BAB II
LANDASAN TEORI

2.1 *Morfologi* Bahasa Indonesia

Morfologi adalah ilmu yang mempelajari seluk beluk kata serta pengaruh perubahan-perubahan bentuk kata terhadap golongan dan arti kata, atau dengan kata lain dapat dikatakan bahwa *morfologi* mempelajari seluk-beluk bentuk kata serta fungsi perubahan-perubahan bentuk kata itu (Ramlan, 1997). *Morfologi* kata bahasa Indonesia bisa terdiri dari struktur *infleksional* dan *derivasional*. *Infleksional* adalah struktur yang paling sederhana yang dinyatakan dalam penambahan *sufiks* dimana tidak mempengaruhi arti sebenarnya dari kata dasar yang dilekati (Tala, 2003). *Sufiks infleksional* dapat dibagi menjadi dua jenis yaitu:

1. *Sufiks* (*-lah, -kah, -pun, -tah*). *Sufiks* ini sebenarnya adalah partikel yang tidak mempunyai arti. Keberadaannya pada suatu kata adalah untuk penekanan.

   Analisis Contoh :

   dia + kah → diakah

   duduk + lah → duduklah

   2. *Sufiks* (*-ku, -mu, -nya*). *Sufiks* ini berfungsi sebagai kata ganti kepunyaan.

   Contoh :

   tas + ku → tasku

   buku + mu → bukumu

Sufiks-sufiks diatas dapat melekat pada kata dasar secara bersama-sama. Adapun aturan urutannya adalah *sufiks* pada jenis kedua selalu diletakkan sebelum *sufiks* jenis pertama. Sehingga struktur *morfologi* pada kata *infleksional* adalah : *Infleksional* = (kata dasar +kata ganti) | (kata dasar + partikel) | (kata dasar + kata ganti + partikel)

Penambahan *sufiks infleksional* tidak akan merubah bentuk dasar dari kata berimbuhan. Dengan kata lain, tidak ada penghilangan atau peleburan kata dasar pada kata berimbuhan. Kata dasar dapat ditentukan dengan mudah.

Pada struktur *infleksional*. Struktur *derivasional* dalam bahasa Indonesia terdiri dari *prefiks*, *sufiks* dan kombinasi dari keduanya. *Prefiks* yang sering dipakai adalah : *ber-, di-, ke-, meng-, peng-, per-, ter-*. Contoh penggunaan *prefiks* adalah :

ber + lari → berlari

di + makan → dimakan

ke + kasih → kekasih

meng + ambil → mengambil

peng + atur → pengatur

per + lebar → perlebar

ter + baca → terbaca

Beberapa *prefiks* seperti *ber-*, *meng-*, *peng-*, *per-*, *ter-* mungkin akan berubah menjadi beberapa bentuk yang berbeda. Bentuk dari setiap *prefiks* bergantung pada karakter pertama dari kata dasar yang dilekatinya. Tidak seperti struktur *infleksional*, pada struktur *derivasional* pengucapan kata mungkin berubah setelah adanya penambahan *prefiks*. Seperti contoh menyapu yang terdiri dari *prefiks meng-* dan kata dasar sapu. *Prefiks* meng- berubah menjadi *meny-* dan karakter pertama dari kata dasar mengalami peleburan. *Sufiks derivasional* adalah *-i, -kan, -an.* Contoh penggunaan *sufiks derivasional* adalah :

gula + i → gulai

makan + an → makanan

sampai + kan → sampaikan

Berbeda dengan penggunaan *prefiks*, penambahan *sufiks* tidak akan mengubah bentuk dasar dari suatu kata. Seperti disebutkan sebelumnya, struktur *derivasional* juga terdiri dari *konfiks*, yaitu gabungan dari *prefiks* dan *sufiks* yang melekat secara bersama-sama pada suatu kata. Contoh :

Per + main + an → permainan

Ke + kalah + an → kekalahan

Ber + jatuh + an → berjatuhan

Meng + ambil + i → mengambili

Struktur *morfologi* pada kata *derivasional* adalah : *Derivasional* = (*prefiks* + kata dasar) | (kata dasar + *sufiks*) | (*prefiks* + kata dasar + *sufiks*) | (*prefiks* 1 + *prefiks* 2 + kata dasar) | (*prefiks* 1 + *prefiks* 2 + kata dasar + *sufiks*).

