Optimasi Pencarian Kata Pada Kamus Aneka Bahasa Menggunakan Algoritma *Levenshtein Distance*

Ida Bagus Ketut Surya Arnawa
STIKOM Bali
Jl.Raya Puputan No. 86 Renon, Denpasar-Bali
arnawa@stikom-bali.ac.id

Abstrak

Sebagian besar masyarakat pulau Bali menggunakan bahasa Bali sebagai alat komunikasi sehari-hari. Banyak cara yang dapat dilakukan oleh masyarakat Bali dan wisatawan untuk salaing memahami dan mengerti bahasa lawan bicara mereka, salah satunya yaitu dengan mempelajari kamus aneka bahasa.Kamus adalah sejenis buku rujukan yang menerangkan makna kata-kata, biasanya disusun menurut abjad beserta penjelasan tentang makna dan pemaikannya.Kamus aneka bahasa (Bali-Indonesia-Inggris-Prancis-Jerman) merupakan salah satu kamus yang menjelaskan makna kata dari satu bahasa ke bahasa yang lainnya.Dalam melakukan pencarian kata terutama pada kamus online, sering kali pengguna menginputkan kata yang bukan merupakan ejaan yang benar atau salah ketik.Sebagai contoh pengguna mengetikkan kata "suksam", padahal ejaan yang benar dalam bahasa Bali adalah "suksma". Tentu saja pengguna akan memperoleh informasi yang kurang lengkap dan bahkan pengguna gagal dalam mendapatkan informasi yang sesuai dengan kata yang ingin dicari.Untuk mengatasi permasalahan yang dialami pengguna dalam melakukan pencarian kata pada kamus aneka bahasa yaitu menggunakan algoritma Levenshtein Distence.Implementasi algoritmalevenshtein pada sistem kamus aneka bahasa sudah dapat mengatasi permasalahan padakesalahan ejaan kata pada saat menterjemahkan kata dengan mekanisme penambahan, penyisipan dan penghapusan karakter.Pengujian yang dilakukan dengan cara membandingkan output dari aplikasi kamus aneka bahasa online yang menggunakan algoritma levenstein distance dan yang tidak menggunakan algoritma levenstein distence, Hasil dari pengujian aplikasi yang mengunakan algoritma levenstein distance dapat mengoptimasi pencarian kata kunci yang mengalami kesalahan ejaan.

Kata kunci:*kamus, levenshtein distance, optimasi pencarian*

Abstract

Most of the island people of Bali use Balinese language as a means of daily communication. Many ways that can be done by the Balinese and tourists to understand and understand the language of their speakers, one of which is by studying the dictionary of various languages. The dictionary is a kind of reference book that explains the meaning of words, usually arranged alphabetically along with an explanation of its meaning and enhancement. Dictionary of various languages (Bali-Indonesia-English-France-Germany) is one dictionary that explains the meaning of words from one language to another. In doing a word search especially on an online dictionary, often users input a word that is not a correct spelling or a typo. For example the user typed the word "suksam", while the correct spelling in Balinese language is "suksma". Of course the user will get information that is not complete and even the user fails in getting the information in accordance with the word you want to search. To overcome the problems experienced by users in performing word search in various language dictionaries that use Levenshtein Distence algorithm Implementation of levenshtein algorithm in various language dictionary system has been able to overcome the problem of spelling mistake of word when translating word with mechanism of addition, insertion and deletion of character.Testing is done by comparing the output of the online language dictionary application using levenstein distance algorithm and which do not use levenstein distence algorithm. The result of application testing using levenstein distance algorithm can optimize the search of keywords that experience spelling mistake..

