BUKU AJAR

PEMROSESAN BAHASA ALAMI

Penyusun:
Dewi Soyusiawaty

KATA PENGANTAR

Puji dan syukur dipanjatkan kepada Allah SWT yang telah memberi rahmat dan hidayahNya sehingga penyusunan buku ajar mata kuliah Pemrosesan Bahasa Alami ini akhirnya bisa diselesaikan. Buku ajar ini disusun sebagai referensi dosen dan mahasiswa untuk perkuliahan Pemrosesan Bahasa Alami di Program Studi Teknik Informatika Universitas Ahmad Dahlan.

Materi yang disajikan sudah diurutkan disesuaikan dengan perencanaan pembelajaran mingguan mata kuliah, sehingga harapannya mahasiswa dapat memiliki referensi belajar untuk tiap materi yang disampaikan.

Penulis menyadari masih banyak ketidaksempurnaan pada penulisan, baik isi maupun redaksi, oleh karenanya kritik dan saran yang membangun diharapkan dapat memperbaiki untuk tahun berikutnya.

Terima kasih kepada semua pihak yang telah membantu baik secara langsung ataupun tidak terhadap terselesaikannya buku ajar ini. Semoga dapat bermanfaat bagi yang membutuhkan.

Yogyakarta, Maret 2023

Penyusun

DAFTAR ISI

KATA PENGANTAR.....2

DAFTAR ISI.....3

BAB 1 PENGANTAR PEMROSESAN BAHASA ALAMI.....4

BAB 2 TEXT PREPROCESSING.....17

BAB 3 STEMMING.....28

BAB 4 N-GRAMS.....53

BAB 5 PEMBOBOTAN KATA.....64

BAB 6 SPELL CHECK.....69

BAB 7 PARSING.....87

BAB 8 KEAMBIGUAN.....95

DAFTAR PUSTAKA.....99

BAB 1 PENGANTAR PEMROSESAN BAHASA ALAMI

A. *Artificial Intelligence* (AI

Artificial Intelligence (AI) adalah bagian dari ilmu komputer yang berhubungan dengan usaha untuk merancang sistem komputer yang memiliki kecerdasan, sistem yang memiliki karakteristik seperti manusia. Kecerdasan buatan (AI) merupakan sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia (Rich & Knight, 1991). Dalam merepresentasikan kecerdasan buatan pengetahuan lebih banyak meggunakan bentuk simbol-simbol daripada bilangan, dan memproses informasi berdasarkan metode heuristic atau dengan berdasarkan sejumlah aturan.

Objektif dari kecerdasan buatan yaitu:

1. Membuat mesin menjadi lebih cerdas (tujuan utama)

   Kecerdasan buatan akan membuat mesin menjadi cerdas yaitu mampu berbuat seperti apa yang dilakukan oleh manusia

2. Memahami apa yang dimaksud dengan “intelligence”.

   Intelligence atau kecerdasan adalah kapasistas untuk berpikir logis, berkomunikasi, belajar, mengetahui, memahami, mengingat, dan mampu memecahkan masalah.

   Ciri-ciri intelligence dapat dilihat sebagai beikut:

   a. Belajar atau memahami dari pengelaman
   b. Memahami makna ganda/ambiguous/pesan salah
   c. Cepat anggap pada situasi baru
   d. Menggunakan alasan untuk menyelesaikan suatu masalah
   e. Menarik kesimpulan logis dari sejumlah fakta besar

f. Dapat memperbaharui pengetahuan

g. Dapat mengingat yang pernah dialami

3. Membuat mesin menjadi lebih berdaya guna

B. Disiplin ilmu *Artificial Intelligence* (AI)

Seperti yang telah disebutkan diatas bahawa *Artificial Intelligence* (AI) merupakan salah satu cabang ilmu komputer. Tetapi karena kompleksitas area *Artificial Intelligence* (AI) maka terdapat sub-sub bagian yang dapat berdiri sendiri dan dapat saling bekerja sama dengan sub bagian lain atau dengan disiplin ilmu lain. Berikut ini beberapa cabang ilmu sub bagain dari *Artificial Intelligence* (AI) :

1. *Expert System*

   Expert System atau sistem pakar merupakan salah satu cabang *Artificial Intelligence* (AI) yang memperlajari pembuatan sebuah sistem yang dapat bekerja layaknya seorang pakar. Sistem pakar dapat menyimpan pengetahuan seorang pakar dan memberikan solusi berdasarkan pengetahuan yang dimilikinya tadi. Sistem pakar juga merupakan salah satu cabang *Artificial Intelligence* (AI) yan sering melakukan kerja sama dengan disiplin ilmu lain karena sifatnya yang dapat menyimpan pengetahuan.

2. *Speech Understanding*

   Speech Understanding merupakan salah satu cabang dari kecerdasan buatan dengan menerapkan fitur pengenalan ucapan. Sebuah sistem komputer dapat menerjemahkan perkataan yang diucapakan oleh manusia menjadi perintah program. Proses komunikasi dilakukan dengan suara tanpa harus mengetik pada *keyoard.*

3. *Robotics and Sensory System*

   Robotics and Sensory System merupakan salah satu cabang kecerdasan buatan yang menggabungkan cabang-cabang kecedasan buatan yang lain untuk membentuk sebuah sistem robotik. Alat atau mesin yang diciptakan dapat diarahkan untuk mengerjakan bermacam tugas tanpa campur tangan manusia.

4. *Computer Vision and Scane Recognition*

   Computer Vision merupakan kemampuan komputer untuk dapat menangkap sinyal yang dikeluarkan oleh kamera dan dapat memahami apa yang dilihat tersebut.

5. *Game Playing*

   Game antara manusia dan komputer, dimana komputer telah dilengkapi data sehingga dapat menganalisa, nalar dan berkomunikasi, membuat keputusan.

6. *Pattern Recognition*
7. *Neural Computing*
8. *News Summarization*
9. *Language Translation*
10. *Fuzzy Logic*
11. *Genetic Algorithm*
12. *Intelligence Software Agents*
13. *Natural Language Processing*

C. Pemrosesan Bahasa Alami atau *Natural Language Processing* (NLP)

Bahasa dapat direpresentasikan dari suatu pesan yang ingin dikomunikasikan antar manusia berupa suara atau ucapan (*spoken languages*) atau tulisan. Bahasa alami atau

bahasa natural adalah suatu bahasa yang diucapkan, ditulis, atau diisyaratkan (secara visual atau isyarat lain) oleh manusia untuk komunikasi umum. Perbedaan bahasa alami dan bahasa buatan adalah sebagai berikut :

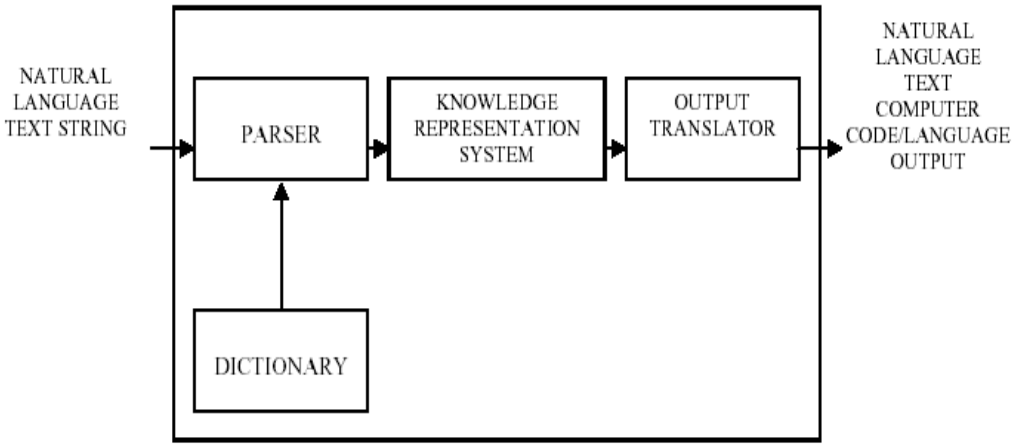
Bahasa Alami	Bahasa Buatan
1. Spontan 2. Bersifat kebiasaan 3. Intuitif 4. Pernyataan secara langsung	1. Berdasarkan pemikiran 2. Sekehendak hati 3. Diskursif (logis, luas, penuh makna) 4. Pernyataan tak langsung

Natural Language Processing (NLP) atau pemrosesan bahasa alami merupakan salah satu cabang *Artificial Intelligence* (AI) yang mempelajari pembuatan sistem untuk menerima masukan bahasa alami manusia. Dalam perkembangannya, NLP berusaha untuk mengubah bahasa alami komputer (*bit* dan *byte*) menjadi bahasa alami manusia yang dapat kita mengerti. NLP merupakan ilmu dasar yang dapat dijadikan jembatan untuk membuat komunikasi antara mesin dengan manusia dengan memproses bahasa, baik lisan atau tulisan yang digunakan oleh manusia dalam komunikasi sehari-hari.

Sistem pemrosesan bahasa alami harus memperhatikan pengetahuan terhadap bahasa itu sendiri, baik dari segi kata yang digunakan, bagaimana kata-kata tersebut digabungkan untuk menghasilkan suatu kalimat, apa arti sebuah kata, apa fungsi sebuah kata dalam sebuah kalimat dan sebagainya. Bahasa buatan dibuat untuk memnuhi kebutuhan tertentu dan dirancang dengan hati-hati agar mematuhi aturan-aturan yang diperlukan untuk kemudahan pemrosesannya. Bahasa alami tumbuh secara alami untuk memnuhi kebutuhan-kebutuhan komunikasi antar manusia. Bahasa alami tidak dirancang

dengan memperhatikan berbagai kendala untuk kemudahan pemrosesan. Sehingga pemrosesan bahasa alami jauh lebih sulit dibandingkan dengan bahasa buatan.

D. Komponen *Natural Language Processing* (NLP)



1. *Dictionary*/Kamus/Leksikon

   Kamus adalah sebuah rujukan yang menerangkan makna kata-kata. Kamus mendaftarkan kata berdasarkan abjad dan informasi tentang bagaimana mereka dipakai. Setiap *entry* dalam kamus merepresentasikan informasi tentang kata benda hidup, kata benda mati, kata sifat, kata kerja, kata keterangan, kata depan, kata sandang, kata petunjuk dan kata ganti yang diketahui untuk sistem pengolahan bahasa alami.

2. Parser

   Parser merupakan elemen yang paling menentukan dalam *Natural Language Processing* (NLP). Parser adalah sepenggal *software* yang dapat menganalisis *input* kalimat secara sintaktik. Parser melakukan idenifikasi tiap-tiap kata dan kemudian membuat peta kata-kata tersebut dalam struktur yang disebut pohon parser.

Pohon parser menunjukkan makna dalam semua kata dan bagaimana cara menggabungkan kata-kata tersebut. parser juga dapat mengidentifikasi prasa kata kerja, prasa kata benda dan selanjutnya memilah-milah kedalam elemen-lemen yang lain.

3. *Knowledge Representation System*/Sistem Representasi Pengetahuan

   Tahap ini digunakan untuk menganalisis *output* parser guna menentukan maknanya. Didalam sistem berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan yang lainnya.

4. *Output Translator*

   Suatu terjemahan yang merepresentasikan sistem pengetahuan dan melakukan langkah-langkah yang bisa berupa jawaban atas bahasa alami atau *output* khusus yang sesuai dengan program komputer lainnya.

E. *Layers of Natural Language Processing* (NLP)

1. *Phonetics & Phonology*

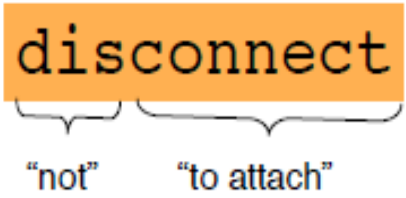
   Phonetics adalah ilmu yang membahas bagaimana suatu suara bisa terbentuk menggunakan beberapa bagian tubuh yaitu bibir, gigi, lidah, faring dan paru-paru. Ilmu ini hanya membahas mengenai hubungan antar bagian tubuh yang menghaslkan suara. Sedangkan *Phonology* merupakan ilmu tentang suatu sistem dala sebua bahasa, ilmu ini salah satu cabang linguistik ang berhubungan dengan cara mengucapkan suatu bahasa. Contonya :

dis-k&-'nekt

disconnect

2. *Morphology*

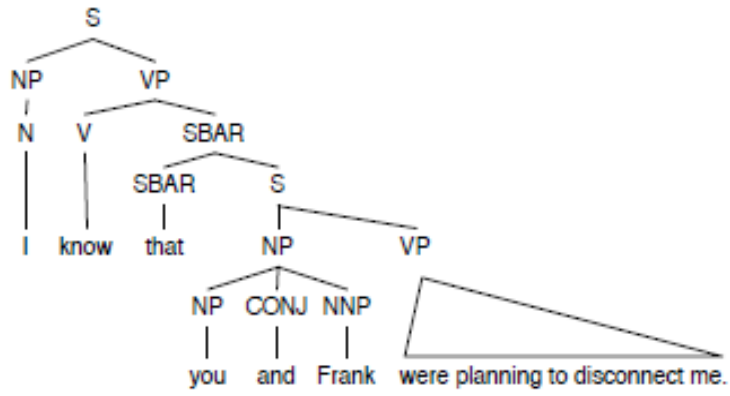
   Morphology atau ilmu bentuk kata adalah cabang linguistik yang mengidentifikasi satuan-satuan dasar bahasa sebagai satuan gramatikal. Contohnya :



3. *Syntax*

   Syntax adalah salah satu dari cabang ilmu linguistik yang mempelajari kaidah yang menentukan bagaimana kata membentuk frasa dan frasa membentuk kalimat. Contohnya :

I know that you and Frank were planning to disconnect me.



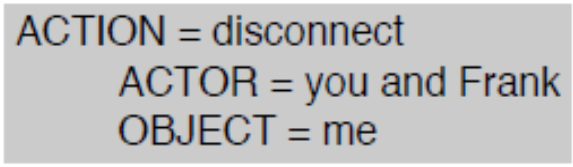
Not same structure:

You know me--Frank and I were planning to disconnect that.

4. *Semantics*

Semantics adalah cabang dari linguistik yang mempelajari arti atau makna yang terkandung pada suatu bahasa, kode, atau jenis representasi lain. Contohnya :

```
I know that you and Frank were planning to disconnect me.
```



5. *Pragmatics*

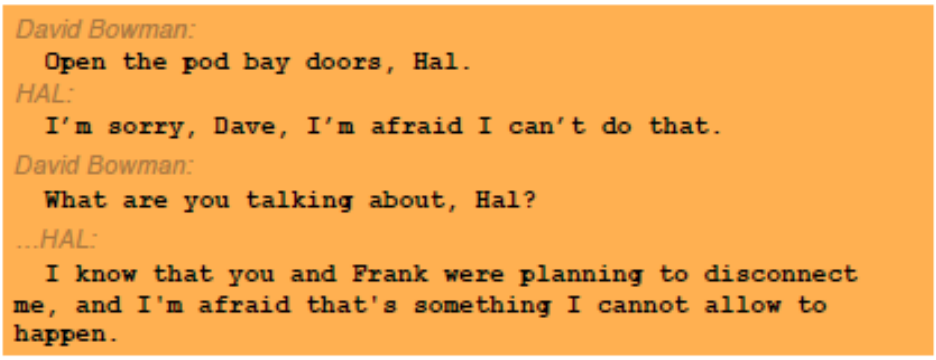
Pragmatics mempelajari hubungan antara konteks luar bahasa dan maksud tuturan. Konteks luar bahasa ialah unsur di luar tuturan yang mempengaruhi maksud tuturan. Dengan *pragmatics* dapat dikatahui tata cara penggunaan bahasa dan pengaruhnya pada pendengar. *Pragmatics* dapat menunjuk kepada apa sebenarnya arti yang diucapkan dan ditulis oleh seseorang. Sebagai contoh pertanyaan "Jam berapa sekarang?", pertanyaan ini sebenarnya tidak jelas, mempunyai makna ganda. Dengan pertanyaan ini belum tentu si penanya ingin mengetahui jam sebenarnya. Tetapi mungkin saja karena Ia merasa sudah terlambat masuk kerja. Contoh lainnya :

```
I'm sorry Dave, I'm afraid I can't do that.
```

6. *Discourse*

Discourse Knowledge melakukan pengenalan apakah suatu kalimat yang sudah dibaca dan dikenali sebelumnya akan mempengaruhi arti dari kalimat selanjutnya.

