Edu Komputika 3 (1) (2016)

Edu Komputika Journal

http://journal.unnes.ac.id/sju/index.php/edukom

Implementasi Otokoreksi Ejaan Bahasa Indonesia pada Kamus Istilah Elektronika

Hilal Aji Wibowo✉ **Hari Wibawanto**

Universitas Negeri Semarang

Info Artikel

Sejarah Artikel:
Diterima Juni 2016
Disetujui Agustus 2016
Dipublikasikan Agustus 2016

Keywords:
autocorrection, levenshtein distance, dictionary, electronics.

Abstrak

Kamus saat ini tersedia secara online maupun offline, pencarian kata secara online mengakibatkan proses pencarian yang lama karena basis datanya tersimpan di server online. Sedangkan aplikasi yang offline masih terdapat beberapa aplikasi yang sedikit menggunakan fitur pencarian tanpa pengoreksi kata maupun otokoreksi. Dengan hal ini perangkat lunak Kamus Istilah Elektronika akan menerapkan fitur otokoreksi dengan algoritma levenshtein distance dalam pencarian kata. Tujuan dari penelitian ini adalah untuk mengetahui bagaimana cara mengimplementasikan fitur otokoreksi dan algoritma levenshtein distance pada aplikasi Kamus Istilah Elektronika. Metode pengembangan aplikasi menggunakan metode waterfall, yang terdiri dari 5 bagian yaitu definisi kebutuhan, rancangan perangkat lunak dan sistem, implementasi, pengujian serta operasi dan pemeliharaan. Pengujian aplikasi Kamus Istilah Elektronika menggunakan pengujian black box, yaitu: pengujian terhadap kemunculan otokoreksi, pengujian terhadap pemberian saran dengan levenshtein distance. Hasil penelitian pada pengujian fitur otokoreksi adalah muncul untuk setiap kata yang dimasukkan dan pada algoritma levenshtein distance saran yang akan muncul sesuai dengan kata masukannya. Fitur otokoreksi diimplementasikan pada aplikasi Kamus Istilah Elektronika dengan menambahkan fitur tersebut pada kotak pencarian edittext, serta algoritma levenshtein distance diimplementasikan pada tombol pencarian.

Abstract

A dictionary now is available online or offline , text search resulted in the process of search is long because the database on the server is online .While application offline there are still several application a little use the feature search without text correction and auto correction text.With this software "Kamus Istilah Elektronika" will apply features autocorrection with algorithm levenshtein distance in search of text .The purpose of this study is to find how to implement features autocorrection and algorithms levenshtein distance on the application "Kamus Istilah Elektronika". Method of development application use waterfall method , consisting of 5 part those definition needs , design software and system , the implementation , testing and operation and maintenance . Testing application "Kamus Istilah Elektronika" use testing black box , : testing to the autocorrection , testing to the provision of advice with levenshtein distance . The results of research on testing features autocorrection is appearing to every word put and on algorithm levenshtein distance advice that will appear corresponding to the input. Features autocorrection implemented on the application "Kamus Istilah Elektronika" by adding the feature in box search edittext , and algorithm levenshtein distance implemented at the button search .

✉ Alamat korespondensi:
Gedung E6 Lantai 2 FT Unnes
Kampus Sekaran, Gunungpati, Semarang, 50229
E-mail: hiraru.setting@gmail.com

ISSN 2252-6811

PENDAHULUAN

Istilah adalah kata atau gabungan kata yang dengan cermat mengungkapkan suatu makna, konsep proses, keadaan, atau sifat yang khas dalam bidang tertentu. Ada dua macam istilah: (1) istilah khusus dan (2) istilah umum. Istilah khusus adalah kata yang pemakaiannya dan maknanya terbatas pada suatu bidang tertentu, misalnya cakar ayam (bangunan), agregat (ekonomi); sedangkan istilah umum ialah kata yang menjadi unsur bahasa umum.misalnya: ambil alih, daya guna, kecerdasan, dan tepat guna merupakan istilah umum, sedangkan radiator, pedagogi, androgogi, panitera, sekering, dan atom merupakan istilah khusus. Istilah dalam bahasa Indonesia bersumber pada: kosa kata umum bahasa Indonesia, kosa kata bahasa serumpun. dan kosa kata bahasa asing.

