization* untuk memotong atau memisahkan paragraf tersebut menjadi beberapa kalimat.

Berikut adalah hasil setelah proses *tokenization*:

Kalimat dari *file* asli:

Memang perkembangan zaman yang modern ini mendekatkan yang jauh tapi menjauhkan yang dekat
Kita semakin jarang untuk berinteraksi dengan sekitar yang nyata dan yang terlihat oleh mata
Suatu saat nanti perkembangan teknologi yang modern bakal membunuh manusia sebagai penciptanya
Itu terbukti kita seakan mati jika kita kehilangan perangkat yang maju tersebut dibandingkan ditinggalkan oleh orang yang kita sayangi.

Kalimat dari *file* plagiat:

Kemajuan di abad yang maju ini mengakrabkan yang jauh tapi menjauhkan yang akrab
Kita semakin kurang berhubungan dengan yang ada sekeliling yang jelas tampak oleh mata
Satu saat manusia akan binasa dan kemajuan teknologi yang maju yang akan melenyapkannya
Itu benar karena orang seakan binasa karena kehilangan alat yang modern dibandingkan ditinggalkan oleh orang yang kita cintai .

Proses selanjutnya adalah proses *stemming* dan *stopwords* yang akan menghilangkan beberapa kata yang tidak terlalu penting sehingga dapat diproses dengan mudah. Berikut adalah hasil setelah di proses *stopwords* dan *stemming*.

Asli	
kalimat	1. Kembang zaman modern dekat jauh
	2. Jarang interaksi
	3. Kembang teknologi modern bunuh manusia cipta
	4. Bukti akan mati hilang angkat maju banding tinggal sayang
Plagiat	
kalimat	1. Maju abad maju akrab jauh akrab
	2. Hubung keliling
	3. Maju binasa maju teknologi maju lenyap
	4. Akan binasa hilang alat modern banding tinggal cinta

Gambar 3.16 Hasil *preprocessing* pada contoh proses sinonim

Proses selanjutnya adalah pencocokkan kalimat yang asli dengan yang plagiat. Kalimat – kalimat dari kedua *file* tersebut akan di potong (*split*) per kata untuk proses pencocokkan secara makna. Berikut adalah proses *split* yang dilakukan.

Kembang	Teknologi	Modern	Bunuh	Manusia	cipt
---------	-----------	--------	-------	---------	------

Proses pencocokkan yang dilakukan dengan cara mencocokkan kata dari *file* asli dan plagiat dan mengecek kedua kalimat tersebut berada dalam satu baris. Jika berada dalam 1 baris maka dianggap sama atau plagiat.

Plagiat					
Maju	Abad	Maju	Akrab	Jauh	akrab

Asli				
Kembang	Zaman	Modern	Dekat	Jauh

Gambar 3.17 Hasil kalimat ke-1 plagiat dengan asli

Plagiat	
Hubung	Keliling

Asli				
Kembang	Zaman	Modern	Dekat	Jauh

Plagiat	
Hubung	Keliling

Plagiat	
Jarang	interaksi

Plagiat	
Hubung	Keliling

Asli					
Kembang	Teknologi	Modern	Bunuh	Manusia	cipta

Plagiat	
Hubung	Keliling

Asli								
Bukti	Akan	Mati	Hilang	Angkat	maju	Banding	tinggal	sayang

Gambar 3.18 Hasil kalimat ke-2 plagiat dengan asli

Plagiat					
Maju	binasa	Maju	teknologi	Maju	lenyap

Asli				
Kembang	Zaman	Modern	Dekat	Jauh

Gambar 3.19 Hasil kalimat ke-3 plagiat dengan asli

Plagiat							
Akan	Binasa	Hilang	Alat	Modern	Banding	Tinggal	cinta

Asli							
Kembang	Zaman	Modern	Dekat	Jauh			

Plagiat							
Akan	Binasa	Hilang	Alat	Modern	Banding	Tinggal	cinta

Asli							
Jarang		interaksi					

Plagiat							
Akan	Binasa	Hilang	Alat	Modern	Banding	Tinggal	cinta

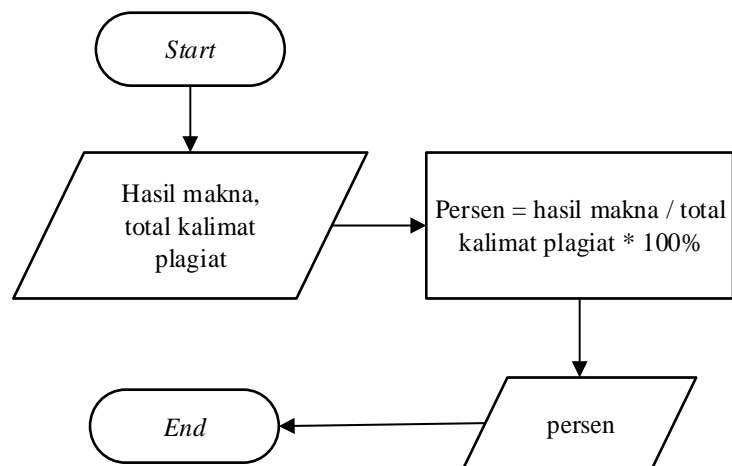
Asli							
Kembang	Teknologi	Modern	Bunuh	Manusia	cipta		

Plagiat							
Akan	Binasa	Hilang	Alat	Modern	Banding	Tinggal	cinta

Asli							
Bukti	Akan	Mati	Hilang	Angkat	maju	Banding	tinggal sayang

Gambar 3.20 Hasil kalimat ke-4 plagiat dengan asli

Proses perhitungan persentase sinonim dapat dilihat pada pemodelan berikut:



Gambar 3.21 *flowchart* proses perhitungan proses sinonim

Proses perhitungan persentase sinonim dimulai dari menghitung jumlah kalimat plagiat yang memiliki makna yang sama dengan kalimat asli. Kemudian proses perhitungan persentase dilakukan dengan persamaan:

$$persen = \frac{\text{hasil makna}}{\text{total kalimat plagiat}} \times 100\%$$

Sebagai contoh Proses perhitungan nilai persentase sinonim dapat dilihat dari contoh kalimat proses sinonim yang berupa kalimat pada gambar 3.17 didapatkan nilai:

Hasil makna = 3, total kalimat plagiat = 4

maka perhitungan persen:

$$persen = \frac{3}{4} \times 100\%$$

$$persen = 75 \%$$

3.2.2 Analisis Kebutuhan

Bagian ini merupakan penjelasan mengenai kebutuhan-kebutuhan yang diperlukan oleh sistem. Bagian analisis dibedakan atas 2 yaitu analisis kebutuhan fungsional dan analisis kebutuhan non fungsional yang diperlukan untuk mencapai tujuan yang ingin dicapai.

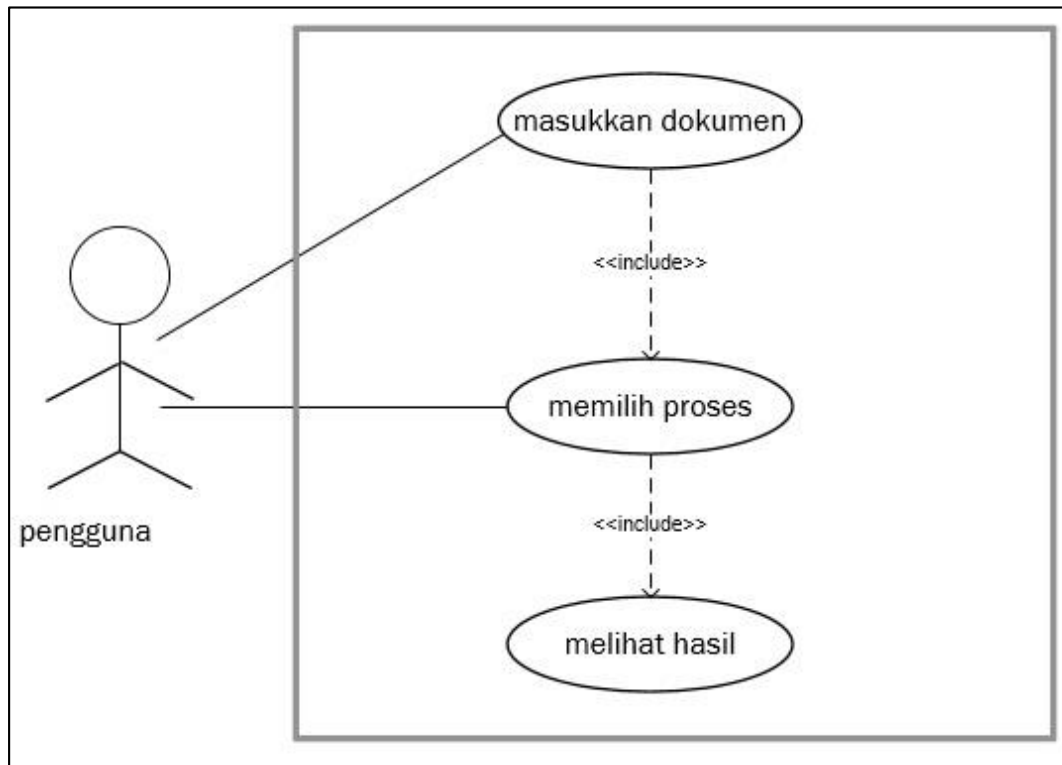
3.2.2.1 Analisis Fungsional

Kebutuhan fungsional adalah pernyataan layanan sistem yang harus disediakan. Bagaimana sistem bereaksi pada *input* tertentu dan bagaimana perilaku sistem pada situasi tertentu. Oleh sebab itu, maka dirumuskan beberapa kebutuhan fungsional pada judul tugas akhir ini:

1. *User* hanya dapat memasukkan *file* berupa pdf pada pengecekan plagiat.
2. *User* hanya bisa memasukkan 2 yaitu 1 dokumen yang original atau asli dan 1 dokumen yang merupakan *file* yang akan diuji.
3. *User* dapat memilih antara 2 pilihan dalam tipe pemrosesan dokumen baik *preprocessing* maupun *non-preprocessing*.
4. Sistem dapat berjalan pada dokumen *file* yang berupa pdf dan dokumen berbahasa Indonesia.
5. Sistem tidak akan berjalan pada *file* berupa pdf yang berisi gambar ataupun hasil *scanning*.