Keywords*: dictionary, levenshtein distance, search optimization*

1. Pendahuluan

Bali yang dijuluki Pulau Dewata merupakan salah satu *icon* pariwisata Indonesia.Bali tidak hanya terkenal di dalam negeri tetapi juga di luar negeri.Bali memiliki kekayaan dan keindahan alam, serta keunikan seni budaya yang menjadi daya tarik utama bagi para wisatawan.Sebagian besar masyarakat pulau Bali menggunakan bahasa Bali sebagai alat komunikasi sehari-hari.Untuk dapat saling berinteraksi dengan baik antara masyarakat Bali dengan wisatawan, maka kedua belah pihak perlu saling memahami dan mengerti bahasa lawan bicara mereka. Banyak cara yang dapat dilakukan oleh masyarakat Bali dan wisatawan untuk saling memahami dan mengerti bahasa lawan bicara mereka, salah satunya yaitu dengan mempelajari kamus aneka bahasa.

Kamus adalah sejenis buku rujukan yang menerangkan makna kata-kata, biasanya disusun menurut abjad beserta penjelasan tentang makna dan pemaikannya.Kamus berfungsi untuk membantu seseorang mengenal perkataan baru.Selain menjelaskan makna kata, kamus juga mempunyai pedoman sebutan, asal-usul (etimologi) suatu dan juga contoh penggunaan bagi suatu perkataan.Untuk memperjelas kadang kala terdapat juga ilustrasi dalam kamus.Kamus adalah kitab yang berisi kata – kata dan arti atau keterangan yang disusun secara alfabetik [3].Berdasarkan isinya kamus dapat dibedakan menjadi beberapa jenis diantaranya kamus ekabahasa yang hanya menggunakan satu bahasa, kamus dwibahasa yaitu kamus yang menggunakan dua bahasa dan kamus aneka bahasa yaitu kamus yang sekurang-kurangnya menggunakan tiga bahasa atau lebih.

Kebutuhan masyarakat terhadap layanan teknologi sangat besar. Salah satunya kebutuhan akan ketersediaan kamus. Kamus aneka bahasa (Bali-Indonesia-Inggris-Prancis-Jerman) merupakan salah satu kamus yang menjelaskan makna kata dari satu bahasa ke bahasa yang lainnya.Seiring berkembangnya ilmu pengetahuan dan teknologi, beberapa jenis kamus tersedia di toko buku, perpustakaan dan banyak juga tersedia kamus *online* dirasa masih kurang memenuhi kebutuhan.Dalam melakukan pencarian kata terutama pada kamus *online*, sering kali pengguna menginputkan kata yang bukan merupakan ejaan yang benar atau salah ketik.Sebagai contoh pengguna mengetikkan kata "*suksam*", padahal ejaan yang benar dalam bahasa Bali adalah "*suksma*". Tentu saja pengguna akan memperoleh informasi yang kurang lengkap dan bahkan pengguna gagal dalam mendapatkan informasi yang sesuai dengan kata yang ingin dicari.Penelitian sebelumnya yang dilakukan oleh B. P. Pratama dan S. A. Pamungkas yaitu membangun sebuah sistem yang mampu mendeteksi tingkat kemiripan antar dokumen teks menggunakan algoritma *Levenshtein distance* dengan menambahkan proses *case folding, tokenizing, stopword removal, stemming,* dan *sorting*. Proses pencocokan *string* pada algoritma ini dapat menghasilkan nilai *distance* yang menjadi penentu persentase bobot *similarity*. Analisa penggunaan *stopword removal, stemming*, dan *sorting* dilakukan untuk melihat pengaruhnya terhadap kinerja algoritma *Levenshtein distance*.[7].

Untuk mengatasi permasalahan yang dialami pengguna dalam melakukan pencarian kata pada kamus aneka bahasa, maka diperlukan suatu metode pendekatan pencarian string agar hasil pencarian dapat maksimal.Ada beberapa algoritma yang dapat diimplementasikan dalam memberikan kata saran yang paling mendekati dari kata yang salah pengetikannya. Salah satu algoritma yang dapat digunakan adalah algoritma *Levenshtein Distence* yang dapat menghitung jarak keterbedaan antar dua string [2]. Algoritma *Levenshtein Distence* merupakan salah satu algoritma terbaik dalam pengecekan kata [4].

