

BAB II

LANDASAN TEORI

2.1 Text Mining

Text Mining dapat didefinisikan sebagai “*penemuan informasi baru dan tidak diketahui sebelumnya oleh computer, dengan secara otomatis mengekstrak informasi dari sumber – sumber teks tak terstruktur yang berbeda*” (Tan, 1999). Definisi singkatnya adalah suatu proses menganalisa teks untuk mengekstrak informasi yang berguna untuk tujuan tertentu. Perbedaan mendasar dari *text mining* dan *data mining* terletak pada sumber data yang digunakan. Pada *data mining* data yang diekstrak berasal dari pola-pola tertentu dan terstruktur, sedangkan *text mining* sumber data yang digunakan berasal dari teks yang relatif tidak terstruktur karena menggunakan tata Bahasa manusia atau biasa disebut (*natural language*). Secara umum basis data didesain untuk program dengan tujuan melakukan pemrosesan secara otomatis, sedangkan teks ditulis untuk dibaca langsung oleh manusia (Herast, 2003). Dalam *text mining* terdapat beberapa tahapan untuk memproses data teks tersebut

1. *Case Folding* dan *Tokenizing*

Case Folding biasa disebut penyeragaman kata dengan cara mengubah seluruh kata menjadi huruf kecil (*lowercase*). Hanya huruf a sampai z yang dapat diterima karakter selain huruf dihilangkan. Terdapat juga kata-kata tertentu yang harus sesuai dengan kaidah yang tidak bisa dilakukan penyeragaman kata seperti kata lembaga atau institusi yang selalui diawali huruf kapital dan juga nama gelar seperti halnya ST, M.Psi dan lain sebagainya. Tergantung dari sumber data yang digunakan untuk diproses.

Tokenizing adalah suatu tahapan pemotongan *string* kata berdasarkan penyusunan kata tersebut.

2. *Filtering*

Filtering adalah pengambilan kata-kata penting dari hasil *Tokenizing* atau bisasa disebut pengeliminasi sebuah kata-kata sesuai dengan kaidahnya.

Algoritma stop-word removal adalah salah satu yang digunakan untuk melakukan tahapan *filtering*

3. *Stemming*

Stemming adalah suatu proses untuk memecah suatu varian-varian kata menjadi kata dasar sesuai dengan kata yang sedang diproses. Jika kata yang diproses adalah Bahasa Indonesia untuk memecah varian kata menjadi kata dasar harus sesuai dengan aturan Bahasa Indonesia salah satu algoritma yang digunakan adalah Nazief & Adriani.

4. *Analyzing*

Analyzing adalah suatu tahapan menganalisa data teks yang sedang diproses untuk menentukan kemiripan antar dokumen teks salah satu metode yang digunakan adalah *cosine similarity*.

2.2 Stop Word

Stop Word adalah beberapa kata umum yang muncul dengan nilai kecil dalam membantu pemilihan kecocokan dokumen yang dibutuhkan pengguna (Eko Prasetyo, 2015). Dalam sebuah rangkaian kalimat atau paragraph dalam Bahasa Indonesia terdapat sebuah kata yang muncul berulang kali. Kemunculan kata tersebut sangat kecil berpengaruh dalam sebuah dokumen dalam hal pencocokan dokumen dikarenakan kata tersebut hanya sebagai penghubung antar kata seperti halnya : dan, ke, atau, yang, atau, sebagai, dari, adalah, kemana, dimana, tetapi, tersebut, walupun, dan lain sebagainya. Dalam hal ini biasanya dibuatkan sistem tersendiri sesuai dengan kebutuhan pengguna, kata tersebut akan dikelompokkan atau dihilangkan. Yang dimaksud dikelompokkan adalah untuk mengetahui berapa kali kemunculan kata penghubung dalam suatu dokumen. Manfaat penggunaan *stop-word* mampu mengurangi jumlah kata dalam suatu kalimat atau paragraph dalam sebuah dokumen untuk membantu pemilihan kecocokan dokumen yang akan diproses.

