Nama : Rini Jannati
NPM : 1706005905

Tugas 2

1. Algoritma Stemming

Pada korpus terdapat kata yang berulang, kata yang berimbuhan dengan partikel(“kah”, “lah”, “tah”, “pun”), kepunyaan(“ku”, “mu”, “nya”), prefiks(“me”, “per”, “di”, “ke”, “ber”, “ter”, “se”), sufiks(“i”, “an”, “kan”) dan sisipan(“el”, “er”, “em”, “in”, “ah”). Berikut adalah aturan algoritma stemming yang telah saya buat.

a. Kata yang akan distemming diperiksa terlebih dahulu apakah kata tersebut mengandung tanda “-” sebagai penanda kata tersebut adalah kata berulang. Jika ya, maka kata tersebut dipisah dan diambil kata pertamanya, kata kedua diabaikan. Jika tidak, maka kata tersebut tetap.
b. Setelah itu, kata tersebut diperiksa imbuhannya.

Dengan menganut prinsip pemakaian imbuhan yang ada paper Adriani et all (2007).

[[[DP+]DP+]DP+] root-word [[+DS][+PP][+P]]

Proses yang pertama kali diperiksa adalah apakah kata tersebut mengandung imbuhan pertikel sebagai imbuhan terluar dari pembentukan kata berimbuhan. Jika mengandung partikel diikuti kepunyaan maka kata tersebut dipotong sesuai dengan jumlah huruf imbuhannya

```
	my @partikel = ("kah","lah","tah","pun","ku","mu","nya");
	foreach my $i (0..@partikel-1){
		if(length($inp2)<7		&&		grep(/^[dmpbkst][ie]/,$inp2)&&
grep(/$partikel[$i]$/,$inp2)){
				$inp2 = $inp2;
		}elsif( length($inp2)>5 && grep(/$partikel[$i]$/,$inp2)){
			my $l = length($partikel[$i]);
			$inp2 = substr($inp2,0,length($inp2)-$l);
			#print"$inp2\ttrue\n";
		}else{
			#print"$inp2\tfalse\n";
			$inp2 = $inp2;
		}
	}
```

Perulangan tersebut akan mengecek apakah kata tersebut mengandung imbuhan tersebut (dapat dilihat pada inisialisasi array @partikel). Kata yang dicek pertama adalah partikel lalu diikuti kepunyaan. Dengan kondisi kata terseput harus lebih dari 5. Karena setelah diteliti, kemungkinan jumlah huruf yang mengandung partikel tersebut berjumlah minimal 6 seperti: apakah, bukuku, sehingga kata celah tidak akan dipotong. Masalah akan terjadi bila kata tersebut memiliki prefiks dan partikel sehingga kata tersebut akan memiliki jumlah huruf lebih dari 5 tetapi kemungkinan kurang dari 7 seperti diolah. Jadi saya membuat kondisi untuk memeriksa jumlah huruf, terkandung imbuhan atau tidak agar kata tsb tidak dilakukan sembarangan pemotongan.

Proses selanjutnya adalah pemeriksaan untuk kata imbuhannya. Saya mengkodingnya dengan mengecek kondisi satu per satu kepemilikan imbuhan. Hal ini dikarenakan ada kondisi-kondisi imbuhan tersebut hanya memiliki sufiks apa saja, kemungkinan adanya

peluruhan dan aturan pemakaian imbuhan. Ada beberapa kondisi untuk prefiks dan sufiks yang tidak diizinkan berpasangan yaitu:

1. be- -i
2. di- -an
3. ke- -i, -kan
4. me- -an
5. se- -i,-kan
6. te- -an

Maka pada program saya membuat kondisi untuk imbuhan

1. di- dan me- hanya bisamenggunakan imbuhan - kan dan -i
2. pe- hanya bisa menggunakan imbuhan -an
3. be- hanya bisa menggunakan imbuhan -an dan -kan
4. ke- hanya bisa menggunakan imbuhan -i dan -an
5. tidak ada imbuhan untuk se dan ter

Untuk imbuhan di- ada aturan pemotongan yang saya buat yaitu:

1. Untuk kata awalan di- yang diikuti kata di- lagi dilakukan pemotongan imbuhan jika banyak huruf tersebut lebih dari 5, sehingga kata didik tidak menjadi dik.
2. Jika imbuhan di- diikuti huruf n dan kata konsonan kecuali y, maka kata tersebut tidak boleh dipotong. Karena tidak ada kata bahasa indonesia yang diawali dengan kata n yang diikuti konsonan selain y.
3. dilakukan pemotongan lagi jika mengandung kata imbuhan lagi seperti ke-, per- dan ber-, dengan kondisi ke diikuti kata konsonan lalu vokal.

