

**PENERAPAN ALGORITMA STEMMING *ENHANCE CONFIX STRIPPING* DAN *JARO WINKLER* UNTUK PENDETEKSIAN DAN PENGOREKSIAN KESALAHAN PENULISAN**

*Diajukan Sebagai Syarat untuk Menyelesaikan  
Pendidikan Program Strata-1 pada  
Jurusan Teknik Informatika Fakultas Ilmu Komputer UNSRI*



Oleh :

AGUM PANJI PERDANA

NIM : 09021181320029

**Jurusan Teknik Informatika  
FAKULTAS ILMU KOMPUTER UNIVERSITAS SRIWIJAYA  
2018**

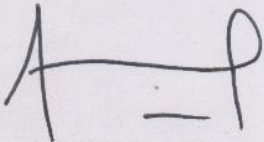
# LEMBAR PENGESAHAN TUGAS AKHIR

## PENERAPAN ALGORITMA STEMMING *ENHANCE CONFLX STRIPPING* DAN *JARO WINKLER* UNTUK PENDETEKSIAN DAN PENGOREKSIAN KESALAHAN PENULISAN

Oleh :

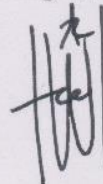
AGUM PANJI PERDANA  
NIM : 09021181320029

Pembimbing I,



M. Fachrurrozi, S.Si., M.T.  
NIP. 198005222008121002

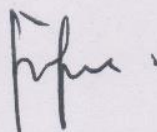
Indralaya, Januari 2018  
Pembimbing II,



Hadipurnawan Satria, Ph.D.  
NIPUS. 198004182015109101

Mengetahui,

Ketua Jurusan Teknik Informatika,



Rifkie Primartha S.T. M.T.  
NIP 197706012009121004

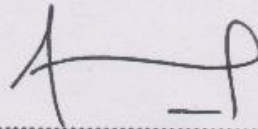
## TANDA LULUS UJIAN SIDANG TUGAS AKHIR

Pada hari Senin tanggal 18 Desember 2017 telah dilaksanakan ujian sidang tugas akhir oleh Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya.

Nama : Agum Panji Perdana  
N I M : 09021181320029  
Judul : Penerapan Algoritma Stemming *Enhance Confix Stripping* dan *Jaro Winkler* untuk Pendeteksian dan Pengoreksian Kesalahan Penulisan

1. Ketua Penguji

M. Fachrurrozi, S.Si., M.T.  
NIP. 198005222008121002



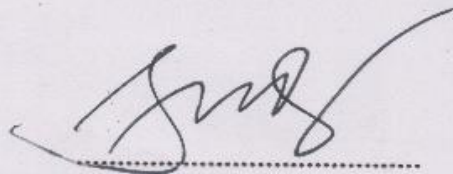
2. Sekretaris

Hadipurnawan Satria, Ph.D.  
NIPUS. 198004182015109101



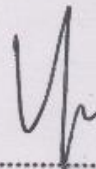
3. Penguji I

Rusdi Efendi, M.Kom.  
NIP.

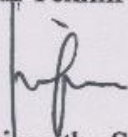


4. Penguji II

Yunita, S.Si., M.Cs  
NIP. 198306062015042002



Mengetahui,  
Ketua Jurusan Teknik Informatika

  
Rifkie Primartha S.T. M.T.  
NIP 197706012009121004

## **MOTTO DAN PERSEMBAHAN**

- “Life is not last forever, so live life to the fullest”

Saya persembahkan kepada :

- ✓ Allah SWT
- ✓ Bapak dan Ibu tercinta
- ✓ Kedua pembimbing
- ✓ Nurma Desty Anggraeni
- ✓ Sahabat–sahabat seperjuangan
- ✓ Almamater

## **Penerapan Algoritma Stemming *Enhance Confix Stripping* dan *Jaro Winkler* untuk Pendeteksian dan Pengoreksian Kesalahan Penulisan**

**Oleh :**

Agum Panji Perdana

09021181320029

Pendeteksian dan pengoreksian kesalahan penulisan telah menjadi kebutuhan dasar dalam mengolah suatu dokumen. Sebuah dokumen bisa dikatakan baik jika memiliki formalitas tinggi, kosa kata yang baik, dan tidak mempunyai kesalahan tata bahasa maupun kesalahan penulisan. Stemming dan *spelling correction* adalah dua proses utama dalam pendeteksian dan pengoreksian kesalahan penulisan. Proses stemming dilakukan dalam pendeteksian kesalahan penulisan sedangkan *spelling correction* dilakukan dalam pengoreksian kesalahan penulisan. Penelitian ini menggunakan algoritma *Enhance Confix Stripping* sebagai algoritma stemming, dan algoritma *Jaro-Winkler* sebagai algoritma *spelling correction*. Hasil dari penelitian ini perangkat lunak mampu mendeteksi kesalahan penulisan dengan rata-rata akurasi sebesar 97,1%, dan mengoreksi kesalahan penulisan dengan akurasi sebesar 78,14%. Kecepatan rata-rata perangkat lunak dalam mendeteksi kedsalahan penulisan adalah sebesar 2380 milisecond atau sekitar 2 detik

**Kata kunci :** Pendeteksian, pengoreksian, kesalahan penulisan pada dokumen, stemming, *spelling correction*, *Enhance Confix Stripping*, dan *Jaro Winkler*.

