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 menganilisa 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, sedangkam *text mining* sumber data yang digunakan berasal dari teks yang relatif tidak testruktur 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 karaktek selain huruf dihilangkan. Terdapat juga kata-kata tertentu yang harus sesuai dengan kaidah yang tidak bisa dilakukan penyeragaman kata sperti 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 berdasrakan penyusunan kata tersebut.

2. *Filtering*

   Filtering adalah pengambilan kata-kata penting dari hasil *Tokenizing* atau biasa 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 sutau 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 menganlisa 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 tesebut sangat kecil berpengaruh dalam sebuah dokumen dalam hal pencocokan deskripsi 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 deskripsi untuk membantu pemilihan kecocokan deskripsi yang akan diproses.

2.3 Algoritma Nazief & Adriani

Menurut Azhar Firdaus, Ernawati dan Arie Vatresia (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 genti 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 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 tidak ditemukan maka lakukan langkah 4.

4. Pada langkah 4 terdapat tiga iterasi.
   1) Iterasi berhenti jika :
      a Ditemukannya kombinasi awalan yang tidak diizikan 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

      b Awalan yang dideteksi saat ini sama dengan awalan yang dihilangkan sebelumnya.
      c Tiga awalan telah dihilangkan.
   2) Identifikasikasikan tipe awalan dan hilangkan. Awalan terdiri dari dua tipe:

a. Standar (“di-“, “ke-“, “se-”) yang dapat langsung dihilangkan dari kata.

b. 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|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|js|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	pelV…	pe-lV… 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 awalannya. 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 sebagi berikut:

$$tf\_wt_{t,d} = \begin{cases} 1 + \log_{10}(tf_{t,d}) & Jika\ tf_{t,d} > 0 \\ 0 & jika\ tf_{t,d} \leq 0 \end{cases} \quad \text{.....(2.1)}$$

$$idf_t = \log_{10}\left(\frac{N}{df_t}\right) \quad \text{.....(2.2)}$$

$$w_{t,d} = \left(1 + \log_{10}(tf_{t,d})\right) \times \log_{10}\left(\frac{N}{df_t}\right) \quad \text{.....(2.3)}$$

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 :

$$sim(d_1, d_2) = \frac{\vec{V}(d_1).\ \vec{V}(d_2)}{|\vec{V}(d_1)|\ \ |\vec{V}(d_2)|} \qquad \text{.....(2.4)}$$

Dimana d1 dan d2 merupakan nilai yang mengekspresikan sudut antar vector jika vector d1 dan d2 memiliki data sebanyak n maka dapat dihitung dengan rumus sebagai berikut :

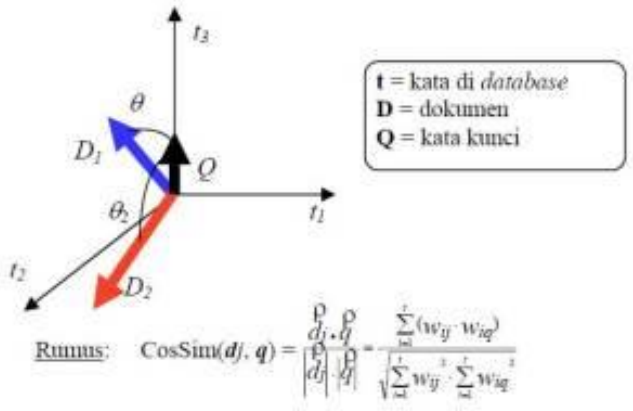
$$D1.D2 = D1_1 D2_1 + \cdots + D1_n D2_n \qquad \text{.....(2.5)}$$

Sedangkan |d1| merupakan panjang vektor. Panjang vektor dapat dihitung dengan rumus sebagai berikut :