2.3 Algoritma Nazief & Adriani

Menurut Azhar Firdaus, Ernawati dan Arie Vatesia (2014). Algoritma stemming Nazief dan Adriani dikembangkan berdasarkan aturan morfologi Bahasa Indonesia yang mengelompokkan imbuhan menjadi awalan (prefix), sisipan (infix), akhiran (suffix) dan gabungan awalan akhiran (confixes). Algoritma ini menggunakan kamus kata dasar dan mendukung recoding, yakni penyusunan kembali kata-kata yang mengalami proses stemming berlebih. Aturan morfologi Bahasa Indonesia mengelompokkan imbuhan ke dalam beberapa kategori sebagai berikut:

1. *Inflection suffixes* yakni kelompok akhiran yang tidak merubah bentuk kata dasar. Sebagai contoh, kata “duduk” yang diberikan akhiran “-lah” akan menjadi “duduklah”. Kelompok ini dapat dibagi menjadi dua:
 - i. *Particle* (P) atau partikel yakni termaksud di dalamnya “-lah”, “-kah”, “-tah” dan “-pun”.
 - ii. *Possessive pronoun* (PP) atau kata ganti kepunyaan, termaksud di dalamnya “-ku”, “-mu” dan “-nya”.
2. *Derivation suffixes* (DS) yakni kumpulan akhiran asli Bahasa Indonesia yang secara langsung ditambahkan pada kata dasar yaitu akhiran “-i”, “-kan”, dan “-an”.
3. *Derivation prefixes* (DP) yakni kumpulan awalan yang dapat langsung diberikan pada kata dasar murni, atau pada kata dasar yang sudah mendapatkan penambahan sampai dengan 2 awalan. Termaksud di dalamnya adalah:
 - i. Awalan yang dapat bermorfologi (“me-“, “be-“, “pe-“ dan “te”).
 - ii. Awalan yang tidak bermorfologi (“di-“, “ke-“ dan “se-“).

Berdasarkan pengklasifikasi imbuhan-imbuhan di atas, maka bentuk kata berimbuhan dalam Bahasa Indonesia dapat dimodelkan sebagai berikut:

[DP+ [DP+ [DP+]]] Kata Dasar [+DS] [+PP]

Gambar 2.1 Model Kata Berimbuhan dalam Bahasa Indonesia

Keterangan dari Gambar 2.1

DP : *Derivation prefixes*

DS : *Derivation suffixes*

PP : *Possessive pronoun*

Dengan model bahasa Indonesia di atas serta aturan-aturan dasar morfologi Bahasa Indonesia, aturan yang digunakan dalam proses algoritma Nazief & Adriani sebagai berikut:

1. Tidak semua kombinasi awalan dan akhiran diperbolehkan. Kombinasi-kombinasi imbuhan yang tidak diperbolehkan, yaitu “be-i”, “ke-i”, “ke-kan”, “me-an”, “se-i”, “se-kan” dan “te-an”.
2. Penggunaan imbuhan yang sama secara berulang tidak diperkenankan.
3. Jika suatu kata hanya terdiri dari satu atau dua huruf, maka proses tidak dilakukan.
4. Penambahan suatu awalan tertentu dapat mengubah bentuk asli kata dasar, ataupun awalan yang telah diberikan sebelumnya pada kata dasar bersangkutan. Sebagai contoh, awalan “me-“ dapat berubah menjadi “meng-“, “men-“, “meny-“, dan “mem-“. Oleh karena itu diperlukan suatu aturan yang mampu mengatasi masalah morfologi ini.

Algoritma Nazief & Adriani memiliki tahap-tahap sebagai berikut:

1. Cari kata dalam kamus jika ditemukan maka diasumsikan bahwa kata tersebut adalah kata dasar. Algoritma berhenti. Jika tidak ditemukan maka lakukan langkah 2.
2. Hilangkan *inflectional suffixes* bila ada. Dimulai dari *inflectional particle* (“-lah”, “-kah”, “-tah” dan “-pun”), kemudian *possessive pronoun* (“-ku”, “-mu” dan “-nya”). Cari kata pada kamus jika ditemukan algoritma berhenti, jika kata tidak ditemukan dalam kamus lakukan langkah 3.
3. Hilangkan *derivation suffixes* (“-an”, “-i” dan “-kan”). Jika akhiran “-an” dihapus dan ditemukan akhiran “-k”, maka akhiran “-k” dihapus. Cari kata pada kamus jika ditemukan algoritma berhenti, jika kata tidak ditemukan maka lakukan langkah 4.
4. Pada langkah 4 terdapat tiga iterasi.