## **Implementation of Enhance Confix Stripping Stemming Algorithm and Jaro Winkler for Detection and Correction of Writing Errors.**

**Oleh :**

Agum Panji Perdana

09021181320029

Detection and correction of writing errors have become a basic requirement in processing a document. A document is good if it has high formality, good vocabulary and no grammatical or writing errors. Stemming and spelling correction are two main processes in the detection and correction of writing errors. Stemming process is done in detecting writing errors while spelling correction is done in correcting writing errors. This research use Enhance Confix Stripping as stemming algorithm, and Jaro winkler as spelling correction algorithm. The result of this research is software is able to detect writing errors with average accuracy of 97,1 %, and correct writing errors with average accuracy of 78,14%. The software's average speed to detect writing errors is 2380 milliseconds or about 2 seconds.

**Keywords :** Detection, correction, writing errors in document, stemming, spelling correction, Enhance Confix Stripping, and Jaro Winkler.

## KATA PENGANTAR

**Assalamu'alaikum wr. wb.**

Puji syukur ke hadirat Allah SWT, karena atas berkat rahmat dan karunia-Nya lah penulis dapat menyelesaikan proposal tugas akhir ini yang berjudul “**Penerapan Algoritma Stemming *Enhance Confix Stripping* dan *Jaro Winkler* Untuk Pendeteksian dan Pengoreksian Kesalahan Penulisan**”. Tujuan dari penulisan tugas akhir ini adalah untuk memenuhi salah satu persyaratan kelulusan pada tingkat sarjana (S1) di Jurusan Teknik Informatika Universitas Sriwijaya.

Dalam menyusun proposal tugas akhir ini, masih banyak kekurangan dan kekeliruan yang perlu disempurnakan, untuk itu penulis sangat mengharapkan kritik dan saran yang bersifat membangun guna menyempurnakan proposal tugas akhir ini. Penulis berharap tugas akhir ini dapat memberikan manfaat bagi para pembaca.

Atas selesainya penulisan tugas akhir ini, penulis mengucapkan terima kasih kepada:

1. Allah SWT yang selalu memberikan berkat dan rahmat-Nya,
2. Ibu dan keluarga tercinta yang selalu memberikan kasih sayang, dukungan baik moral maupun materiil serta doanya untuk penulis,
3. Bapak Jaidan Jauhari M.T., selaku Dekan Fakultas Ilmu Komputer Universitas Sriwijaya,

4. Bapak Rifkie Primarta M.T., selaku Ketua Jurusan Teknik Informatika Fakultas Komputer Universitas Sriwijaya,
5. Bapak M. Fachrurrozi, M.T. dan bapak Hadipurnawan Satria. Ph.D. selaku pembimbing yang telah memberikan banyak masukan, kritik, dan saran yang bermanfaat bagi penulis dalam menyelesaikan proposal tugas akhir ini.
6. Bapak dan Ibu Dosen di Jurusan Teknik Informatika Universitas Sriwijaya yang telah memberikan bekal ilmu kepada penulis selama perkuliahan.
7. Seluruh karyawan di Fakultas Ilmu Komputer yang telah membantu penulis mengurus urusan administrasi,
8. Nurma Desty Anggraeni yang memberikan banyak dukungan pada penulis.
9. Sahabat seperjuangan yang bersedia membantu dalam penyusunan proposal tugas akhir ini.

Penulis menyadari masih banyak terdapat kekurangan dalam penulisan proposal tugas akhir ini. Selain itu penulis juga mengharapkan saran dan kritik yang bersifat membangun demi sempurnanya tugas akhir ini. Akhir kata penulis mengharapkan semoga proposal tugas akhir ini dapat bermanfaat bagi kita semua.