6. Sistem akan mengambil informasi dari kamus data seperti kata-kata *stopwords* maupun kata dasar untuk mempermudah proses *preprocessing* seperti *stopwords* dan *stemming* dan proses pengecekan makna.

Untuk lebih jelasnya kebutuhan fungsional, dapat dilihat *use case* pada gambar berikut ini.



Gambar 3.22 *Use Case diagram* Pengujian Deteksi Tingkat Plagiarisme dengan Mempertimbangkan Makna Kata dengan Menggunakan Algoritma *Levenshtein distance*

Penjelasan dari *use case* dari gambar diatas dibuat dalam sebuah narasi yang dapat dilihat pada tabel dibawah ini.

Tabel 3.1 Narasi *Use Case* Masukkan Dokumen

Nama UseCase	Masukkan dokumen.	
Aktor	Pengguna.	
Prakondisi	Pengguna sudah harus memilih <i>file</i> yang akan diuji.	
Deskripsi	Proses ini adalah tindakan untuk memasukkan <i>file</i> yang akan diuji.	
Sasaran	Isi dari <i>file</i> tersebut akan ditampilkan.	
	Aksi Aktor	Respon Sistem
	1. Menekan tombol load. 3. Mencari dan memilih Dokumen yang akan diuji.	2. Menampilkan tampilan <i>file</i> select dialog box. 4. Menampilkan isi <i>dokumen</i> .

Tabel 3.2 Narasi Memilih Proses

Nama UseCase	Memilih Proses.	
Aktor	Pengguna.	
Prakondisi	Pengguna sudah dapat memastikan dokumen yang akan diuji.	
Deskripsi	Proses ini adalah proses untuk memilih tipe pengecekan atau tipe pengujian.	
Sasaran	Pengguna dapat melihat detil proses yang dilakukan pada saat pengecekan atau pengujian.	
	Aksi Aktor	Respon Sistem

	1. Memilih antara 2 <i>radiobutton</i> yang berupa <i>Preprocessing</i> dan <i>non-preprocessing</i> . 3. Menunggu proses hingga selesai. 5. Melihat setiap langkah dalam pengujian.	2. Menampilkan tombol proses. 4. Menampilkan tahapan proses dari tipe pengujian yang telah dipilih.
--	--	--

Tabel 3.3 Narasi *Use Case* melihat hasil

Nama UseCase	Melihat Hasil.	
Aktor	Pengguna.	
Prakondisi	Pengguna harus menekan tombol Result.	
Deskripsi	Proses ini adalah proses yang menunjukkan hasil akhir dari pengujian atau pengecekan yang dilakukan.	
Sasaran	Menampilkan hasil akhir dalam bentuk persentase.	
	Aksi Aktor	Respon Sistem
		1. Menampilkan persentase nilai dari hasil perhitungan persamaan kata dan perhitungan <i>levenshtein</i> . 2. Menentukan suatu <i>file</i> dianggap plagiat atau tidak.

	3. Melihat hasil akhir dari proses pengujian yang dilakukan.	
--	--	--

3.2.2.2 Analisis Non Fungsional

Kebutuhan non fungsional adalah kebutuhan yang secara tidak langsung memberikan peran dalam fitur tertentu didalam sistem. Kebutuhan yang tidak memerlukan tahapan peng-*input*-an, tahapan proses maupun tahapan *output* kebutuhan non fungsional sebaiknya dipenuhi, karena dapat menentukan sistem akan digunakan oleh pengguna atau tidak. Berikut adalah kebutuhan non-fungsional pada judul tugas ini:

1. Sistem dapat menampilkan detil proses (tahapan) dari setiap tipe pemrosesan yang telah dipilih.
2. Sistem dapat menampilkan informasi jumlah kata / kalimat dari kedua dokumen yang sedang diuji.
3. Sistem dapat memberikan peringatan ketika *user* melakukan pengujian ulang terhadap dokumen.

3.3 Perancangan

3.3.1 Kamus data

Kamus data atau *data directory* yang digunakan dalam sistem ini hanya dalam proses pengecekan sehingga tidak memiliki hubungan (relasi) yang tercipta dalam kamus data tersebut sehingga tidak diperlukan proses normalisasi. Kamus data tersebut terdiri dari informasi-informasi yang didapatkan berasal dari kamus thesaurus Bahasa Indonesia, kata dasar Bahasa Indonesia dan *stopwords list* Bahasa Indonesia. Khusus *tabel unneeded* pada kamus data digunakan untuk menginsert kata-kata baru atau kata-kata yang tidak terdapat di *tabel* kamus thesaurus bahasa Indonesia.

Berikut ini adalah kamus data dalam sistem yang diusulkan:

1. Kamus_Tesaurus = kata_u + persamaan 1 + + persamaan 147

2. *Stopwords list* = ID + List

3. Rootword = rootword

4. *unneeded* = Added

Dari kamus data tersebut, maka didapatkan struktur tabel yang diimplementasikan kedalam DBMS Microsoft Access 2013.

Tabel 3.4 Struktur tabel *kamus_tesaurus*

Field	Type	Keterangan
kata_u	Short text	-
Persamaan 1	Short text	-
⋮	⋮	⋮
persamaan 147	Short text	-

Tabel 3.5 Struktur tabel *stopwords list*

Field	Type	Keterangan
ID	Auto number	-
List	Short text	-

Tabel 3.6 Struktur tabel *tb_rootword*

Field	Type	Keterangan
Rootword	Short text	-

Tabel 3.7 Struktur tabel *unneeded*

Field	Type	Keterangan
Added	Short text	-

3.3.2 Perancangan *User Interface*

User interface merupakan salah satu bagian penting dalam sebuah sistem, karena bagian ini yang menghubungkan pengguna dengan sistem. Tampilan user

interface yang baik dan menarik akan mempermudah pengguna dalam menggunakan sistem yang akan dibuat.

3.3.2.1 Form Input File

The screenshot shows a window titled 'PLAGIARISM CHECKER' with three tabs: 'Input File', 'Detil Proses', and 'Hasil Akhir'. The 'Input File' tab is active. It contains the following elements:

- 1**: A 'Load' button next to the 'File Original' label.
- 2**: A text box for displaying information from the original document.
- 3**: A large text area for displaying the original document's content in detail.
- 4**: A 'Load' button next to the 'File yang akan diuji' label.
- 5**: A text box for displaying information about the document to be tested.
- 6**: A large text area for displaying the document to be tested in detail.
- 7**: An 'Options' section with two radio buttons: 'Preprocessing' and 'Non Preprocessing'.
- 8**: A 'Proses' button.
- A progress bar at the bottom of the form.

Gambar 3.23 Form Input file

Pada gambar 3.23 yang ada diatas merupakan *form* untuk melakukan peng-*input*-an dokumen yang akan dilakukan pengecekan.

Keterangan:

1. Pada *form* tampilan *input file* merupakan tombol untuk meng-*input* atau memasukkan dokumen. Dokumen yang di-*input* / dimasukkan merupakan dokumen yang menjadi acuan atau dokumen yang original.
2. Pada *form* tampilan *input file* merupakan *textbox* atau tempat dimana informasi dari dokumen yang telah di masukkan sebelumnya ditampilkan.
3. Pada *form* tampilan *input file* merupakan *text area* atau tempat dimana isi dokumen yang berupa teks akan ditampilkan secara rinci.
4. Pada *form* tampilan *input file* merupakan tombol untuk meng-*input* atau memasukkan dokumen yang akan diuji.

5. Pada *form* tampilan *input file* merupakan *textbox* atau tempat dimana informasi dari dokumen ditampilkan.
6. Pada *form* tampilan *input file* merupakan *text area* atau tempat dimana isi dokumen yang berupa teks akan ditampilkan secara rinci.
7. Pada *form* tampilan *input file* merupakan *radiobutton* yang digunakan untuk memilih antara 2 tipe proses pengecekan yaitu proses pengecekan secara *preprocessing* dan *non-preprocessing*.
8. Pada *form* tampilan *input file* merupakan tombol untuk memulai proses pengecekan terhadap kedua dokumen yang telah dimasukkan.

3.3.2.2 Form Detil Proses (*Preprocessing*)

The screenshot shows a software window titled 'PLAGIARISM CHECKER' with three tabs: 'Input File', 'Detil Proses', and 'Hasil Akhir'. The 'Detil Proses' tab is selected. It displays the results of the preprocessing steps for a document. The steps are numbered 1 through 7:

- 1. Tokenization: Shows a paragraph of text split into individual words.
- 2. Stopwords: Shows words that have been removed from the text.
- 3. Stemming: Shows words reduced to their root form.
- 4. Levenshtein distance: Shows a list of distances between words.
- 5. Sinonim: Shows a list of synonyms for the words.
- 6. Hasil Akhir: The final result of the preprocessing.
- 7. Kembali: A button to return to the previous screen.

Gambar 3.24 *Form Detil Proses (Preprocessing)*

Pada gambar 3.24 yang ada diatas merupakan *form* yang muncul jika pengguna memilih tipe pengecekan dengan *preprocessing*. *Form* ini menampilkan setiap tahapan proses pengecekan secara detil.