2. Tinjauan Pustaka

2.1 Algoritma Levenshtein

Algoritma *Levenshtein* ditemukan oleh ilmuan asal Rusia bernama Vladimir Levenshtein pada tahun 1963, algoritma ini juga disebut dengan algoritma *Edit Distance*. Perhitungan *edit distance* didapatkan dari matriks yang digunakan untuk menghitung jumlah perbedaan *string* antara dua *string*, sebagai contohhasil penggunaan algoritma ini, *string* "komputer" dan "*computer*" memiliki *distance* 1 karena hanya perlu dilakukan satu operasi saja untuk mengubah satu *string* ke *string* yang lain. Dalam kasus dua *string* di atas, *string* "*computer*" dapat menjadi "komputer" hanya dengan melakukan satu penukaran karakter "c" menjadi "k" [2].

Algoritma *Levenshtein* digunakan secara luas dalam berbagai bidang, misalnya mesin pencari, pengecek ejaan (*spell checking*), pengenal pembicaraan (*speech recognition*), pengucapan dialek, analisis DNA, pendeteksi pemalsuan, dan lain-lain.Algoritma ini menghitung jumlah operasi *string* paling sedikit yang diperlukan untuk mentransformasikan suatu *string* menjadi *string* yanglain[1].Algoritma *Levenshtein* bekerja dengan menghitung jumlah minimum pentranformasian suatu *string* menjadi *string* lain yang meliputi penghapusan, penyisipan, dan penukaran.

Selisih perbedaan antar *string* dapat diperoleh dengan memeriksa apakah suatu *string* sumber sesuai dengan *string* target. Nilai selisih perbedaan ini disebut juga *edit distance* atau jarak *Levenhstein.* Jarak *Levenshtein* antar *string* "s" dan *string* "t" tersebut adalah fungsi D yang memetakan (s,t) ke suatu bilangan *real* nonnegatif, sebagai contoh diberikan dua buah *string* s = s(1)s,(2),s(3),...,s(m) dan t =

t(1),t(2),t(3),...,t(n) dengan | s | = m dan | t | = n sepanjang alfabet V berukuran r sehingga “s” dan “t” anggota dari V*. s(j) adalah karakter pada posisi ke-j pada *string* “s” dan t(i) adalah karakter pada posisi ke-i pada string “t”. Sehingga jarak Levenshtein dapat didefinisikan sebagai.

$$D(s,t) = d(s1,t1) + d(s2,t2) + \ldots + d(s1,t1)$$

D(s,t) adalah banyaknya operasi minimum dari operasi penghapusan, penyisipan dan penukaran untuk menyamakan string s dan t. Pada implementasi pencocokan antar string, ketiga operasi tersebut dapat dilakukan sekaligus untuk menyamakan string sumber dengan string target.