**Wassalamu'alaikum wr. wb..**

Indralaya Januari 2018

Penulis



## DAFTAR ISI

	Halaman
HALAMAN JUDUL .....	i
LEMBAR PENGESAHAN TUGAS AKHIR .....	ii
TANDA LULUS UJIAN SIDANG TUGAS AKHIR .....	iii
MOTTO DAN PERSEMBAHAN .....	iv
ABSTRAK .....	v
ABSTRACT .....	vi
KATA PENGANTAR .....	vii
DAFTAR ISI .....	ix
DAFTAR GAMBAR .....	xii
DAFTAR TABEL .....	xiii
DAFTAR LAMPIRAN .....	xiv

### BAB I PENDAHULUAN

1.1 Latar Belakang .....	I-1
1.2 Rumusan Masalah .....	I-4
1.3 Tujuan .....	I-4
1.4 Manfaat .....	I-4
1.5 Batasan Masalah .....	I-4
1.6 Metodologi Penelitian .....	I-5
1.6.1 Unit Penelitian .....	I-5
1.6.2 Metode Pengumpulan Data .....	I-5
1.6.2.1 Jenis Data .....	I-5
1.6.2.2 Sumber Data .....	I-6
1.6.2.3 Teknik Pengumpulan Data .....	I-6
1.6.3 Metodologi Penelitian .....	I-6
1.6.4 Peralatan Penelitian .....	I-6
1.6.4.1 <i>Hardware</i> .....	I-6
1.6.4.2 <i>Software</i> .....	I-7
1.6.5 Metode Pengembangan Perangkat Lunak .....	I-7
1.7 Sistematika Penulisan .....	I-9

## BAB II TINJAUAN PUSTAKA

2.1 Penelitian Terkait .....	II-1
2.2 Kesalahan Penulisan .....	II-3
2.2.1 Pendeteksian Kesalahan Penulisan .....	II-3
2.2.2 Pengoreksian Kesalahan Penulisan .....	II-4
2.3 Bahasa Indonesia .....	II-5
2.3.1 Huruf .....	II-5
2.3.2 Morfem .....	II-6
2.3.3 Kata Dasar dan Imbuhan .....	II-6
2.3.4 Morfofonemik Pada Awalan .....	II-7
2.3.4.1 Morfofonemik Awalan Meng- .....	II-7
2.3.4.2 Morfofonemik Awalan Per- .....	II-8
2.3.4.3 Morfofonemik Awalan Ber- .....	II-9
2.3.4.4 Morfofonemik Awalan Ter- .....	II-10
2.4 Tokenisasi .....	II-10
2.5 Stemming .....	II-10
2.5.1 Algoritma Stemming Confix-Stripping .....	II-13
2.5.2 Enhance Confix-Stripping Stemmer .....	II-18
2.6 String Metric .....	II-21
2.6.1 Algoritma Jaro-Winkler .....	II-22
2.7 Pengujian .....	II-23
2.7.1 Recall .....	II-23
2.7.2 Precision .....	II-23
2.7.3 F-measure .....	II-24

## BAB III ANALISIS DAN PERANCANGAN

3.1 Analisis Masalah .....	III-1
3.1.1 Analisis Data .....	III-1
3.1.2 Analisis Arsitektur Perangkat Lunak .....	III-2
3.1.3 Analisis Prapengolahan .....	III-4
3.1.3.1 Analisis Segmentasi Kalimat .....	III-5
3.1.3.2 Analisis <i>Case Folding</i> .....	III-6
3.1.3.3 Analisis Tokenisasi .....	III-7
3.1.3.4 Analisis <i>Filtering</i> .....	III-8
3.1.4 Analisis Deteksi Kesalahan Penulisan .....	III-9
3.1.4.1 Analisis Algoritma Stemming <i>Enhanced Confix Stripping</i> .....	III-9
3.1.5 Analisis Pengoreksian Kesalahan Penulisan .....	III-12
3.1.5.1 Analisis Algoritma <i>Jaro-Winkler</i> .....	III-12
3.2 Analisis Perancangan Perangkat Lunak .....	III-16
3.2.1 Deskripsi Umum Perangkat Lunak .....	III-16
3.2.2 Spesifikasi Kebutuhan Perangkat Lunak .....	III-16
3.3 Perancangan Perangkat Lunak .....	III-17
3.3.1 Model <i>Use Case</i> .....	III-17

3.3.1.1 Diagram <i>Use Case</i> Pendeteksian dan Pengoreksian Kesalahan Penulisan .....	III-17
3.3.1.2 Tabel Definisi Aktor .....	III-18
3.3.1.3 Tabel Definisi <i>Use Case</i> .....	III-19
3.3.1.4 Skenario <i>Use Case</i> .....	III-19
3.3.1.5 Kelas Analisis Melakukan Deteksi Kesalahan Penulisan dan Pengoreksian Kesalahan Penulisan .....	III-21
3.3.1.6 Kelas Diagram .....	III-25
3.3.1.7 Diagram Sekuensial .....	III-26
3.3.1.7.1 Diagram Sekuensial Melakukan Preprocassing .....	III-26
3.3.1.7.2 Diagram Sekuensial Melakukan Pendeteksian .....	III-28
3.3.1.7.3 Diagram Sekuensial Melakukan Pengoreksian .....	III-29
3.3.2 Perancangan Antarmuka .....	III-30