Keterangan:

1. Pada *form* detil proses (*Preprocessing*) merupakan tempat yang menampilkan tampilan hasil *tokenization* atau proses pemotongan sebuah paragraf menjadi per kalimat pada dokumen yang dimasukkan sebelumnya.

2. Pada *form* detil proses (*preprocessing*) merupakan tempat untuk menampilkan tampilan hasil *stopwords* atau proses penghapusan kata-kata yang tidak penting dalam sebuah kalimat.
3. Pada *form* detil proses (*preprocessing*) merupakan tempat untuk menampilkan tampilan hasil *stemming* atau proses penghilangan imbuhan yang terdapat pada setiap kata pada kalimat.
4. Pada *form* detil proses (*preprocessing*) merupakan tempat untuk menampilkan hasil dari proses pengecekan kata yang dianggap plagiat dari segi kesamaan kata atau *string matching*.
5. Pada *form* detil proses (*preprocessing*) merupakan tempat untuk menampilkan hasil dari proses pengecekan kata yang dianggap plagiat dari segi persamaan kata atau sinonim.
6. Pada *form* detil proses (*preprocessing*) merupakan tombol untuk melanjutkan ke *form* selanjutnya yaitu *form* hasil.
7. Pada *form* detil proses (*preprocessing*) merupakan tombol untuk kembali ke *form* input file.

3.3.2.3 Form Detil Proses (Non-preprocessing)

PLAGIARISM CHECKER

Input File Detil Proses Hasil Akhir

Memangg perkembangan zaman yang modern ini mendekatkan yang jauh tapi menjauhkan yangdekat. Kita semakin jarang untuk berinteraksi dengan sekitar yang nyata dan yang terlihat oleh mata. Suatu saat nanti

Kemajuan di abad yang maju ini mengakrabkan yang jauh tapi menjauhkan yang akrab. Kita semakin kurang berhubungan dengan yang ada sekeliling yang jelas tampak oleh mata. Satu saat manusia akan binasa dan kemajuan teknologi yang maju

kembang
zaman
modern
dekat

maju
abad
akrab
jauh

Tokenization

Levenshtein distance

5
4
5
5

Sinonim

3

7 Kembali

6 Hasil Akhir

Gambar 3.25 Form Detil Proses (Non-preprocessing)

Pada gambar 3.25 yang ada diatas merupakan *form* yang muncul jika pengguna memilih tipe pengecekan dengan *non-preprocessing*. *Form* ini menampilkan setiap tahapan proses pengecekan secara detil.

Keterangan:

1. Pada *form* detil proses (*non-preprocessing*) merupakan tempat yang menampilkan tampilan hasil *tokenization* atau proses pemotongan sebuah paragraf menjadi per kalimat pada dokumen yang dimasukkan sebelumnya.
2. Pada *form* detil proses (*non-preprocessing*) merupakan tempat untuk menampilkan hasil dari proses pengecekan kata yang dianggap plagiat dari segi kesamaan kata atau *string matching*.
3. Pada *form* detil proses (*non-preprocessing*) merupakan tempat untuk menampilkan hasil dari proses pengecekan kata yang dianggap plagiat dari segi persamaan kata atau sinonim.
4. Pada *form* detil proses (*non-preprocessing*) merupakan tombol untuk melanjutkan ke *form* selanjutnya yaitu *form* hasil.
5. Pada *form* detil proses (*non-preprocessing*) merupakan tombol untuk kembali ke *form input file*.

3.3.2.4 Form Hasil

PLAGIARISM CHECKER

Input File | **Detil Proses** | Hasil Akhir

File Original

Nama file: Asli.pdf
directory: c:/dokumen
ukuran file: 96kb

File yang diuji

Nama file: plagiat.pdf
directory: c:/dokumen
ukuran file: 96kb

Hasil pengecekan

Levenshtein: 86 %
Synonym: 13 %

Levenshtein: 8 Kata
Synonym: 6 Kata

Total kata file original: 115 Kata
Total kata file plagiat: 96 Kata

Info

0% Tidak ada plagiat
<30% Plagiat rendah
30%-70% Plagiat menengah
71%-100% Plagiat tinggi

plagiat tinggi
plagiat rendah

Kembali

cek ulang

Gambar 3.26 Form Hasil

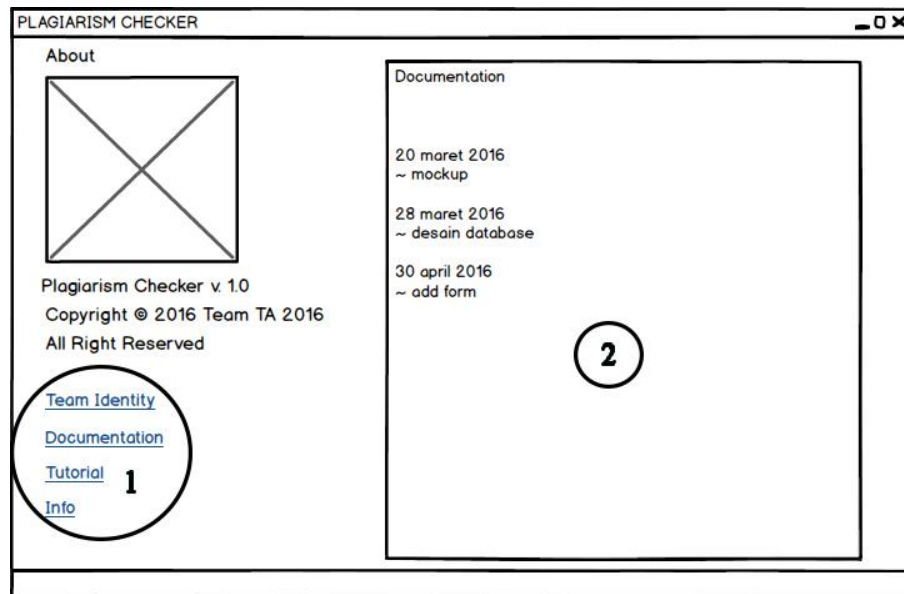
Pada gambar 3.26 yang ada diatas merupakan *form* yang menampilkan hasil pengecekan dokumen.

Keterangan:

1. Pada *form* hasil merupakan tampilan yang menampilkan informasi dari setiap dokumen.
2. Pada *form* hasil merupakan tampilan yang menampilkan hasil pengecekan dalam bentuk persentase.
3. Pada *form* hasil merupakan tampilan yang menampilkan jumlah kata pada setiap dokumen baik dokumen yang original dengan dokumen yang diuji, dan menampilkan jumlah kata menurut kesamaan kata (*string matching*) dan persamaan kata (sinonim)
4. Pada *form* hasil merupakan *label* yang akan menunjukkan status dari dokumen dapat dikatakan plagiat atau tidak
5. Pada *form* hasil merupakan tampilan yang menampilkan informasi tingkatan dari plagiat
6. Pada *form* hasil merupakan tombol untuk melakukan pengecekan ulang terhadap dokumen yang dicek ataupun melakukan pengecekan dokumen lain

7. Pada *form* hasil merupakan tombol untuk kembali ke *form* detail proses

3.3.2.5 Form about



Gambar 3.27 *Form About*

Pada gambar 3.27 yang ada diatas merupakan *form* yang menampilkan informasi tentang perangkat lunak.

Keterangan:

1. Pada *form about* merupakan *link label* yang menjadi penghubung untuk menampilkan setiap informasi sesuai dengan kategori yang ada.
2. Pada *form about* merupakan *text area* atau bagian yang akan menampilkan setiap informasi yang muncul pada saat *link label* di klik.

BAB IV

HASIL DAN PENGUJIAN

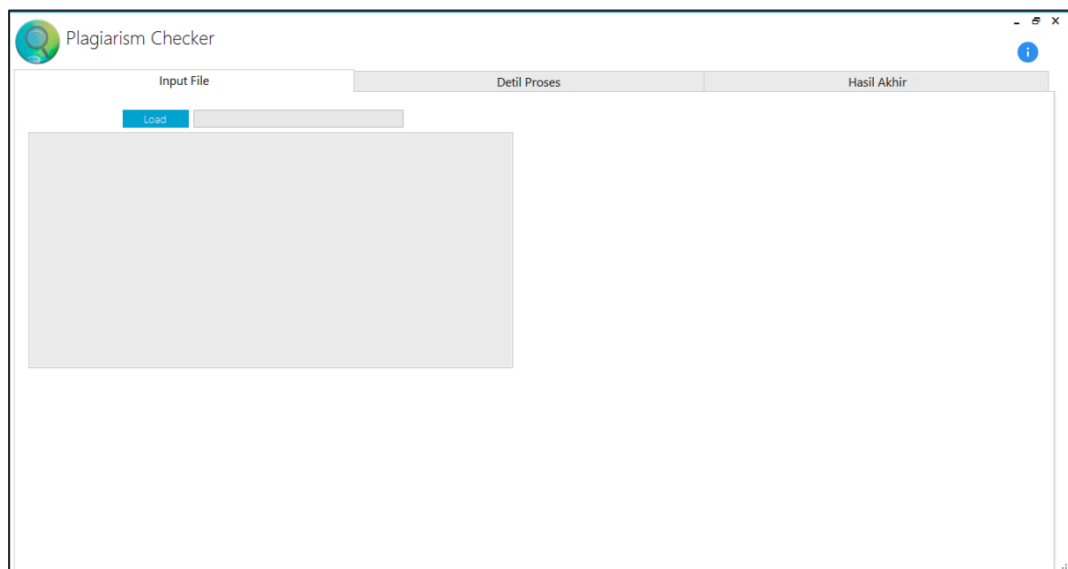
Pada bagian bab ini akan dijelaskan tentang langkah-langkah dalam menggunakan sistem serta melakukan pengujian terhadap kemampuan sistem. Sistem yang dibahas adalah “Sistem Pengujian Deteksi Tingkat Plagiarisme dengan Mempertimbangkan Makna Kata dengan Menggunakan Algoritma *Levenshtein distance*. Bahan untuk pengujian berupa *file* dokumen berformat *.pdf.