2.2 UML (*Unifield Modeling Language*)

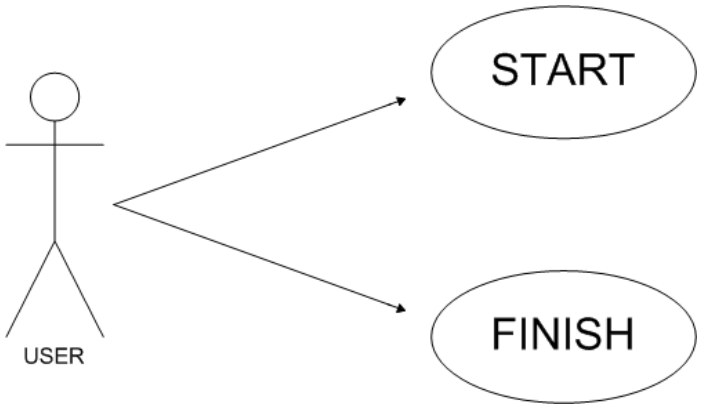
Notasi UML dibuat sebagai kolaborasi dari Grady Booch, DR. James Rumbough, Ivar Jacobson, Rebecca Wirfs-Brock, Peter Yourdon and lainnya. Jacobson menulis tentang pendefinisian persayaratan-persayaratan sistem yang disebut use case. Juga mengembangkan sebuah metode untuk perancangan sistem yang disebut *Object-Orioented Software Enginnering* (OOSE) yang berfokus pada analisis.Boorch, Rumbough dan Jacobson bisa disebut tiga sekawan (*tree amigos*).Semuanya bekerja di *Rational Software Corporation* dan berfokus pada standarisasi dan perbaikan ulang UML.Penggabungan beberapa metode menjadi UML dimulai 1993.Pada akhir tahun 1995 *Unifield Method* versi 0.8 diperkenalkan.*Unifield Method* diperbaiki dan diubah menjadi UML pada tahun 1996 UML 1.0 disahkan dan diberikan pada *Object Technology Group* (OTG) pada tahun 1997 dan pada tahun itu juga beberapa perusahaa pengembang utama perangkat lunak mulai mengadopsinya. Pada tahun yang sama*OMG* merilis UML 1.1 sebagai standar industri [5]. UML merupakan metode pemodelan secara visual sebagai sarana untuk merancang *software* berorientasi objek dimanasemua elemen dan diagram yang terdapat didalamnya berbasiskan pada pradigman objek oriented. UML tidak hanya bahasa pemrograman visual saja namun juga dapat dihubungkan dengan bahasa pemrograman lainnya sepert *Java* C++ VIsual *Basic* atau dihubungkan secara langsung kedalam *object oriented database*.

2.2.1 Diagram-diagram dalam UML

UML menyediakan beberapa diagram visual yang menunjukan berbagai aspek dalam sistem dimana fungsinya agar pengguna mendapatkan banyak pandangan terhadap sistem informasi yang akan dibangun. Ada beberapa diagram yang disediakan dalam UML :

a. Diagram Use Case

Diagram use case atau *use case diagram* adalah sebuah kegiatan yang dilakukan oleh sistem, biasanya dalam menanggapi permintaan dari pengguna sistem [6]. *Diagram use case* atau *use case diagram* menyajikan interaksi antara *use case* dan aktor. Dimana aktor dapat berupa orang peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. *Use Case* menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai. Contoh *diagram use case*sederhana : Diagram *usecase* dapat dilihat pada gambar 1.



Gambar 1*Diagram Use Case*

b. Diagram Aktifitas

Diagram aktivitas atau *activity diagram* adalah sebuah diagram alur kerja yang menjelaskan berbagai kegiatan pengguna (atau sistem), orang yang melakukan masing-masing aktivitas, dan aliran sekuensial dari aktivitas-aktivitas tersebut. Diagram aktivitas atau *activity diagram* menggambarkan aliran fungsionalitas sistem. Pada tahap pemodelan bisnis diagram aktifitas dapat digunakan untuk menunjukan aliran kerja bisnis (*business work flow*). Dapat juga digunakan untuk menggambarkan aliran kejadian (*flow of events*) dalam *use case*. Diagram aktifitas menunjukan informasi yang sama sebagaimana dalam aliran kejadian dengan teks.

c. Diagram Sekuensial

Diagram sekuensial atau *sequence diagram* digunakan untuk menunjukan aliran fungsionalitas dalam *use case*. Misalkan dalam *use case* "menarik uang" mempunyai beberapa kemungkinan, seperti penarikan uang secara normal, percobaan penarikan uang tanpa kecukupan ketersediaan dana penarikan dengan penggunaan PIN yang salah dan lainnya.

d. Diagram Kolaborasi

Diagram kolaborasi atau *collaborating diagram* menunjukan informasi yang sama persis dengan diagram sekuensial, tetapi bentuk dan tujuan yang berbeda. Pada diagram sekuensial, keseluruhan interaksi berdasarkan urutan waktu, tetapi pada diagram kolborasi, interaksi antar objek atau aktor ditunjukan dengan arah panah tanpa keterangan waktu.