#### BAB IV IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Perangkat Lunak .....	IV-1
4.1.1 Lingkungan Implementasi .....	IV-1
4.1.2 Implementasi Kelas .....	IV-2
4.1.3 Implementasi Antarmuka .....	IV-3
4.2 Pengujian Perangkat Lunak .....	IV-5
4.2.1 Lingkungan Pengujian .....	IV-5
4.2.2 Rencana Pengujian .....	IV-5
4.2.3 Kasus Uji .....	IV-7
4.3 Hasil dan Analisis Pengujian .....	IV-9
4.3.1 Hasil dan Analisis Pengujian Tingkat Akurasi Pendeteksian Kesalahan Penulisan .....	IV-9
4.3.2 Hasil dan Analisis Pengujian Tingkat Akurasi Pengoreksian Kesalahan Penulisan .....	IV-14
4.3.3 Hasil dan Analisis Pengujian Kecepatan Pendeteksian Kesalahan Penulisan .....	IV-19

#### BAB V KESIMPULAN DAN SARAN

5.1 Kesimpulan .....	V-1
5.2 Saran .....	V-2

DAFTAR PUSTAKA .....	xv
----------------------	----

## DAFTAR GAMBAR

	Halaman
I-1 Disiplin dan Fase dari RUP .....	I-7
II-1 Diagram Alir Algoritma Confix-Stripping .....	II-17
III-1 <i>Flow chart</i> Proses Perangkat Lunak .....	III-3
III-2 <i>Flow chart</i> Prapengolahan .....	III-4
III-3 <i>Flow chart</i> Pengembalian Akhiran .....	III-10
III-4 Diagram Use Case .....	III-18
III-5 Kelas Analisis <i>Use Case</i> Melakukan Preprocessing .....	III-22
III-6 Kelas Analisis <i>Use Case</i> Melakukan Pendeteksian Kesalahan Penulisan .....	III-23
III-7 Kelas Analisis <i>Use Case</i> Melakukan Pengoreksian Kesalahan Penulisan .....	III-24
III-8 Kelas Diagram .....	III-25
III-9 Diagram Sekuensial Melakukan Preprocessing .....	III-27
III-10 Diagram Sekuensial Melakukan Pendeteksian .....	III-28
III-11 Diagram Sekuensial Melakukan Pengoreksian .....	III-29
III-12 Perancangan Antarmuka .....	III-30
IV-1 Antarmuka MainUIBoundary .....	IV-4
IV-2 Tingkat Akurasi Pendeteksian Berdasarkan Jumlah Kata .....	IV-12
IV-3 Tingkat Akurasi Pendeteksian Berdasarkan Jumlah Kata Salah .....	IV-13
IV-4 Tingkat Akurasi Berdasarkan Jenis Kesalahan .....	IV-16
IV-5 Tingkat Akurasi Berdasarkan Jenis Kata .....	IV-17
IV-6 Kecepatan Sistem Dalam Mendeteksi Kesalahan Penulisan Berdasarkan Jumlah Kata dalam Dokumen .....	IV-21

## DAFTAR TABEL

	Halaman
II-1 Perubahan Awalan Meng- .....	II-7
II-2 Perubahan Awalan Per- .....	II-9
II-3 Perubahan Awalan Ber- .....	II-9
II-4 Perubahan Awalan Ter- .....	II-10
II-5 Kombinasi Prefiks dan Sufiks yang Dilarang .....	II-14
II-6 Aturan Pemotongan Prefiks .....	II-19
II-7 Aturan Tambahan Algoritma Enhance Confix-Stripping .....	II-21
III-1 Teks Input .....	III-5
III-2 Hasil Segmentasi Kalimat .....	III-5
III-4 Contoh Proses Tokenisasi .....	III-7
III-5 Contoh Hasil Stemming .....	III-11
III-6 Contoh Jaro Winkler.....	III-13
III-7 Kebutuhan Fungsional .....	III-17
III-8 Kebutuhan Non-Fungsional .....	III-17
III-9 Definisi Aktor .....	III-18
III-10 Definisi Use Case .....	III-19
III-11 Skenario Use Case Melakukan Preprocessing .....	III-19
III-12 Skenario Use Case Melakukan Deteksi Kesalahan Penulisan .....	III-20
III-13 Skenario Use Case Melakukan Pengoreksian Kesalahan Penulisan .....	III-19
IV-1 Spesifikasi Perangkat Keras .....	IV-1
IV-2 Spesifikasi Perangkat Lunak .....	IV-1
IV-3 Daftar Kelas .....	IV-2
IV-4 Rencana Pengujian <i>Use Case</i> Melakukan Preprocessing .....	IV-5
IV-5 Rencana Pengujian <i>Use Case</i> Melakukan Pendeteksian .....	IV-5
IV-6 Rencana Pengujian <i>Use Case</i> Melakukan Pengoreksian .....	IV-6
IV-7 Kasus Uji <i>Use Case</i> Melakukan <i>Preprocessing</i> .....	IV-7
IV-8 Kasus Uji <i>Use Case</i> Melakukan Pendeteksian .....	IV-7
IV-9 Kasus Uji <i>Use Case</i> Melakukan Pengoreksian .....	IV-8
IV-10 Tingkat Akurasi Berdasarkan Jumlah Kata Dokumen .....	IV-11
IV-11 Tingkat Akurasi Berdasarkan Jumlah Kata Salah pada Dokumen .....	IV-11
IV-12 Perhitungan Pengoreksian Berdasarkan Jenis Kesalahan .....	IV-15
IV-13 Contoh Kata-Kata Substitution .....	IV-16
IV-14 Perhitungan Pengoreksian Berdasarkan Jenis Kata .....	IV-17
IV-15 Contoh Kata-Kata Berimbuhan yang Gagal Dikoreksi .....	IV-18
IV-16 Kecepatan Pendeteksian Berdasarkan Jumlah Kata pada Dokumen .....	IV-19