4.1 Hasil

Pada subbab ini akan dijelaskan tampilan dari sistem serta pembahasan mengenai cara menggunakan sistem. Berikut tampilan sistem:

1. Tampilan *input file*

Berikut adalah tampilan awal dari aplikasi pada saat dijalankan, dapat dilihat pada gambar 4.1. Pada tampilan ini pengguna harus melakukan klik pada tombol *load* untuk menginput *file* yang menjadi acuan dalam proses pengecekan yang akan dilakukan.

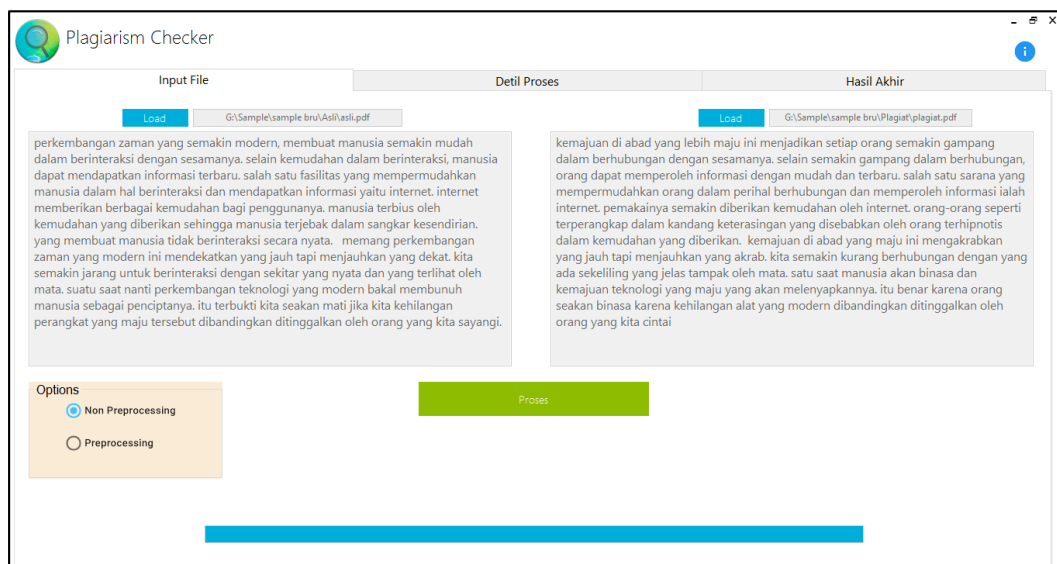


Gambar 4.1 Tampilan *input file*

2. Tampilan proses input *file* dan pemilihan tipe pengecekan

Berikut adalah tampilan pada saat *file* telah di-*input* dan telah melakukan pemilihan tipe pengecekan. Pada tampilan dibawah adalah tampilan pada saat pengguna telah menginput *file* yang menjadi acuan dan menginput *file* yang akan diuji.

Pada saat kedua *file* yang akan dicek telah di-*input* maka akan keluar pilihan tipe pemrosesan yang terdiri dari tipe *non-preprocessing* dan *preprocessing*. Kedua tipe tersebut memiliki tujuan untuk mempersiapkan kedua *file* yang akan diuji untuk lanjut ke proses selanjutnya yaitu proses *levenshtein* dan sinonim

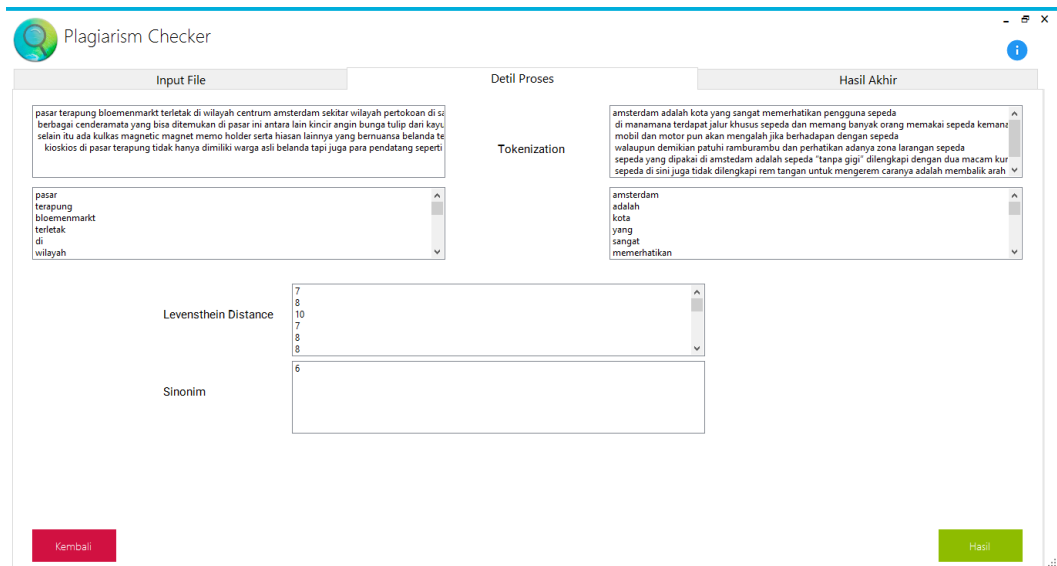


Gambar 4.2 Tampilan proses *input file* dan pemilihan tipe pengecekan

3. Tampilan Detil Proses (*non-preprocessing*)

Berikut tampilan detil proses pada saat melakukan pemilihan tipe pengecekan *non-preprocessing*. Pada tampilan detil proses yang *non-preprocessing* ini akan menampilkan tiap – tiap proses yang seharusnya berada di belakang layar. Tetapi ditampilkan untuk menunjukkan kepada pengguna proses secara detil. Terdapat 2 *listbox* yang berisi kata – kata, itu merupakan tampilan kalimat yang telah di *split* atau dipisah per spasi sehingga kata terurut per baris.

Pada bagian bawah terdapat 2 *listbox* yang berisi angka – angka, itu merupakan hasil dari proses pengecekan dengan menggunakan algoritma *levenshtein* dan pengecekan secara sinonim. Pada bagian *listbox* pertama merupakan hasil pengecekan *levenshtein* yang mengecek *string* / kata yang sama atau *string matching*. Jika ada kata yang sama secara *string* maka hasil di *listbox* tersebut adalah ‘0’ dan jika ada kata yang tidak sama secara *string* maka hasil di *listbox* tersebut sesuai jumlah perbedaan huruf pada kata.



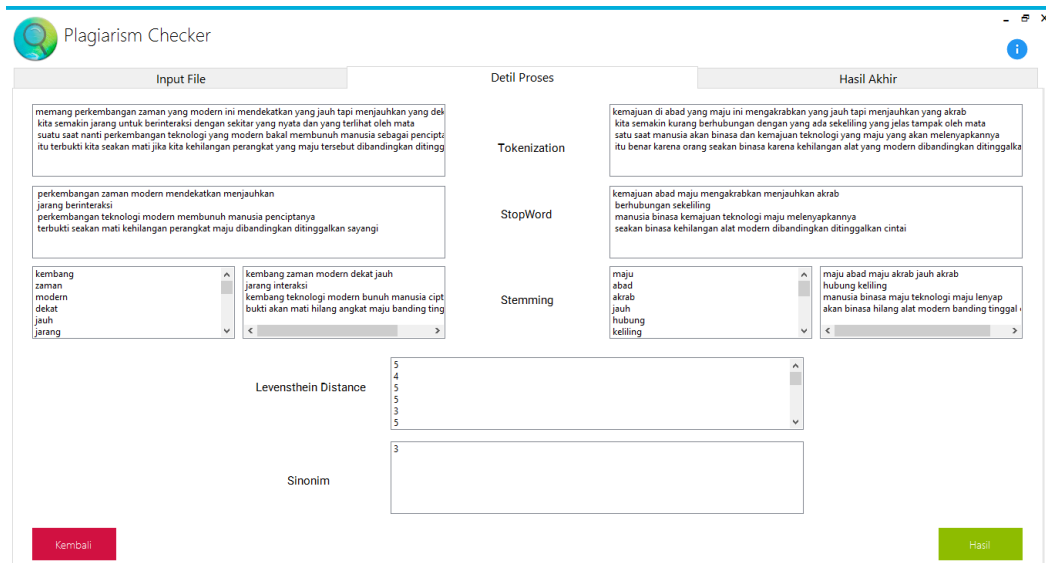
Gambar 4.3 Tampilan Detil Proses (*non-preprocessing*)

4. Tampilan Detil Proses (*Preprocessing*)

Berikut tampilan detil proses pada saat melakukan pemilihan tipe pengecekan *preprocessing*. Pada tampilan detil proses ini akan menampilkan tiap – tiap proses dalam pengecekan *file*. Proses – proses pada *preprocessing* ditampilkan secara menyeluruh mulai dari proses *tokenization*, *stopwords*, *stemming* serta hasil pengecekan awal dari algoritma *levenshtein* dan sinonim.