Informasi ini penting diketahui untuk melakukan pengolahan arti terhadap kata ganti orang dan untuk engartikan aspek sementara dari informasi. Contohnya :



F. Kesulitan dalam *Natural Language Processing* (NLP)

1. Ambiguity

   Ambiguity artinya mempunyai makna lebih dari satu. Keambiguan ini dapat menimbulkan keraguan atau ketidakjelasan dalam kalimat yang diucapkan atau ditulis. Contohnya kata "bisa" yang bisa jadi memiliki arti "racun" atau "dapat". Hal lain yang dapat menjadi ambigu yaitu simbol huruf dan tanda baca (simbol titik tidak selalu berfungsi sebagai akhir kaliat, tetapi dapat menjadi bagian dari singkatan). Contohnya "Ir., Dr., S.T, dan masih banyak lagi"

2. Jumlah kosa kata (*vocabulary*) dalam bahasa alami sangat besar

3. *Text segmentation*

   Sebagai contoh beberapa bahasa tulisan seperti Chinese, Japanese, dan Thai tidak memiliki baasan antara satu kata dengan yang lain.

4. *Word sense disambiguation*

   Yaitu kata-kata yang mempunyai lebih dari satu arti

5. *Syntactic ambigulity*

Yaitu terdapat banyaknya kemungkinan pohon parsing untuk satu kalimat

G. Teknologi *Natural Language Processing* (NLP)

1. Sub Sistem *Natural Language Processing* (NLP)

   Yaitu pemrosesan secara simbolik terhadapat bahasa tulisan

2. Sub sistem TTS (Text to Speech)

   Yaitu pemrosesan text (bahasa tulisan) menjadi ucapan (bahasa lisan)

3. Sub sistem SR (Speech Recognition)

   Yaitu pemrosesan ucapan (bahasa lisan) menjadi text (bahasa tulisan)

H. Kategori Aplikasi *Natural Language Processing* (NLP)

1. Natural Language Translator

   Translator bahasa alami bukan hanya kamus yang menerjemahkan kata per kata, tetapi harus juga mentranslasikan sintaks dari bahasa asal ke bahasa tujuannya.

2. Translator bahasa alami ke bahasa buatan

   Translator yang mengubah perintah-perintah dalam bahasa alami menjadi bahasa buatan yang dapat dieksekusi oleh mesin atau komputer.

3. Text Summarization

   Sistem yang dapat “membuat ringkasan” hal-hal yang penting dari suatu wacana yang diberikan.

I. Aplikasi Teknologi Bahasa

1. Alat bantu baca tunanetra

Alat bantu membaca bagi tunanetra mempunyai masukan berupa teks tercetak (misalnya buku) dan mempunyai keluaran berupa ucapan dari teks tercetak yang diberikan. Pada prinsipnya ada dua komponen utamanya, yaitu bagian "pengenal karakter" yang menggunakan teknologi OCR *(Optical Character Recognition),* serta bagian TTS. Dengan alat bantu ini, orang tunanetra dapat membaca suatu buku atau dokumen. Bahkan, jika teks yang ingin dibacakan sudah tersedia di dalam komputer, dengan teknologi Text to Speech dapat langsung diucapkan.

2. Alat bantu bicara tunawicara

   Alat bantu membaca bagi tunawicara mempunyai masukan posisi tangan yang dideteksi oleh suatu sensor dan unit identifikasi. Rangkaian huruf yang diidentifikasikan akan disusun membentuk suatu kata yang pada akhirnya akan diumpankan pada bagian TTS.

3. Online translator

   Online translator yang dimaksud disini adalah translator yang secara otomatis dapat menerjemahkan kalimat lisan dari suatu bahasa alami (misalnya Bahasa Inggris) menjadi ucapan hasil terjemahannya dalam bahasa alami lainnya (misalnya Bahasa Indonesia). Online translator terdiri dari 3 bagian. Bagian pertama, *speech recognition,* berfungsi untuk mengenali rangkaian kata dari bahasa sumber menjadi teks dalam bahasa sumber. Bagian berikutnya adalah translator teks ke teks. Hasil bagian kedua ini adalah kalimat bahasa tujuan yang masih berupa teks. Bagian ketiga berupa sistem TTS dalam bahasa tujuan. Aplikasi seperti ini mungkin untuk dikembangkan, karena teknologi *speech recognition* sudah banyak dikembangkan. Translator bahasa pun sudah banyak dikembangkan, termasuk translator Bahasa Inggris ke Indonesia.

4. *Talking email*

   TTS juga memungkinkan diintegrasikan dengan berbagai program aplikasi, seperti email, web browser, aplikasi-aplikasi multimedia atau aplikasi-aplikasi lainnya.

5. Aplikasi telephony

   TTS dapat digunakan pada aplikasi telephony, seperti sistem informasi billing atau sistem informasi lainnya yang diucapkan secara lisan. TTS juga dapat digunakan untuk konversi dari SMS *(Short Message System)* ke ucapan sehingga pesan SMS dapat didengar. Dengan demikian memungkinkan untuk mendengar pesan SMS sambil melakukan aktivitas yang menyulitkan untuk membacanya, seperti sedang mengendarai mobil. Dengan TTS tersebut, memungkinkan pula untuk meneruskan pesan SMS ke sistem telepon biasa (PSTN). Speech Recognition memungkinkan pencarian informasi secara lisan.

J. Aplikasi *Natural Language Processing* (NLP)

1. Small
   a. Spelling correction; Grammar Checking
2. Medium
   a. Word sense disambiguation
   b. Named entity recognition
   c. Information retrieval from books, news, papers , web pages etc
   d. Better search engine
3. Large
   a. Information Extraction
   b. Question Answering
   c. Machine Translation

d. Dialogue System (automated customer services)

e. Reading and interpretting a textbook

f. Speech Recognition

g. Text to speech

h. Automatic Summarization

i. Foreign Language Reading Aid

j. Foreign Language Writing Aid

BAB 2 TEXT PREPROCESSING

A. Karakteristik Dokumen Texts

Karakteristik adalah kualitas tertentu atau ciri yang khas dari sesuatu. Berikut karakteristik dokumen text menurut Loretta Auvil dan Duane Searsmith :

1. *Database text* berukuran besar
2. Memliki dimensi yang tinggi, artinya satu kata merupakan satu dimensi
3. Mngandung kumpulan kata yang saling terkait (*frase*) dan antara kumpulan kata satu dengan lain dapat memiliki arti yang berbeda
4. Banyak mengandung kata ataupun arti yang bias (*ambiguity*)
5. Dokumen email merupakan dokumen yang tidak memiliki struktir bahasa yang baku, karena didalamnya terkadang muncul istilah slank. Conoth istilah slank yaitu "r u there?", "hellllooo bosss, whatzzzzzzz up?", dan masih banyak lagi.

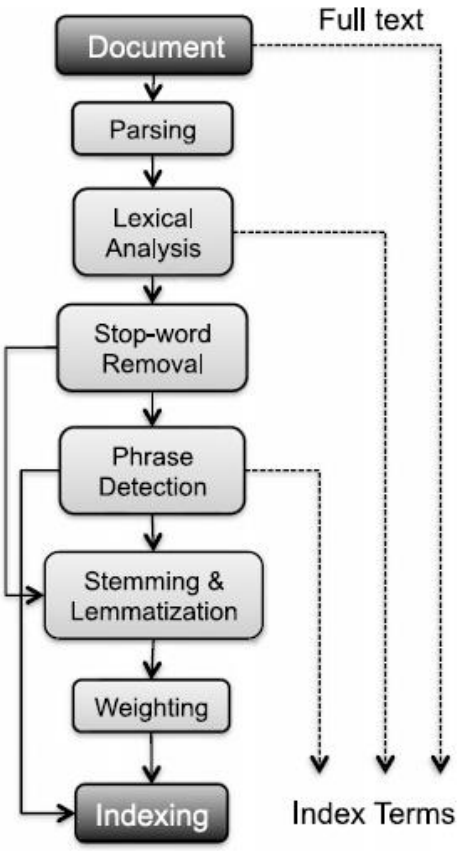
B. Pemrosesan Teks (*Text Preprocessing*)

Text atau dokumen pada umumnya memiliki struktur kalimat yang tidak baik dan banyak *noise* sehingga informasi di dalamnya tidak bisa diekstrak secara langsung. Tidak semua kata mencerminkan makna/isi yang terkandung dalam sebuah dokumen. Untuk mendapatkan ekstraksi fitur pada *text* maka harus dilakukan *text preprocessing*. *Preprocessing* diperlukan untuk memilih kata yang akan digunakan sebagai indeks. Indeks adalah kata-kata yang mewakili dokumen yang nantinya digunakan untuk membuat pemodelan untuk *Information Retrieval* maupun aplikasi *teks mining* lain.

Pemrosesan Teks (*Text Preprocessing*) adalah suatu proses pengubahan bentuk data yang belum terstruktur menjadi data yang terstruktur sesuai dengan kebutuhan untuk

proses mining yang lebih lanjut (sentiment analysis, peringkasan, clustering dokumen, etc.). Singkatnya *Preprocessing* adalah merubah teks menjadi term index. Tujuannya untuk menghasilkan sebuah set term index yang bisa mewakili dokumen.

Langkah-langkah yang dapat dilakukan pada text pre-processing yaitu :



1. Langkah 1 : Parsing

   Tulisan dalam sebuah dokumen bisa jadi terdiri dari berbagai macam bahasa, character sets, dan format. Sering juga, dalam satu dokumen yang sama berisi tulisan dari beberapa bahasa. Misal, sebuah email berbahasa Indonesia dengan lampiran PDF berbahasa Inggris.

Parsing Dokumen berurusan dengan pengenalan dan "pemecahan" struktur dokumen menjadi komponen-komponen terpisah. Pada langkah preprocessing ini, kita menentukan mana yang dijadikan satu unit dokumen. Contohnya :

a. Email dengan 4 lampiran bisa dipisah menjadi 5 dokumen : 1 dokumen yang merepresentasikan isi (body) dari email dan 4 dokumen dari masing-masing lampiran.
b. Buku dengan 100 halaman bisa dipisah menjadi 100 dokumen; masing-masing halaman menjadi 1 dokumen.
c. Satu tweet bisa dijadikan sebagai 1 dokumen. Begitu juga dengan sebuah komentar pada forum atau review produk.

2. Langkah 2 : *Lexical Analysis*

Lexical analysis lebih populer disebut *Lexing* atu Tokenization/Tokenisasi. Tokenisasi adalah proses pemotongan string input berdasarkan tiap kata penyusunnya. Pada prinsipnya proses ini adalah memisahkan setiap kata yang menyusun suatu dokumen. Pada proses ini dilakukan penghilangan angka, tanda baca dan karakter selain huruf alfabet, karena karakter-karakter tersebut dianggap sebagai pemisah kata (delimiter) dan tidak memiliki pengaruh terhadap pemrosesan teks.

Pada tahapan ini juga dilakukan beberapa proses seperti *case folding*, dimana semua huruf diubah menjadi huruf kecil. Selain *case folding* terdapat juga proses *Cleaning*, dimana *Cleaning* adalah proses membersihkan dokumen dari komponen-komponen yang tidak memiliki hubungan dengan informasi yang ada pada dokumen, seperti tag html, link, dan script, dsb.

Token adalah kata-kata yang dipisah-pisah dari teks aslinya tanpa mempertimbangkan adanya duplikasi. Contohnya :

Text	Hasil Token
"apakah culo dan boyo bermain bola di depan rumah boyo?"	"culo", "dan", "boyo", "bermain", "bola", "di", "depan", "rumah", "boyo"

Type adalah token yang memperhatikan adanya duplikasi kata. Ketika ada duplikasi hanya dituliskan sekali saja. Contohnya :

Text	Hasil Type
"apakah culo dan boyo bermain bola di depan rumah boyo?"	"culo", "dan", "boyo", "bermain", "bola", "di", "depan", "rumah"

Term adalah type yang sudah dinormalisasi (dilakukan stemming, filtering, dsb). Contohnya :

Text	Hasil Term
"apakah culo dan boyo bermain bola di depan rumah boyo?"	"culo", "boyo", "main", "bola", "depan", "rumah"

Berikut contoh tokenisasi text bahasa inggris dan bahasa indonesia :

Text English	They are applied to the words in the texts.
Tokens	they
	are

	applied
	to
	the
	words
	in
	the
	texts

Text Bahasa	Namanya adalah Santiago. Santiago sudah memutuskan untuk mencari sang alkemis
Tokens	namanya
	adalah
	santiago
	santiago
	sudah
	memutuskan
	untuk
	mencari
	sang
	alkemis

3. Langkah 3 : *Stopword Removal*

Stopword Removal disebut juga Filtering. Filtering adalah tahap pemilihan kata-kata penting dari hasil token, yaitu kata-kata apa saja yang akan digunakan

untuk mewakili dokumen. Algoritma yang digunakan pada Filtering yaitu stoplist dan wordlist.

Stoplist atau stopword adalah kata-kata yang tidak deskriptif (tidak *penting*) yang dapat dibuang dengan pendekatan *bag-of-words*. Kita memiliki database kumpulan kata-kata yang tidak deskriptif (tidak *penting*), kemudian kalau hasil tokenisasi itu ada yang merupakan kata tidak penting dalam database tersebut, maka hasil tokenisasi itu dibuang. Contohnya adalah i'm, you, one, two, they, are, to, the, in, dst.

Hasil Token	Hasil Filtering
they	-
are	-
applied	applied
to	-
the	-
words	words
in	-
the	-
texts	text

Contoh lainnya yaitu untuk, sang, sudah, adalah, dst.

Hasil Token	Hasil Filtering
namanya	namanya
adalah	-

santiago	santiago
santiago	santiago
sudah	-
memutuskan	memutuskan
untuk	-
mencari	mencari
sang	-
alkemis	alkemis

Wordlist adalah kata-kata yang deskriptif (*penting*) yang harus disimpan dan tidak dibuang dengan pendekatan *bag-of-words*. Kita memiliki database kumpulan kata-kata yang deskriptif (*penting*), kemudian kalau hasil tokenisasi itu ada yang merupakan kata penting dalam database tersebut, maka hasil tokenisasi itu disimpan. Contoh wordlist adalah applied, words, texts, dst.

Hasil Token	Hasil Filtering
they	-
are	-
applied	applied
to	-
the	-
words	words
in	-
the	-

texts	text

Contoh wordlist adalah santiago, namanya, mencari, memutuskan, alkemis, dst.

Hasil Token	Hasil Filtering
namanya	namanya
adalah	-
santiago	santiago
santiago	santiago
sudah	-
memutuskan	memutuskan
untuk	-
mencari	mencari
sang	-
alkemis	alkemis

Kebanyakan aplikasi text mining ataupun IR bisa ditingkatkan performanya dengan penghilangan stopword. Akan tetapi, secara umum Web search engines seperti *google* sebenarnya tidak menghilangkan stop word, karena algoritma yang mereka gunakan berhasil memanfaatkan stopword dengan baik.

4. Langkah 4 : *Phrase Detection*

Langkah ini bisa menangkap informasi dalam teks melebihi kemampuan dari metode tokenisasi / bag-of-word murni. Pada langkah ini tidak hanya dilakukan tokenisasi per kata, namun juga mendeteksi adanya 2 kata atau lebih yang menjadi frase. Contoh, dari dokumen ini : *"search engines are the most visible*

information retrieval applications" . Terdapat dua buah frase, yaitu *"search engines"* dan *"information retrieval".*

Phrase detection bisa dilakukan dengan beberapa cara : menggunakan rule/aturan (misal dengan menganggap dua kata yang sering muncul berurutan sebagai frase), bisa dengan *syntactic analysis*, and kombinasi keduanya. Metode umum yang diguakan adalah penggunaan thesauri untuk mendeteksi adanya frase. Contoh : Pada thesauri tersebut terdapat daftar frase-fase dalam bahasa tertentu, kemudia kita bandingkan kata-kata dalam teks apakah mengandung frase-frase dalam thesauri tersebut atau tidak.