Struktur lain yang mungkin terjadi dalam *morfologi* bahasa Indonesia adalah penambahan *sufiks infleksional* pada struktur *derivasional*, yang dinamakan *multiple suffiks* atau *sufiks* bertingkat. Sehingga dapat disimpulkan secara umum struktur *morfologi* kata bahasa Indonesia adalah : Struktur *morfologi* = [*prefiks* 1] + [*prefiks* 2] + kata dasar + [*sufiks*] + [kata ganti] + [partikel].

2.2 *Preprocessing*

Prepocessing merupakan proses awal yang akan mengubah data masukan menjadi data yang sesuai dan siap untuk diproses. *Prepocessing* terdiri dari *case folding, tokenizing, stopword removal* dan *stemming*.

2.2.1 Case Folding

Case folding adalah proses penyamaan huruf dengan mengubahnya menjadi huruf kecil (*lowercase)*. Pada kata yang dimasukkan terdapat huruf besar (kapital) dan kecil. Perbedaan bentuk huruf pada kata tersebut akan menganggu proses selanjutnya maka dari itu diperlukan proses *case folding* untuk merubah kata menjadi huruf kecil semua.

2.2.2 Tokenizing (Pemisahan rangkaian kata)

Tahap *tokenizing* yakni tahap pemotongan string inputan berdasarkan kata yang menyusunnya. Pada dasarnya proses *tokenizing* adalah pemenggalan kalimat menjadi kata.

2.2.3 Penyaringan (*Stop word removal*)

Penyaringan atau *Filtration* merupakan pengambilan 'kata' penting dari hasil token, menggunakan *algoritma stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). '*Stoplist*' atau '*Wordlist*' adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag of word*. Kata yang tidak penting adalah hasil *parsing* dicek dengan kamus (kumpulan

kata) *stopword*. Jika *parsing* ada yang sama dengan *stopword* maka akan dibuang atau dihapus.

2.2.4 Stemming

Stemming merupakan sebuah cara untuk menghilangkan imbuhan seperti berupa awalan dan akhiran agar didapatkan kata dasar (*root word*) (Novitasari, 2016). Proses *stemming* memiliki aturan atau *algoritma* yang berbeda dalam menghilangkan imbuhan disetiap bahasanya, seperti bahasa inggris memiliki perbedaan aturan penggunaan tata bahasa dengan bahasa indonesia.

2.2.5 Stemming Nazief&Adriani

Algoritma Nazief & Adriani memperhatikan kemungkinan adanya partikel-partikel yang mungkin mengikuti suatu kata berimbuhan. Sehingga kita dapat melihat pada rumus untuk *algoritma* ini yaitu adanya penempatan *possesive pronoun* dan juga partikel yang mungkin ada pada suatu kata berimbuhan (Agusta, 2009).

Algoritma Nazief & Adriani yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut (Agusta, 2009) :

1. Cari kata yang akan *distemming* dalam kamus kata dasar. Jika ditemukan maka diasumsikan kata adalah kata dasar. Maka *algoritma* berhenti.

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

3. Hilangkan *Derivation Suffixes* (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka *algoritma* berhenti. Jika tidak maka ke langkah 3a.

    a. 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.

b. Akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah 4.

a. Langkah 4 berhenti jika :

1. Terjadi kombinasi awalan dan akhiran.

2. Awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya.

3. Tiga awalan telah dihilangkan.

Tabel 2.1 Kombinasi Awalan-Akhiran yang tidak diizinkan Awalan Akhiran yang tidak diijinkan.