Masalah muncul untuk imbuhan di ketika:

dialog-dialog	alog
dibantah	ban
diberangkatkan	angkat
dikelilingi	liling
dikelola	lola
dikelompokkan	lompok
dikeluarkannya	luar
dikenali	nali
dikenang	nang
dikepung	pung
dikeroyok	royok
diketuai	tua
dimensinya	mensi

Untuk Imbuhan me- ada aturan pemotongan yang saya buat yaitu:

1. Jika me- diikuti huruf *l, n, q, r, w* lalu diikuti dengan huruf vokal, maka hanya me- yang dihapus

2. Jika me- diikuti huruf *m* lalu diikuti huruf *b*, *f*,, *v* maka yang dihapus adalah huruf mem-
3. Jika me- diikuti huruf *n* lalu diikuti huruf *c*, *d*, *j*, *t* maka yang dihapus adalah huruf men-
4. Jika me- diikuti huruf *ng* lalu diikuti huruf konsonan maka yang dihapus adalah huruf meng-
5. Jika me- diikuti huruf *m* lalu diikuti huruf vokal maka imbuhan huruf mem- diganti jadi p
6. Jika me- diikuti huruf *n* lalu diikuti huruf vokal maka imbuhan huruf men- diganti jadi t
7. Jika me- diikuti huruf *ny* lalu diikuti huruf vokal maka imbuhan huruf meny- diganti jadi s
8. Jika me- diikuti huruf *ng* lalu diikuti huruf vokal maka imbuhan huruf meng- diganti jadi k

Pada imbuhan me- kata juga bisa ditambah oleh imbuhan lain seperti memper-, member- dan memer-

Masalah terjadi jika imbuhan me- ini pada peluruhan untuk huruf p seperti

memaafkan	paaf
memadai	padai
memadu	padu
memainkan	pain
memaknai	pakna
memanfaatkan	panfaat
memanipulasi	panipulas
memanusiakan	panusia
memasak	pasak
memasukkan	pasuk
memasyarakatkan	pasyarakat
mematikannya	pati

Kasus imbuhan pe- hampir sama dengan imbuhan me-, namun pe hanya bisa digabung dengan pember-.

Untuk imbuhan ber- ada aturan pemotongan yang saya buat, yaitu:
1. Pada program saya melakukan hardcoding khusus untuk kata belajar karena dari beberapa artikel web dan buku referensi bahasa indonesia yang saya baca hanya belajarlah yang memiliki imbuhan bel+ajar.
2. Jika imbuhan ber- diikuti oleh huruf vokal, maka hanya be- saja yang dihapus. Namun, ada kasus jika ber- yang diikuti huruf vokal lalu diikuti huruf *r*, *b*, *d*, *l*, maka huruf ber- dihapus.
3. Jika imbuhan ber- diikuti oleh huruf konsonan, maka huruf ber- dihapus.
4. Pada imbuhan ber- prefiks yang mengikutinya bisa menjadi berke-, berpe- dengan aturan imbuhan berke- dapat dipotong jika diikuti huruf konsonan kecuali n, karena jika diperhatikan jarang ada kata yang dapat ditambahkan imbuhan berke pada huruf berawalan n, dan m yang diikuti oleh konsonan sedangkan imbuhan

pen- jika diikuti konsonan maka imbuhan berpen dapat dihapus jika diikuti vokal maka diganti jadi t.

Masalah terjadi jika imbuhan ber ini diikuti huruf vokal seperti:

beragama	ragama
berakar	rakar
beralasan	alas
beranak	ranak
beraneka	raneka
berantakan	antak
berasal	rasal
berasosiasi	rasosiasi
berasumsi	rasumsi
beratkan	atkan
beraturan	ratur
berawal	rawal
berendam	endam
bereskan	eskan
berinisiatif	rinisiatif
berisikan	risik
berita-berita	rita
berjumlah	jum
berkas-berkas	kas
berkeliling	liling
berkelompok	lompok
berlaku	berla
beroperasinya	roperasi
berpedoman	pedom
berpengaruh	aruh
bersihnya	sih
bertanya-tanya	berta
bertemu	berte
bertingkah	ting
beruji	ruji
berukuran	rukur
berupaya	rupaya
berusia	rusia

Untuk imbuhan ke-, ter- dan se- tidak ada aturan khusus.