Pada *listbox* bagian paling atas adalah proses *tokenization* yang membagi sebuah paragraf menjadi beberapa kalimat. Pada bagian bawahnya merupakan *listbox* untuk proses *stopwords* yang menghapus atau menghilangkan sebagian kata yang tidak diperlukan. Pada bagian bawah dari *listbox stopwords* adalah *listbox* untuk proses *stemming* dibagian terdapat masing – masing 2 *listbox* antara yang *file*

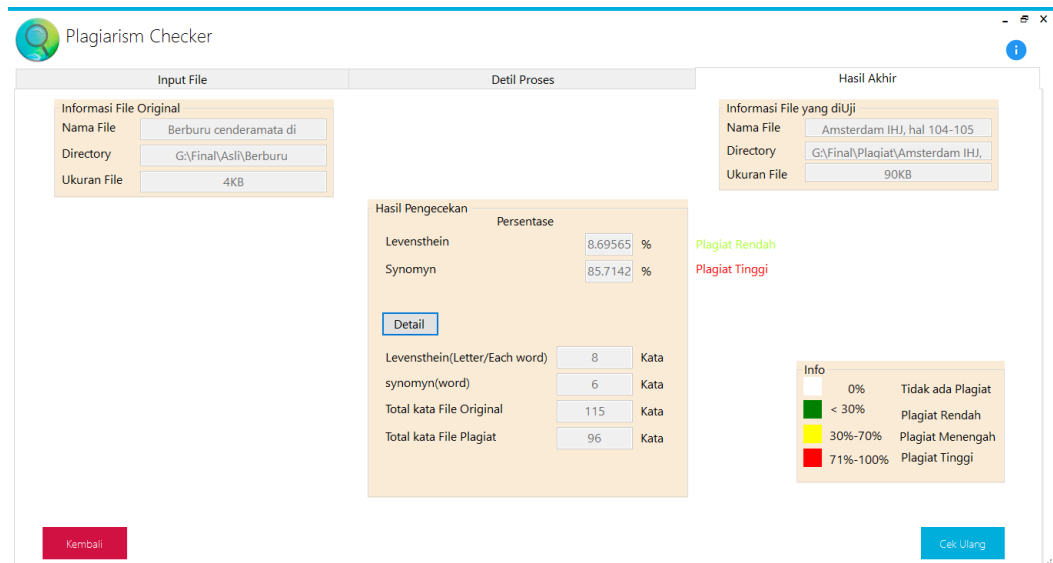
acuan dan *file* plagiat. Bagian *listbox* yang menampilkan kalimat yang di *split* per kata dan dibuat berurut dimaksudkan untuk proses pengecekan dengan menggunakan algoritma *levenshtein* untuk mencari kesamaan pada kata atau *string*. Sedangkan *listbox* yang menampilkan kalimat dimaksudkan untuk proses pengecekan sinonim.



Gambar 4.4 Tampilan Detil Proses (*preprocessing*)

5. Tampilan hasil akhir

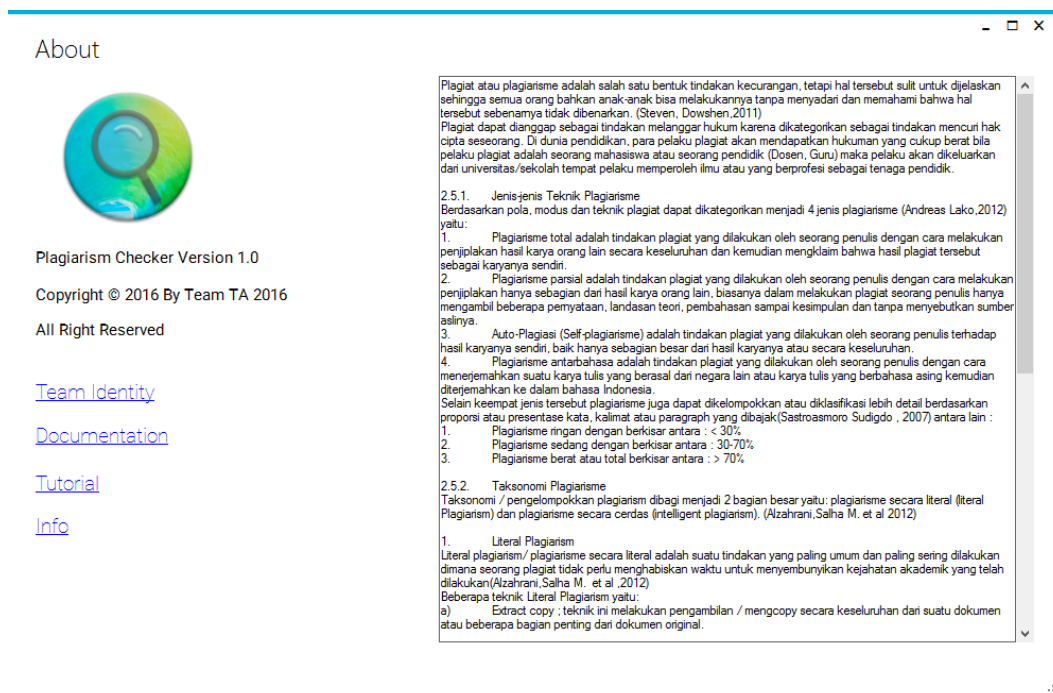
Berikut tampilan hasil akhir dari proses pengecekan yang telah dilakukan. Pada bagian tampilan ini akan menampilkan hasil perhitungan dari proses pengecekan dengan menggunakan *levenshtein* maupun pengecekan secara sinonim. Pada tampilan ini juga menampilkan informasi dari *file* yang telah di-*input*, menampilkan jumlah kata pada kedua *file* yang telah di-*input*.



Gambar 4.5 Tampilan hasil akhir

6. Tampilan *about*

Berikut tampilan *about* dari aplikasi, pada tampilan ini akan menampilkan informasi sekilas mengenai plagiarisme, dokumentasi atau track record dalam pengembangan sistem. Selain menampilkan dokumenstasi dan informasi juga dapat menampilkan tutorial atau tata cara penggunaan pada sistem.



Gambar 4.6 Tampilan *about*

4.2 Pengujian

Untuk menguji kemampuan sistem dalam menentukan suatu *file* dinyatakan sebagai plagiat atau tidak maka dilakukan pengujian dengan menggunakan 10 *file* dokumen yang original dan 10 *file* yang dianggap plagiat. Prosedur pengujian adalah memasukkan *file* yang original dan *file* yang akan diuji kedalam sistem, Kemudian proses pengecekan akan dimulai. Hasil dari pengecekan dalam bentuk persentase (%). Ada 3 hasil yang akan didapatkan yaitu hasil dari *levenshtein distance*, hasil dari sinonim, dan total.

Berikut ini adalah tabel daftar dokumen asli dan dokumen pembanding (plagiat) yang akan diuji. Proses pengujian dilakukan sebanyak 2 kali yaitu pengujian normal antara dokumen asli dan yang dianggap plagiat dan pengujian secara random antara dokumen asli dan dokumen pembanding.

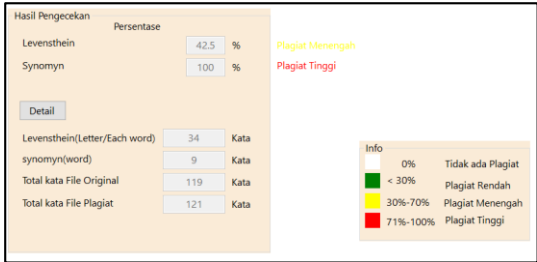
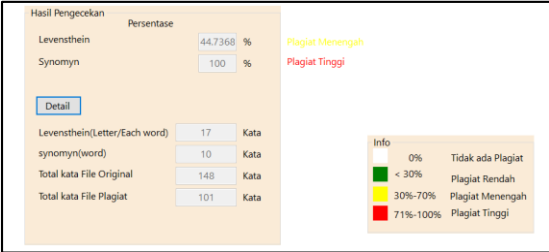
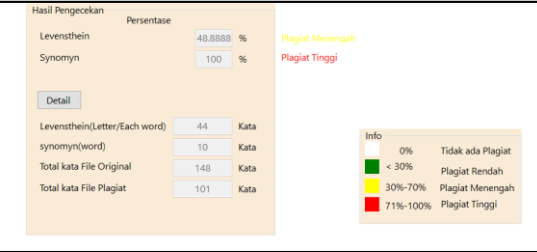
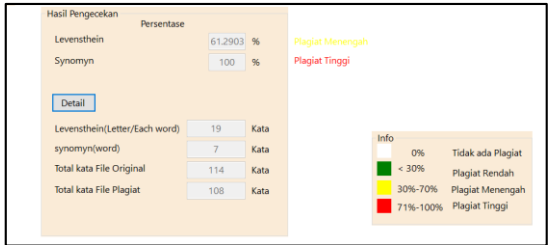
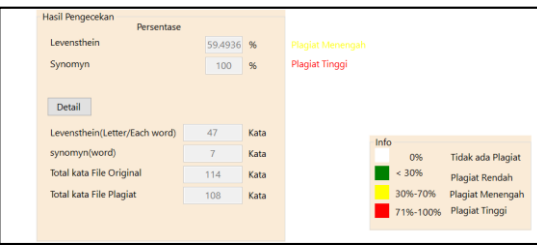
Tabel 4.1 Daftar dokumen asli dan dokumen plagiat (pembanding)