Awalan	Akhiran yang tidak di izinkan
be-	-i
di-	-an
ke-	-I, -kan
me-	-an
se-	-I, -kan
te-	-an

b. Identifikasi tipe awalan dan hilangkan. Tipe awalan ada 2 yaitu:

1. Standart : “di-”, “ke-”, “se-” dapat langsung dihilangkan dari kata

2. Kompleks : “me-”, “be”, “pe-”, “te-” adalah tipe awalan yang dapat *bermorfologi* sesuai kata dasar yang mengikutinya. Oleh karena itu, gunakan aturan pada Tabel 2.1 untuk mendapatkan pemenggalan yang tepat.

c. Cari kata yang telah dihilangkan awalannya ini di dalam kamus. Apabila tidak ditemukan, maka langkah 4 diulangi kembali. Apabila ditemukan maka keseluruhan proses selesai.

5. Apabila setelah langkah 4 kata dasar masih belum ditemukan, maka proses *recoding* dilakukan dengan mengacu pada aturan pada Tabel 2.3 *Recoding* dilakukan dengan menambahkan karakter *recoding* di awal kata yang dipenggal. Karakter *recoding* adalah huruf kecil setelah tanda hubung (‘-’) dan

terkadang berada sebelum tanda kurung. Sebagai contoh, kata “menangkap” (aturan 15), setelah dipenggal menjadi “nangkap”. Karena tidak *valid*, maka *recoding* dilakukan dan menghasilkan kata “tangkap”.

Tabel 2.2 Aturan Pemenggalan awalan *Stemmer* Nazief & Adriani

Aturan	Format Kata	Pemenggalan
1	berV…	ber-V … | be-rV
2	berCAP…	ber-CAP… dimana C != ‘r’ & P != ‘er’
3	berCAerV	ber-CaerV… dimana C != ‘r’
4	Belajar	bel-ajar
5	berC1erC2…	be-C1erC2… dimana C1 != ‘r’ | ‘l’
6	terV…	ter-V… | te-rV…
7	terCerV…	ter-CerV dimana C != ‘r’
8	terCP…	Ter-CP… dimana C!=’r’ dan P !=’er’
9	teC1erC2…	Te-C1erC2… dimana C1 != ‘r’
10	me{l|r|w|y}V…	me – {l|r|w|y} V…
11	mem{b|f|v}…	mem-{b|f|v}…
12	Mempe	mem-pe…
13	mem{rV|V}…	me-m{rV|V}… | me-p{rV|V}
14	men{c|d|j|s|z}…	men-{c|d|j|s|z}…
15	menV…	me-nV… | me-tV
16	meng{g|h|q|k}…	meng-{g|h|q|k}…
17	mengV…	meng-V… | meng-kV…| mengV-... jika V=’e’
18	menyV…	meny-sV….
19	mempA…	mem-pA… dimana A != ‘e’
20	pe{w|y}V…	pe-{w|y}V…
21	perV…	per-V… | pe-rV…
22	perCAP…	per-CAP… dimana C != ‘r’ dan P != ‘er’
23	perCAerV…	per-CAerV… dimana C != ‘r’
24	pem{b|f|V}…	pem-{b|f|V}…
25	pem{rV|V}…	pe-m{rV|V}… | pe-p{rV|V}…

26	pen{c|d|j|z}…	pen-{c|d|j|z}…
27	penV…	pe-nV… | pe-tV…
28	pengC…	peng-C…
29	pengV…	peng-V… | peng-kV… | pengV-... jika V=’e’
30	penyV…	peny-sV…
31	pelV…	pe-lV… kecuali “pelajar” yang menghasilkan “ajar”
32	peCerV…	Per-erV … dimana C!= {r|w|y|l|m|n}
33	peCP	Pe-CP… dimana C!={r|w|y|l|m|n}dan P!= ‘er’
34	terC1erC2...	ter-C1erC2... dimana C1!= ‘r’
Keterangan simbol huruf C : huruf konsonan V : huruf vocal A : huruf vocal atau konsonan P : partikel atau fragmen dari setiap kata, misalnya “er”		

6. Jika semua langkah telah selesai tetapi tidak juga berhasil maka kata awal diasumsikan sebagai kata dasar. 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 2.2 Hapus awalan jika ditemukan.