Kelemahanya, tahap ini butuh komputasi yang cukup lama. Kebanyakan aplikasi teks mining atau IR tidak menggunakan *Phrase Detection*, sudah cukup dengan Token per kata. Akan tetapi, sebenarnya pemanfaatan *Phrase* akan meningkatkan akurasi

5. Langkah 5 : *Stemming*

Stemming adalah proses pengubahan bentuk kata menjadi kata dasar atau tahap mencari root kata dari tiap kata hasil filtering. Dengan dilakukanya proses stemming setiap kata berimbuhan akan berubah menjadi kata dasar, dengan demikian dapat lebih mengoptimalkan proses teks mining.

Hasil Token	Hasil Filtering	Hasil Stemming
they	-	-
are	-	-
applied	applied	apply
to	-	-

the	-	-
words	words	word
in	-	-
the	-	-
texts	text	text

Hasil Token	Hasil Filtering	Hasil Stemming
namanya	namanya	nama
adalah	-	-
santiago	santiago	santiago
santiago	santiago	santiago
sudah	-	-
memutuskan	memutuskan	putus
untuk	-	-
mencari	mencari	cari
sang	-	-
alkemis	alkemis	alkemis

Hasil Token	Hasil Filtering	Hasil Stemming	Type	Term
they	-	-	-	-
are	-	-	-	-
applied	applied	apply	apply	apply
to	-	-	-	-

the	-	-	-	-
words	words	word	word	word
in	-	-	-	-
the	-	-	-	-
texts	text	text	text	text

Hasil Token	Hasil Filtering	Hasil Stemming	Type	Term
namanya	namanya	nama	nama	nama
adalah	-	-	-	-
santiago	santiago	santiago	santiago	santiago
santiago	santiago	santiago	-	-
sudah	-	-	-	-
memutuskan	memutuskan	putus	putus	putus
untuk	-	-	-	-
mencari	mencari	cari	cari	cari
sang	-	-	-	-
alkemis	alkemis	alkemis	alkemis	alkemis

BAB 3 STEMMING

Pengolahan bahasa alami banyak dilibatkan dalam suatu bidang Sistem Temu Kembali Informasi. Kehandalan suatu sistem pencarian dapat dilihat dari nilai efektifitas yang dihasilkan. Nilai efektifitas model sistem temu kembali informasi (*information retrieval system*) menunjukkan tingkat keberhasilan performansi suatu mesin pencari. Banyak faktor yang berpengaruh terhadap diperolehnya nilai efektifitas tersebut. Salah satunya pengaruh proses *stemming* (pencarian kata dengan menghilangkan *prefiks, infix* dan *suffix*). Contoh bahasa Indonesia yaitu :

1. Main
2. Bermain
3. Mainan
4. Permainan
5. Dipermainkan
6. Dimainkan
7. Memainkan
8. mempermainkan

Contoh bahasa inggris yaitu :

1. *Connect*
2. *Connecting*
3. *Connected*
4. *InConnected*
5. *Connectivity*
6. *Connection*

7. *Disconnect*

Preprocessing diperlukan untuk memilih kata yang akan digunakan sebagai indeks. Indeks ini adalah kata-kata yang mewakili dokumen yang nantinya digunakan untuk membuat pemodelan untuk NLP, Information Retrieval, maupun aplikasi teks mining lain. Langkah-langkah pada Preprocessing yaitu :

1. Langkah 1 : Parsing
2. Langkah 2 : *Lexical Analysis/* Tokenization/Tokenisasi
3. Langkah 3 : *Stopword Removal*
4. Langkah 4 : *Phrase Detection*
5. Langkah 5 : *Stemming*

Implementasi proses stemming sangat beragam , tergantung dengan bahasa dari dokumen. Beberapa metode untuk Stemming :

a. Porter Stemmer (English & Indonesia)
b. Stemming Arifin-Setiono (Indonesia)
c. Stemming Nazief-Adriani (Indonesia)
d. Khoja (Arabic)
e. Algoritma Idris dan Mustofa
f. Algoritma Vega
g. Algoritma Ahmad, Yussof dan Sembok

A. Metode Stemming

1. Metode Algorithmic

   Metode Algorithmic dilakukan dengan membuat sebuah algoritma yang mendeteksi imbuhan. Jika ada awalan atau akhiran yang seperti imbuhan, maka

akan dibuang. Kelebihan metode algorithmic yaitu relatif cepat. Sedangkan kekurangannya beberapa algoritma terkadang salah mendeteksi imbuhan, sehingga ada beberapa kata yang bukan imbuhan tapi dihilangkan. Contohnya :

makan -> mak; an dideteksi sebagai akhiran sehingga dibuang

2. Metode Lemmatization

Metode Lemmatization proses Stemming yang dilakukan berdasarkan kamus, menggunakan *vocabulary* dan *morphological analysis* dari kata untuk menghilangkan imbuhan dan dikembalikan ke bentuk dasar dari kata. Stemming ini bagus untuk kata-kata yang mengalami perubahan tidak beraturan (terutama dalam english). Contohnya :

“see” -> “see”, “saw”, atau “seen”
Jika ada kata “see”, “saw”, atau “seen”, bisa dikembalikan ke bentuk aslinya yaitu “see”

B. Algoritma Porter Stemming

Algoritma Porter Stemming merupakan algoritma yang paling populer. Ditemukan oleh Martin Porter pada tahun 1980. Mekanisme algoritma tersebut dalam mencari kata dasar suatu kata berimbuhan, yaitu dengan membuang imbuhan–imbuhan (atau lebih tepatnya akhiran pada kata–kata bahasa Inggris karena dalam bahasa Inggris tidak mengenal awalan). Contohnya :

Porter' s algorithm

Rule			Example		
SSES	→	SS	caresses	→	caress
IES	→	I	ponies	→	poni
SS	→	SS	caress	→	caress
S	→		cats	→	cat

C. Kesalahan Stemming

Kesalahan yang terjadi pada proses stemming yaitu Overstemming, Understemming, Unchange, dan Spelling exception. Berikut penjelasannya :

1. Overstemming

   Overstemming dapat diartikan sebagai pemenggalan yang melebihi seharusnya. Contohnya :

   masalah menjadi masa

   Kesalahan timbul karena bentuk kata yang menyerupai imbuhan.

2. Understemming

   Understemming dapat diartikan sebagai pemenggalan yang terlalu sedikit. Contohnya :

   belajar menjadi lajar

   Kesalahan timbul dikarenakan kekurangan pada aturan imbuhan yang didefinisikan

3. Unchange

   Kesalahan unchange artinya tidak terjadi pemenggalan. Contohnya :

   telapak

4. Spelling exception

   Spelling exception artinya huruf pertama kata dasar yang didapat tidak benar akibat pemenggalan awalan. Contohnya :

   memukul menjadi ukul

   Contoh lainnya yaitu beberapa imbuhan berubah setelah ditempel (ber + ajar, pen + lihat, pen + sakit). Ada imbuhan yang huruf pertama kata dasarnya luluh (meng/peng ditambah karang menjadi mengarang, men + tuai menjadi menuai).

D. Algoritma Stemming untuk Bahasa Indonesia

Algoritma *stemming* untuk bahasa yang satu berbeda dengan algoritma stemming untuk bahasa lainnya. Bahasa Inggris memiliki morfologi yang berbeda dengan bahasa Indonesia sehingga algoritma *stemming* untuk kedua bahasa tersebut juga berbeda. Proses *stemming* pada teks berbahasa Indonesia lebih rumit/kompleks karena terdapat variasi imbuhan yang harus dibuang untuk mendapatkan *root word* (kata dasar) dari sebuah kata.

Beberapa Contoh Algoritma Yang Digunakan Untuk Melakukan Proses Stemming Pada Bahasa Indonesia Dan Bahasa Inggris :

a. Algoritma Nazief dan Andriani, algoritma ini mengacu pada aturan morfologi bahasa Indonesia yang mengelompokkan imbuhan, yaitu imbuhan yang diperbolehkan atau imbuhan yang tidak diperbolehkan. Pengelompokan ini termasuk imbuhan di depan (awalan), imbuhan kata di belakang (akhiran), imbuhan kata di tengah (sisipan) dan kombinasi imbuhan pada awal dan akhir kata (konfiks). Algoritma ini menggunakan kamus kata keterangan yang digunakan untuk mengetahui bahwa proses stemming telah mendapatkan kata dasar.
b. Algoritma Arifin dan Setiono, Arifin and Setiono mengajukan skema algoritma yang

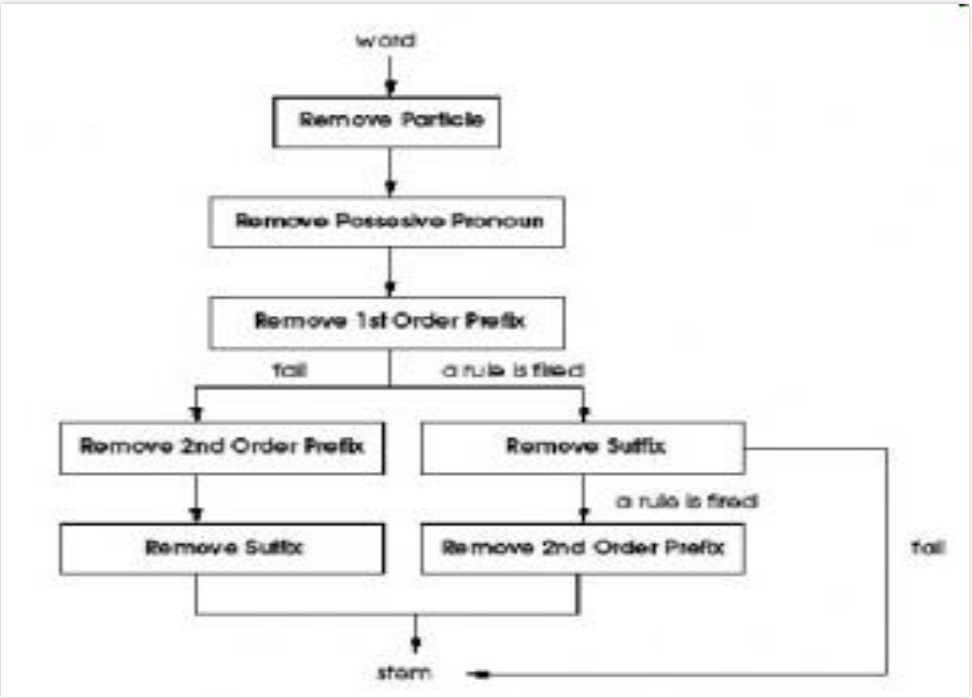
sedikit lebih sederhana dibandingkan algoritma Nazief and Adriani tetapi yang menjadi kesamaan diantara kedua algoritma tersebut adalah kedua algoritma tersebut menggunakan kamus dan sama-sama menyediakan fungsi recoding.

c. Algoritma Vega, algoritma Vega tidak menggunakan kamus. Urutan penghilangan imbuhan yang melekat pada kata untuk algoritma Vega ini, sama seperti urutan penghilangan imbuhan yang diterapkan pada algoritma Arifin dan Setiono.
d. Algoritma Porter, algoritma Porter ditemukan oleh Martin Porter 1980. Algoritma ini terkenal digunakan sebagai stemmer untuk bahasa Inggris, kemudian karena proses stemming bahasa inggris berbeda dengan bahasa indonesia maka, dikembangkan algoritma porter khusus untuk bahasa indonesia (Porter Stemmer for Bahasa Indonesia) oleh W.B. Frakes pada tahun 1992.
e. Algoritma Confix Stripping (CS), Confix stripping (CS) stemmer adalah metode stemming pada Bahasa Indonesia yang diperkenalkan oleh Jelita Asian yang merupakan pengembangan dari metode stemming yang dibuat oleh Nazief dan Adriani (1996).
f. Algoritma Enhanded Confix Stripping (ECS), algoritma ini perupakan pengembangan dari algoritma Confix stripping (CS). Dalam algoritma Confix stripping (CS) terdapat kelemahan atau tidak dapat mengstemming kata-kata tertentu, pada algoritma ini Menambahkan suatu algoritma tambahan untuk mengatasi kesalahan pemenggalan akhiran yang seharusnya tidak dilakukan.

Pada umumnya kata dasar pada bahasa Indonesia terdiri dari kombinasi :

Prefiks 1 + *Prefiks 2* + *Kata dasar* + *Sufiks 3* + *Sufiks 2* + *Sufiks 1*

Berikut Algoritma Porter Stemming untuk bahasa indonesia :



Langkah-langkah yang harus dilakukan pada algoritma Porter Stemming untuk bahasa Indonesia yaitu :

1. Hapus Particle
2. Hapus Possessive Pronoun
3. Hapus awalan pertama, jika tidak ada lanjut ke langkah 4a, jika ada cari lanjutkan ke 4b
4. Berikut langkahnya
    a. Hapus awalan kedua, lanjut ke 5a
    b. Hapus akhiran, jika tidak ditemukan maka kata diasumsikan sebagai rootword, jika ditemukan maka lanjut ke 5b
5. Berikut langkahnya
    a. Hapus akhiran, kemudian kata akhir diasumsikan sebagai rootword

b. Hapus awalan kedua, kemudian kata akhir diasumsikan sebagai rootword

Partikel dan possesive pronoun

Akhiran	Replacement	Contoh
Partikel		
-kah	Null	bukukah
-lah	Null	Pergilah
-pun	Null	Bukupun
Possesive pronoun		
-ku	Null	Bukuku
-mu	Null	Bukumu
-nya	Null	bukunya

First order derivational prefix

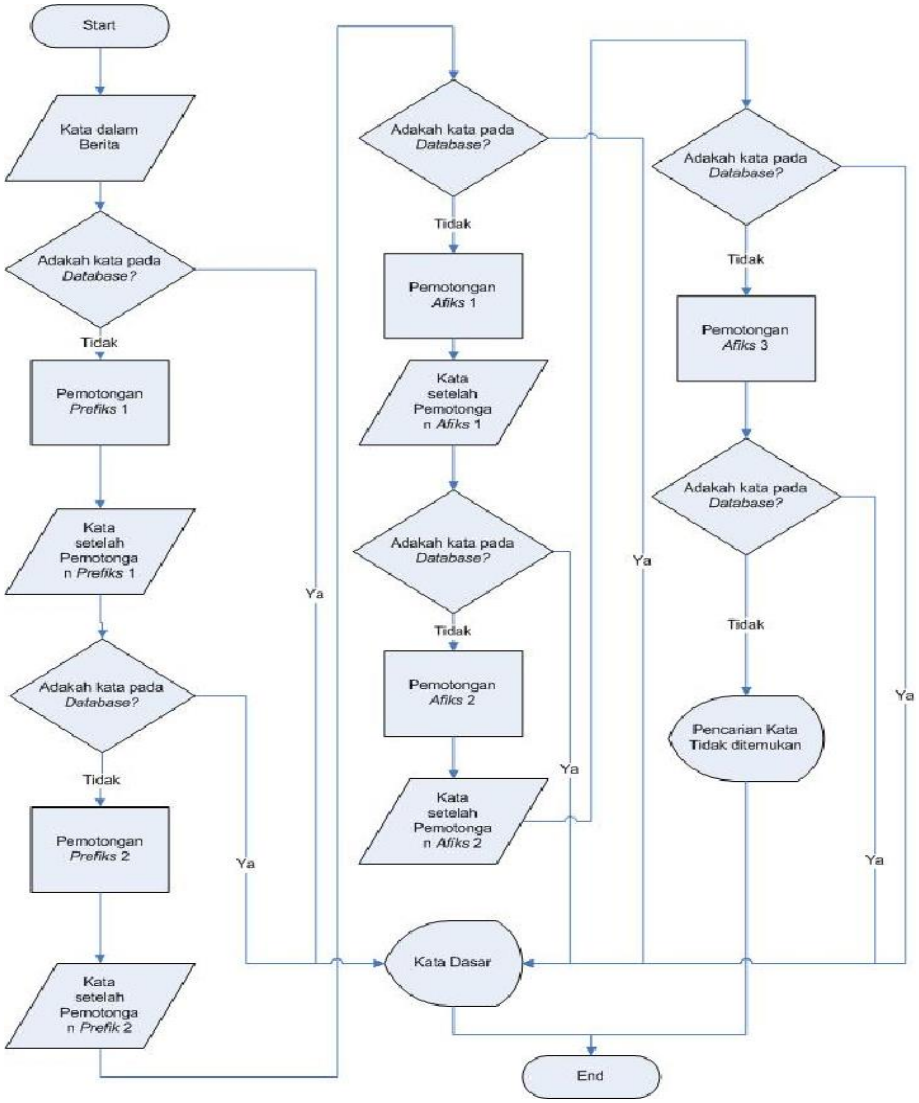
Awalan	Replacement	Additional condition	Contoh
Meng-	Null	Null	Mengukur -> ukur
Meny-	S	V...	Menyapu -> sapu
Men-	Null	Null	Menduga -> duga
Mem-	P	V...	Memaksa -> paksa
Mem-			
Me	Null		
Peng-	S		
Peny-	Null		