Proses pembentukan istilah dilakukan melalui pemadanan atau penerjemahan, misalnya *busway* menjadi jalur bus; penyerapan kosa kata asing, misalnya *camera* menjadi kamera dan gabungan penerjemahan dan penyerapan, misalnya *subdivision* menjadi subbagian.

Sedangkan kamus merupakan buku acuan yang memuat kata dan ungkapan. Berfungsi untuk membantu mengenal perkataan atau istilah baru. Selain menerangkan maksud kata, kamus juga mungkin mempunyai pedoman sebutan, asal-usul (etimologi) sesuatu perkataan dan juga contoh penggunaan bagi sesuatu perkataan. Untuk memperjelas kadang kala terdapat juga ilustrasi di dalam kamus.

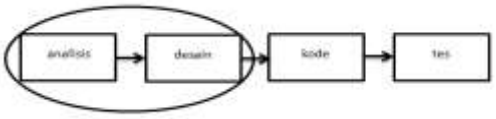
Biasanya penyusunan kata dan ungkapan disesuaikan dengan urutan abjad berikut dengan keterangan, pemakaian atau terjemaahan. Kamus terdiri dari beberapa jenis, antara lain : Kamus Istilah, Kamus Etimologi, Kamus Tesaurus, Kamus Peribahasa atau Simpulan Bahasa, Kamus Kata Nama Khas, Kamus Terjemahan, Kamus Kolokasi. Dari beberapa jenis kamus tersebut dalam penelitian ini dibuat kamus isitilah pada bidang ilmu elektronika secara *offline*.

Kamus Istilah Elektronika yang akan dibuat dilengkapi fitur otokoreksi sebagai fitur penunjang kecepatan kinerja pencarian istilah kata dalam Kamus Istilah Elektronika. Fitur otokoreksi akan menggunakan algoritma *levenshtein distance.*

METODE PENELITIAN

Metode pengembangan sistem yang digunakan adalah *System Development Life Cycle (SDLC)* yang merupakan pengembangan perangkat lunak yang terdiri dari beberapa tahapan. Terdapat banyak model yang berbasis SDLC, dalam penelitain ini menggunakan model sekuensial linear atau sering disebut *waterfall*. *Waterfall* dipilih karena dalam Pressman (2002:38) disebutkan bahwa model ini adalah rekayasa perangkat lunak yang paling luas dipakai dan paling tua, jadi sudah teruji kegunaanya dalam merancang sebuah perangkat lunak karena banyak yang sudah menggunakan.

Beberapa tahapan yang dilakukan dalam metode ini adalah sebagai berikut:



Gambar 1. Tahapan metode pengembangan sistem *Waterfall*

1. Analisis

Tahap analisis dilakukan untuk memahami alur program atau sistem yang dibutuhkan untuk memahami sifat program yang dibangun berdasarkan data dan informasi yang sebelumnya dilakukan pada pengumpulan data. Analisis sistem yang digunakan ialah metode analisis berorientasi pada objek atau OOA (*Object Oriented Analysis*) dengan menggunakan *tools* UML (*Unified Modelling Language)*. (Pressman, 2002:37).

2. Desain

Desain perangkat lunak merupakan suatu proses yang terdiri dari tahapan-tahapan sehingga perangkat lunak yang dibuat berjalan sesuai dengan tujuan perangkat lunak tersebut. Dalam desain perangkat lunak digunakan

metode perancangan perangkat lunak yaitu metode perancangan berorientasi *object* (OOD). Dalam metode perancangam berorientasi *object* menggunakan alat bantu UML yang digunakan untuk memvisualisasikan sistem usulan perangkat lunak yang dirancang. Dari UML tersebut dapat dirancang basisdata dan interface perangkat lunak.

3. Kode

Tahap pengkodean ini merupakan tahap dimana penerjemahan dari desain perangkat lunak yang telah dibuat sebelumnya menjadi sebuah kode di dalam program.

Dalam tahap pengkodean, digunakan berbagai alat diantaranya menggunakan *Visual Basic 6* dan *Microsoft Windows Common Control 6.0 (SP6)*.