No.	Dokumen asli	Isi dokumen Asli	Dokumen plagiat	Isi dokumen plagiat
1	Asli.pdf	“Dampak dari perkembangan zaman yang modern”	Plagiat.pdf	“kemajuan di abad yang lebih maju”
2	Asli2.pdf	“Guru pahlawan nasional”	Plagiat2.pdf	“pengajar pahlawan bangsa”
3	Asli3.pdf	“Nasionalisme bangsa”	Plagiat3.pdf	“Semangat kebangsaan indonesia”
4	Asli4.pdf	“Akibat dari perang”	Plagiat4.pdf	“Bentrokan senjata ”
5	Pidato Michelle Obama (25 agustus 2008)	“Pidato di democratic national convention 2008”	Pidato Melania Trump (18 juli 2016)	“Pidato di republic national convention 2016”
6	Rashomona kutagawa Ryunosuke	“Kota Kyoto yang sepi”	Perempuan Tua dalam rashomon datang ari murtono	“Kota Kyoto yang ramai dan permai mulai sepi”

7	Menguliti Bakal Calon Presiden Indonesia 2014 (Satriawan kompas, 19 januari 2012)	“Pemilu untuk tahun 2014”	Nanda Dyani Amilla, Menguliti Bakal Calon Presiden 2014 (11 maret 2014)	“Kasak-kusuk tentang siapa calon presiden Indonesia tahun 2014”
8	Berburu cenderamata di bloemenmarkt (kompas 6 agustus 2011)	“Berburu cenderamata di bloemenmarkt”	Novel amsterdam IHJ, hal 238 (arumi, E 2013)	“oleh –oleh dari bloemenmarkt”
9	jalan-jalan di amsterdam (Javamilk, 29 january 2009)	“jalan-jalan di amsterdam”	Amsterdam IHJ, hal 104-105 (arumi,E ,2013)	“perjalanan di kota amsterdam”
10	Keukenhof Secuil Taman Surga Di Daratan Eropa (rohman,2010)	“Keukenhof Secuil Taman Surga Di Daratan Eropa (Belanda)”	Amsterdam Ik Hou Van Je (Arumi E,2013)	Keindahan taman di kota keukenhof

Berikut ini adalah *tabel* hasil pengujian dengan menggunakan aplikasi pengujian deteksi tingkat plagiarisme dengan mempertimbangkan makna kata dengan menggunakan algoritma *levenshtein distance*.

Tabel 4.2 pengujian dokumen asli dengan dokumen plagiat

No .	Dokumen asli	Dokumen plagiat	<i>Preprocessing</i> (Persentase <i>Levenshtein</i> dan sinonim)	<i>Non-preprocessing</i> (Persentase <i>Levenshtein</i> dan sinonim)
1	Asli.pdf	Plagiat.pdf		
2	Asli2.pdf	Plagiat2.pdf		
3	Asli3.pdf	Plagiat3.pdf		

8	Berburu cenderamata di bloemenmarkt (kompas 6 agustus 2011)	Novel amsterdam IHJ, hal 238 (arumi, E 2013)	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>83.0508 %</div> </div> <div> <div>Synonymn</div> <div>66.6666 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>49 Kata</div> </div> <div> <div>synonymn(word)</div> <div>2 Kata</div> </div> <div> <div>Total kata File Original</div> <div>115 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>99 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagiat Tinggi</div> <div>Plagiat Menengah</div> </div> </div> </div>	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>79.3478 %</div> </div> <div> <div>Synonymn</div> <div>100 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>73 Kata</div> </div> <div> <div>synonymn(word)</div> <div>3 Kata</div> </div> <div> <div>Total kata File Original</div> <div>115 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>99 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagit Tinggi</div> <div>Plagiat Tinggi</div> </div> </div> </div>
9	jalan-jalan di amsterdam (Javamilk, 29 january 2009)	Amsterdam IHJ, hal 104-105 (arumi,E ,2013)	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>70.2702 %</div> </div> <div> <div>Synonymn</div> <div>42.8571 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>26 Kata</div> </div> <div> <div>synonymn(word)</div> <div>3 Kata</div> </div> <div> <div>Total kata File Original</div> <div>107 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>96 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagiat Menengah</div> <div>Plagiat Menengah</div> </div> </div> </div>	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>75.5814 %</div> </div> <div> <div>Synonymn</div> <div>85.7142 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>65 Kata</div> </div> <div> <div>synonymn(word)</div> <div>6 Kata</div> </div> <div> <div>Total kata File Original</div> <div>107 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>96 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagit Tinggi</div> <div>Plagiat Tinggi</div> </div> </div> </div>
10	Keukenhof Secuil Taman Surga Di Daratan Eropa (rohman,2010)	Amsterdam Ik Hou Van Je (Arumi E,2013)	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>88.8888 %</div> </div> <div> <div>Synonymn</div> <div>0 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>24 Kata</div> </div> <div> <div>synonymn(word)</div> <div>0 Kata</div> </div> <div> <div>Total kata File Original</div> <div>48 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>56 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagiat Tinggi</div> </div> </div> </div>	<div> <div> <div>Hasil Pengecekan</div> <div> <div>Persentase</div> <div> <div>Levensthein</div> <div>72.5490 %</div> </div> <div> <div>Synonymn</div> <div>100 %</div> </div> </div> <div> <div>Detail</div> <div> <div>Levensthein(Letter/Each word)</div> <div>37 Kata</div> </div> <div> <div>synonymn(word)</div> <div>3 Kata</div> </div> <div> <div>Total kata File Original</div> <div>48 Kata</div> </div> <div> <div>Total kata File Plagiat</div> <div>56 Kata</div> </div> </div> <div> <div>Info</div> <div> <div>0%</div> <div>Tidak ada Plagiat</div> </div> <div> <div>< 30%</div> <div>Plagiat Rendah</div> </div> <div> <div>30%-70%</div> <div>Plagiat Menengah</div> </div> <div> <div>71%-100%</div> <div>Plagiat Tinggi</div> </div> </div> <div> <div>Plagit Tinggi</div> <div>Plagiat Tinggi</div> </div> </div> </div>

Tabel 4.3 Pengujian terhadap dokumen asli dan pembanding

No	File Asli	File Plagiat	Preprocessing		Non-preprocessing	
			Levenshtein (%)	Sinonim (%)	Levenshtein (%)	Sinonim (%)
1	Asli.pdf	Plagiat.pdf	44,117	77,777	42,5	100
2	Asli2.pdf	Plagiat2.pdf	44,736	100	48,888	100
3	Asli3.pdf	Plagiat3.pdf	61,290	100	59,493	100
4	Asli4.pdf	Plagiat4.pdf	47,222	100	43,478	100
5	Pidato Michelle Obama (25 agustus 2008)	Pidato Melania Trump (18 juli 2016)	60,714	0	59,420	100
6	Rashomona kutagawa Ryunosuke	Perempuan Tua dalam rashomon dadang ari murtono	76,119	90	62,616	90
7	Menguliti Bakal Calon Presiden Indonesia 2014 (Satriawan kompas, 19 januari 2012)	Nanda Dyani Amilla, Menguliti Bakal Calon Presiden 2014 (11 maret 2014)	84,905	71,428	76,530	85,714
8	Berburu cenderamata di bloemenmarkt (kompas 6 agustus 2011)	Novel amsterdam IHJ, hal 238 (arumi, E 2013)	83,050	66,666	79,347	100
9	jalan-jalan di amsterdam (Javamilk, 29 january 2009)	Amsterdam IHJ, hal 104-105 (arumi,E ,2013)	70,2702	42,857	75,581	85,714
10	Keukenhof Secuil Taman Surga Di Daratan Eropa (rohman,2010)	Amsterdam Ik Hou Van Je (Arumi E,2013)	88,888	0	72,549	100

Dari tabel 4.2 dapat disimpulkan bahwa pada pengujian yang dilakukan dengan menggunakan dokumen asli dan dokumen pembanding (plagiat) menunjukkan hasil yang berbeda sesuai dengan tipe pemrosesan yang digunakan dalam pengecekan. Pada proses *preprocessing*, proses-proses yang dilakukan adalah proses *tokenization*, *stopwords* dan *stemming*. Sedangkan proses *non-preprocessing*, proses yang dilakukan hanya *tokenization* untuk memecahkan sebuah paragraf menjadi beberapa kalimat dan juga kata-katanya. Pengujian pada *non-preprocessing* menggunakan proses *tokenization* mempermudah pengerjaan proses *levenshtein distance* dan sinonim tanpa menghilangkan informasi yang ada.

Pada tabel pengujian tersebut menunjukkan dengan menggunakan tipe pemrosesan *preprocessing* dokumen yang memiliki kemiripan dari segi kata (*string matching*) menghasilkan nilai persentase menengah ke atas antara 44% hingga 88 persen, sedangkan dokumen yang memiliki kemiripan dari segi makna (sinonim) menghasilkan nilai persentase yang cukup tinggi antara 40% hingga 100% ini menunjukkan bahwa dokumen yang diuji memiliki kesamaan makna kata dalam kalimat cukup tinggi. Adapun dokumen yang menunjukkan nilai persentase 0% itu menunjukkan tidak adanya kata yang memiliki makna yang sama tetapi pada hasil *levenshtein distance* menunjukkan persentase yang tinggi sehingga dapat disimpulkan penjiplakannya kata-kata hampir semuanya seperti pada pengujian no 10.

Sedangkan dengan menggunakan tipe pemrosesan *non-preprocessing* dokumen yang memiliki kemiripan dari segi kata (*string matching*) menghasilkan persentase yang lebih kecil dibanding hasil pada tipe pemrosesan *preprocessing* antara 42% hingga 79% ini disebabkan karena adanya proses-proses untuk mempersiapkan dokumen untuk dilakukan proses perhitungan lebih lanjut. Proses-proses yang dimaksud adalah proses *stopwords* dan *stemming* yang mengakibatkan sebagian kata mengalami penghilangan atau penghapusan sehingga membuat sebagian informasi hilang, sedangkan dokumen yang memiliki kemiripan dari segi makna menghasilkan persentase yang lebih besar dibanding hasil pada tipe pemrosesan secara *preprocessing* yaitu antara 82% hingga 100% ini disebabkan

tidak adanya kehilangan informasi karena tidak proses *stopwords* dan *stemming* pada tipe pemrosesan *non-preprocessing*.