Tabel 2.3 Cara Menentukan Tipe Awalan Untuk awalan "te-"

Following Characters				**Tipe Awalan**
Set 1	**Set 2**	**Set 3**	**Set 4**	
"-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 "-r-")	Not "-er-"	-	Ter
Not (vowel or "-r-")	"-er-"	Vowel	-	None
Not (vowel or "-r-")	"-er-"	Not Vowel	-	None

Table 2.4 Jenis awalan berdasarkan tipe awalannya

Tipe Awalan	**Awalan Yang Harus Di hapus**
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.

   a. 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".

b. 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.

a. Untuk tipe awalan “mem-“, kata yang diawali dengan awalan “memp-” memiliki tipe awalan “mem-”.

b. 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 + 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

2.3 *N-Gram* Metode

Nilai-nilai N-gram merupakan salah satu proses yang secara luas digunakan dalam *text mining* (pengolahan teks) dan pengolahan bahasa. Secara N-gram merupakan sekumpulan kata yang diberikan dalam sebuah paragraf dan ketika menghitung n-gram biasanya dilakukan dengan menggerakkan satu kata maju ke depan (Meskipun dalam prosesnya terdapat suatu proses dimana kata yang dimajukan sejumlah X kata). Sebagai contoh terdapat sebuah kaliman “Berjalan”. Jika N=2 makan dikenal dengan *bigram*. Dimana ngram menjadi :

- Be
- Er
- Rj
- Ja
- Al
- La
- An

Bisa dilihat dari contoh diatas, dimana kita memiliki 5 n-gram dalam kasus tersebut. Perhatikan bahwa terjadi pergerakan kata tiap dua kata yaitu dari Be -> E ke E –> Er ke R -> *over* dst. Dimana kejadiannya terjadi secara berurutan dengan cara tiap kata berpindah maju satu kata ke depan untuk membangkitkan n-gram selanjutnya.

Jika terdapat N=3 maka n-gram menjadi sebagai berikut :

- Ber
- Erj
- Rja
- Jal
- Ala
- Lan

Berdasarkan kasus di atas terdapat 6 n-gram. Sedangkan jika N=1 maka bisa disebut dengan ***Uni-Gram*** yang pada dasarnya hanya terdiri dari satu kata dalam sebuah kalimat. Ketika terdiri dari N=2 maka disebut dengan ***Bi-Gram*** , dan ketika terdir dari N>1 bisa disebut dengan *four gram*, *five gram* dan seterusnya.

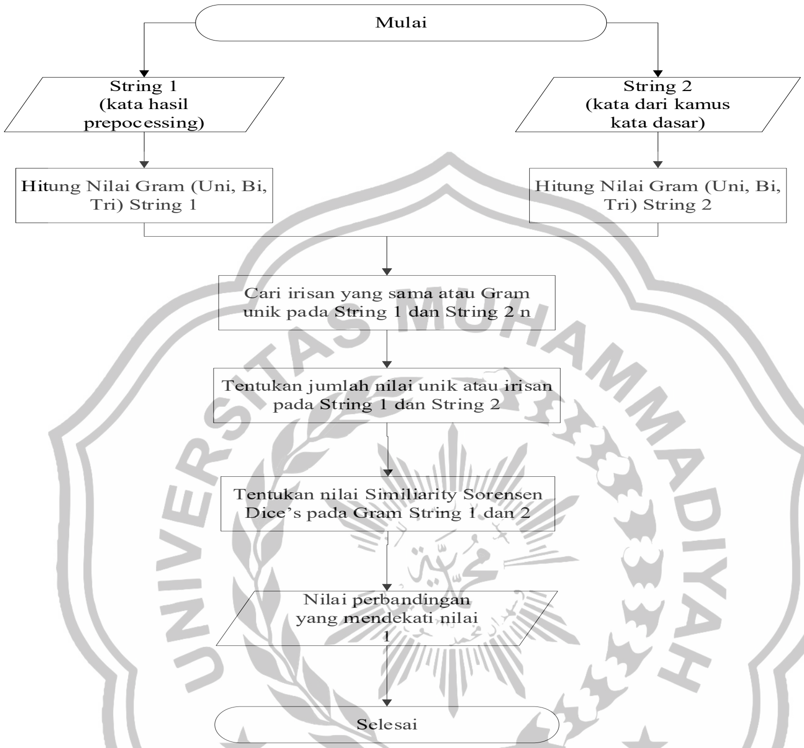
Jika dimisalkan X adalah jumlah kata dalam suatu kalimat K, maka jumlah n-gram dari kalimat K dapat dilihat pada rumus 2.1