4. Tes

Setelah tahap pengkodean, maka dilakukan tahap pengujian dengan tujuan untuk menguji sistem yang telah dirancang sebelumnya. Pengujian yang dipakai adalah pengujian *black box*, dimana pengujian *black box* adalah pengujian yang menguji fungsionalitas dari sistem yang sudah dirancang dan dibuat. Dilakukan pengujian fungsionalitas yang terdapat dalam sistem untuk meengetahui apakah masih terdapat kesalahan atau tidak dalam setiap fungsi yang ada pada sistem.

INSTRUMEN PENGUJIAN

Pada prinsipnya meneliti adalah melakukan pengukuran terhadap fenomena sosial maupun alam. Karena pada prinsipnya meneliti adalah melakukan pengukuran, maka harus ada alat ukur yang baik. Alat ukur dalam penelitian biasanya dinamakan instrumen penelitian. Jadi instrumen penelitian adalah alat ukur yang digunakan mengukur fenomena alam maupun sosial yang diamati.(Sugiyono, 2012:147-148).

Proses pengujian merupakan proses untuk mengetahui apakah perangkat lunak sudah sesuai dengan yang diharapakan atau belum. Pengujian dilakukan dengan metode *black box* dimana menurut Hanif Al Fatta (2007:12) *black box* merupakan pengujian yang dilakukan dengan menjalankan suatu unit ataupun modul, yang kemudian diamati apakah hasil dari unit tersebut sesuai dengan yang diinginkan atau tidak.

Dalam pengujian *black box*, peneliti merancang skenario yang berupa lembar pengujian, di mana di dalam lembar pengujian tersebut berisi aspek-aspek fungsionalitas perangkat lunak. Pada lembar pengujian tersebut, aspek fungsionalitas akan mendapat penilaian dari pengguna perangkat lunak, apakah sudah sesuai dengan fungsionalitas atau tidak. Pengguna perangkat lunak juga dapat memeberi saran pada setiap aspek fungsionalitas jika memang diperlukan.

1. Uji Fungsi

Pada pengujian awal perangkat lunak Kamus Istilah Elektronika adalah menguji semua fungsi dasar dari aplikasi Kamus Istilah Elektronika. Fungsi dalam hal ini adalah kotak utama aplikasi, menu keluar aplikasi, sugesti teks, pemberian saran. Untuk lebih jelasnya akan dijabarkan dalam tabel 1.

Tabel 1. Pengujian Aplikasi Kamus Istilah Elektronika

No	Hal yang Diuji	Butir Pengujian	Luaran yang Diharapkan	Hasil
1.	Eksekusi Aplikasi	*Double-Click Icon* Aplikasi	Aplikasi dapat berjalan dengan lancar	
2.	Kotak teks masukan	Dimasukkan kata atau istilah yang diinginkan sesuai basis data	Menampilkan kata sesuai masukan di *listview* saran kata	
		Dimasukkan kata atau istilah yang tidak ada di	Menampilkan Saran kata	

		dalam basis data	terdekat dari nilai *edit distance* terkecil
		Fungsi *Key Enter*	Setelah pengguna memasukkan kata, fungsi enter dapat menampilkan saran
3.	Tombol Search	*Single-Click Action*	Menampilkan saran kata ataupun deskripsi istilah
4.	*Listview* Saran Kata	*Single-Click Action*	Dapat menampilkan deskripsi dari saran kata yang di-klik
5.	Tombol Keluar	*Single-Click Action*	Keluar dari Aplikasi

2. Pengujian Fitur Otokoreksi

Dalam menguji fitur otokoreksi penguji akan diberikan kata yang sengaja disalahkan yang digunakan untuk menguji kata yang akan disrankan dan sebarapa tingkat kebenaran dari fitu otokoreksi dalam memberikan koreksi terhadap kata yang salah. Kata yang akan dimasukkan oleh penguji dipaparkan dalam tabel 2.