- 1) Iterasi berhenti jika :
- Ditemukannya kombinasi awalan yang tidak diizinkan berdasarkan awalan

Tabel 2.1 Kombinasi Awalan Akhiran yang Tidak Diizinkan

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

- Awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya.
 - Tiga awalan telah dihilangkan.
- 2) Identifikasikan tipe awalan dan hilangkan. Awalan terdiri dari dua tipe:
- Standar (“di-“, “ke-“, “se-“) yang dapat langsung dihilangkan dari kata.
 - Kompleks (“me-“, “be-“, “pe-“, “te-“) adalah tipe-tipe awalan yang dapat bermorfologi sesuai kata dasar yang mengikutinya. Oleh karena itu dibutuhkan aturan pada tabel 2.2 untuk mendapatkan hasil pemenggalan yang tepat.

Tabel 2.2 Aturan Pemenggalan Awalan

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 w y}V	me - {l 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-...
18	menyV...	meny-sV....

Lanjutan dari tabel 2.2

Aturan	Format Kata	Pemenggalan
19	mempA...	mem-pA... dimana A != 'e'
20	pe{w y}V...	pe-{w y}V...
21	perV...	per-V... pe-rV...
23	perCAP...	per-CAP... dimana C != 'r' dan P !=
24	perCAerV...	per-CAerV... dimana C != 'r'
25	pem{b f V}...	pem-{b f V}...
26	pem{rV V}...	pe-m{rV V}... pe-p{rV V}...
27	pen{c d j z}...	pen-{c d j z}...
28	penV...	pe-nV... pe-tV...
29	pengC...	peng-C...
30	pengV...	peng-V... peng-kV... pengV-... jika
31	penyV...	peny-sV...
32	peIV...	pe-IV... kecuali "pelajar" yang
33	peCerV...	Per-erV ... dimana C != {r w y l m n}
34	peCP	Pe-CP... dimana C != {r w y l m n} dan
35	terC1erC2...	ter-C1erC2... dimana C1 != 'r'
36	peC1erC2...	pe-C1erC2... dimana C1 != {r w y l m n}

Keterangan simbol huruf

C : huruf konsonan

V : huruf vocal

A : huruf vocal atau konsonan

P : partikel atau fragmen dari setiap kata, misalnya "er"

- 3) Cari kata yang telah dihilangkan awalnya. Apabila tidak ditemukan, maka langkah 4 diulang kembali. Apabila ditemukan, maka algoritma berhenti.
5. Apabila setelah langkah 4 kata dasar masih belum ditemukan, maka proses *recording* dilakukan dengan mengacu pada aturan tabel 2.4. *Recording* dilakukan dengan menambahkan karakter *recording* di awal kata yang dipenggal. Pada tabel 2.4, karakter *recording* adalah huruf kecil setelah tanda hubung ('-') dan terkadang berada sebelum tanda kurung. Sebagai contoh, kata "menangkap" (aturan 15) pada tabel 2.4, setelah dipenggal menjadi "nangkap". Karena tidak valid, maka *recording* dilakukan dan menghasilkan kata "tangkap".
6. Jika semua langkah gagal, maka input kata yang diuji pada algoritma ini di anggap sebagai kata dasar.

2.4 Term Frequency dan Invers Document Frequency (TF-IDF)

Metode tf-idf merupakan salah satu metode untuk menghitung bobot setiap kata yang digunakan. Pada penelitian sebelumnya dilakukan perhitungan dengan metode *binary term* sedangkan pada Metode *tf-idf* terkenal mudah, efisien dalam hal melakukan perhitungan bobot kemunculan kata pada sebuah dokumen dengan rumus sebagai berikut:

$$tf_wt_{t,d} = \begin{cases} 1 + \log_{10}(tf_{t,d}) & \text{Jika } tf_{t,d} > 0 \\ 0 & \text{jika } tf_{t,d} \leq 0 \end{cases} \dots\dots\dots(1)$$

$$idf_t = \log_{10} \left(\frac{N}{df_t} \right) \dots\dots\dots(2)$$

$$w_{t,d} = (1 + \log_{10}(tf_{t,d})) \times \log_{10} \left(\frac{N}{df_t} \right) \dots\dots\dots(3)$$