Pen-	P		
Pem-	Null		
Di-	Null		
Ter-	Null		
Ke-	Null	Null	Kekasih -> kasih

Second order derivational prefix

Awalan	Replacement	Additional condition	Contoh
Ber-	Null	Null	Berlari -> lari
Bel-	Null	Ajar	Belajar -> ajar
Be-	Null	Ker	Bekerja -> kerja
Per-	Null	Null	Perjelas -> jelas
Pel-	Null	Ajar	Pelajar -> ajar
Pe-	Null	Null	Pekerja -> kerja

Berikut alur dari pada proses stemming



E. Algoritma Nazief & Adriani

Algoritma Nazief & Adriani yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut :

1. Pertama cari kata yang akan diistem dalam kamus kata dasar. Jika ditemukan maka diasumsikan kata adalah *root word*. Maka algoritma berhenti.
2. *Inflection Suffixes* (“-lah”, “-kah”, “-ku”, “-mu”, atau “-nya”) dibuang. Jika berupa

particles (“-lah”, “-kah”, “-tah” atau “-pun”) maka langkah ini diulangi lagi untuk menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada.

3. Hapus *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah 3a
   g. Jika “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 3b.
   h. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.
4. Hapus *Derivation Prefix*. Jika pada langkah 3 ada sufiks yang dihapus maka pergi ke langkah 4a, jika tidak pergi ke langkah 4b.

   a. Periksa tabel kombinasi awalan-akhiran yang tidak diijinkan. Jika ditemukan maka algoritma berhenti, jika tidak pergi ke langkah 4b.

   b. For i = 1 to 3, tentukan tipe awalan kemudian hapus awalan. Jika root word belum juga ditemukan lakukan langkah 5, jika sudah maka algoritma berhenti. Catatan: jika awalan kedua sama dengan awalan pertama algoritma berhenti.
5. Melakukan Recoding.
6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai *root word*. Proses selesai.

Tipe awalan ditentukan melalui langkah-langkah berikut:

1 Jika awalannya adalah: “di-”, “ke-”, atau “se-” maka tipe awalannya secara berturut-turut adalah “di-”, “ke-”, atau “se-”.

2 Jika awalannya adalah “te-”, “me-”, “be-”, atau “pe-” maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.

3 Jika dua karakter pertama bukan “di-”, “ke-”, “se-”, “te-”, “be-”, “me-”, atau “pe-” maka berhenti.

4 Jika tipe awalan adalah “none” maka berhenti. Jika tipe awalan adalah bukan “none” maka awalan dapat dilihat pada Tabel 2. Hapus awalan jika ditemukan.

Tabel 1. Kombinasi Awalan Akhiran Yang Tidak Diijinkan

Awalan	Akhiran yang tidak diizinkan
be-	-i
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan

Tabel 2. Cara Menentukan Tipe Awalan Untuk awalan “te-”

Following Characters				Tipe
Set 1	Set 2	Set 3	Set 4	Awalan
“-r-“	“-r-“	-	-	none
“-r-“		-	-	ter-luluh
“-r-“	not (vowel or “-r-”)	“-er-“	vowel	ter
“-r-“	not (vowel or “-r-”)	“-er-“	not vowel	ter-
“-r-“	not (vowel or	not “-er-“	-	ter

	“-r-”)			
not (vowel or “-r-”)	“-er-“	vowel	-	none
not (vowel or “-r-”)	“-er-“	not vowel	-	te

Tabel 3. Jenis Awalan Berdasarkan Tipe Awalannya

Tipe Awalan	Awalan yang harus dihapus
di-	di-
ke-	ke-
se-	se-
te-	te-
ter-	ter-
ter-luluh	ter

Untuk mengatasi keterbatasan pada algoritma di atas, maka ditambahkan aturan-aturan dibawah ini:

1. Aturan untuk reduplikasi.

1 Jika kedua kata yang dihubungkan oleh kata penghubung adalah kata yang sama maka *root word* adalah bentuk tunggalnya, contoh : “buku-buku” *root word*-nya adalah “buku”.

2 Kata lain, misalnya “bolak-balik”, “berbalas-balasan, dan ”seolah-olah”. Untuk mendapatkan *root word*-nya, kedua kata diartikan secara terpisah. Jika keduanya memiliki *root word* yang sama maka diubah menjadi bentuk tunggal, contoh: kata

"berbalas-balasan", "berbalas" dan "balasan" memiliki *root word* yang sama yaitu "balas", maka *root word* "berbalas-balasan" adalah "balas". Sebaliknya, pada kata "bolak-balik", "bolak" dan "balik" memiliki *root word* yang berbeda, maka *root word*-nya adalah "bolak-balik".

2. Tambahan bentuk awalan dan akhiran serta aturannya.

1 Untuk tipe awalan "mem-", kata yang diawali dengan awalan "memp-" memiliki tipe awalan "mem-".

1 Tipe awalan "meng-", kata yang diawali dengan awalan "mengk-" memiliki tipe awalan "meng-".

Berikut contoh-contoh aturan yang terdapat pada awalan sebagai pembentuk kata dasar.

1. Awalan SE-

Se + semua konsonan dan vokal tetap tidak berubah

Contoh :

- Se + bungkus = sebungkus
- Se + nasib = senasib
- Se + arah = searah
- Se + ekor = seekor

2. Awalan ME-

Me + vokal (a,i,u,e,o) menjadi sengau "meng"

Contoh :

- Me + inap = menginap
- Me + asuh = mengasuh
- Me + ubah = mengubah

- Me + ekor = mengekor
- Me + oplos = mengoplos

Me + konsonan b menjadi “mem”

Contoh :

- Me + beri = member
- Me + besuk = membesuk

Me + konsonan c menjadi “men”

Contoh :

- Me + cinta = mencinta
- Me + cuci = mencuci

Me + konsonan d menjadi “men”

Contoh :

- Me + didik = mendidik
- Me + dengkur = mendengkur

Me + konsonan g dan h menjadi “meng”

Contoh :

- Me + gosok = menggosok
- Me + hukum = menghukum

Me + konsonan j menjadi “men”

Contoh :

- Me + jepit = menjepit
- Me + jemput = menjemput

Me + konsonan k menjadi “meng” (luluh)

Contoh :

- Me + kukus = mengukus
- Me + kupas = mengupas

Me + konsonan p menjadi “mem” (luluh)

Contoh :

- Me + pesona = mempesona
- Me + pukul = memukul

Me + konsonan s menjadi “meny” (luluh)

Contoh :

- Me + sapu = menyapu
- Me + satu = menyatu

Me + konsonan t menjadi “men” (luluh)

Contoh :

- Me + tanama = menanam
- Me + tukar = menukar

Me + konsonan (l,m,n,r,w) menjadi tetap “me”

Contoh :

- Me + lempar = melempar
- Me + masak = memasak
- Me + naik = menaik
- Me + rawat = merawat
- Me + warna = mewarna

3. Awalan KE-

Ke + semua konsonan dan vokal tetap tidak berubah

Contoh :

- Ke + bawa = kebawa
- Ke + atas = keatas

4. Awalan PE-

Pe + konsonan (h,g,k) dan vokal menjadi “per”

Contoh :

- Pe + hitung + an = perhitungan
- Pe + gelar + an = pergelaran
- Pe + kantor + = perkantoran

Pe + konsonan “t” menjadi “pen” (luluh)

Contoh :

- Pe + tukar = penukar
- Pe + tikam = penikam

Pe + konsonan (j,d,c,z) menjadi “pen”

Contoh :

- Pe + jahit = penjahit
- Pe + didik = pendidik
- Pe + cuci = pencuci
- Pe + zina = penzina

Pe + konsonan (b,f,v) menjadi “pem”

Contoh :

- Pe + beri = pemberi
- Pe + bunuh = pembunuh

Pe + konsonan “p” menjadi “pem” (luluh)

Contoh :

- Pe + pikir = pemikir
- Pe + potong = pemotong

Pe + konsonan “s” menjadi “peny” (luluh)

Contoh :

- Pe + siram = penyiram
- Pe + sabar = penyabar

Pe + konsonan (l,m,n,r,w,y) tetap tidak berubah

Contoh :

- Pe + lamar = pelamar
- Pe + makan = pemakan
- Pe + nanti = penanti

Pe + wangi = pewangi

F. PENERAPAN STEMMING

1. PENERAPAN ALGORITMA STEMMING NAZIEF & ADRIANI DAN SIMILARITY PADA PENERIMAAN JUDUL THESIS

Pendahuluan

Pencarian Informasi berupa teks atau dokumen yang dikenal dengan istilah Invormation Retrieval (IR) merupakan proses pemisahan dokumen-dokumen yang dianggap relevan dari sekumpulan dokumen yang tersedia. Semakin banyaknya jumlah dokumen thesis yang tersedia memungkinkan terjadinya kesamaan tema atau topik pembahasan yang diangkat sebagai judul thesis.

Algoritma Stemming adalah salah satu algoritma yang digunakan untuk meningkatkan performa IR dengan cara mentrasformasikan semua kata dalam teks dokumen kedalam

“rootword” atau kata dasarnya. Algoritma Stemming untuk satu bahasa satu dan bahasa lainnya berbeda, sebagai contoh Bahasa Inggris memiliki morfologi yang berbeda dengan Bahasa Indonesia. Proses stemming untuk Bahasa Indonesia lebih kompleks karena terdapat lebih banyak variasi imbuhan yang harus dibuang untuk mendapatkan rootword[1]. Penggunaan algoritma Stemming dalam Bahasa Indonesia sebelumnya pernah diteliti oleh Nazief dan Adriani yang berdasarkan dari Algoritma Stemming Porter. Similarity atau tingkat kesamaan dokumen lama dengan dokumen baru dihitung untuk mengetahui tingkat kesamaan topik melalui judul dan abstraksi dokumen yang diambil berdasarkan indexnya. Similarity digunakan untuk mencegah terjadinya duplikasi dan plagiatisme. Pada papper ini hanya akan dibahas mengenai konsep algoritma stemming dan similiraity pada penerimaan judul thesis dengan Bahasa Indonesia.

Thesis adalah istilah yang digunakan di Indonesia untuk mengilustrasikan suatu karya tulis ilmiah berupa paparan tulisan hasil penelitian sarjana S2 yang membahas suatu permasalahan/fenomena dalam bidang ilmu tertentu dengan menggunakan kaidah-kaidah yang berlaku.

Landasan Teori

Stemming

Stemming merupakan suatu proses yang terdapat dalam sistem IR yang mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (rootword) dengan menggunakan aturan- aturan tertentu. Sebagai contoh, kata bersama, kebersamaan, menyamai, akan distem ke root wordnya yaitu “sama”. Proses stemming pada teks Bahasa Indonesia berbeda dengan stemming pada teks berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks.

Sedangkan pada teks berbahasa Indonesia, selain sufiks, prefiks, dan konfiks juga dihilang-kan[2].

Similarity

Konsep similarity sudah menjadi isu yang sangat penting hampir setiap bidang ilmu pengetahuan.[3] Terdapat tiga macam teknik yang dibangun untuk menentukan nilai similarity:

1. Distance-based similarity measure Distance-based similarity measure mengukur tingkat kesamaan dua buah objek dari segi jarak geometris dari variabel-variabel yang tercakup di dalam kedua objek tersebut. Metode ini meliputi : Menkowski Distance, Manhatann/ City Block Distance, Eulidean Distance, Jaccard Distance, Dicee's Coefficient, Cosine similarity, Levenshtein Distance, Hamming distance, dan Soundex distance.

2. Feature-basedsimilaritymeasure Feature-based similarity measure melakukan perhitungan tingkat kemiripan dengan merepresentasikan objek ke dalam bentuk feature- feature yang ingin dibandingkan. Feature-based ini banyak digunakan pada pengklasifikasian atau patern matching untuk gambar dan teks.

3. Probabilistic-basedsimilaritymeasure Probabilistic-based similarity measure mengukur tingkat kemiripan dua objek dengan me- representasikan dua set objek yang dibandingkan dalam bentuk probability. Metode ini mencakup Kullback Leibler Distance dan Posterior Probability. Perhitungan Similarity pada jarak antara dua entitas informasi adalah syarat inti pada semua kasus penemuan informasi, seperti pada Information Retrieval yang kemudian digunakan untuk pendeteksi plagiarisme. Mengukur kesamaan semantik (Semantic Similarity) merupakan tugas yang sulit dalam membandingkan kesamaan kata-kata dalam dokumen. Kesamaan kata-kata dalam dokumen diukur berdasarkan indeks kesamaan jumlah kata yang mungkin.

2. STEMMER UNTUK BAHASA MADURA DENGAN MODIFIKASI METODE ENHANCED CONFIX STRIPPING STEMMER

PENDAHULUAN

Indonesia merupakan negara yang tidak hanya luas wilayahnya, akan tetapi juga kaya akan budaya, seni dan bahasa. Indonesia memiliki 707 bahasa daerah [1], sedangkan menurut Badan Bahasa Kemendiknas mencatat sejumlah 617 bahasa daerah yang tersebar di seluruh Indonesia, terdapat sejumlah 139 bahasa daerah yang terancam punah, dan sejumlah 15 bahasa daerah dinyatakan punah [2]. Ancaman tersebut kemungkinan disebabkan minimnya penggunaan bahasa daerah dalam kehidupan sehari-hari, selain itu penelitian terkait bahasa daerah kurang menarik untuk diteliti, terutama jika dikaitkan dengan teknologi informasi.

Menurut Lauder dalam [3] mengatakan bahwa bahasa Madura(bM) merupakan bahasa daerah keempat terbesar dari 13 besar bahasa daerah di Indonesia, sekitar 13,7 juta jiwa jumlah penuturnya. Bahasa Madura adalah bahasa yang komplek dan unik, baik dari sisi sosiolinguistik, morfologi dan fonologi, bahasa Madura terdiri empat dialek utama yaitu (1) dialek Sumenep, (2) dialek Pamekasan, (3) dialek Bangkalan, dan (4) dialek kangean, dan dua dialek tambahan, serta memiliki delapan tingkat tutur atau ondhâghân bhâsa yang terdiri dari empat tingkat tutur utama yaitu (1) enjâ'-iyâ, (2) engghè-enten, (3) engghi-enten, dan (4) èngghi-bhunten, dan empat tingkat tutur turunan [3] [4].