Untuk imbuhan sufiks saya membuat aturan untuk menghilangkan imbuhan -i dan -an

Pada imbuhan -an, saya hanya mengizinkan pemotongan ketika banyak kata lebih dari 5 untuk menghindari kata makan, jaman.
Pada imbuhan -kan, saya hanya memeriksa jika setelah imbuhan -an dipotong maka saya memeriksa huruf k- nya karena ada kemungkinan kata ajakan bisa dipotong menjadi aja. Jadi ketika sebelum huruf k adalah huruf konsonan, maka huruf k dihapus. Namun, masalah terjadi ketika muncul kata-kata ini:

adegan	adeg
agendakan	agendak
ajukan	ajuk
berisikan	risik
buktikan	buktik
demikianlah	demiki
gandakan	gandak
gunakan	gunak
halamannya	halam
matikan	matik

Pada imbuhan -i, saya banyak membuat aturan karena banyak kata dasar yang berakhiran i tanpa harus ditambahkan sufiks -i.
1. Kata harus lebih dari 4 (tidak berlaku untuk cuci, maki, laki, gaji dll)
2. Ketika kata tersebut diawali dengan konsonan, lalu sebelum huruf i ada 2 huruf vokal dengan jumlah katanya ganjil maka huruf akhiran -i dibuang. (pakai, ramai dll)
3. Ketika kata tersebut lebih dari 5 yang diawali dengan konsonan kecuali *d* dan *m* lalu diakhiri vokal-konsonan dan -i, maka sufiks dihapus. (untuk kata seperti fasilitas, kenali dll)
4. Ketika jumlah kata lebih dari 5 tetapi ada kata awalan vokal yang memiliki akhiran huruf s dan akhiran i, maka sufiks tersebut tidak dihapus (contoh kata instruksi, asosiasi, asumsi. Tetapi menghindari kata seperti awasi, atasi agar sufiks dihapus).
5. ketika ada awalan konsonan tetapi berakhiran si agar tidak dihapus -i nya (untuk kata definisi dll).
6. ketika sebelum akhiran -i terdapat 2 huruf konsonan, huruf -i tidak boleh dihapus.
7. ketika ada kata yang diawali huruf konsonan, tetapi diakhiri dengan vokal-konsonan-i, maka tidak dihapus.
8. selain itu dihapus.
Masalah terjadi ketika

dikenai	kenai
diketuai	tuai
dilaluinya	lalui
edisinya	edis
emisinya	emis
memadai	padai
materinya	mater
melukai	lukai

Setelah melakukan pengecekan imbuhan selanjutnya adalah pengecekan sisipan. Pada sisipan saya membuat aturan jika diawal kata adalah konsonal alu diikuti dengan *el*, *er*, *em*, *in*, *ah* lalu diikuti oleh huruf vokal, maka kata peluruhan tersebut dihapus.
Dari aturan stem yang telah saya buat, untuk ukuran 1000 kata dengan model kata-kata tersebut maka akurasi yang dapat dihasilkan adalah 92,1%. Algoritma dapat lihat pada file T2_1706005905_RiniJannati_Stemming.pl. Hasil stemming dapat dilihat pada file hasil-stemming.csv.

2. Algoritma Soundex

Saya membuat sebuah program yang dapat menginput kata untuk menguji algoritma soundex yang telah saya buat. Alur program sama seperti algortima yang diberikan. Namun pada prosedur ketika ada angka yang berurutan, saya menggantinya dengan angka “0” lalu seluruh angka “0” dihapus. Penulisan program dapat dilihat pada file T2_1706005905_RiniJannati_Soundex.pl

hasil:

```
>> Inputkan sebuah kata: Jakarta
J263
 >> Inputkan sebuah kata: herman
H655
>> Inputkan sebuah kata: adinda
A353
>> Inputkan sebuah kata: bank
B520
>> Inputkan sebuah kata: raya
R000
```

3. Inverted Index

Pertama yang saya lakukan saat mengkoding soal no.3, yang saya lakukan adalah menyimpan isi file tersebut dalam array. Array menyimpan data baris demi baris. Lalu saya menggabungkan array tersebut dalam sebuah scalar. Lalu scalar tersebut saya pisah untuk diambil datanya perdokumen dengan mensplit skalar tersebut dengan “</doc>” sehingga isi array tersebut adalah data per dokumen. Saya juga melakukan token seluruh kata untuk seluruh dokumen dan saya lakukan stemming. Token seluruh kata ini berguna untuk melakukan pengindexan.