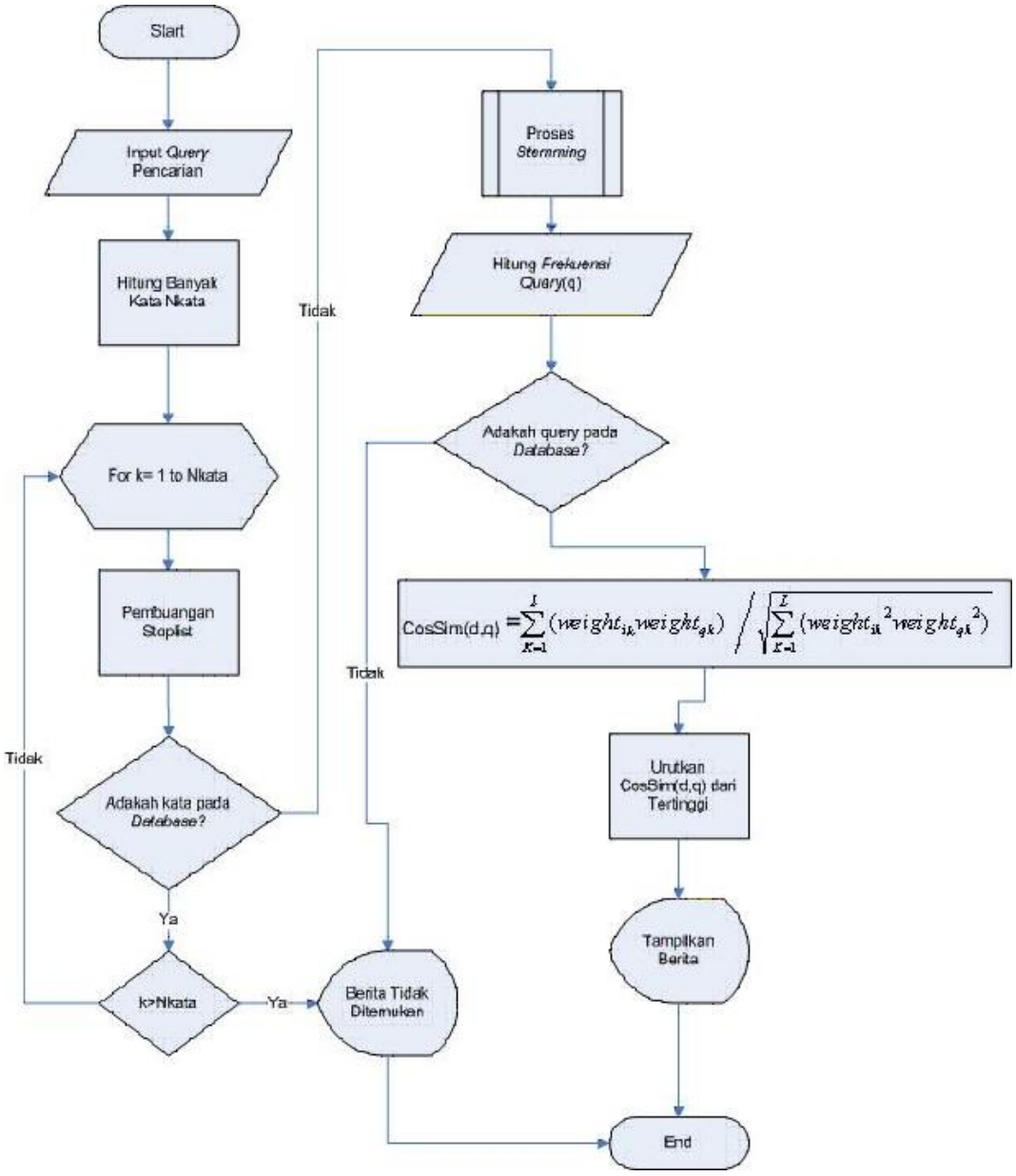
$$|D1| = \sqrt{a^2 + a^2 + a^2 + \cdots + a^n} \qquad \text{.....(2.6)}$$

2.6 Pengertian Vector Space Model (VSM)

Vector Space Model (VSM) adalah suatu model yang digunakan untuk mengukur kemiripan antara suatu dokumen dengan suatu query. Query dan dokumen dianggap sebagai vektor-vektor pada ruang n-dimensi, dimana t adalah jumlah dari seluruh term yang ada dalam leksikon. Leksikon adalah daftar semua term yang ada dalam indeks. Selanjutnya akan dihitung nilai cosinus sudut dari dua vektor, yaitu W dari tiap dokumen dan W dari kata kunci.