Keunikan lain dari bahasa Madura dari unsur fonologi khususnya dari variasi fonem, yakni pada huruf vokal dan konsonan [5]. Bahasa Madura memiliki enam buah vokal, yakni /a/, /i/, /u/, /e/, /ə/ dan /o/ dimana keenam vokal tersebut terdiri dari lima belas alofon, alofon unik terdapat pada vokal /a/, /i/, /u/ /e/ dan /o/, sedangkan pada konsonan

memiliki 31 buah dengan beberapa fonem yang unik, yaitu /‟/, /bha/, /dha/, /gha/, /jha/, /th/.

Penelitian tentang stemming bahasa Daerah khususnya bahasa Madura masih sangat minim dilakukan, yakni dilakukan oleh Sholihin dkk [6] dengan memodifikasi metode Enhanced Confix Stripping yang dikembangkan oleh Arifin dkk [7], tetapi penelitian ini terbatas dalam penggunaan kata yang umum digunakan dalam dialek Bangkalan, yakni sejumlah 400 kata.

Keunikan fonem dalam bahasa Madura berpengaruh dalam pelafalan/artikulasi dan penulisan kata, kesalahan dalam penulisan kata(typo) khususnya penggunaan fonem yang kurang tepat akan berpengaruh pelafalan dan arti kata, banyak kata dalam bahasa Madura yang memiliki kemiripan dalam penulisan[5] diantaranya pada fonem vokal /â/ dan /a/ pada kata rajâ(besar) dan raja(raja) sedangkan pada fonem konsonan /k/ dan /‟/ pada kata katok(bersinggungan) dan kato’(celana dalam), konsonan /b/ dan /bha/ pada kata bâja (saat, waktu) dan bhâjâ (buaya), konsonan /d/ dan /dha/ pada kata dâpa’(sampai) dan dhâpa’(telapak). Fonem konsonan lainnya seperti pada /g/ dan /gha/, /j/ dan /jha/ dll. Keunikan dan kemungkinan kesalahan penulisan tersebut memungkinkan menimbulkan ambiguitas, overstemming atau understemming pada saat proses stemming, misalnya pada kalimat “Bâdâ bâddhâ beddhâ'na Raja bheddhâ rajâ , ” (Ada wadah bedaknya raja robek besar).

Dalam rencana penelitian ini akan dikembangkan sebuah algortima stemmer untuk bahasa Madura dengan menerapkan algoritma Enhanced Confix Stripping Stemmer dimana rule base pada algoritma acuan tersebut disesuaikan dengan morfologi bahasa Madura, dataset yang digunakan adalah kata-kata dalam dialek Sumenep, Pamekasan dan Bangkalan yakni dialek paling sering digunakan di dalam bahasa Madura, yakni kata-kata

dari puisi berbahasa Madura, pemilihan puisi sebagai dataset/datauji karena variasi kata-katanya lebih beragam, terutama kata yang mengandung sisipan.

TINJAUAN PUSTAKA

Stemming

Stemming merupakan proses ektraksi suatu kata dalam suatu dokumen digital yang bertujuan untuk mendapat kata dasarnya, misalnya dalam bahasa Indonesia kata „menendang", „tendangan", „penendang", „menendangi" kata dasarnya adalah „tendang", contoh dalam bahasa Madura „ter- penter"(pintar-pintar), „mamenter'(membuat pintar), „penterran'(lebih pintar), „terpenterran'(paling pintar), „termapenter'(berlagak pintar) kata dasarnya adalah „penter"(pintar). Stemming ini nantinya bisa digunakan untuk penelitian di bidang temu kembali infomasi(information retrieval), misalnya translasi, ringkasan dan klasifikasi suatu dokumen[8].

Teknik stemming yang paling populer digunakan adalah teknik Porter Stemmer terutama untuk bahasa Inggris, yakni sebuah teknik stemmer dengan pendekatan rule base berdasarkan struktur morfologi bahasa. Teknik tersebut diadopsi oleh Tala untuk stemming bahasa Indonesia [9], teknik stemming bahasa Indonesia lainnya adalah Config-Stripping yang dibuat oleh Adriani dkk [8], kedua teknik stemmer bahasa Indonesia tersebut sama-sama menggunakan rule base, yang membedakan keduanya adalah penggunaan kamus bahasa Indonesia sebagai acuan, pada teknik stemmer Config-Stripping menggunakan kamus sebagai acuan sedangkan pada teknik Tala tidak. Perbaikan Config- Stripping diberi nama Enhanced Confix Stripping Stemmer(ECS) [7], algoritma ini memperbaiki beberapa aturan agar proses stemming berhasil untuk kata-kata dengan format "mem+p-", "mem+s-", "menge-", "penge-", "peng+k-", serta

penambahan algoritma untuk mengatasi kesalahan overstemming yang diberi nama “iterasi pengembalian akhiran”.

Morfologi BahasaMadura

Dalam morfologi bahasa Madura banyak macamnya, menurut asal pembentukannya (kadhâddhiânna oca’) dibagi menjadi tiga [4], yakni:

1. Kata dasar(oca’ asal), ialah kata belum berubah dari asalnya, contoh: obâng(uang), bâto(batu), berraâ’(berat), dhâddhi(jadi), kakan(makan)
2. Kata ubahan(oca’ obâ’an), ialah kata yang sudah berubah dari asalnya setelah ditambahkan imbuhan.

Imbuhan dalam bahasa Madura diantaranya:

a. Awalan/infiks(ter-ater), yakni imbuhan di awal kata dasar, diantaranya: N-, a-, ta-, ka-, pa-, é-, éka, épa-, ma-, sa-, pan-, pam-, pang-, pé- contoh: atellor(bertelur), épésa(dipisah), takébâ (terbawa), kaburu (kerburu), sabungko (serumah), pabhersé (bersihkan), panjâgâ (penjaga), pambâgi (pembagi), pangghâluy (pengaduk), pétotor (penutur), épabagus (dibuat bagus), ékandi’ (dimiliki), saséba’(sebagian).
b. Akhiran/sufiks(panoténg), yakni imbuhan di akhir kata dasar, diantaranya: -a, -é, -aghi, -an, - en, -na dan, contoh: tédunga(mau tidur), toro’é(ikuti), sambiaghi (bawakan), nangésan (cengeng), kalagguen (kepagian), parlona (perlunya)

c. Imbuhan dan akhiran/sufiks(ter-ater ban panoténg), yakni imbuhan yang diberikan pada awal dan akhir dari kata dasar, diantaranya: N-e, N-aghi, N-ana, a-e, a-aghi, ma-e, ma-an, ma-ana, ma-aghi, é-é, e-na, e-aghi, eka-e, eka-ana, eka- aghi, apa-an, epa-e, epa-aghi, ka-e, ka-aghi, pa-e, pa-aghi, pa-an, par-an, pa-na, jha-na, sa-an, sa-na, cé‟-

na, contoh: ékala'aghi (diambilkan), katédungan (tertidur), akorosan (lebih kurus), pacolo'an (cerewet), ce'saké'na (begitu sakitnya), épaté'é (dibunuh), épangangguaghi(dipakaikan)

b. Tandhuk, yakni imbuhan yang meluruhkan kata dasarnya, diantaranya: ma-, na-, nya-, nga-, contoh: masang (memasang), mukka'(membuka), noles (menulis), namen (menanam), nyaba' (menaruh), nyokor (mencukur), ngarang (mengarang), ngakan (memakan)

c. Sisipan(sesselan), yakni kata yang mendapatkan sisipan di tengah kata dasar misalnya: al, ar, en, om, um, contoh: bhâlâtra(menjadi rata), karépé' (terhimpit), pénalang (penghalang), tomekka(terkabul), ghumancang(cepat).

d. Kata ulang / Reduplikasi (oca' rangkebbhân) pada bahasa Madura terdiri dari tiga-macam kata ulang, yaitu: (a) Kata ulang sempurna (Oca' rangkep buto), yakni kata ulang dengan pengulangan penuh pada bentuk kata dasarnya, contoh: kéra-kéra (kira-kira), moghâ-moghâ (moga-moga); (b) Kata ulang depan (Oca' rangkep ada'), yakni kata ulang tidak sempurna yang suku kata depan diambil untuk diulang, contoh: lalaké (laki-laki), papareng(pemberian). (c) Kata ulang belakang(Oca‟ rangkep budhi), yakni kata ulang tidak sempurna yang suku kata belakang diambil untuk diulang, contoh: ca‟oca‟(kata-kata), nibenni(bukan-bukan)

BAB 4 N – GRAMS

C. Pengantar N-Grams

Word prediction task adalah potongan n karakter dalam suatu string tertentu atau potongan n kata dalam suatu kalimat tertentu, atau dapat dikatakan sebagai sebuah sub urutan dari sejumlan n elemen dari urutan yang diberikan. Elemen yang dimaksud dapat berupa fonem, huruf, kata tergantung kebutuhan aplikasi. Untuk mendukung *word prediction task* maka dilakukan penggabungan metode N-grams dengan fungsi *scoring* seperti *language model*.

Pemodelan bahasa atau *language model* dapat memberikan nilai probabilitas untuk setiap kata yang berada dalam suatu *ruang sample* atau *text dataset* atau *dokumen*. *Language model* didasarkan pada urutan kata dan kata yang paling sering digunakan dalam inputan text. *Language model* adalah metode input berdasarkan frekuensi kata dan tidak peka terhadap konteks dan bobot yang ditenukan dengan mengiterasi subset dari kombinasi yang mungkin.

Language model yang sudah dikenal dan banyak digunakan oleh peneliti adalah N-grams. N-grams adalah kumpulan dari item sejumlah n yang disusun secara berurutan dari text atau speech. Representasi N-grams dapat diketahui sebagai berikut :

1. N = 1 disebut dengan Unigram
2. N = 2 disebut dengan Bigram
3. N = 3 disebut dengan Trigram
4. Dst

Contoh N-grams dalam kalimat

Kalimat = “Andi suka bermain sepakbola di lapangan Senayan”

1. Unigram (N=1)

   andi, suka, bermain, sepakbola, di, lapangan, senayan

2. Bigram (N=2)

   andi suka, suka bermain, bermain sepakbola, sepakbola di, di lapangan, lapangan senayan

3. Trigram (N=3)

   andi suka bermain, suka bermain sepakbola, bermain sepakbola di, sepakbola di lapangan, di lapangan senayan

Contoh N-grams dalam kata

Kata = “pemerintah”

1. Unigram (N=1) : p, e, m, e, r, i, n, t, a, h
2. Bigram (N=2) : pe, em, me, er, ri, in, nt, ta, ah
3. Trigram (N=3) : pem, eme, mer, eri, rin, int, nta, tah

Untuk megetahui banyaknya N-grams dalam sebuah kalimat maka dapat menggunakan rumus dibawah ini. Jika diasumsikan X adalah jumlah kata dalam suatu kalimat K, maka jumlah n-gram dari kalimat K adalah :

$$N\textit{grams}K = X - (N - 1)$$

Contohnya :

Kalimat = “The cow jumps over the moon”

X = 6

N = 2

Bigram = The cow, Cow jumps, Jumps over, Over the, The moon

N*grams*K = X – (N – 1)

N*grams*K = 6 – (2 - 1) = 5

Language model adalah salah satu bagian terpenting dari *natural Language Processing* (NLP) modern. Ada banyak jenis aplikasi untuk pemodelan bahasa, seperti :

1. prediksi kata yang akan digunakan, seperti pada fitur auto-correct
2. pencarian kata kunci tertentu di dalam Google Search
3. Automatic speech recognition
4. Handwriting and character recognition
5. Spelling correction
6. Machine translation
7. Dst.

Dalam *Word prediction task* Kita dapat memodelkan tugas prediksi kata sebagai kemampuan untuk menilai probabilitas bersyarat dari suatu kata yang diberikan kata-kata sebelumnya dalam urutan $P(w_{1,}w_{2,},\dots,w_{n-1})$. Berikut rumus yang digunakan untuk menghitung nilai probabilitas :

Kalimat adalah urutan dari kata-kata	$S = w_1, w_2, \dots, w_n$
Probabilitas dari kata w dari sejumlah history h	$P(w\|h)_,$

Anggap h = pergi ke pasar	$P(bersama \mid pergi \quad ke \quad pasar)$
Probabilitas kata berikutnya adalah bersama	$P(bersama \mid pergi \quad ke \quad pasar) = \frac{C(pergi \quad ke \quad pasar \quad bersama)}{C(pergi \quad ke \quad pasar)}$
Sequence dari N kata	$w_1, \ldots, w_n$ atau w_1^n
Join probability dari tiap kata dalam sequence	$P(X = w_1, \quad Y = w_2, \quad Z = w_3, \ldots, W = w_n)$ $P(w_1, w_2, \ldots, w_n)$

Sedangkan untuk menghitung probabilitas dari seluruh sequence dapat menggunakan Chain Rule of Probability sebagai berikut :

$$P(X_1, \ldots X_n) = P(X_1)P(X_2|X_1)P(X_3|X_1^2)\ldots P(X_n|X_1^{n-1}) = \prod_{k=1}^{n} P(X_k|X_1^{k-1})$$

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

Chain rule menunjukkan hubungan antara perhitungan joint probability dari suatu sequence dan perhitungan conditional probability dari sebuah kata diberikan kata sebelumnya.

Jadi untuk menghitung probabilitas dari kalimat P(pergi ke pasar bersama) =

$$P(pergi) * P(ke|pergi) * P(pasar|ke, pergi) * P(bersama|pasar, ke, pergi)$$

Kalau kita perhatikan dengan menggunakan chain rule, semakin banyak N yang dimiliki suatu kalimat, akan semakin sulit perhitungannya. pemodelan N-Gram(dengan N=1) menjadi:

$$P(w_1, w_2, \ldots, w_T) \approx P(w_1)P(w_2), \ldots, P(w_T)$$

Model n-gram yang hanya melibatkan 2 elemen menggunakan teorema bayes yaitu :

P = Probabilitas kata yang diberikan oleh kata sebelumnya

$$P(W_n|W_{n-1}) = \frac{P(W_{n-1}, W_n)}{P(W_{n-1})} \qquad P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$$

contoh probabilitas :

Probabilitas kalimat : I want to eat Chinese Food

P(I want to eat Chinese Food) = P(I | <start>) *

P(want | I) *

P(to | want) *

P(eat | to)*

P(Chinese | eat) *

P(Food |Chinese) *

Berikut a bigram grammar fragment from BERP

Eat on	.16	Eat Thai	.03
Eat some	.06	Eat breakfast	.03
Eat lunch	.06	Eat in	.02
Eat dinner	.05	Eat Chinese	.02
Eat at	.04	Eat Mexican	.02

Eat a	.04	Eat tomorrow	.01
Eat Indian	.04	Eat dessert	.007
Eat today	.03	Eat British	.001
<start> I	.25	Want some	.04
<start> I'd	.06	Want Thai	.01
<start> Tell	.04	To eat	.26
<start> I'm	.02	To have	.14
I want	.32	To spend	.09
I would	.29	To be	.02
I don't	.08	British food	.60
I have	.04	British restaurant	.15
Want to	.65	British cuisine	.01
Want a	.05	British lunch	.01
		Chinese Food	.2

P(I want to eat British food) = P(I|<start>) *

P(want|I) *

P(to|want) *

P(eat|to) *

P(British|eat) *

P(food|British)

= .25*.32*.65*.26*.001*.60

= .0000080

P(I want to eat Chinese Food) = P(I | <start>) *

P(want | I) *

P(to | want) *

P(eat | to)*

P(Chinese | eat) *

P(Food |Chinese) *

= .25*.32*.65*.26*.02*.2

= 0.000054

D. Penerapan N-Gram

1. DETEKSI BAHASA UNTUK DOKUMEN TEKS BERBAHASA INDONESIA

Deteksi Bahasa dengan n-gram Penggunaan n-gram untuk deteksi bahasa didasarkan pada anggapan bahwa pola sebaran n-gram dari suatu bahasa bersifat unik karena ini terkait dengan frekuensi penggunaan huruf, atau pasangan huruf baik itu vokal atau konsonan dari suatu bahasa yang umumnya berbeda dengan bahasa yang lain. Untuk unigram misalnya, yang jika dihitunmg frekuensinya adalah frekuensi keumunculan huruf dalam teks bahasa tertentu yang akan uniq untuk bahasa yang berbeda. Untuk teks bahasa Indonesia vokal a akan merupakan vokal yang frekuensi munculnya paling tinggi, sementara untuk bahasa inggris vokal e merupakan vokal yang frekuensinya paling tinggi. Demikian juga jika digunkan abi-gram dan tri-gram, keunikan pola n-gram dari suatu bahasa akan nampak lebih menonjol.