e. Diagram Kelas

Diagram kelas atau *class diagram* menunjukan interaksi antar kelas dan sistem. Sebagai contoh, nomor *account* milik X adalah sebuah objek dari kelas *account*. Kelas mengandung informasi dan tingkah laku (*behavior*) yang berkaitan dengan informasi tersebut. Kelas *account* mengandung nomor PIN pengguna dan tingkah laku untuk mengecek PIN. Sebuah kelas pada diagram kelas dibuat untuk setiap tipe objek pada diagram sekuensial atau diagram kolaborasi.

f. Diagram *Statechart*

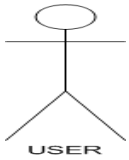
Diagram statechart atau satechart diagram menyediakan sebuah cara meodelkan bermacam-macam keadaan yang memungkinkan dialami oleh sebuah objek. Jika didalam diagram kelas menunjukan gambaran statis kelas-kelas dan relasinya, diagram *statechart* digunakan untuk memodelkan tingkah laku dinamik sistem.

2.2.2 Konsep Pemodelan *UseCase*

Konsep dasar pemodelan *use case*meliputi :*use case*, aktor, relasi, diagram aktifitas dan *diagram use case*. Pemodelan *use case* hampirsama dengan pemodelan bisnis hanya saja perbedaan utama adalah jika pemodelan bisnis berfokus pada organisasi, sedangkan pemodelan sistem berkonsentrasi pada sistem yang sedang dibangun.

a. Aktor

Aktor adalah seseorang atau apa saja yang berhubungan dengan sistem yang dibangun. Dalam UML, aktor dipresentasikan menggunakan notasi berikut :*Use case* aktor dapat dilihat pada gambar 2.



Gambar 2 Use Case Aktor

Ada 3 tipe aktor: pengguna sistem, sistem lain yang berhubungan dengan sistem yang sedang dibangun, dan waktu. Tipe pertama aktor secara fisik atau seorang pengguna, ini adalah gambaran seorang aktor yang secara langsung berhubungan dengan sistem. Tipe aktor kedua adalah sistem lain, misalkan dalam sistem persoaalan apotik mungkin memerlukan antar muka dengan aplikasi external untuk memvalidasi pembelian menggunakan kartu kredit. Maka dalam kontek ini, sistem aplikasi kartu kredit termasuk aktor terhadap sistem penjualan obat di apotik. Tipe aktor yang ketiga adalah waktu, waktu menjadi aktor ketika pada waktu tertentu memicu beberapa kejadian dalam sistem

b. Use Case

Use case adalah bagian tingkat tinggi dari fungsionalitas yang disediakan oleh sistem. Dengan kata lain, *use case* menggambarkan bagaimana seseorang menggunakan sistem. Untuk mengidentifikasi *use case*, dapat dilakukan dengan menjawab pertanyaan : apa yang masing-masing aktor kerjakan dalam sistem. Dalam UML, *use case* disimbolkan sebagai berikut : Notasi *use case* dapat dilihat pada gambar 3.

Gambar 3 Notasi *Use Case*

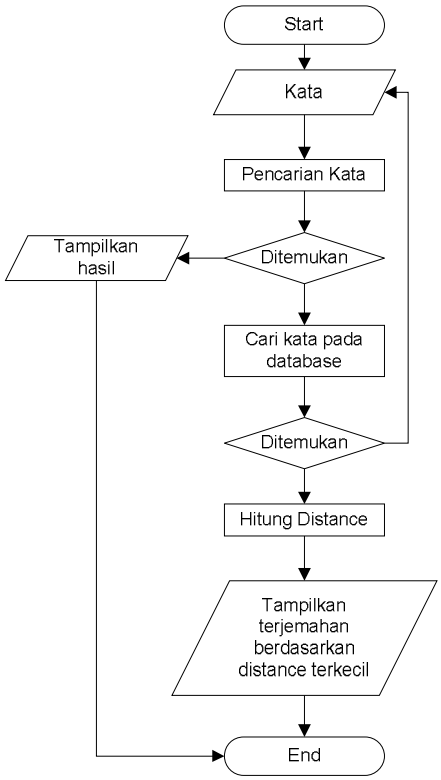
3. Metode Penelitian

3.1Tempat Waktu Penelitian

Penelitian dilakukan selama 9 bulan di STMIK-STIKOM Bali Jalan Raya Puputan No 86 Renon Denpasar Bali.