Pada pengujian no 1 sampai no 4 adalah sampel yang dibuat sendiri untuk menguji makna dengan merubah kata yang memiliki makna sama pada *file* asli ke *file* pembandingnya. Hasil pengujiannya cukup sesuai yang diharapkan yaitu pada proses *preprocessing* persentasenya 77.77%-100% dan pada proses *non-preprocessing* 100%.

Tabel 4.4 pengujian dokumen asli dengan dokumen plagiat secara acak

No .	Dokumen asli	Dokumen plagiat	Preprocessing (Persentase <i>Levenshtein</i> dan sinonim)	Non-preprocessing (Persentase <i>Levenshtein</i> dan sinonim)
1	Asli.pdf	Plagiat 4.pdf	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 14.2857 %</p> <p>Synonym 60 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 5 Kata</p> <p>synonym(word) 3 Kata</p> <p>Total kata File Original 119 Kata</p> <p>Total kata File Plagiat 78 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 12.5 %</p> <p>Synonym 100 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 10 Kata</p> <p>synonym(word) 5 Kata</p> <p>Total kata File Original 119 Kata</p> <p>Total kata File Plagiat 78 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>
2	Asli 2.pdf	Pidato Melania Trump(18 juli 2016).pdf	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 13.1579 %</p> <p>Synonym 40 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 5 Kata</p> <p>synonym(word) 2 Kata</p> <p>Total kata File Original 148 Kata</p> <p>Total kata File Plagiat 92 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 18.8888 %</p> <p>Synonym 100 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 17 Kata</p> <p>synonym(word) 5 Kata</p> <p>Total kata File Original 148 Kata</p> <p>Total kata File Plagiat 92 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>
3	Asli 3.pdf	Perempuan Tua dalam rashomon dadang ari murtono.pdf	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 0 %</p> <p>Synonym 30 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 0 Kata</p> <p>synonym(word) 3 Kata</p> <p>Total kata File Original 114 Kata</p> <p>Total kata File Plagiat 129 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levenshtein 10.2803 %</p> <p>Synonym 90 %</p> <p>Detail</p> <p>Levenshtein(Letter/Each word) 11 Kata</p> <p>synonym(word) 9 Kata</p> <p>Total kata File Original 114 Kata</p> <p>Total kata File Plagiat 129 Kata</p> </div> <div> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> </div>

4	<p>Berburu cenderamata di bloemenmarkt(k ompas 6 agustus 2011).pdf</p>	<p>Amsterdam IHJ, hal 104-105(arumi,E ,2013).pdf</p>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 3.38983 %</p> <p>Synonym 28.5714 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 2 Kata</p> <p>synonym(word) 2 Kata</p> <p>Total kata File Original 115 Kata</p> <p>Total kata File Plagiat 96 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> <p>Plagiat Rendah</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 8.69565 %</p> <p>Synonym 85.7142 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 8 Kata</p> <p>synonym(word) 6 Kata</p> <p>Total kata File Original 115 Kata</p> <p>Total kata File Plagiat 96 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> <p>Plagiat Tinggi</p> </div> </div>
5	<p>jalan-jalan di amsterdam(Java milk, 29 january 2009).pdf</p>	<p>Nanda Dyani Amilla, Menguliti Bakal Calon Presiden 2014(11 maret 2014).pdf</p>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 0 %</p> <p>Synonym 14.2857 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 0 Kata</p> <p>synonym(word) 1 Kata</p> <p>Total kata File Original 107 Kata</p> <p>Total kata File Plagiat 116 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 12.7659 %</p> <p>Synonym 85.7142 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 12 Kata</p> <p>synonym(word) 6 Kata</p> <p>Total kata File Original 107 Kata</p> <p>Total kata File Plagiat 116 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> <p>Plagiat Tinggi</p> </div> </div>
6	<p>Keukenhof Secuil Taman Surga Di Daratan Eropa(rohman,2 010).pdf</p>	<p>Novel amsterdam IHJ, hal 238(arumi, E 2013).pdf</p>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 3.63636 %</p> <p>Synonym 33.3333 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 2 Kata</p> <p>synonym(word) 1 Kata</p> <p>Total kata File Original 48 Kata</p> <p>Total kata File Plagiat 99 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> <p>Plagiat Menengah</p> </div> </div>	<div> <div> <p>Hasil Pengecekan</p> <p>Persentase</p> <p>Levensthein 9.63855 %</p> <p>Synonym 66.6666 %</p> <p>Detail</p> <p>Levensthein(Letter/Each word) 8 Kata</p> <p>synonym(word) 2 Kata</p> <p>Total kata File Original 48 Kata</p> <p>Total kata File Plagiat 99 Kata</p> <p>Info</p> <p>0% Tidak ada Plagiat</p> <p>< 30% Plagiat Rendah</p> <p>30%-70% Plagiat Menengah</p> <p>71%-100% Plagiat Tinggi</p> </div> <div> <p>Plagiat Rendah</p> <p>Plagiat Menengah</p> </div> </div>

Tabel 4.5 Pengujian dengan sample yang di acak

No	File Asli	File Plagiat	Preprocessing		Non-preprocessing	
			Levenshtein (%)	Sinonim (%)	Levenshtein (%)	Sinonim (%)
1	Asli.pdf	Plagiat 4.pdf	14,285	60	12,5	100
2	Asli 2.pdf	Pidato Melania Trump(18 juli 2016).pdf	13,157	40	18,888	100
3	Asli 3.pdf	Perempuan Tua dalam rashomon dadang ari murtono.pdf	0	30	10,280	90
4	Berburu cenderamata di bloemenmarkt(kompas 6 agustus 2011).pdf	Amsterdam IHJ, hal 104-105(arumi,E ,2013).pdf	3,389	28,571	8,695	85,714
5	jalan-jalan di amsterdam(Javamilk, 29 january 2009).pdf	Nanda Dyani Amilla, Menguliti Bakal Calon Presiden 2014(11 maret 2014).pdf	0	14,285	12,765	85,714
6	Keukenhof Secuil Taman Surga Di Daratan Eropa(rohman,2010).pdf	Novel amsterdam IHJ, hal 238(arumi, E 2013).pdf	3,636	33,333	9,638	66.666
7	Menguliti Bakal Calon Presiden Indonesia 2014(Satriawan kompas, 19 januari 2012).pdf	Amsterdam Ik Hou Van Je(Arumi E,2013).pdf	0	33,333	6,122	66.666
8	Pidato Michelle Obama(25 agustus 2008).pdf	Perempuan Tua dalam rashomon dadang ari murtono.pdf	0	30	5,607	100
9	Rashomona kutagawa Ryunosuke.pdf	Nanda Dyani Amilla, Menguliti Bakal Calon Presiden 2014(11 maret 2014).pdf	1,818	42,857	7,4468	100
10	Asli 4.pdf	Plagiat.pdf	13,888	66,666	14,102	100

Pada tabel 4.3 pengujian dilakukan dengan dokumen yang di acak pengujian ini berbeda dengan pengujian pada tabel 4.2. Pada tabel pengujian 4.3 dapat disimpulkan bahwa meskipun dokumen diuji secara acak masih menghasilkan presentase yang cukup tinggi. Itu terbukti dari pengujian dengan menggunakan tipe pemrosesan *non-preprocessing* meskipun dokumen yang diuji berbeda tetap menghasilkan nilai yang tinggi pada pengujian dokumen secara kemiripan makna hingga mencapai persentase 100%. Ini disebabkan karena proses pembobotan atau pemberian nilai pada setiap kalimat dan kebanyakan makna yang sama yaitu kata-kata dari *stopwords*, misalnya “di”, “dan”, “sebab”, “adalah”, “dengan”, dan lain-lainnya. Pada saat proses pengecekan sinonim setiap kalimat yang memiliki 1 kata yang makna nya sama dengan kalimat yang diuji akan diberi bobot atau *point* 1. Sehingga pada saat menggunakan tipe pemrosesan ini, nilai persentase sinonim tinggi. Sedangkan pada nilai dari segi kemiripan kata (*string matching*) menghasilkan nilai yang rendah dikarenakan tidak ada kata yang sama dalam jumlah yang besar.

Pada tipe pemrosesan *preprocessing* persentase dari segi kemiripan kata (*string matching*) tetap rendah dikarenakan isi dari dokumen telah kehilangan sebagian kata karena adanya proses *stopwords* dan *stemming* persentase nilai *levenshtein* berkisar antara 1,8% hingga 14%. Adapun hasil yang mendapatkan nilai 0% karena tidak kata yang sama. Sedangkan nilai persentase dari kemiripan makna (sinonim) juga menghasilkan nilai yang relative tinggi, persentase berkisar antara 14% hingga 60 persen disebabkan karena proses pembobotan pada proses pengecekan kalimat.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian yang dilakukan terhadap kinerja sistem, dapat diambil kesimpulan sebagai berikut:

1. Dari pengujian yang dilakukan sistem dapat mengenali berapa banyak kalimat yang memiliki kemiripan makna (sinonim). Serta dapat mengenali dan mendeteksi kata yang mirip (*string matching*).
2. Nilai persentase sinonim yang dihasilkan cukup baik pada pengecekan dokumen secara normal maupun pengecekan secara *random*, meskipun pada pengecekan *random* nilai persentase sinonim tetap tinggi dikarenakan pemberian bobot pada kalimat yang memiliki makna yang sama dengan kalimat yang diuji akan diberi bobot 1 sehingga menghasilkan persentase yang tinggi.
3. Nilai persentase *levensthein* yang dihasilkan cukup memuaskan dengan rata-rata 66% dengan proses *preprocessing* dan *non-preprocessing* didapatkan rata-rata 62%. Pada pengecekan secara *random* hasil nilai persentase dibawah 14% pada proses *preprocessing* dan pada proses *non-preprocessing* hasil nilai persentase dibawah 18%. Meskipun dengan menggunakan tipe pemrosesan yang berbeda tetapi hasil yang diberikan cukup stabil.
4. Pengecekan dokumen yang tidak di-*random* atau menguji dokumen yang memiliki kesamaan makna maupun isi seperti dokumen "asli2" dan "plagiat2" yang memiliki makna yang sama sehingga akan menghasilkan nilai presentase sinonim yang akurat.