2. PENGGUNAAN N-GRAM PADA ANALISA SENTIMEN PEMILIHAN KEPALA DAERAH JAKARTA MENGGUNAKAN ALGORITMA NAÏVE BAYES

Berdasarkan penelitian di atas yang sejenis penelitian ini mencoba melakukan analisa sentimen dengan menggunakan Algoritma Naïve Bayes untuk mengklasifikasikan data twitter mengenai topik pemilihan kepala daerah Jakarta dengan membagi pendapat menjadi 2 yaitu positif dan negatif dengan menerapkan penggunaan N-gram. Tujuannya yaitu untuk melihat tingkat akurasi klasifikasi sistem dengan penggunaan N-gram pada Algoritma Naïve Bayes, sebagaimana pada penelitian Afshoh (2017) penggunaan fitur N-gram sangat berpengaruh dalam perhitungan ketepatan nilai akurasi klasifikasi Algoritma Naïve Bayes.

Tahapan yang dilakukan :

1. Data Twitter

   Tahap pertama yang dilakukan pada penelitian yaitu mengumpulkan data twitter yang diambil secara acak tentang pemilihan kepala daerah Jakarta dari 3 calon pasangan, kemudian untuk setiap calon pasangan masing-masing terdiri dari 100 data positif dan 100 data negatif. Data twitter tersebut berisi kalimat pendapat dari masyarakat mengenai ketiga calon pasangan gubernur Jakarta yang dicari menggunakan #dukung untuk mencari data positif dan #tolak untuk mencari data negatif. Total data twitter yang berhasil dikumpulkan yaitu 600 data yaitu 100 data positif dan 100 data negatif untuk pasangan calon gubernur pertama, 100 data positif dan 100 data negatif untuk pasangan calon gubernur kedua, serta 100 data positif dan 100 data negatif untuk pasangan calon ketiga.

2. Preprocessing

   Tahap kedua yaitu dengan melakukan preprocessing data. Preprocessing

merupakan pengolahan awal data dan mempersiapkan data teks untuk dilakukan proses klasifikasi, yaitu dengan melakukan metode:

1. Normalisasi

   Metode normalisasi merupakan metode untuk menormalisasikan data teks twitter menjadi data teks normal. Karena keterbatasan twitter yang membatasi karakternya banyak pengguna menuliskan kata-kata gaul seperti “TDK” jika dinormalisasikan menjadi “TIDAK”.

2. Transfrom Case

   Merupakan metode untuk mengubah data teks twitter yang ditulis dengan huruf besar (upper case) menjadi huruf kecil semua (lower case).

3. Tokenisasi

   Tokenisasi merupakan metode pengambilan data teks pada suatu dokumen untuk dipisahkan menjadi beberapa karakter/token.

4. Generate N-gram

   Setelah data teks twitter di normalisasi dan transform case berikutnya yaitu dengan tokenisasi menggunakan jenis token unigram, bigram dan trigram sama seperti pada penelitian Nurfalah & Ardiyanti (2017) yang melakukan pembagian N-gram menjadi tiga jenis. N-gram merupakan penggabungan kata sifat yang sering muncul untuk menunjukan suatu sentimen. Pada penelitian menggunakan jenis token unigram yaitu token data teks twitter yang hanya terdiri dari satu kata, kemudian bigram yaitu token data teks twitter yang terdiri dari dua kata dan trigram yaitu token data teks twitter yang terdiri dari tiga kata. Penerapan N-gram dapat dilihat seperti berikut :

Contoh kalimat : Pemilihan kepala daerah Jakarta tahun ini tidak begitu ramai dibandingkan dengan tahun sebelumnya.

Unigram

Pemilihan, kepala, daerah, Jakarta, tahun, ini, tidak, begitu ramai, dibandingkan, dengan, tahun, sebelumnya.

Bigram

Pemilihan kepala, kepala daerah, daerah Jakarta, Jakarta tahun, tahun ini, ini tidak, tidak begitu, begitu ramai, ramai dibandingkan, dibandingkan dengan, dengan tahun, tahun sebelumnya.

Trigram

Pemilihan kepala daerah, kepala daerah Jakarta, daerah Jakarta tahun, Jakarta tahun ini, tahun ini tidak, ini tidak begitu, tidak begitu ramai, begitu ramai dibandingkan, ramai dibandingkan dengan, dibandingkan dengan tahun, dengan tahun sebelumnya.

Tujuan pemakaian N-gram dilakukan pada penelitian ini karena dalam bahasa Indonesia banyak frase yang tidak hanya terdiri dari satu kata.

3. DETEKSI KEMIRIPAN DOKUMEN TEKS DENGAN METODE N-GRAM BERBASIS PANJANG KATA

Ada berbagai metode deteksi plagiarisme dokumen teks yang sudah diteliti dan dikembangkan. Salah satu metode bagi penjiplakan yang tidak terlalu banyak melakukan perubahan dari teks asli (disebut plagiarisme literal), adalah metode deteksi berbasis leksikal, pada umumnya teknik n-gram berbasis karakter dan n-gram berbasis kata. Teknik n-gram berbasis karakter dan n-gram berbasis kata memakan waktu yang lama dengan bertambahnya jumlah dokumen, sehingga muncul berbagai variasi teknik n-gram, seperti

n-gram berbasis stop word (kata-kata umum) dan n-gram berbasis panjang kata. Teknik n-gram berbasis panjang kata memiliki keunggulan yaitu membutuhkan ruang dan waktu komputasi yang lebih kecil, karena hanya meng-encode panjang kata (jumlah huruf per kata).

BAB 5 PEMBOBOTAN KATA

Dapat diketahui bahwa Pemrosesan Bahasa Alami beririsan dengan Sistem Temu Kembali Informasi (STKI), sedangkan STKI berhadapan dengan pencarian informasi yang sesuai dengan query pengguna dari koleksi dokumen. Koleksi dokumen tersebut terdiri dari dokumen-dokumen yang beragam panjangnya dengan kandungan term yang berbeda pula. Pada dokumen yang besar, skema yang paling sukses dan secara luas digunakan untuk pemberian bobot term adalah skema pembobotan *term weightingi* (TF-IDF).

Hal yang perlu diperhatikan dalam pencarian informasi dari koleksi dokumen yang heterogen adalah pembobotan term. Term dapat berupa kata, frase atau unit hasil indexing lainnya dalam suatu dokumen yang dapat digunakan untuk mengetahui konteks dari dokumen tersebut, maka untuk setiap kata tersebut diberikan indikator, yaitu *term weight*.

Contoh penerapan TF-IDF yaitu pada pencarian karya ilmiah, yaitu dengan melakukan pengukur tingkat similaritas antara dokumen dengan keyword yang didapat dari ekstraksi teks pada dokumen sehingga mendapatkan data yang terurut dari kemiripannya paling tinggi. Setelah bobot w masing2 dokumen diketahui selanjutnya dilakukan pengurutan dimana semakin besar w, semakin besar tingkat similatiras dokumen terhadap kata kunci dan begitupun sebailknya.

TF merupakan frekuensi kemunculan *term* (t) pada dokumen (d). Semakin besar jumlah kemunculan suatu term (TF tinggi) dalam dokumen, semakin besar pula bobotnya atau akan memberikan nilai kesesuaian yang semakin besar. Pada TF, terdapat beberapa jenis formula yang dapat digunakan yaitu :

1. TF biner (binary TF), hanya memperhatikan apakah suatu kata atau term ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak diberi nilai nol (0).
2. TF murni (raw TF), nilai TF diberikan berdasarkan jumlah kemunculan suatu term di dokumen. Contohnya, jika muncul lima (5) kali maka kata tersebut akan bernilai lima (5).
3. TF logaritmik, hal ini untuk menghindari dominansi dokumen yang mengandung sedikit term dalam query, namun mempunyai frekuensi yang tinggi. Berikut adalah rumus TF logaritmik :

$$TF = \begin{cases} 1 + log_{10}\left(f_{t,d}\right), & f_{t,d} > 0 \\ 0, & f_{t,d} = 0 \end{cases}$$

dimana $f_{t,d}$ adalah frekuensi term (t) pada document (d)

Jadi jika suatu kata atau term terdapat dalam suatu dokumen sebanyak 5 kali maka diperoleh bobot = 1 + log (5) = 1.699. Tetapi jika term tidak terdapat dalam dokumen tersebut, bobotnya adalah nol (0).

IDF (Inverse Document Frequency) merupakan sebuah perhitungan dari bagaimana term didistribusikan secara luas pada koleksi dokumen yang bersangkutan. IDF menunjukkan hubungan ketersediaan sebuah term dalam seluruh dokumen. Semakin sedikit jumlah dokumen yang mengandung term yang dimaksud, maka nilai IDF semakin besar. Sedangkan untuk IDF dihitung dengan menggunakan rumus sebagai berikut :

$$IDF_j = \log\left(\frac{D}{df_j}\right)$$

dimana D adalah jumlah semua dokumen dalam koleksi sedangkan df_j adalah jumlah dokumen yang mengandung term (tj).

Jenis formula TF yang biasa digunakan untuk perhitungan adalah TF Murni (raw TF). Demikian dengan rumus umum untuk TF-IDF adalah penggabungan dari formula

perhitungan raw TF dengan formula IDF dengan cara mengalikan nilai TF dengan nilai IDF. Berikut rumus untuk menghitung TF-IDF :

$$W_{ij} = tf_{ij} \, . idf_j$$

$$W_{ij} = tf_{ij} \, . \log\left(\frac{D}{df_j}\right)$$

dimana W_{ij} adalah bobot term (tj) terhadap dokumen (di). Sedangkan tf_{ij} adalah jumlah kemunculan term (tj) dalam dokumen (di). D adalah adalah jumlah semua dokumen yang ada dalam database dan df_j adalah jumlah dokumen yang mengandung term (tj).

Berapapun besarnya nilai tf_{ij}, apabila D = df_j, maka akan didapatkan hasil 0 (nol), dikarenakan hasil dari log 1, untuk perhitungan IDF. Untuk itu dapat ditambahkan nilai 1 pada sisi IDF, sehingga perhitungan bobotnya menjadi sebagai berikut :

$$W_{ij} = tf_{ij} \, . \log\left(\frac{D}{df_j}\right) + 1$$

query : *gold silver truck*

Sehingga didapatkan query terms (Q) :

Q
gold
silver
truck

koleksi dokumennya terdapat :

dokumen 1 (d1) : Shipment of gold damaged in a fire

dokumen 2 (d2) : Delivery of silver arrived in a silver truck

dokumen 3 (d3) : Shipment of gold arrived in a truck

Jadi total jumlah dokumen dalah koleksi dokumen (D) = 3

Setelah melalui proses ini, maka kata “of”, “in”, dan “a” pada ketiga dokumen dihapus lalu di-stemming sehingga didapatkan term-term (documents terms) sebagai berikut:

term
ship
gold
damage
fire
deliver
silver
arrive
truck

Tahapan proses keseluruhan diatas dinamakan text preprocessing.

Semakin besar nilai perhitungan bobot yang diperoleh maka semakin tinggi tingkat similaritas dokumen terhadap query. Contohnya untuk perhitungan bobot (w) term query silver dalam dokumen2 (d2) = Delivery of silver arrived in a silver truck, yaitu: jumlah kemunculan term silver dalam dokumen 2 (d2) adalah sebanyak dua kali (tf = 2), total dokumen yang ada di koleksi sebanyak tiga dokumen (D)=3, dari ketiga dokumen dalam koleksi, term silver muncul pada dokumen 2 (d2) saja, sehingga total dokumen yang mengandung term silver adalah satu dokumen (df)=1, sehingga dapat diperoleh nilai bobot term silver pada dokumen 2 (d2)

$$W_{ij} = tf_{ij} \,.\log\left(\frac{D}{df_j}\right) + 1$$

$$W_{ij} = 2 \,.\log\left(\frac{3}{1}\right) + 1$$

$W_{ij} = 2 . (0{,}477 + 1)$

$W_{ij} = 2{,}954$

	tf			df	D/df	IDF	IDF+1	W = tf * (IDF + 1)		
Q	d1	d2	d3					d1	d2	d3
Gold	1	0	1	2	1.5	0.176	1.176	1.176	0	1.176
Sliver	0	2	0	1	3	0.477	1.477	0	2.954	0
truck	0	1	1	2	1.5	0.176	1.176	0	1.176	1.176
								Sum(d1)	Sum(d2)	Sum(d3)
Nilai bobot tiap dokumen								1.176	4.130	2.352

BAB 6 SPELL CHECK

C. Pemeriksaan Ejaan

Dalam teknologi komputasi, pemeriksa ejaan adalah aplikasi yang memeriksa semua kata dalam sebuah dokumen untuk menghindari kesalahan pengejaan. Pemeriksa ejaan dapat berupa aplikasi mandiri atau sebagai bagian dari aplikasi yang lebih besar, seperti aplikasi pengolah kata, klien surel, kamus elektronik, atau mesin pencari web. Beberapa hal kegunaan Pemeriksa ejaan dasar yaitu :

1. memindai kata-kata pada suatu naskah dan mengekstraknya
2. membandingkan kata yang salah dengan memberikan pilihan kepada pengguna terhadap kata-kata yang diketahui ejaannya oleh pemeriksa ejaan tersebut. Ini mungkin hanya akan menampilkan beberapa daftar kata, atau juga mengandung beberapa informasi tambahan, seperti kata hubung serta atribut leksikal dan gramatikal.

Dalam beberapa kasus, pemeriksa ejaan memberikan saran kata yang salah, ini dikarenakan ketidakakuratan kata yang terdapat dalam program tersebut. Berikut adalah latar belakang dan solusi dari spell check yaitu :

1. kesalahan penulisan pada editor teks
2. Perlunya kontrol pengecekan ejaan yang salah
3. Pengecekan kesalahan secara manual akan menghabiskan banyak waktu
4. Membutuhkan sumber pasti sebagai acuan banhwa kata tersebut memang salah dalam penulisannya

5. Sistem pengecekan ejaan sangat diperlukan

Kesalahan pengetikan atau *Typographical Error* (*typo*) adalah praktek yang biasa terjadi kala membuat suatu tulisan. Namun, apa sajakan faktor-faktor yang dapat mempengaruhi kesalahan pada pengetikan, berikut faktor kesalahan pada pengetikan :

1. Letak huruf pada keyboard yang berdekatan
2. Kesalahan karena kegagalan mekanis atau slip tangan atau jari
3. Kesalahan karena ketidak sengajaan

D. Levenstein Distance

Levenstein Distance dibuat oleh Vladimir Levenstein pada tahun 1965. Levenstein Distance digunakan untuk mengoreksi kesalahan penulisan dengan menemukan kesalahan berdasarkan kemiripan penulisan antara teks yang tertulis dengan database program. Pengoreksian tersebut dilakukan melalui matriks yang digunakan untuk menghitung jumlah perbedaan string antara dua string. Perhitungan jarak antara dua string ditentukan dari jumlah minimum operasi perubahan untuk membuat string A menjadi string B. contohnya disaat salah mengetikan kata “sayt” sedangkan kata yang mirip dengan kata tesebut yaitu : “saya”, “sayat”, “sayap”, “sapa”, “sayur”.