Keterangan :

tf_wt = nilai bobot atau score

idf = nilai idf

df = jumlah kemunculan kata pada tiap dokumen

t = jumlah kemunculan kata

d = dokumen

w = bobot akhir nilai tf X idf

N = jumlah keseluruhan dokumen

2.5 Cosine Similarity

Cosine Similarity adalah salah satu metode dalam menentukan nilai kemiripan antar dua objek. Salah satu contoh penerapan adalah penentuan kemiripan pada sidik jari manusia. *Cosine Similarity* dapat diterapkan dalam menentukan nilai kemiripan pada dua berkas dokumen teks. Parameter yang digunakan adalah jumlah kata-kata pada dua dokumen teks yang dibandingkan. *Cosine Similarity* menggunakan dua vektor yang mempresentasikan dua dokumen teks dengan rumus sebagai berikut :

$$sim(d_1, d_2) = \frac{\vec{v}(d_1) \cdot \vec{v}(d_2)}{|\vec{v}(d_1)| |\vec{v}(d_2)|} \dots\dots\dots(4)$$

Dimana d_1 dan d_2 merupakan nilai yang mengekspresikan sudut antar vector jika vector d_1 dan d_2 memiliki data sebanyak n maka dapat dihitung dengan rumus sebagai berikut :

$$D_1 \cdot D_2 = D_{1_1}D_{2_1} + \dots + D_{1_n}D_{2_n} \dots\dots\dots(5)$$

Sedangkan $|d_1|$ merupakan panjang vektor. Panjang vektor dapat dihitung dengan rumus sebagai berikut :

$$|D_1| = \sqrt{a^2 + a^2 + a^2 + \dots + a^n} \dots\dots\dots(6)$$

2.6 Analisa Optimasi

Analisa optimasi dilakukan untuk mengetahui perhitungan manual dari metode atau algoritma yang akan digunakan untuk menentukan kemiripan dokumen. Diasumsikan terdapat sebuah data dengan sebuah kalimat sebagai berikut

D1 : komputer membantu pekerjaan manusia menjadi lebih mudah

D2 : pekerjaan manusia menjadi lebih mudah bila dibantu oleh komputer

Table 2.3 nilai perhitungan

NO	Kata	D1	D2	tf_{wt1}	tf_{wt2}	df	idf	$tfidf_{d1}$	$tfidf_{d2}$
1	Bantu	1	1	1	1	2	1	1	1
2	Bila	0	1	0	1	1	0,301	0	0,301
3	Komputer	1	1	1	1	2	1	1	1
4	Jadi	1	0	1	0	1	0,301	0,301	0
5	Lebih	1	1	1	1	2	1	1	1
6	Manusia	1	1	1	1	2	1	1	1
7	Mudah	1	1	1	1	2	1	1	1
8	Oleh	0	1	0	1	1	0,301	0,301	0
9	Kerja	1	1	1	1	2	1	1	1

Bedasarkan table perhitungan maka akan didapatkan hasil tf-idf dari kedua dokumen. Kemudian hasil tf-idf kita analisa dengan metode *cosine similarity* dengan perhitungan sebagai berikut :

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 0,301\ 1\ 1\ 1\ 0,301\ 1}{\sqrt{1^2 + 0^2 + 1^2 + 0,301^2 + 1^2 + 1^2 + 1^2 + 0,301^2 + 1^2}}$$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 0,301\ 1\ 1\ 1\ 0,301\ 1}{\sqrt{1 + 0 + 1 + 0,090601 + 1 + 1 + 1 + 0,090601 + 1}}$$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 0,301\ 1\ 1\ 1\ 0,301\ 1}{\sqrt{6,181202}}$$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 0,301\ 1\ 1\ 1\ 0,301\ 1}{2,48620}$$

$$= \{0,40222\} \{0\} \{0,40222\} \{0,12107\} \{0,40222\} \{0,40222\} \{0,40222\} \{0,12107\} \{0,40222\}$$

$$\vec{V}(D_2) = \frac{1\ 0,301\ 1\ 0\ 1\ 1\ 1\ 0\ 1}{\sqrt{1^2 + 0,301^2 + 1^2 + 0^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2}}$$

$$\vec{V}(D_2) = \frac{1\ 0,301\ 1\ 0\ 1\ 1\ 1\ 0\ 1}{\sqrt{1 + 0,090601 + 1 + 0 + 1 + 1 + 1 + 0 + 1}}$$