Tabel 2. Pengujian Fitur Otokoreksi

No	Kata Masukan	Muncul /Tidak	Luaran yang diharapkan	Saran yang muncul
1.	Abberaton	Muncul /Tidak	Abberation	
2.	Basso	Muncul /Tidak	Bass	
3.	Callibrat	Muncul /Tidak	Callibrate	
4.	Datas	Muncul /Tidak	Data	
5.	Electro	Muncul /Tidak	Electron	
6.	Fas	Muncul /Tidak	Fan	
7.	Gas	Muncul /Tidak	Gap	
8.	Ham	Muncul /Tidak	Ham	
9.	Impuls	Muncul /Tidak	Impulse	
10.	Kelvon	Muncul /Tidak	Kelvin	
11.	Lamberer	Muncul /Tidak	Lambert	

3. Pengujian Algoritma *Levenshtein distance*

Dalam pengujian ini akan dilakukan 5 kategori pengujian yang sesuai dengan operasi yang ada pada algoritma levenshtein distance. Kaegori yang akan diujikan adalah penggantian satu huruf, penukaran dua huruf, kesalahan lebih dari satu huruf. Dengan 5 kategori tersebut penguji akan diberikan kata salah untuk diamsukkan dalam aplikasi Kamus Istilah Elektronika.

Tabel 3. Pengujian Pemberian Saran dengan levenshtein distance

Kategori	Kata Masukan	Kata yang Diingikan	Saran yang Muncul
Penggantian satu	Ampeve	Ampere	

huruf		
Penyisipan satu huruf	Accers	Access
Penghapusan satu huruf	Vot	Volt
Penggantian dua huruf	Vlot	Volt
Kesalahan lebih dari satu huruf	Abbetarion	Abberation

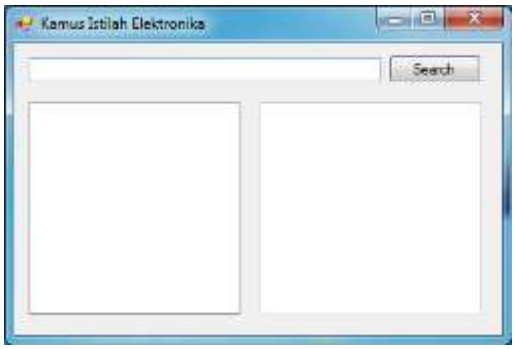
HASIL PENELITIAN DAN PEMBAHASAN

HASIL ANTAR MUKA

Berikut ini merupakah hasil dari penelitian yang dilakukan

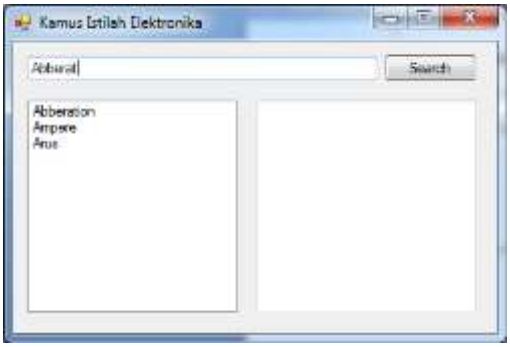
1. Rancangan Antar Muka

Pada sub bab ini akan dijelaskan tentang semua tampilan Graphical User Interface yang terdapat di dalam Aplikasi Kamus Istilah Elektronika :



Gambar 2. Tampilan Awal Aplikasi Kamus Istilah Elektronika

Pada gambar merupakan tampilan dalam pemberaina saran dari kata masukan "abberati" dengan menampilkan tiga saran yaitu "abberation", "lambert", "camera".



Gambar 3. Tampilan Pemberian Saran Kata

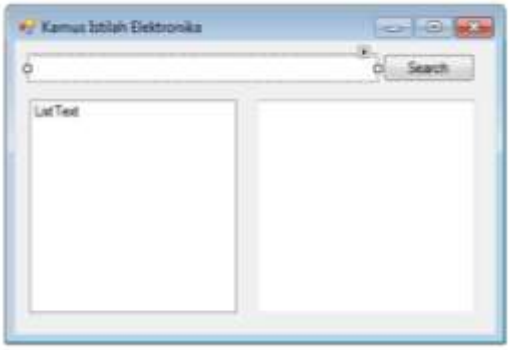
Dalam menampilkan deskripsi dari saran yang dikehendaki pengguna hanya perlu meng-klik kata yang dikehendaki dan dicari. Berikut tampilan dari deskripsi dari istilah yang dikehendaki.