## DAFTAR LAMPIRAN

1. Daftar Kata Salah yang Terdeteksi
2. Daftar Pengoreksian Kata Salah
3. Kode Program
4. Kata Terdeteksi dengan Salah

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Sebuah dokumen bisa dikatakan baik jika memiliki formalitas tinggi, kosa kata yang baik, dan tidak mempunyai kesalahan tata bahasa maupun kesalahan penulisan. Sebuah dokumen harus dicek dan diteliti huruf demi huruf berulang-ulang untuk menghindari kesalahan penulisan (Widjaja & Hansun, 2015). Hal tersebut membutuhkan waktu yang sangat banyak dan juga rawan kesalahan jika dilakukan oleh manusia. Untuk mempercepat proses pengecekan kesalahan penulisan, dokumen bisa diproses secara digital dengan bantuan program berbasis *Natural Language Processing* (NLP) untuk mendeteksi kesalahan penulisan yang ada, salah satu algoritma NLP yang bisa dipakai adalah algoritma stemming. Dengan stemming kesalahan penulisan dapat dideteksi dengan cara membandingkan kata-kata pada dokumen yang sudah distemming dengan kata dasar yang ada pada kamus.

Stemming adalah kegiatan untuk mencari suatu kata dasar dari sebuah kata dengan cara menghilangkan awalan dan akhiran (Lovins, 1968). Dengan stemming pencarian kata dasar bisa dilakukan secara cepat dan akurat daripada manual. Stemming dapat digunakan untuk mendeteksi kesalahan penulisan pada dokumen dengan tingkat keakuratan yang tinggi. Penggunaan Stemming untuk melakukan pengecekan terhadap kesalahan penulisan sudah pernah dilakukan oleh (Widjaja & Hansun, 2015) dalam penelitian yang berjudul "*Implementation of porter's*

*modified stemming algorithm in an Indonesian word error detection plugin application*” dengan menggunakan metode modifikasi porter. Penelitian tersebut menunjukkan bahwa stemming sangat akurat dalam mendeteksi kesalahan penulisan dengan tingkat akurasi 96.31%, akan tetapi aplikasi yang dikembangkan belum bisa memperbaiki kesalahan penulisan yang ditemukan.

Algoritma yang bisa digunakan dalam stemming bahasa Indonesia antara lain adalah *Confix-Stripping*, *Enhanced Confix-Stripping*, Porter, Nazief & Andriani. Algoritma *Enhanced Confix-Stripping* adalah algoritma perkembangan dari algoritma *Confix-Stripping* yang juga merupakan perkembangan dari algoritma Nazief & Andriani. Algoritma *Enhanced Confix-Stripping* (ECS) dikembangkan dan digunakan dalam penelitian (Arifin, 2009) dalam penelitian berjudul “*Enhanced Confix Stripping Stemmer and Ants Algorithm for Classifying News Document in Indonesia Language*” sebagai proses untuk mengklasifikasikan Dokumen berbahasa Indonesia. Pada penelitian tersebut Algoritma ECS memiliki tingkat akurasi yang lebih tinggi 2% dibanding algoritma asalnya yaitu *Confix-Stripping* yang memiliki tingkat akurasi 95.7%. Sehingga dapat disimpulkan algoritma ECS merupakan algoritma stemming yang paling baik dalam stemming Bahasa Indonesia.