Operasi yang dapat dilakukan pada Levenstein Distance yaitu :

1. Pengubahan Karakter

   Menulis ‘yang’ menjadi ‘yamg’, maka m ditukar dengan n

2. Penambahan Karakter

   String ‘kepad’ menjadi ‘kepada’, menambahkan ‘a’ di akhir string dan penambahan bisa di posisi mana saja

3. Penghapusan Karakter

Menghilangkan karakter ‘r’ pada kata ‘barur’

Sedangkan tahapan yang perlu dilakukan pada Levenstein Distance yaitu :

1. Menelusuri teks dan memisahkan teks per kata
2. Membandingkan kata-kata yang diperiksa dengan sebuah kamus
3. Apabila suatu kata tidak terdapat dalam kamus
4. Namun mirip dengan kata tertentu, dapat dikatakan kata tersebut merupakan kata yang salah

Selisih perbedaan antar string dapat diperoleh dengan memeriksa apakah suatu string sumber (s) sesuai dengan string target (t). Nilai selisih perbedaan ini disebut juga Edit Distance/jarak Levenshtein. Jarak Levenshtein antar string (s) dan string (t) tersebut adalah fungsi D yang memetakan (s,t) ke suatu bilangan real non-negatif

$$D(s,t) = d(s1,t1) + d(s2,t2) + d(s3,t3) + ... + d(sm,tn)$$

Dimana d(sj,ti) = 1 jika sj ≠ ti

d(sj,ti) = 0 jika sj = ti

berikut contoh string input dan string yang dibandingkan :

String sumber (s) = RONALDINHO

String target (t) = ROLANDO

Panjang string |s| = 10

Panjang string |t| = 7

D(s,t) = d(r,r) + d(o,o) + d(n,l) + d(a,a) + d(l,n) + ... + d(n,*) + d(h,*) + d(o,*)

D(s,t) = 0 + 0 + 1 + 0 + 1 + 0 + 1 +1 + 1 + 1 = 6

Artinya jarak perbedaan antara string tersebut adalah 6. Terdapat 6 operasi yang dilakukan untuk mengubah string sumber ke string target. 6 Operasi string yang dapat dilakukan pada string RONALDINHO menjadi ROLANDO yaitu :

1. Mensubstitusikan N dengan L

   RO**N**ALDINHO -> RO**L**ALDINHO

2. Mensubstitusikan L dengan N

   ROLA**L**DINHO -> ROLA**N**DINHO

3. Mensubstitusikan I dengan O

   ROLAND**I**NHO -> ROLAND**O**NHO

4. Menghapus O

   ROLANDONH**O** -> ROLANDONH

5. Menghapus H

   ROLANDON**H** -> ROLANDON

6. Menghapus N

   ROLANDO**N** -> ROLANDO

Untuk menghitung jaraknya tanpa menggunakan proses rekursif, digunakan matriks *(n + 1) × (m + 1)*. Dimana n adalah panjang *string* s1 dan m adalah panjang *string* s2. Algoritma ini berjalan mulai dari pojok kiri atas sebuah array dua dimensi yang telah diisi sejumlah karakter sring awal dan string target dan diberikan nilai *cost.* Elemen baris 1 kolom 1 (M[1,1]) adalah jumlah operasi yang diperlukan untuk mengubah *substring* dari kata ROLANDO yang diambil mulai dari *karakter awal sebanyak 1* (R) ke *substring* dari kata RONALDINHO yang diambil mulai dari *karakter awal sebanyak 1* (R). Elemen M[3,5] adalah jumlah operasi antara ROL (*substring* yang diambil mulai dari karakter awal sebanyak 3) dengan RONAL (*substring* yang diambil mulai dari karakter awal sebanyak 5). Berarti

elemen M[p,q] adalah jumlah operasi antara *substring kata pertama yang diambil mulai dari awal sebanyak p* dengan *substring kata kedua yang diambil dari awal sebanyak q*.

		R	**O**	**N**	**A**	**L**	**D**	**I**	**N**	**H**	**O**
	0	1	2	3	4	5	6	7	8	9	10
R	1	0	1	2	3	4	5	6	7	8	9
O	2	1	0	1	2	3	4	5	6	7	8
L	3	2	1	1	2	3	4	5	6	7	8
A	4	3	2	2	1	2	3	4	5	6	7
N	5	4	3	3	2	2	3	4	5	6	7
D	6	5	4	4	3	3	2	3	4	5	6
O	7	6	5	5	4	4	3	3	4	5	6

Elemen terakhir (yang paling kanan bawah) adalah elemen yang nilainya menyatakan jarak kedua *string* yang dibandingkan.

Implementasi Levenstein Distance yaitu dapat mencari seluruh kata pada kamus yang dijadikan sebagai pembanding dan koreksi kesalahan ejaan. Setiap kata yang memiliki perbedaan jarak (distance) minimum, akan mendapat pertimbangan solusi untuk saran perbaikan. Jadi setiap kata yang memiliki tingkat kemiripan yang tinggi (distance terendah) terhadap kata yang tidak sesuai dengan kamus bahasa Indonesia, maka akan menjadi saran utama dalam perbaikan.

Terdapat kesalahan lain nya seperti Kurang spasi, sehingga kata tidak memiliki arti. Misal tertulis : 'burungnuri', ika dicari dalam kamus Bahasa Indonesia tentu saja tidak memiliki arti. Cara kerja metode ini adalah mencoba semua kemungkinan. Memisahkan string menjadi beberapa kemungkinan kata kemudian dicocokkan ke dalam basis data. Penyelesaian yang dapat dilakukan yaitu sebagai berikut :

String 1	String 2
b	Urungnuri
Bu	Rungnuri
bur	Ungnuri
buru	Ngnuri
burun	Gnuri
burung	Nuri
burungn	Uri
burungnu	Ri
burungnur	is

E. PENERAPAN SPELL CHECK

1. IMPLEMENTASI ALGORITMA LEVENSHTEIN DISTANCE DAN METODE EMPIRIS UNTUK MENAMPILKAN SARAN PERBAIKAN KESALAHAN PENGETIKAN DOKUMEN BERBAHASA INDONESIA

Bahasa merupakan alat komunikasi lingual manusia baik secara lisan maupun tulisan. Peran Bahasa Indonesia yang baik dan benar sebagai bahasa resmi nasional memiliki arti yang sangat penting. Apalagi dalam penyusunan karya ilmiah. Hal ini disebabkan adanya data dan informasi yang ada didalamnya digunakan sebagai acuan untuk penelitian atau pengkajian selanjutnya bagi ilmuan lainnya. Hal itu tentu saja menimbulkan beberapa masalah seperti kesalahan pengetikan. Kontrol pegecekan bahasa yang baik dan benar jika dilakukan secara manual pastinya akan menghabiskan banyak waktu, apalagi dijaman serba modern seperti saat ini.

Kesalahan penulisan pada umumnya disebabkan kedekatan letak keyboard, slip jari, ataupun karena dua karakter yang letaknya tertukar. Untuk permasalahan itu muncullah ide untuk membuat Sistem Pengecekan Ejaan Bahasa Indonesia yang dikembangkan dengan berbasis web menggunakan bahasa pemrograman PHP dan menggunakan basis data kumpulan kata berbahasa Indonesia yang mengacu pada KBBI. Algoritma yang digunakan untuk memberikan saran perbaikan adalah Levenshtein Distance yang dapat menghitung keterkaitan antar string dan menghitung jumlah keterbedaan antar dua string. Disamping itu kesalahan penulisan juga dapat disebabkan kurangnya spasi antar kata sehingga kata tersebut tidak mengandung makna maka dengan menerapkan metode empiris diharapkan dapat menjadi jalan keluarnya.

1. PENDAHULUAN

Kesalahan pengetikan dokumen memang sering sekali terjadi. Apalagi belakangan ini kesadaran masyarakat untuk menuangkan idenya ke dalam artikel, jurnal ilmiah, tugas kuliah ataupun dokumen lainnya mengalami peningkatan. Tentu saja dalam penulisan dokumen tersebut adanya kesalahan pengetikan yang disebabkan oleh beberapa faktor seperti :

a. Letak huruf pada keyboard yang berdekatan,

b. Kesalahan karena kegagalan mekanis atau slip dari tangan atau jari,

c. Kesalahan yang disebabkan oleh ketidaksengajaan.

Proses pengecekan pengetikan dengan cara manual akan menghabiskan banyak waktu dan membutuhkan suatu sumber pasti sebagai acuan bahwa kata tersebut memang salah dalam proses penulisannya. Efisiensi waktu yang dibutuhkan jika dilakukan dengan manual tentunya tidak akan optimal dan cukup membosankan sehingga kemungkinan adanya human error dapat mengakibatkan proses pengecekan kata menjadi tidak optimal(Wardiana, 2002).

Beberapa algoritma pun dapat diimplementasikan dalam memberikan kata saran yang paling mendekati dari kata yang salah pengetikannya. Salah satunya algoritma Levenshtein Distance yang dapat menghitung jarak keterbedaan antara dua string (Andhika,2010). Selain itu ada juga kesalahan pengetikan karena kurangnya spasi sehingga kata tersebut tidak memiliki arti. Untuk permasalahan kata yang berdempetan tersebut digunakan metode empiris yaitu dengan cara memecah kata menjadi dua bagian dan mencocokkan kata ke dalam basis data dan memberikan saran kemungkinan adanya kata yang diketik tanpa spasi.

2. ANALISIS KINERJA ALGORITMA LEVENSHTEIN DISTANCE DALAM MENDETEKSI KEMIRIPAN DOKUMEN TEKS

Akhir-akhir ini, plagiarisme pada dokumen teks merupakan salah satu permasalahan akademik yang semakin meningkat. Secara umum, tindak plagiarisme dilakukan dengan mengubah struktur kata dalam kalimat dan struktur kalimat dalam paragraf, serta melakukan penyisipan, penghapusan, atau penggantian kata. Pada penelitian ini, dibangun 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. Simulasi algoritma ini dilakukan terhadap dua data set dan satu data real. Pada data set 1 dilakukan pengubahan struktur kata dalam kalimat, pada data set 2 pengubahan struktur kalimat dalam paragraf, dan pada data real tidak dilakukan pengubahan apapun karena data real merupakan dokumen abstrak dari jurnal penelitian. Hasil simulasi menunjukkan bahwa penggunaan sorting sangat berpengaruh bagi algoritma Levenshtein distance. Hasil terbaik pada data set 1 ditunjukan pada proses yang menggunakan stopword removal, stemming, dan sorting sekaligus. Hasil terbaik pada data set 2 diperlihatkan pada proses yang menggunakan stopword dan stemming yang

digabungkan dengan sorting. Hasil terbaik pada data real diperlihatkan pada proses stemming-sorting. Kata kunci: plagiarisme, Levenshtein distance, Stopword removal, Stemming, sorting.

PENDAHULUAN

Teknologi dalam bidang informasi telah banyak dimanfaatkan kelebihannya untuk mempermudah suatu pekerjaan menjadi lebih efektif dan lebih cepat. Namun, hal ini tidak serta-merta memberikan dampak positif. Salah satu wujud nyata dampak negatif dari perkembangan teknologi adalah terjadinya tindakan mengambil sebagian atau seluruh ide seseorang berupa teks dokumen tanpa mencantumkan sumber pengambilan, yang sering disebut dengan plagiarisme. Plagiarisme adalah tindakan penyalahguanaan, pencurian/perampasan, penerbitan, pernyataan, atau menyatakan sebagai milik sendiri sebuah pikiran, ide, tulisan atau ciptaan yang sebenarnya milik orang lain.

Tindak plagiarisme secara perlahan dapat ditekan dan untuk mendukung penekanan tindak plagiarisme ini, maka dibutuhkan adanya sistem yang memudahkan dalam mendeteksi dan mengukur kemiripan dokumen. Selain dapat mengetahui tindak plagiarisme, pengukuran kemiripan dokumen ini dapat membantu dalam pengelompokan dokumen. Sebagian besar kasus plagiarisme ditemukan dibidang akademisi, berupa essai, jurnal, laporan penelitian, dan sebagainya. Deteksi plagiarisme dilakukan dengan membandingkan sebuah dokumen dengan dokumen lainnya. Tingkat kesamaan dokumen tersebut akan menjadi dasar pendeteksian plagiarisme. Terdapat tiga metode dalam pendeteksian plagiarisme, yaitu metode pencarian kata kunci, perbandingan teks lengkap dan dokumen fingerprinting. Pada metode pencarian kata kunci, metode ini mencari kesamaan kata-kata yang sering muncul dalam suatu dokumen dan kemudian akan dibandingkan dengan kata-kata yang sering muncul pada dokumen lain. Sedangkan pada

metode perbandingan teks lengkap adalah dengan membandingkan seluruh isi teks. Pada metode dokumen fingerprinting yaitu dengan mengubah struktur kalimat menjadi sebuah angka-angka yang kemudian dibandingkan nilainya dengan dokumen lain yang sudah diubah juga ke dalam bentuk angka-angka.

Beberapa penelitian telah dilakukan untuk mendeteksi plagiarisme. Pandawa [6] menggunakan algoritma Winowwing untuk mendeteksi kemiripan isi dokumen teks. Algoritma tersebut menggunakan metode fingerprinting. Nafik [5] menggunakan algoritma Levenshtein distance untuk mencari nilai edit distance untuk penilaian jawaban esai. Pada penelitian tersebut dilakukan uji coba untuk melihat kemampuan membandingkan kemiripan jawaban esai dengan kunci jawaban, kemudian dilihat pengaruh stemming terhadap perubahan presentase yang dihasilkan dan hasilnya dibandingkan dengan hasil penilaian secara manual.

Pada penelitian ini, penulis merancang sebuah aplikasi deteksi kemiripan dokumen berbasis web yang dibuat untuk membandingkan satu dokumen dengan dokumen lain. Perbedaan penelitian ini dengan penelitian sebelumnya adalah uji coba menggunakan algoritma Leveshtein distance preprocessing untuk mendeteksi kemiripan dokumen teks menggunakan metode pencarian kata kunci dengan menambahkan proses converting, tokenizing, stopword removal, stemming dan sorting. Kemudian hasilnya dibandingkan dengan algoritma Levenshtein distance standard, yaitu tanpa proses stopword removal, stemming dan sorting, penggunaan database untuk menyimpan kata berimbuhan atau term beserta kata dasarnya pada tabel stem, dan menyimpan kata-kata yang sering muncul pada tabel stop.

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

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.

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].

BAB 7 PARSING

A. Pengertian

Parser, yaitu program yang melakukan proses parsing. Parsing, yaitu proses mengambil kalimat input bahasa alami dan menguraikannya menjadi beberapa kata dan beberapa bagian gramatikal (kata benda, kata kerja, kata sifat, dan lain-lain). *Constituent*, yaitu unsur-unsur pembentuk kalimat yang dapat berdiri sendiri, contohnya *noun phrase*, *verb phrase* dan sebagainya. CFG, yaitu kaidah tata bahasa yang terdiri atas 2 bagian dimana bagian kirinya hanya terdiri atas satu simbol non terminal.