Gambar 4. Tampilan Deskripsi Istilah

2. Implementasi Fitur Otokoreksi

Fitur otokoreksi sendiri memanfaatkan algoritma *levenshtein distance* dengan menampilkan hasil perhitungan nilai terkecil dari perbandingan *edit distance* sebagai pencarian terdekat dengan kata masukan dari pengguna yang salah. Berikut ini adalah alur secara garis besar fitur otokoreksi pada aplikasi Kamus Istilah Elektronika :

Membuat InputBox "Text" yang akan digunakan sebagai tempat pengguna memasukkan kata. Seperti yang dipaparkan dalam gambar dibawah yang merupakan kotak masukan kata yang akan dicari.



Gambar 5. Tampilan Box Kata Masukan

HASIL PENGUJIAN BLACK BOX

1. Hasil Uji Fungsi

Berikut ini hasil dari pengujian yang dilakukan dengan metode *black box* pada aplikasi Kamus Istilah Elektronika :

Tabel 4. Hasil Pengujian Fungsi Sistem Aplikasi

No	Hal yang Diuji	Butir Pengujian	Luaran yang Diharapkan	Hasil
1.	Eksekusi Aplikasi	*Double-Click Icon* Aplikasi	Aplikasi dapat berjalan dengan lancar	Start awal aplikasi tidak ada masalah. Sesuai dengan yang diharapkan
2.	Kotak teks masukan	Dimasukkan kata atau istilah yang diinginkan sesuai basis data	Menampilkan kata sesuai masukan di *listview* saran kata	Dapat menampilkan saran sesuai basis data
		Dimasukkan kata atau istilah yang tidak ada di dalam basis data	Menampilkan Saran kata terdekat dari nilai *edit distance* terkecil	Dapat menampilkan sesuai hasl yang diharapkan
		Fungsi *Key Enter*	Setelah pengguna memasukkan kata, fungsi enter dapat menampilkan saran	Fungsi *Key Enter* berfungsi dengan baik.
3.	Tombol Search	*Single-Click Action*	Menampilkan saran kata ataupun deskripsi istilah	Tombol *Search* berjalan sesuai.
4.	*Listview* Saran Kata	*Single-Click Action*	Dapat menampilkan deskripsi dari saran kata yang di-klik	Dengan satu klik pada kata yang diharapkan dapat memunculkan deskripsi
5.	Tombol Keluar	*Single-Click Action*	Keluar dari Aplikasi	Dapat keluar dengan baik

2. Hasil Pengujian Fitur Otokoreksi

Beberapa kata dimasukkan dengan sengaja disalahkan untuk menguji fitur otokoreksi di dalam perangkat lunak Kamus Istilah Elektronika. Dalam pengujian kata yang sengaja disalahkan dicoba sesaui dengan teori operasi levenshtein distance. Berikut ini adalah beberapa kata - kata salah yang dimasukkan oleh penguji. Untuk lebih jelas hasil dan saran yang muncul bisa melihat Tabel 5.

Tabel 5. Penggunaan Fitur Otokoreksi

No	Kata Masukan	Muncul/ Tidak	Luaran yang diharapkan	Saran yang muncul
1	Abberaton	Muncul/ Tidak	Abberation	Abberation, Generator,

				Lambert
2	Basso	Muncul/ Tidak	Bass	Bass, Bar, Band
3	Callibrat	Muncul/ Tidak	Callibrate	Callibrate, Callibrator, Camera
4	Datas	Muncul/ Tidak	Data	Data, Draft, FET
5	Electro	Muncul/ Tidak	Electron	Electron, Electrode, Elektron
6	Fas	Muncul/ Tidak	Fan	Fan, Bar
7	Gas	Muncul/ Tidak	Gap	Gap, Bar
8	Ham	Muncul/ Tidak	Ham	Ham
9	Impuls	Muncul/ Tidak	Impulse	Impulse, Arus, Coil
10	Kelvon	Muncul/ Tidak	Kelvin	Kelvin, Kilo, Kerma
11	Lambrer	Muncul/ Tidak	Lambert	Lambert, Cable, Camera

Dalam menampilkan saran, fitur otokoreksi lebih condong membandingkan antara perbandingan edit distance apabila terdapat perbedaan antara kata asal dan kata target. Bukan berdasarkan abjad awal dari kata asal.