Setelah itu dilakukan looping per dokumen untuk menoken kata dan melakukan pengindexan. Pengindexan saya lakukan dengan array dua dimensi dengan
indeks[term][dokumen]

Hasil pengindexan dokumen saya simpan dalam hasil-index.csv yang berisikan token kata dan jumlah kata perdokumen.

Untuk melihat listing file per token dapat dilihat pada file list-index.txt yang berisikan file seperti yang diminta pada soal.

Sayangnya model koding yang saya buat memiliki kompleksitas waktu lama. Kira-kira prosesnya sekitar 10 menit untuk mendapatkan hasil ini.

Pada boolean retrieval saya memanfaatkan index yang telah dihasilkan. Pertama kali yang saya lakukan untuk boolean retrieval adalah mengecek kata tersebut ada pada token, jika tidak ada maka tidak ada dokumen yang mengandung kata tsb, jika ada maka nomor dokumen disimpan di dalam array. Setelah itu jika yang diminta adalah AND, maka array kedua kata tersebut saya periksa untuk mencari nomor dokumen yang sama, jika ada maka nomor dokumen ditampilkan, jika tidak ada maka memberikan pesan bahwa tidak ada yang mengandung kedua query. Berbeda dengan OR, isi index dokumen saya gabungkan lalu saya tampilkan nomor dokumen tersebut (nomor dokumen tidak berulang).

Hasil pencarian query:

contoh AND:

```
>> Pencarian QUERY
Anda dapat mencari dokumen dengan 2 kata sebagai kata kunci
Gunakan penghubung antara 2 kata tersebut dengan AND atau OR
contoh:
1. Kata1 AND kata2
2. Kata1 OR Kata2
Silahkan masukkan QUERY: jakarta and gedung
Query jakarta and gedung ada pada dokumen ke:
105|132|158|178|191|196|24|245|248|257|303|305|34|346|368|374|389|
392|405|420|421|43|5|52|68|70|85|
```

contoh OR:

```
Pencarian QUERY
Anda dapat mencari dokumen dengan 2 kata sebagai kata kunci
Gunakan penghubung antara 2 kata tersebut dengan AND atau OR
contoh:
1. Kata1 AND kata2
2. Kata1 OR Kata2
Silahkan masukkan QUERY: jakarta or gedung
Query jakarta or gedung ada pada dokumen ke:
1|105|106|11|110|112|115|116|117|121|132|139|14|141|142|143|144|14
5|147|152|153|154|156|158|16|160|161|162|164|167|168|174|176|178|1
79|180|181|182|183|184|185|187|189|191|196|2|201|204|205|206|207|2
12|216|217|22|220|221|226|238|24|242|244|245|247|248|254|257|258|2
7|274|275|276|278|280|282|286|288|289|29|293|294|295|296|298|299|3
|301|303|305|313|314|315|316|317|320|321|322|323|328|337|34|340|34
1|344|345|346|347|35|350|352|353|357|358|359|36|361|363|365|366|36
7|368|369|37|370|372|374|375|38|380|383|385|386|389|39|390|392|393
|4|401|405|408|409|41|411|412|413|414|416|419|42|420|421|423|424|4
26|43|44|45|5|52|58|6|62|64|65|68|69|7|70|76|78|79|8|81|82|84|85|8
7|88|90|92|93|94|95|97|
```

4. Soal Bonus

Masalah 1:

Cara menggunakan soundex pada proses pembuatan inverted index.

1. Keseluruhan token diubah ke karakter soundex, lalu disimpan dalam memori yang sama dengan inverted index.
2. Ketika ada query seperti robah, dilihat karakter soundex yang sama pada array inverted index, untuk melihat apakah ada kemungkinan kata robah adalah kata yang salah, karena robah dan rubah memiliki karakter soundex yang sama, sehingga robah bisa digantikan menjadi rubah.

Masalah 2:

Solusi sinonim:

Kalau menggunakan algoritma soundex, kata sinonim kemungkinan tidak bisa ditemukan karena kemungkinan besar memiliki karakter soundex yang berbeda. Solusinya adalah menggunakan korpus yang berisi sinonim kata-kata tersebut. Karena jika kita sebagai manusia harus mengingat kata sinonim tersebut, maka kita harus menyimpan kata-kata sinomin tersebut dalam memori jika kita ingin komputer mengetahuinya.