B. Objek Parsing

1. Morphological parsing:

   Menganalisis kata menjadi morfem dan affixes

2. Phonological parsing

   Menganalisis suara menjadi kata dan frasa

3. Syntactic parsing:

   Mengidentifikasi bagian-bagian komponen dan kaitannya

   Untuk mengetahui apakah suatu kalimat gramatikal

E. Jenis Parsing atau Analisis Struktur Kalimat

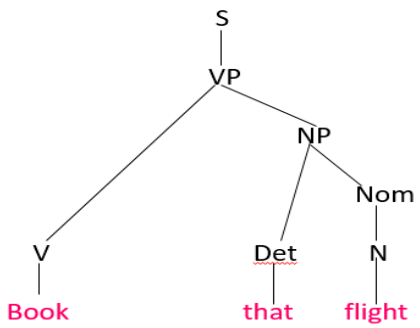
1. Top Down

Pembentukan non terminal symbol pada sebelah kiri dari kaidah CFG yang kemudian ditransformasikan ke bagian kanan terus menerus sampai ditemukan terminal symbol. Menguraikan sebuah kalimat mulai dari constituent yang terbesar sampai menjadi constituent yang terkecil. Hal ini dilakukan terus-menerus sampai semua komponen yang dihasilkan ialah constituent terkecil dalam kalimat, yaitu kata. Hasil analisis kalimat membentuk struktur pohon yang disebut Parse Tree. Left Recursive Harus dihindari. Mengakibatkan penelusuran yang tidak kunjung selesai

CFG for Fragment of English

S → NP VP	VP → V
S → Aux NP VP	PP -> Prep NP
S → VP	N → book | flight | meal | money
NP → Det Nom	V → book | include | prefer
NP →PropN	Aux → does
Nom → N Nom	Prep →from | to | on
Nom → N	PropN → Houston | TWA
Nom → Nom PP	Det → that | this | a
VP → V NP	

Pohon frase untuk kalimat “Book that flight”



Rule Expansion

S → NP VP	VP → V
S → Aux NP VP	PP -> Prep NP
S → VP (1)	N → book | flight | meal | money
NP → Det Nom (3)	V → book | include | prefer
NP →PropN	Aux → does
Nom → N Nom	Prep →from | to | on
Nom → N (4)	PropN → Houston | TWA
Nom → Nom PP	Det → that | this | a
VP → V NP (2)	

2. Bottom Up

Dimulai dari terminal simbol atau Bagian kanan dari kaidah CFG ditransformasikan ke bagian kiri. Bottom-up parser bekerja dengan cara mengambil satu demi satu kata dari kalimat yang diberikan, untuk dirangkaikan menjadi constituent yang lebih besar. Hal ini dilakukan terus-menerus sampai constituent yang terbentuk ialah sentence atau kalimat. Dengan demikian metode bottom-up bekerja dengan cara yang terbalik dari top-down. Left Corner

Secara garis besar adalah bottom up tetapi diadakan suatu prediksi yang dilakukan dari atas ke bawah atau perpaduan.

BAB 7 KEAMBIGUAN

Merupakan makna ganda atau membingungkan. Banyak ambigu yang di temukan pada bahasa alami. Sedangkan dalam sistem komputer tidak diperkenankan makna ganda sehingga perlu dijabarkan secara rinci informasi mengenai unsur untuk memudahkan analisis.

A. Jenis Keambiguan

1. Keambiguan Leksikal

Menyangkut “Kelas Kata” (Parts of Speech)

Apa bedanya?

The water is boiling [1]

They water the plant twice a day [2]

Solusi

[1] Nomina

[2] Verba

Apa bedanya?

Peristiwa genting [4]

Atap genting [5]

Solusi

[4] Adjectiva

[5] Nomina

2. Keambiguan Struktural

The girl put the book on the table [6]

The girl saw the book on the table [7]

Pendaftaran anggota baru akan kami lakukan minggu depan [8]

[6] 'on the table' terkait dengan 'put'

[7] 'on the table' terkait dengan 'the book'

[8] baru sebagai adjectiva yang terkait dengan anggota dan dapat berupa adverb yang terkait dengan 'minggu depan'

Bagaimana makna kalimat ini? "Young men and women must participate"

Keambiguan struktural teridentifikasi dengan lebih dari 1 pohon parsing yang dihasilkan

CFG :

NP → A NP

NP → NP C NP

NP → N

A → {young}

N → {men, women}

C → {and}

Apa maknanya? They are visiting professors

Makna 1 : Mereka adalah guru besar yang berkunjung

Makna 2 : Mereka sedang mengunjungi guru besar

CFG :

S → NP VP

VP → V NP

NP → N

NP → A N

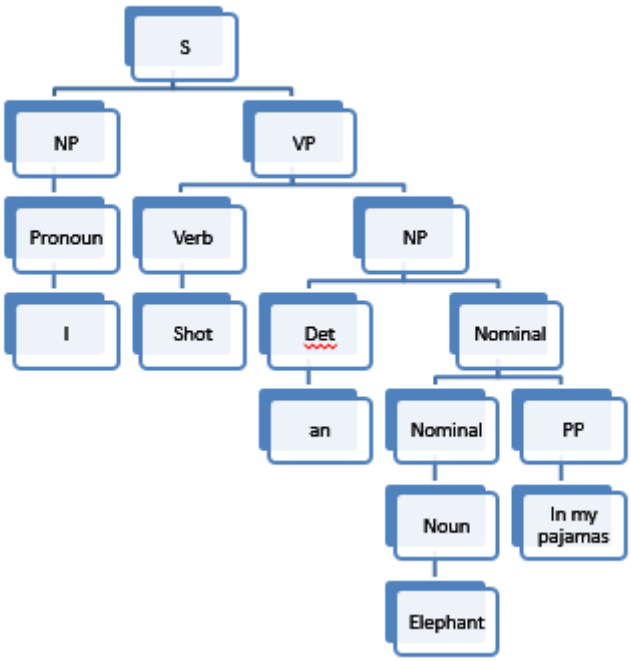
V → Aux V

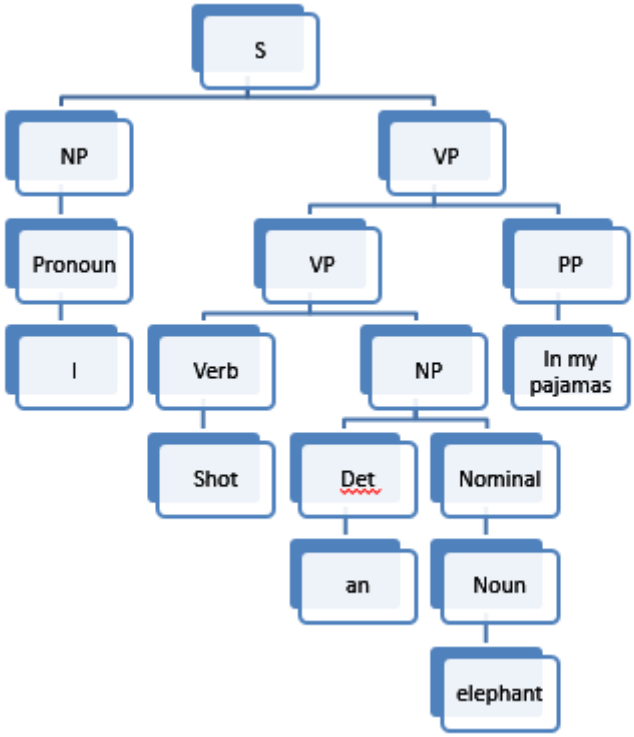
V → Aux

N → {they, professors}

V → {are, visiting}

Aux → {are}





B. Faktor-Faktor Penyebab Keambiguan

Adapun beberapa faktor penyebab munculnya keambiguan yaitu diantaranya sebagai berikut :

1. Faktor Morfologi

Faktor Morfologi merupakan salah satu faktor Keambiguan yang muncul diakibatkan dari pembentukan kata itu sendiri.

Contoh :

Permen Karet itu tertelan olehku

Permen itu tidak sengaja tertelan, atau

Permen itu akhirnya dapat ditelan.

b. **Faktor Sintaksis**

Faktor Sintaksis ini dapat terjadi karena terdapat sebuah susunan kata di dalam suatu kalimat yang masih kurang jelas.

Contoh :

Gigit jari

Ayu hanya dapat *gigit jari* melihat barang yang ia inginkan tidak bisa menjadi miliknya.

Aira tidak sengaja *menggigit jarinya* hingga keluar darah.

Kata dari *gigit jari* diatas mempunyai dua makna yaitu sebuah keputus asaan atau sungguh menggigit jarinya.

c. **Faktor Struktural**

Faktor Struktural adalah merupakan salah satu faktor yang menyebabkan keambiguitasan yang disebabkan dari struktur kalimat itu sendiri.

Contoh :

Pembacaan puisi baru.

Pembacaan Puisi Baru, akan dilaksanakan pada hari sabtu "yang dibaca adalah puisi baru".

Pembacaan Puisi, baru akan dilaksanakan pada hari sabtu "yang dibaca hari sabtu dalah puisi".

C. Pemecahan Keambiguan

1. Permasalahan bagi pendengar, penerjemah atau penganalisis suatu teks

2. Dapat mengaburkan pengertian
3. Memperhitungkan konteks kebahasaan dan situasi pragmatik
4. Pra-suntingan dan Pasca Suntingan

Contoh :

a. Kata “kali”

Keambiguan Leksikal

1) Sungai
2) Frekuensi
3) Suatu perlakuan matematik

Bukan tiga kalis, ……. [‘sungai’]

Bukan tiga kalif, ……. [‘frekuensi’]

Bukan tiga kalim, …… [perlakuan matematik]

Solusi

Menandai makna kata dengan suatu symbol dengan subskrip

Keambiguan Leksikal

b. It was painted by_1 the new artist

It was painted by_2 the new techniques

Jika by diikuti oleh kata induk yang mempunyai ciri semantik manusia, maka by berarti ‘oleh’

Jika by diikuti frase yang berkata induk tak bernyawa, maka by berarti ‘dengan’

c. That’s a green house

Keambiguan Leksikal

Rumah yang berwarna hijau (a)

Rumah pemeliharaan tanaman (b)

Solusi

Frase dengan makna (a) ditulis terpisah

Frase dengan makna (b) ditulis rapat

d. **Anisa membaca buku sejarah Indonesia baru.**

Kalimat diatas dapat menimbulkan pertanyaan apakah bukunya yang baru, atau sejarahnya yang baru. Untuk menghindari keambiguan pada kalimat tersebut maka cara penulisannya adalah sebagai berikut :

Anisa membaca buku-sejarah-Indonesia yang baru "jika menjelaskan bukunya yang baru".

Anisa membaca buku tentang sejarah-Indonesia yang baru "jika menjelaskan sejarahnya yang baru".

e. **Mobil Mentri baru sedang diperbaiki di bengkel.**

Kalimat diatas masih belum jelas apakah mobilnya yang baru atau Mentrinya yang baru. Untuk menghindari keambiguan pada kalimat tersebut maka cara penulisannya adalah sebagai berikut :

Mobil, Mentri yang baru sedang diperbaiki di bengkel "jika menjelaskan mobilnya yang baru".

Mobil Mentri, baru itu sedang diperbaiki di bengkel “jika menjelaskan pegawainya yang baru”.

f. Sumbangan ke dua sekolah tersebut sudah dikirimkan.

Kalimat tersebut juga menimbulkan keambiguitasan yaitu, apakah itu merupakan sumbangan yang ke dua kalinya, dan sumbangan yang diberikan dari dua sekolah yang berbeda ataupun sumbangan kepada dua sekolah yang berbeda.

Oleh sebab itu untuk menghindari keambiguitasan terhadap kalimat tersebut maka cara penulisannya sebagai berikut :

Sumbangan yang kedua kalinya dari Sekolah tersebut sudah dikirimkan “jika menjlaskan sumbangannya yang kedua kali”.

Sumbangan untuk dua Sekolah tersebut sudah dikirimkan “jika menjelaskan sumbangan tersebut untuk dua Sekolah”.

Sumbangan dari kedua-sekolah tersebut sudah dikirimkan “jika menjelaskan dua Sekolah yang menyumbang”.

D. PENERAPAN

1. ALGORITMA PENENTUAN KELAS KATA PADA KATA-KATA YANG MEMILIKI SIFAT AMBIGU DALAM PART-OF-SPEECH TAGGING BAHASA INDONESIA DENGAN METODE RULE BASED

Bahasa Indonesia memegang peran fundamental dalam komunikasi masyarakat indonesia. Namun, ada permasalahan yang muncul Dalam penggunaan Bahasa Indonesia, yakni tentang ambiguitas. Ambiguitas adalah sifat atau hal yang memiliki pengertian atau bermakna lebih dari satu. dalam kajian Natural Language Processing, Part of Speech (POS) tagging memegang peran dalam penentuan label kelas kata. Penelitian ini bertujuan untuk mengetahui algoritma penentuan kelas kata pada kata-kata yang memiliki sifat ambigu dalam POS tagging bahasa Indonesia dengan metode rule based. Rule based merupakan salah satu metode dalam sistem pakar yang menggunakan aturan if-then untuk menyajikan pengetahuannya.

Penelitian ini tergolong ke dalam penelitian dasar dengan desain riset eksperimental, dengan tahapan-tahapan sebagai berikut: 1) inventarisasi pengetahuan, 2) penyusunan algoritma, 3) implementasi, 4) Pengujian, 5) Analisis, 6) Penarikan kesimpulan, dan 7) Penyusunan laporan. Penelitian ini menggunakan kerangka eksplanatori dengan metode analitis. Eksplanatori adalah kerangka berpikir yang menjelaskan sifat-sifat objek yang diteliti. Metode analisis yang digunakan dalam penelitian ini adalah analisis deskriptif, yakni dengan menggambarkan kenyataan situasi atau peristiwa dengan runtut dan sistematis.

Hasil dari penelitan ini adalah sebuah algoritma penentuan kelas kata yang dihasilkan dari beberapa tahapan sebagai berikut: memahami pola kalimat bahasa Indonesia, menentukan strategi penalaran, menghimpun kata-kata ambigu, membuat pola keambiguitasan menyusun aturan, berkonsultasi dengan pakar, dan fiksasi algoritma. Alur dalam algoritma tersebut meliputi: 1) input kalimat, 2) segmentasi kalimat menjadi kata, 3) pengecekan keambiguitasan, 4) jika tidak ambigu, maka dilakukan pelabelan kata, 6) jika ambigu kata diuji dengan aturan- aturan sintaksis, 7) tampilkan hasil

pelabelan kelas kata. Algoritma yang telah disusun diimplementasikan pada aplikasi POS tagging bahasa Indonesia yang dikembangkan oleh kelompok belajar NLP UIN Sunan kalijaga. Aplikasi ini dikembangkan dengan bahasa pemrograman PHP dan DBMS mySQL. Hasil pengujian menunjukkan bahwa implementasi algoritma ini berhasil memberikan pelabelan 92 dari 100 kata yang diujikan (92%). Hasil implementasi dipengaruhi oleh ketersediaan aturan dalam menangani keambiguitasan kelas kata dalam berbagai bentuk, seperti algoritma masih belum bisa menangani dua kata ambigu yang berdampingan dan kata ambigu berada di awal kalimat.

Kata Kunci: Algoritma, Ambigu, POS tagging, Rule based.

2. WORD SENSE DISAMBIGUATION DENGAN ALGORITMA LESK (SIMPLIFIED LESK)

Ambiguitas makna pada kalimat bahasa inggris adalah gejala terjadinya tafsiran lebih dari satu makna. Hal ini dapat terjadi baik dalam ujaran lisan maupun tulisan. Tafsiran lebih dari satu ini dapat menimbulkan keraguan dan kebingungan dalam mengambil keputusan tentang makna yang dimaksud. Untuk menaggulangi permasalahan ini digunakan metode pencarian makna terhadap kata ambigu pada kalimat bahasa inggris dengan menggunakan algoritma simplified lesk. Algoritma ini digunakan karena kesederhanaannya dalam menentukan makna ambigu, yaitu menghitung jumlah kesamaan kata antara konteks kalimat dengan gloss (definisi) makna kata ambigu beserta examples (contoh) kalimat yang diambil dari WordNet. Algoritma ini digunakan untuk menentukan makna kata ambigu pada masing-masing kalimat yang terdapat pada sample dataset yang telah disediakan sebanyak 100 kalimat bahasa inggris. Kemudian evaluasi dilakukan dengan tujuan mencari precision, recall, f-measure dan

accuracy dari algoritma simplified lesk. Dari hasil evaluasi didapatkan precision dari algoritma simplified lesk adalah 46,5%, recall 40%, f-measure 43% dan accuracy 47%.

Kata Kunci : Bahasa Inggris, Ambiguitas, Simplified Lesk, WordNet

DAFTAR PUSTAKA

Manning, Christopher D and Hinrich Schiitze. 1999. Foundation of Statistical Natural Language Processing. MIT Press Cambridge, Massasuchet, London, England.

Jurafsky, Daniel and James H Martin. 2000. Speech and Language Processing, Prentice Hall.


LABORATORIUM
TEKNIK INFORMATIKA
UNIVERSITAS AHMAD DAHLAN


UNIVERSITAS AHMAD DAHLAN