Hal tersebut dapat dilihat dari nomer 1, 4, 6, 7, 9 dan 11 dengan munculnya kata - kata saran yang meleset jauh dari abjad yang seahrusnya yaitu "A". Dalam perbandingan fiur otokoreksi di dalam aplikasi Kamus Istilah Elektronika bukan per karakter awal tetapi menggunakan nilai edit distance terkecil.

Perhitungan pada nomer 1 jumlah kata asal "abberaton" adalah 9 dan kata target "abberation" berjumlah 10. Secara kasat mata membutuhkan satu operasi penambahan karakter "i" di dalam kata asal "abberaton". Karena hanya menggunkan stau operasi maka dapat dikatakan nilai edit disance adalah 1.

Secara perhitungan matriks dengan Aplikasi Microsoft Excel yang telah dibuat untuk perhitungan edit distance bernilai 1.

		A	B	B	E	R	A	T	O	N
	0	1	2	3	4	5	6	7	8	9
A	1	0	1	2	3	4	5	6	7	8
B	2	1	0	1	2	3	4	5	6	7
B	3	2	1	0	1	2	3	4	5	6
E	4	3	2	1	0	1	2	3	4	5
R	5	4	3	2	1	0	1	2	3	4
A	6	5	4	3	2	1	0	1	2	3
T	7	6	5	4	3	2	1	0	1	2
I	8	7	6	5	4	3	2	1	1	2
O	9	8	7	6	5	4	3	2	1	2
N	10	9	8	7	6	5	4	3	2	1

Gambar 6. Perbandingan Matriks "Abberaton" dan "Abberation"

Sedangkan perbandingan untuk "Abberaton" dengan "Generator" dan "Lambert". Secara jumlah karakter atau panjang kata adalah 9 : 9 : 7. Untuk "Generator" secara panjang kata sama dengan "Abberaton" tetapi panjang kata yang dibutuhkan adalah 10. Dengan operasi di dalam perbandingan dua kata tersebut lebih dari 1. Dengan kata lain sudah melibihi nilai edit distance dari "abberation" yang mempunyai nilai 1.

Karakter dalam "Generator" hanya memiliki kemiripan dari target yaitu di bagian tengah karakter "ERAT". Selain itu masih menggunakan 5 operasi untuk bisa memenuhi padanan kata targer yaitu "Abberation".

Dari gambar tersebut terlihat nilai edit distance yang melebihi dari "Abberation". Perhitungan tabel sudah dianggap valid dikarenakan perhitungan tersebut sudah dibuat dalam bentuk digital dari perhitungan manual.

		G	E	N	E	R	A	T	O	R
	0	1	2	3	4	5	6	7	8	9
A	1	1	2	3	4	5	5	6	7	8
B	2	2	2	3	4	5	6	6	7	8
B	3	3	3	3	4	5	6	7	7	8
E	4	4	3	4	3	4	5	6	7	8
R	5	5	4	4	4	3	4	5	6	7
A	6	6	5	5	5	4	3	4	5	6
T	7	7	6	6	6	5	4	3	4	5
I	8	8	7	7	7	6	5	4	4	5
O	9	9	8	8	8	7	6	5	4	5
N	10	10	9	8	9	8	7	6	5	5

Gambar 7. Perbandingan "Abberation" dengan "Generator"

Perhitungan gambar 4.6 nilai edit distance dari perbandingan "Abberation" dan "Generator" bernilai 5. Perbandingan nilai dengan "Abberation" yang hanya satu membuat sistem memilih kata "Abberation". Karena sistem yang dibuat memilih nilai terkecil dalam perbandingan kata. Untuk "Lambert" juga berlaku sama dengan nilai edit distance melibihi nilai edit distance dari "Generator" sehingga muncul di saran nomer 3.

Dalam menampilkan saran fitur otokoreksi memang sengaja hanya memunculkan tiga saran kata. Hal tersebut disengaja karena untuk menampilkan saran kata yang lain yang mendekati. Dan untuk melihat perbandingan antara semua kata yang ada di basis data.

Untuk kemungkinan jumlah basis data yang belum memenuhi jumlah yang dikehendaki juga dapat memunculkan saran kata lain. Dikarenakan jumlah panjang kata yang hampir sama hanya sedikit. Sehingga kemunculan kata saran hanya jumlah panjang kata yang hampir sama jumlah panjang katanya.