5.2 Saran

Adapun saran dari hasil pengujian dan pengembangan dari sistem adalah sebagai berikut:

1. Perlu dioptimalkan proses untuk mendeteksi kata dari segi makna kata. Pada penelitian saat ini pengecekan kata dari segi kata masih menggunakan teknik manual atau *brute force* sehingga membutuhkan waktu yang lama dalam

memproses sebuah *file* atau dokumen dengan jumlah kalimat yang banyak. Perlu adanya algoritma yang dapat mempersingkat proses pencarian makna kata tersebut.

2. Sistem ini masih memiliki kekurangan dalam proses *stemming* atau penghilangan kata imbuhan (afiks). Yang masih memiliki kesalahan dalam proses penghilang kata imbuhan (afiks) seperti kata “mengerti” menjadi “erti”. Perlu adanya peningkatan dalam hal pengenalan kata yang memiliki imbuhan.
3. Sistem ini juga tidak mengenali sebuah kalimat atau kata yang memiliki makna denotasi, konotasi maupun memiliki makna kiasan. Mungkin diperlukan penambahan beberapa *database* yang menampung kata-kata yang memiliki makna selain sinonim.
4. Sistem memiliki kelemahan dalam hal menangani kata yang memiliki perulangan seperti “janji-janji”, pada kata tersebut pada di lakukan proses *preprocessing* garis penghubung kata hilang tapi kata menyatu menjadi 1 seperti “janjijanji”. Maka perlu peningkatan dalam menangani kata yang memiliki perulangan agar tidak memunculkan kata yang bias atau kata yang tidak memiliki bahkan bukan termasuk dalam Kamus Besar Bahasa Indonesia (KBBI).
5. Proses pemberian bobot pada proses pengecekan sinonim perlu dioptimalisasi agar dapat menghasilkan nilai persentase yang sesuai.

DAFTAR PUSTAKA

- Adiwidya, B.M.D., 2009, Algoritma *Levenshtein* Dalam Pendekatan *Approximate String Matching*. Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Bandung.
- Adriyani, Ni Made Murni, et al., 2012, Implementasi Algoritma *Levenshtein distance* dan Metode Empiris untuk Menampilkan Saran Perbaikan Kesalahan Pengetikan Dokumen Berbahasa Indonesia. Universitas Udayana.
- Ariyani, Na'firul Hasna, et al. 2016, "Aplikasi Pendeteksi Kemiripan Isi Teks Dokumen Menggunakan Metode *Levenshtein distance*" . Fakultas Teknik, Universitas Halu Oleo, Kendari
- Aggarwal, Charu C. dan Zhai, Chen Xiang, 2012, *ASurvey of Text Classification Algorithms* (Online), Chapter 6, halaman 52, tersedia pada: [http: / / link.springer.com / chapter / 10.1007%2F978-1-4614-3223-4_6](http://link.springer.com/chapter/10.1007%2F978-1-4614-3223-4_6),
- Chaer, Abdul, 2007, *Linguistik Umum*, Jakarta: Rineka Cipta.
- Dr. S. Vijayarani ed al., 2011, *Preprocessing Techniques for Text Mining - An Overview*, *International Journal of Computer Science & Communication Networks*, Volume 5, halaman 7-16.
- Driscoll Lyn, Dana dan Brizee, Allen., 2013, *Quaoting, Paraphrasing, and Summarizing*. Tersedia pada: [https: / / owl.english.purdue.edu / owl / resource / 563 / 1 /](https://owl.english.purdue.edu/owl/resource/563/1/), tanggal akses: 18 Mei 2016.
- Dola, Abdullah., 2010, *Tataran Sintaksis dalam Gramatikal Bahasa Indonesia*, Makassar: Badan Penerbit Universitas Negeri Makassar.
- Haddi, Emma, ed al., 2013, *The Role of Text Pre-processing in Sentiment Analysis*.
- Gurusamy, Vairaprakash, 2014, *Preprocessing Techniques for Text Mining*.
- Gilleland, M., 2009. *Levenshtein distance, in three flavors*. Merriam Park Software: [http: / / www.merriampark.com / ld. htm](http://www.merriampark.com/ld.htm).
- Galiotou, Eleni ed al., 2013, *On the Effect of Stemming Algorithms on Extractive Summarization: A Case Study*.
- Jivani , Anjali Ganesh Ms., 2011, *A Comparative Study of Stemming Algorithms*.

- Juri D. Apresjan¹, ed al., 2009, *Semantic Paraprasing for Information Retrieval and Extraction*, tersedia pada:
http://link.springer.com/chapter/10.1007/978-3-642-04957-6_44.
- Lestarini, Ade Hapsari, 2014, Sederet Kasus Plagiarisme di Kampus, <http://news.okezone.com/read/2014/02/25/373/946214/sederet-kasus-plagiarisme-di-kampus>, Diakses 28 Maret 2016.
- Manning, Christopher D. ed al., 2009, *An Introduction to Information Retrieval*.
- Marsa, A.H. 2009. Ayo Mengenal Paragraf. Jakarta: PT Wangsa Jatra Lestari.
- Mayantara, 2012, Menerjemahkan Karya Sastra,
<http://mayantara.sch.id/artikel/menerjemahkan-karya-sastra.htm#comments>, Diakses 18 Mei 2016.
- McCabe, Donald L, 2005, *Cheating among college and university students: A North American perspective*, International Journal of Academic Integrity.
- Nasucha, Rohmadi dan Wahyudi, 2009, Bahasa Indonesia Untuk Karya Tulis Ilmiah, Yogyakarta: Media Perkasa.
- paice, c. d., 2009. *stemming*. Dalam: l. liu dan M. T. Özsu, penyunt. *encyclopedia of database systems*. s.l.:springer, halaman 2790-2793.
- Pyriana, Dewi Rokhmah, et al., 2013, Program Aplikasi Editor Kata Bahasa Indonesia menggunakan Metode *Approximate String Matching* dengan Algoritma *Levenshtein distance* Berbasis Java, Program Studi Teknik Informatika Universitas Brawijaya.
- Rohmadi, Muhammad dan Nasucha, Yakub, 2010, Paragraf Pengembangan dan Implementasi, Yogyakarta: Media Perkasa.
- Sastroasmoro, Sudigno, 2007, Beberapa Catatan Tentang Plagiarisme, Departemen Ilmu Kesehatan Anak, Fakultas Kedokteran Universitas Indonesia, Jakarta.
- Stein, B., Meyer, S. zu Eissen., 2006, *Near Similarity Search and Plagiarism Analysis*, 29th Annual Conference of the German Classification Society (GfKI), Magdeburg, halaman 430 – 437.
- George R.S Weir, 2004, *Work In progress Technology in plagiarism detection and management*, 34th ASEE / IEEE Frontiers in Education Conference.

DAFTAR RIWAYAT HIDUP

Nama : Denny Gunawan
Umur : 22 tahun
Tempat / Tanggal Lahir : Lubuk Pakam / 20 Maret 1994
Jenis Kelamin : Pria
Agama : Buddha
Tempat Tinggal : Jln. Perintis Kemerdekaan no. 12 Galang

Pendidikan:

1. Tamatan SD Perguruan Kristen Methodist Galang tahun 2006
2. Tamatan SMP Perguruan Kristen Methodist Lubuk Pakam Tahun 2009
3. Tamatan SMA Perguruan Kristen Methodist Lubuk Pakam Tahun 2012

Demikian Daftar Riwayat hidup ini saya perbuat dengan sesungguhnya.

Hormat Saya,

DAFTAR RIWAYAT HIDUP

Nama : Rivaldi Warman
Umur : 22 tahun
Tempat / Tanggal Lahir : Medan / 25 Juli 1994
Jenis Kelamin : Pria
Agama : Buddha
Tempat Tinggal : Jln. Mayjend Sutoyo no. 32 Binjai Barat

Pendidikan:

1. Tamatan SD Perguruan Kristen Methodist Binjai tahun 2006
2. Tamatan SMP Perguruan Kristen Methodist Binjai Tahun 2009
3. Tamatan SMA Perguruan Kristen Methodist Binjai Tahun 2012

Demikian Daftar Riwayat hidup ini saya perbuat dengan sesungguhnya.

Hormat Saya,

DAFTAR RIWAYAT HIDUP

Nama : Wisely Jansen Hadi
Umur : 22 tahun
Tempat / Tanggal Lahir : Lubuk Pakam / 11 Juni 1994
Jenis Kelamin : Pria
Agama : Buddha
Tempat Tinggal : Jln. Jemadi Gg. Kesuma Komplek Griya no 34 Brayan

Pendidikan:

1. Tamatan SD Swasta Wiyata Dharma tahun 2006
2. Tamatan SMP Sutomo 2 Tahun 2009
3. Tamatan SMA Sutomo 2 Tahun 2012

Demikian Daftar Riwayat hidup ini saya perbuat dengan sesungguhnya.

Hormat Saya,