Untuk mempercepat proses perbaikan, selain menggunakan stemming untuk pendeteksian kesalahan penulisan pada dokumen diperlukan juga metode untuk melakukan *spelling correction*. Sehingga perbaikan dokumen juga bias dilakukan dengan cepat. *Spelling correction* adalah kegiatan untuk mengoreksi atau menyugestikan kata yang dianggap salah dalam sebuah sistem. Algoritma yang



digunakan dalam *spelling correction* antara lain N-gram, Hamming-distance, Lavehstain distance, dan Jaro Winkler. Metode-metode tersebut sudah pernah di uji coba untuk Bahasa Indonesia pada penelitian oleh (Rochmawati & Kusumaningrum, 2016) berjudul “*Studi Perbandingan Algoritma Pencarian String dalam Metode Approximate String Matching untuk Identifikasi Kesalahan Pengetikan Teks*”

Jaro Winkler adalah algoritma yang dikembangkan dari Jaro distance metric untuk menghitung tingkat kemiripan antara 2 string. Algoritma Jaro Winkler sering dipakai untuk *spelling correction* dan pendeteksi plagiarisme seperti yang dilakukan oleh Kornain (2014) sebagai algoritma untuk sistem pendeteksi plagiarisme pada dokumen teks berbahasa Indonesia. Jaro Winkler merupakan salah satu algoritma yang paling efektif dalam menghitung tingkat kemiripan. Dalam penelitian yang dilakukan oleh (Rochmawati & Kusumaningrum, 2015) berjudul “*Studi Perbandingan Algoritma Pencarian String dalam Metode Approximate String Matching untuk Identifikasi Kesalahan Pengetikan Teks*”, metode Jaro Winkler merupakan algoritma terbaik dalam melakukan *spelling correction* pada kata berbahasa Indonesia. Dengan menggabungkan metode ini dengan algoritma stemming ECS diharapkan dapat mendeteksi dan memperbaiki kesalahan yang ada pada dokumen secara cepat dan akurat.

## **1.2 Rumusan Masalah**

Rumusan masalah pada penelitian ini adalah bagaimana cara untuk melakukan penyuntingan dokumen secara tepat serta akurat dalam perbaikan kesalahan penulisan.

## **1.3 Tujuan**

1. Melakukan implementasi algoritma *Enhanced Confix-Stripping* dan Jaro Winkler pada sebuah perangkat lunak yang mampu mendeteksi dan mengoreksi kesalahan penulisan dengan akurat;
2. Untuk mengukur tingkat akurasi algoritma stemming ECS dengan algoritma Jaro Winkler dalam mendeteksi dan mengoreksi kesalahan penulisan pada dokumen.
3. Untuk mengukur kecepatan algoritma *Enhance Confix Stripping* dan *Jaro Winkler*

## **1.4 Manfaat**

1. Untuk membantu penulis dokumen dalam menemukan dan mengoreksi kesalahan penulisan pada dokumen.

## **1.5 Batasan Masalah**

Batasan masalah pada penelitian ini adalah :

1. Perangkat lunak hanya menerima masukan dokumen berbahasa Indonesia.
2. Format berkas dokumen masukan adalah \*.txt (teks file)

3. Jumlah maksimal kata yang bisa di proses pada dokumen maksimal 3000 kata.

## **1.6 Metodologi Penelitian**

### **1.6.1 Unit Penelitian**

Unit penelitian pada tugas akhir ini adalah area Fakultas Ilmu Komputer Universitas Sriwijaya.

### **1.6.2 Metode Pengumpulan Data**

#### **1.6.2.1 Jenis Data**

Jenis data yang dipakai pada penelitian ini adalah data primer dan data sekunder. Data primer berupa dokumen latar belakang Tugas Akhir dan jurnal yang mengandung kesalahan penulisan berformat \*.txt. Data sekunder berupa *database* kata dasar yang diambil dari Kamus Besar Bahasa Indonesia.

#### **1.6.2.2 Sumber Data**

Data primer diperoleh dari 28 orang mahasiswa Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Sriwijaya, dan 2 jurnal dari ecm.org. Data sekunder menggunakan *database* kata dasar bahasa Indonesia yang diambil dari Kamus Besar Bahasa Indonesia

#### **1.6.2.3 Teknik Pengumpulan Data**

Teknik pengumpulan data primer adalah dengan mengumpulkan dokumen yang berasal dari mahasiswa Teknik Informatika dan icm.org.

Sedangkan teknik pengumpulan data sekunder adalah studi pustaka. Semua kata dasar diambil dari Kamus Besar Bahasa Indonesia yang akan digunakan sebagai kamus acuan untuk mesin stemming.