3. Hasil Pengujian Algoritma *Levenshtein distance*

Berikut hasil dari pengujian algoritma levenshtein distance beserta edit distance :

Tabel 6. Pengujian Pemberian Saran dengan *Levenshtein distance*

Kategori	Kata Masukan	Kata yang Diingikan	Saran yang Muncul
Pengganti an satu huruf	Ampeve	Ampere	Ampere
Penyisipa n satu huruf	Accers	Access	Access
Penghapu san satu huruf	Vot	Volt	Volt
Pengganti an dua huruf	Vlot	Volt	Volt
Kesalahan lebih dari satu huruf	Abbetari on	Abberati on	Abberati on

PEMBAHASAN

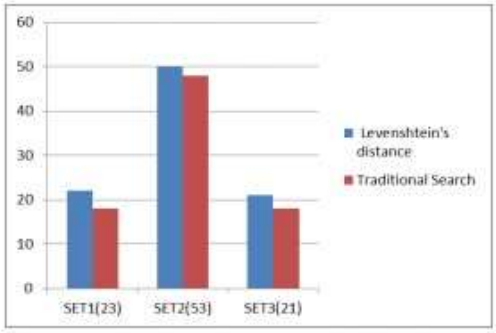
Penerapan fitur otokoreksi dalam aplikasi Kamus Istilah Elektronika memberikan kemudahan bagi pengguna dalam proses pencarian kata maupun dalam menghindari kesalahan penulisan dari pengguna. Hal tersebut dapat diartikan bahwa fitur otokoreksi ini sangat efektif untuk digunakan dalam aplikasi Kamus Istilah Elektronika sebagai fitur pembantu dalam mempercepat pencarian kata dan mengurangi kesalahan penulisan dari masukan pengguna. Seperti yang dipaparkan dalam penelitian yang dilakukan oleh Dwiastuti, Rachamania Nur. dkk (2013) dan Pyriana, Dewi Rokhmah. dkk (2013) menyimpulkan bahwa penggunaan algoritma *levenshtein distance* dapat mengurangi kesalahan yang muncul dari pengguna dengan implementasi dari algoritma *levenshtein distance*. Dan penelitian yang dilakukan oleh Dwitiyastuti, Rachmania Nur. dkk (2013) menyebutkan koreksi yang dapat dilakukan mencapai persentase 100% akurat.

Selanjutnya dalam penelitian yang dilakukan oleh Pyriana, Dewi Rokhmah. dkk (2013) penggantian kata dengan tingkat kompleksitas tinggi membutuhkan banyak basis data karena dalam membandingkan *string* hanya

mengandalkan nilai *edit distance* terendah. Sehingga apabila ada kata yang salah menurut sistem maka akan diambil kata dengan nilai *edit distance* terendah.

Untuk kecepatan dan ketepatan sudah layak untuk digunakan dalam aplikasi. Dalam penggantian kata yang salah tidak membutuhkan waktu yang terlalu lama karena perhitungan yang dilakukan oleh algoritma *levenshtein distance* terhitung cepat. Seperti halnya yang ditulis oleh Pyriana, Dewi Rokhmah. dkk (2013) "Waktu yang dibutuhkan untuk eksekusi berbanding lurus dengan jumlah input kata yang dibutuhkan untuk ekseskusi. Semakin banyak kata yang dimasukkan, semakin banyak pula waktu yang dibutuhkan untuk eksekusi. Sedangkan jumlah kesalahan ejaan kata pada suatu teks tidak berpengaruh pada waktu eksekusi."

Pada penelitia yang dilakukan oleh Abraham, Riya Mary (2014) algoritma *levenshtein distance* diimplementasikan dalam pencarian tradisional, algoritma ini sedikit unggul. Sehingga tidak salah jika algoritma ini efektif untuk digunakan dalam berbagai aplikasi atau kegunaan lain. Gambar di bawah merupakan grafik dari perbandingan algoritma *levenshtein distance* dan pencarian tradisional yang dilakukan oleh Abraham, Riya Mary (2014) :



Gambar 8. Grafik Perbandingan Algoritma dan Pencarian Tradisional

Dari penelitian Abraham, Riya Mary (2014) menyimpulkan algoritma *levenshtein distance* memiliki nilai tinggi dan dalam memprediksi kemiripan kata yang dimasukkan pengguna. Jika dalam aplikasi Kamus Istilah Elektronika tidak terdapat pengguna. Jika dalam aplikasi Kamus Istilah Elektronika tidak terdapat perbandingan penggunaan otokoreksi dengan algoritma *levenshtein distance*, karena keduanya salaing bekerja sama dalam proses pencarian kata.