3.2Data Penelitian

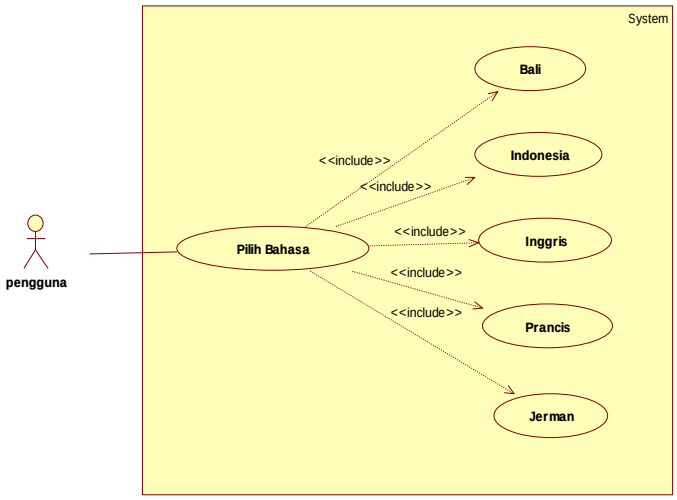
Data yang digunakan dalam penelitian ini adalah data yang terdapat kamus bahasa asing.Pada aplikasi user menginputkan sebuah kata kunci, kemudian aplikasi mengolah kata tersebut untuk mendapatkan bobot kemiripan.Berikut ini adalah flowchart penerjemahan kata dengan penerepan algoritma *levenstein distance.*



4. Hasil dan Pembahasan

4.1 Usecase diagram

Berikut ini adalah *usecase* diagram untuk aplikasi kamus aneka bahasa. *Usecase* aplikasi kamus aneka bahasa dapat dilihat pada gambar 4.Pada *usecase* tersebut pengguna dapat memilih salah satu bahasa yang ingin diterjemahkan dari sekian bahasa yang tersedia.



Gambar 4.*Use Case Diagram*

4.2Implementasi Sistem

4.2.1 Halaman Utama Sistem

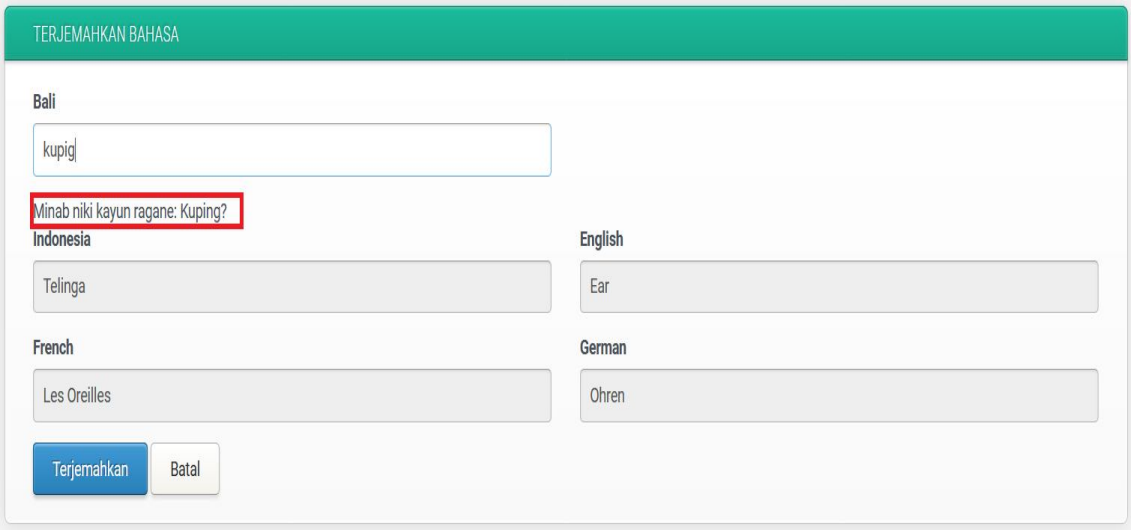
(a) (b)