$$\vec{V}(D_2) = \frac{1\ 0,301\ 1\ 0\ 1\ 1\ 1\ 0\ 1}{\sqrt{6,090601}}$$

$$\vec{V}(D_2) = \frac{1\ 0\ 1\ 0,301\ 1\ 1\ 1\ 0,301\ 1}{2,46791}$$

$$= \{0,40520\} \{0,12196\} \{0,40520\} \{0\} \{0,40520\} \{0,40520\} \{0,40520\} \{0\} \{0,40520\}$$

$$= (D1_1 * D2_1) + (D1_2 * D2_2) + \dots + (D1_9 * D2_9)$$

$$= \mathbf{0,9779}$$

Berdasarkan perhitungan yang dilakukan dengan menggunakan pembobotan tf-idf dengan metode *cosine similarity* didapatkan hasil kemiripan dokumen adalah 0,9779

2.7 Penelitian Sebelumnya

Penelitian sebelumnya yang dilakukan oleh Azhar Firdaus, Ernawati, dan Arie Vatesia dari program studi Teknik Informatika, Fakultas Teknik, Universitas Bengkulu mendapatkan hasil nilai kemiripan dokumen adalah 0,8018 dengan menggunakan pembobotan *binary term* dengan metode *cosine similarity*.

Tabel 2.4 Pembobotan *Binary Term*

No	Kata	Dokumen Pertama (D1)	Dokumen kedua (D2)
1	Bantu	1	1
2	Bila	0	1
3	komputer	1	1
4	Lebih	1	1
5	manusia	1	1
6	menjadi	1	0
7	mudah	1	1
8	Oleh	0	1
9	kerja	1	1

Tabel 2.5 Vektor berdasarkan Aturan metode *cosine similarity*

No	Kalimat	Vektor yang dibentuk
D1	<i>Komputer bantu pekerjaan manusia jadi lebih mudah</i>	$A = (1,0,1,1,1,1,1,0,1)$
D2	<i>Pekerjaan manusia jadi lebih mudah bila bantu oleh komputer</i>	$B = (1,1,1,1,1,0,1,1,1)$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}{\sqrt{1^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2}}$$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}{\sqrt{7}}$$

$$\vec{V}(D_1) = \frac{1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}{2,64575}$$

$$D1 = \{0,37796\} \{0\} \{0,37796\} \{0,37796\} \{0,37796\} \{0,37796\} \{0,37796\} \{0\} \{0,37796\}$$

$$\vec{V}(D_2) = \frac{111110111}{\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2 + 1^2 + 1^2}}$$

$$\vec{V}(D_2) = \frac{111110111}{\sqrt{8}}$$

$$\vec{V}(D_2) = \frac{111110111}{2,82842}$$

$$D2 = \{0,35355\} \{0,35355\} \{0,35355\} \{0,35355\} \{0,35355\} \{0\} \{0,35355\} \{0,35355\} \{0,35355\}$$

$$= (D1_1 * D2_1) + (D1_2 * D2_2) + \dots + (D1_9 * D2_9)$$

$$\text{Sim}(D1, D2) = \mathbf{0,8018}$$

Table 2.6 Hasil Nilai perhitungan Dengan Batas Threshold

Dokumen Teks (D1)	Dokumen Teks (D2)	Hasil Nilai Sebelumnya	Hasil Nilai Similarity Selanjutnya	Batas Threshold	Status Dokumen
komputer membantu pekerjaan manusia menjadi lebih mudah	pekerjaan manusia menjadi lebih mudah bila dibantu oleh computer	0.8018	0,9779	< 0.50 = tidak mirip > 0.50 = mirip	Mirip

Berdasarkan hasil penelitian terdapat perbedaan hasil *similarity* dari dokumen yang sedang dibandingkan dengan menggunakan pembobotan yang berbeda. Pembobotan yang dilakukan dengan menggunakan binary term adalah 0,8018 sedangkan pembobotan yang dilakukan menggunakan tf-idf adalah 0,9779. Terdapat peningkatan nilai ketelitian kemiripan dokumen dengan selisih 0,1761 dan *threshold* 0.50 dasar pemberian *threshold* didapatkan dari *Based on slides of* W. Arms, Thomas Hofmann, Ata Kaban, Melanie Martin.