N*grams*K = X – (N – 1)……..(2.1)

N-gram digunakan dalam berbagai macam proses pengolahan bahasa, *N-gram* yang digunakan tidak hanya menggunakan unigram tapi model bigram dan trigram juga digunakan. *Microsoft* dan *Google* menggunakan model ini untuk pengembangan *website* mereka beberapa diantaranya digunakan untuk beberapa tugas seperti *spelling correction*, *word breaking* dan *text summarization* dalam web mereka

2.4 *Sorensen-Dice Coefficient*

Sorensen-Dice Coefficient, atau biasa disebut *Sorensen Index* atau *Dice's Coefficient* ditemukan oleh Throvald Sorensen dan Lee Raymond Dice. Alur proses dalam perhitungan kedekatan dengan *Sorensen-Dice* seperti pada gambar.



Gambar 2. 1 Alur proses metode *Sorensen-Dice Coefficient*

Rumus untuk menghitung *Sorensen Dice's Similiarity* dapat dilihat pada rumus 2.2 sebagai berikut

$$S = \frac{2|A \cap B|}{|A| + |B|} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

Dimana S adalah nilai *Similiarity* |A|, dan |B| merupakan jumlah *N-gram* yang unik dari *String* pertama dan *String* kedua. |A|∩|B| merupakan jumlah *N-Gram* yang unik dan sama dari masing-masing *String* yang dibandingkan. Sebagai contoh, jumlah bigram dari Parang dan Perang adalah:

|A| = 5

|B| = 4

|A∩B| = 4

$$S = \frac{2|5|\cap|4|}{|5| + |4|}$$

8/9 = 0,8889

Sehingga dapat diperoleh S=0,8889 atau presentase *similiarity* dari kedua kata tersebut adalah 90%.

2.5 Penelitian Sebelumnya

Penelitian sebelumnya dilakukan oleh Andita Dwiyoga Tahitoe dan Diana Purwitasari (Purwitasari, 2010) dari Institut Teknologi Sepuluh November telah melakukan penelitian yaitu "Modifikasi *Enhanced Confix Stripping Stemmer* Untuk Bahasa Indonesia dengan Metode Corpus Based Stemming" pada penelitian tersebut mampu memperbaiki kesalahan overstemming dan understemming yang dilakukan oleh *algoritma Enhanced Confix Stripping Stemmer*. Metode *Enhanced Confix Stripping* adalah penggabungan pengembangan dari dua mteode sebelumnya yaitu nazie-adriani dan Arifin. Penggunaan metode *corpus based stemming* dapat digunakan untuk memilih hasil *stemming* yang tepat berdasarkan koleksi dokumen yang digunakan.

Penelitian lainnya dilakukan oleh Erick Alfons Lisangan (2013). Mahasiswa Universitas Atma Jaya telah melakukan penelitian yaitu "Implementasi *N-Gram Technique* Dalam Deteksi Plagiarisme Pada Tugas Mahasiswa"pada penelitian tersebut mampu menemukan kata yang relevan dari dokumen 1 dengan dokumen yang lain berdasarkan nilai *n-Gram* yang diperoleh dari kedua dokumen yang dibandingkan.