### **1.6.3 Metodologi Penelitian**

Tahapan penelitian yang telah dan akan dilakukan dalam tugas akhir ini adalah :

1. Studi literatur mengenai bahasa Indonesia, algoritma stemming *Confix-Stripping*, *Enhanced Confix-Stripping*, dan *Jaro Winkler*.
2. Pengumpulan data untuk *database* kamus kata dasar.
3. Pengumpulan data uji berupa dokumen latar belakang berformat \*.txt.
4. Merancang input dan output dari hasil proses stemming
5. Mengembangkan perangkat lunak pendeteksian dan pengoreksian kesalahan penulisan huruf menggunakan algoritma Stemming *Enhanced Confix-Stripping* dan *Jaro Winkler*.
6. Melakukan percobaan terhadap perangkat lunak yang dikembangkan.
7. Melakukan analisis dari hasil percobaan

### **1.6.4 Peralatan Penelitian**

#### **1.6.4.1 Hardware**

*Hardware* yang digunakan pada penelitian ini adalah komputer yang digunakan untuk pengembangan perangkat lunak dan pengujian perangkat lunak.

Spesifikasi komputer yang digunakan :

- a. Processor Intel Core i7 4790K @4.00GHz

b. Ram 16 GB

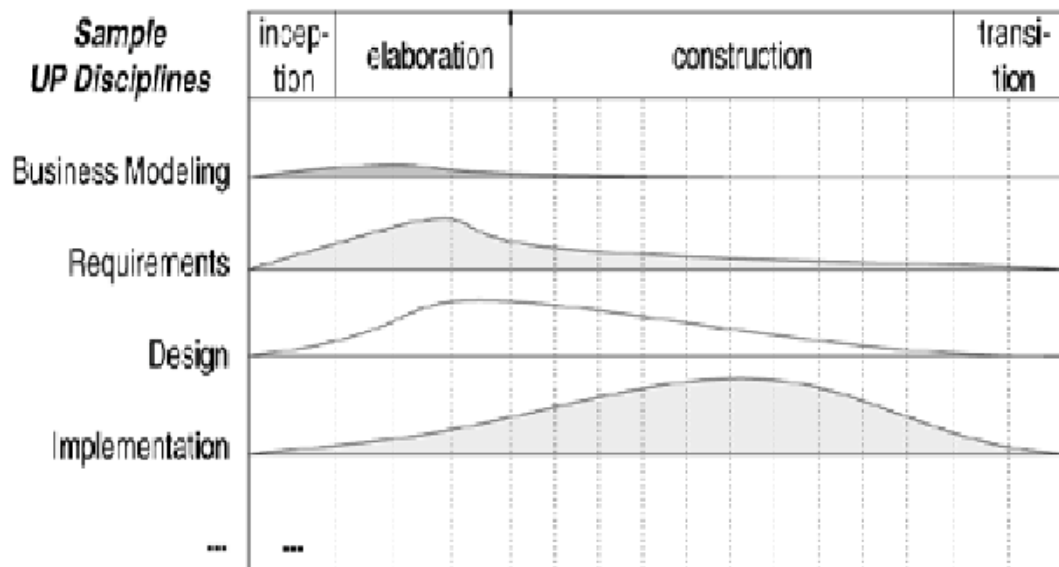
#### 1.6.4.2 Software

*Software* yang digunakan pada penelitian ini adalah :

1. Sistem Operasi Windows 10 Professional 64-Bit
2. NetBean 8.0.2

#### 1.6.5 Metode Pengembangan Perangkat Lunak

Metode yang digunakan dalam mengembangkan Perangkat Lunak ini adalah berorientasi objek dengan menggunakan metode *Rational Unified Process*. Metode ini dipilih karena dianggap sebagai metode yang paling baik dalam mengembangkan perangkat lunak berorientasi objek.



Gambar I-1 Disiplin dan Fase dari RUP (Larman, 2004)

Pada gambar dimensi horizotal merupakan fase-fase iteratif yang ada dalam pengembangan perangkat lunak. Sementara dimensi vertikal merupakan tugas yang akan dilakukan pada tiap iterasi yang dikelompokkan menjadi disiplin.

## **1.7 Sistematika Penulisan**

Berikut adalah sistematika penulisan dari Tugas Akhir ini

### **BAB I. PENDAHULUAN**

Pada bab ini dijelaskan mengenai latar belakang, perumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, metodologi penelitian, dan sistematika penulisan.

### **BAB II. TINJAUAN PUSTAKA**

Pada bab ini dijelaskan dasar teori yang berguna untuk analisis, perancangan, dan implementasi metode penelitian pada bab selanjutnya.

### **BAB III. ANALISIS DAN PERANCANGAN**

Pada bab ini dijelaskan tentang analisis dan perancangan perangkat lunak dari algoritma yang digunakan. Analisis dan perancangan ini berguna untuk membantu implementasi pada bab berikutnya.

### **BAB IV. IMPLEMENTASI DAN PENGUJIAN**

Pada bab ini membahas tentang lingkungan implementasi dari analisis dan perancangan perangkat lunak, implementasi program, hasil eksekusi, dan hasil pengujian.