Gambar 3.(a) Tampilan awal versi*Desktop* (b) Tampilan awal versi*Mobile*

Gambar 3 merupakan halaman utama yang digunakan user untuk memilih bahasa yang tersedia pada kamus aneka bahasa untuk melakukan terjemahan.

4.2.2 Halaman Terjemahkan Bahasa

Proses penterjemahan pada aplikasi dengan menggunakan kata “kupng”, tidakmenampilkan hasil dikarenakan kata tersebut tidak sesuai. Pada kondisi seperti ini aplikasiakanmengganggap kata yang digunakan mengalami kesalahan pada proses pengejaan sehingga aplikasi memberikan saran kata kunci “kuping” dan hasil pencarian yang menggunakan saran kata dapat dilihat pada Gambar 4.



(a)

Gambar 4.Halaman Terjemahan dari Bahasa Bali

Pada saat melakukan penerjemahan pada aplikasi menggunakan kata yang tidak sesuai yang disebabkan oleh kesalahaan pada pengejaan, maka aplikasi akan memberikan saran kata yang merupakan hasil dari perbaikan ejaan kata yang sebelumnya. Aplikasi memberikan saran kata "kuping" yang merupakan perbaikan ejaan dari kata "kupng". Adapun proses perbaikan ejaan dapat dijelaskan sebagai berikut :

1. Mengkonversi kata kunci "kupng" ke dalam array.
2. Melakukan proses seleksi pada semua kata yang tersimpan pada tabel. Kata yang digunakan perbandingan adalah kata yang memiliki panjang karakter (P) antara Pkata – 3 sampai Pkata + 3. Sehingga kata kunci "kupng" dengan panjang karakter 5, maka kata kata yang digunakan sebagai pembanding adalah kata yang memiliki panjang karakter diantara 2 – 8.
3. Melakukan perhitungan jarak dengan menggunakan metode *Levenshtein* terhadap kata "kupng" dengan setiap kata yang terpilih pada tabel. Sebagai contoh tiga kata kunci yang dipilih sebagai perbandingan dan akan dilakukan perhitungan jarak yaitu kata "kelod" dengan panjang karakter = 5, kata "kuping" dengan panjang karakter = 6 dan kata "kauh" dengan panjang karakter = 4.