Penelitian yang dilakukan oleh Lhoussain, Aouragh Si. (2015) tentang implementasi *levenshtein distance* pada pengoreksian ejaan kontekstual. Lhoussain melakukan pengenalan metode *bi-gram* pada algoritma *levenshtein distance.* Hasilnya didapatkan bahwa kombinasi tersebut dapat meningkatkan kepuasan dalam solusi metode tersebut.

SIMPULAN

Pengunaan fitur otokoreksi dalam pencarian kata istilah dapat memudahkan pengguna dalam mencari istilah dan memudahkan hasil pencarian istilah terdekat apabila terjadi kesalahan dalam penulisan masuka kata yang akan dicari. Fitur otokoreksi yang diimplementasikan dalam Kamus Istilah Elektronika sangat efektif untuk digunakan.

DAFTAR PUSTAKA

Abraham, Riya Mary. 2014. *Use of Edit distance Algorithm to Search A Keyword In Cloud Environment.* International Journal of Database Theory And Application. (7) 6 : 223-232.

Chen, Yoke Yie., dkk. 2014. *E-Mail Hoax Detection System Using Levenshtein distance Method.* Journal of Computer. (9) 2 : 445-446.

Dwitiyastuti, Rachmania Nur,. dkk. 2013. *Pengoreksi Kesalahan Ejaan Bahasa Indonesia Menggunakan Metode Levenshtein distance.* http://www.eletronika.studentjournal.ub.ac.id/. Diakses pada 10 Februari 2015.

Hakim, Ahmad Luqman, Ferdian Rikza. 2013. *Sistematika Penulisan Kamus Berdasarkan Entri, Jumlah Bahasa, dan Metode Masa/Periode.* http://www.fai.uad.ac.id/. Diakses pada 3 Februari 2015

Harfatiani, Rina Rizky. 2007. *Teknik Riset Operasi*. Surabaya: Kartika.

Ilmy, Muhammad Bahari., dkk. 2006. *Penerapan Algortima Levenshtein distance untuk Mengoreksi Kesalahan Pengejaan pada Editor Teks.*

http://www.informatika.stei.itb.ac.id/. Diakses pada 10 Februari 2015.

Ladjamuddin, Al-Bahra bin. 2005. *Analisis dan Desain Sistem Informasi.* Yogyakarta: Graha Ilmu.

Lhoussain, Aouragh Si. 2005. *Adapting The Levenshtein distance to Contextual Spelling and Correction.* International Journal of Computer Science and Application. (12) 1 : 127-133.

Pressman, Roger S. 2002. *Rekayasa Perangkat Lunak: Pendekatan Praktisi (Buku 1).* Yogyakarta: Andi.

Pyriana, Dewi Rokhmah., dkk. 2013. *Program Aplikasi Editor Kata Bahasa Indonesia Menggunakan Metode Approximate String Matching dengan Algoritma Levenshtein distance Berbasis Java.* http://www.filkom.ub.ac.id/. Diakses pada 10 Februari 2015.

Rosa A, S dan M Salahuddin. 2011. *Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek).* Bandung: Informatika.

Sugiyono. 2012. *Metode Penelitian Kuantitatif Kualitatif dan R&D.* Bandung: Alfabeta.

Wahyudi, Bambang. 2003. *Pengantar Struktur Data dan Algoritma.* Yogyakarta: Andi.

Widodo, P.D. 2003. *Kamus Istilah Internet dan Komputer.* Jombang: Lintas Media.

Wiyanto, Asul., dkk. 2005. *Mampu Berbahasa Indonesia.* Jakarta: Grasindo.

Yao, Zhiqiang. 2013. *Implementation of The Autocomplete feature of The Textbox Based on Ajax and Web Service.* Journal of Computer (8) 9 : 2199-2201.