## **BAB V. KESIMPULAN DAN SARAN**

Bab ini merupakan penutup yang berisi kesimpulan dari pembahasan bab-bab sebelumnya, dan saran-saran yang berguna untuk pengembangan perangkat lunak lebih lanjut.

## DAFTAR PUSTAKA

- Adriani, M. et al. 2007. Stemming Indonesian: A confix-stripping approach. ACM J. Educ. Resour. Comput. 6, 4, Article 13 (December 2007), 33 pages. DOI = 10.1145/1316457.1316459.  
<http://doi.acm.org/10.1145/1316457.1316459>
- Agusta, L. 2009. PERBANDINGAN ALGORITMA STEMMING PORTER DENGAN ALGORITMA NAZIEF & ADRIANI UNTUK STEMMING DOKUMEN TEKS BAHASA INDONESIA. Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Bali, Indonesia.
- Arifin, A.Z., Mahendra, P.A.K, dan H.T. Ciptaningtyas. 2009. ENHANCED CONFIX STRIPPING STEMMER AND ANTS ALGORITHM FOR CLASSIFYING NEWS DOCUMENT IN INDONESIAN LANGUAGE. The 5th Internasional Conference on Information & Communication Technology and System: 149-158.
- Asian, J., Williams, H.E., dan S.S.M. Tahaghoghi. 2004. Stemming Indonesian. School of Computer Science and Information Technology, RMIT University, Melbourne, Australia.
- Asian, J. 2007. Effective Techniques for Indonesian Text Retrieval. School of Computer Science and Information Technology, Science, Engineering, and Technology Portfolio, RMIT University, Melbourne, Victoria, Australia
- Buckland, M., 1998. "What is a Digital Document?" <http://people.ischool.berkeley.edu/~buckland/digdoc.html>. Diakses pada tanggal 11 Maret 2016.
- Damerau, F.J. 1964. A Technique for Computer Detection and Correction of Spelling Errors. Communications of the ACM(7): 171-176.
- Departemen Pendidikan Nasional. 2000. Pedoman Umum Ejaan Bahasa Indonesia Yang Disempurnakan. Panitia Pengembangan Bahasa Indonesia.
- Hapsari, R.K., dan J.Y. Santoso. 2015. STEMMING ARTIKEL BERBAHASA INDONESIA DENGAN PENDEKATAN CONFIX-STRIPPING. Teknik Informatika, Institut Teknologi Adhi Tama Surabaya, Surabaya, Indonesia.



- Koriyanti, E. 2003. Studi dan Implementasi Algoritma Stemming Untuk Bahasa Indonesia. Thesis Teknk Informatika, Institut Teknologi Bandung.
- Kornain, A., Yansen, F., dan Tinaliah. 2014. Penerapan Algoritma Jaro-Winkler Distance untuk Sistem Pendeteksi Plagiarisme pada Dokumen Teks Berbahasa Indonesia. STMIK GI MDP.
- Krisdalaksana, H. 1996. Pembentukan kata dalam Bahasa Indonesia. Jakarta: Gramedia Pustaka Utama.
- Kurniati, A., Puspitodjati, Sulisty., dan S. Rahman. 2010. Implementasi Algoritma Jaro-Winkler Distance untuk Membandingkan Dokumen Berbahasa Indonesia. Proceeding Seminar Ilmiah Nasional KOMMIT 2010. Universitas Gunadarma.
- Larman, C. 2004. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition. Boston: Addison Wesley Professional.
- Lovins, J.B. 1968. Development of Stemming Algorithm. Mechanical Translation and Computational Linguistics(11): 22-31.
- Pahdi, A. 2016. Koreksi Ejaan Istilah Komputer Berbasis Kombinasi Algoritma Damerau Levenshtein dan Algoritma Soundex. Journal Sentra Penelitian dan Edukasi(8).
- Pusat Bahasa Departemen Pendidikan Nasional. 2008. Kamus Besar Bahasa Indonesia. Jakarta :Balai Pustaka.
- Rochmawati, Y., dan R, Kusumaningrum. 2016. Studi Perbandingan Algoritma Pencarian String dalam Metode Approximate String Matching untuk Identifikasi Kesalahan Pengetikan Teks. Jurnal Buana Informatika(7) : 125-134.
- Widjaja, M. dan S. Hansun. 2015. IMPLEMENTATION OF PORTER'S MODIFIED STEMMING ALGORITHM IN AN INDONESIAN WORD ERROR DETECTION PLUGIN APPLICATION. Faculty of Information and Communication Technology, Universitas Multimedia Nusantara, Tangerang, Indonesia.
- Winkler, W.E. 1995. Matching and Record Linkage. Conference of Business Survei Methods : 355-384.