Tabel 1. Menghitung nilai jarak kata kupng dengan kata kelod

		K	U	P	N	G
	0	1	2	3	4	5
K	1	0	1	2	3	4
E	2	1	1	2	3	4
L	3	2	2	2	3	4
O	4	3	3	3	3	4
D	5	4	4	4	4	4

Tabel 2. Menghitung nilai jarak kata kupng dengan kata kuping

		K	U	P	N	G
	0	1	2	3	4	5
K	1	0	1	2	3	4
U	2	1	0	1	2	3
P	3	2	1	0	1	2
I	4	3	2	1	1	2
N	5	4	3	2	1	2
G	6	5	4	3	2	1

Tabel 3. Menghitung nilai jarak kata kupng dengan kata kauh

		K	U	P	N	G
	0	1	2	3	4	5
K	1	0	1	2	3	4
A	2	1	1	2	3	4
U	3	2	2	2	3	4
H	4	3	3	3	3	4

Dari perhitungan yang telah dilakukan melalui Tabel.1, Tabel.2 dan Tabel.3 makadiperoleh nilai jarak untuk masing-masing kata yang dibandingkan yaitu sebagai berikut :
Nilai jarak (kupng, kelod) = 4
Nilai jarak (kupng, kuping) = 1
Nilai jarak (kupng, kauh) = 4
Hasil nilai jarak yang didapatkan kemudian diurutkan mulai dari nilai jarak yang terkecil sampai nilaijarak yang terbesar.Kata yang berada pada urutan teratas atau kata yang memiliki nilai jarak terkecilmerupakan kata yang terpilih sebagai kata yang disarankan.Sehingga kata yang terpilih sebagai katasaran pada kasus diatas adalah kata “kuping”. Jika tidak dilakukan optimasi pada kata kunci yang mengalami kesalahan ejaan maka kata kunci tersebut tidak akan mendapatkan hasil sesuai dengan yang diinginkan.

4.2.3 Pengujian

Aplikasi kamus *online* aneka bahasa ini diuji dengan membandingkan hasil *output* dari aplikasi kamus *online* aneka bahasa yang menggunakan algoritma *levenstein distence* dengan aplikasi kamus *online* aneka bahasa yang tidak menggunakan algoritma *levenstein distence*. Berikut ini adalah hasil perbandingan :

Input Kata	Ejaan Yang Benar	Hasil	
		Tanpa Levenstein Distance	Dengan Levenstein Distance
Kupng	Kuping	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Kuping
Kajeng	Ajeng	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Ajeng
Siduk	Seduk	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Seduk
Tidr	Tidur	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Tidur
Laper	Lapar	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Lapar
Seltan	Selatan	Tidak menampilkan hasil	Menampilkan hasil terjemahan dari

		terjemahan	kata Selatan
Sory	Sorry	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Sorry
Nort	North	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata North
Stomace	Stomach	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Stomach
Someil	Sommeil	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Sommeil
Affam	Affame	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Affame
Ouest	Oueste	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Ouest
Westent	Westen	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Westen
Haban	Haben	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Haben
Entschudigung	Entschuldigung	Tidak menampilkan hasil terjemahan	Menampilkan hasil terjemahan dari kata Entschuldigung

5. Simpulan

Berdasarkan uraian laporan penelitian diatas yang berjudul "Optimasi Pencarian Kata Pada Kamus Aneka Bahasa Menggunakan Algoritma *Levenshtein Distance*", penulis dapat menyimpulkan sebagai berikut:

1. Algoritma *Levenshtein* dapat melakukan optimasi pencarian kata pada aplikasi kamus aneka bahasa.
2. Sistem ini dapat membantu *user* dalam melakukan menterjemahkan kata dengan hasil terjemahaan yang optimal.

Daftar Pustaka

[1] Adiwidya, B. M . Algoritma Levenshtein Dalam Pendekatan Approximate String Matching.. Institut Teknologi Bandung, Bandung. 2009.

[2] Andhika, Fatardhi Rizky. "Penerapan String Suggestion dengan Algoritma Levenshtein Distance dan Alternatif Algoritma Lain dalam Aplikasi"..Bandung : Institut Teknologi Bandung.2010.

[3] Teguh Setiawan.Leksikologi.Jakarta: Penerbit Ombak (Anggota IKAPI).2015.

[4] Yeny Rochmawati. *Studi Perbandingan Algoritma Pencarian String dalam Metode Approximate String Matching untuk Identifikasi Kesalahan Pengetikan Teks*. Jurnal Buana Informatika. Vol.7. No.2016

[5] Zufria, Ilka. (2013). Pemodelan Berbasis UML (Unified Modeling Language) dengan Strategi Teknik Orientasi Objek User Centered Design(UCD) dalam Sistem Administrasi Pendidikan. Processor.

[6] Triandini, Evi. Suardika, I Gede. Step By Step Desain Proyek Menggunakan UML. Yogyakarta: Andi. 2012.

[7] B. P. Pratama dan S. A. Pamungkas.2016. *Analisis Kinerja Algoritma Levenshtein Distance Dalam Mendeteksi Kemiripan Dokumen Teks*. Jurnal Logika. Vol 6. No 2.