Vector space model (VSM) solusi atas permasalah yang dihadapi jika menggunakan algoritma TF/IDF. Karena pada algoritma TF/IDF terdapat kemungkinan antar dokumen memiliki bobot yang sama, sehingga ambigu untuk diurutkan. Adapun Flowchart dari pencarian menggunakan algortima Vector space model sebagai berikut:

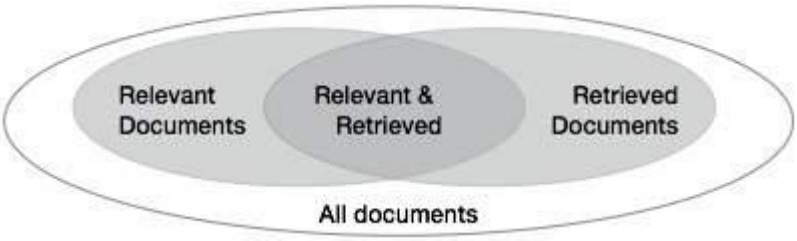


Gambar 2.1 Flowchart dari pencarian menggunakan algortima VSM

2.7 Recall dan Precision

Ada beberapa alasan yang berbeda mengapa tahap evaluasi Sistem Temu Kembali Informasi (Information Retrieval) adalah sesuatu yang penting. Sebagai contoh, penyedia sumber informasi membutuhkan informasi tentang penggunaan sumber daya oleh user, dan organisasi yang bekerja untuk meningkatkan kinerja search engine perlu metode-metode yang efektif untuk mengevaluasi perubahan yang dilakukan untuk algoritma dan user interface. Dengan demikian, tujuan evaluasi adalah untuk menghasilkan perbaikan pada proses pengambilan informasi. Di sisi lain, tujuan evaluasi biasanya tergantung pada penelitian, beberapa peneliti mungkin berpendapat bahwa tujuan utama dari evaluasi adalah untuk mengevaluasi kekuatan metodologi pengindeksan dan pencarian, namun beberapa fokus lain dari penelitian evaluasi information retrieval adalah proses kognitif pengguna, antarmuka manusia komputer, dan karakteristik database (Zhang, 2008).

Information retrieval system mengembalikan satu set dokumen sebagai jawaban atas *user's query*. Terdapat dua kategori dokumen yang dihasilkan oleh sistem Sistem Temu Kembali Informasi terkait pemrosesan query, yaitu relevant document (dokumen yang relevan dengan query) dan retrieved document (dokumen yang diterima pengguna). Hubungan antara kedua kategori ini digambarkan menggunakan diagram Venn pada gambar 2.7. (Cios, 2007):



Gambar 2.2 Relasi antara *relevant* dan *retrieve* dokumen.

Ukuran umum yang digunakan untuk mengukur kualitas dari text retrieval adalah kombinasi precision dan recall. Pada dasarnya, nilai precision dan recall bernilai antara 0-1. Oleh karena itu, suatu sistem Sistem Temu Kembali Informasi yang baik diharapkan untuk dapat memberikan nilai precision dan recall mendekati 1. Precision dan recall adalah faktor penting dalam mengevaluasi

sistem Sistem Temu Kembali Informasi, Kondisi antara precision dan recall mengakibatkan terjadi 2 situasi ekstrim berikut (Cios, 2007):

1. Precision terlalu tinggi dan recall rendah. Sistem mengembalikan beberapa dokumen dan Sistem Temu Kembali Informasi semuanya relevan, tetapi sejumlah besar dokumen relevan lain terabaikan.
2. Recall sangat tinggi dan precision relatif rendah. Sistem mengembalikan sejumlah besar dokumen yang mengikutsertakan Sistem Temu Kembali Informasi semua dokumen relevan tetapi juga mencakup sebagian besar dokumen yang tak diharapkan.

Pengujian kemampuan sistem information retrieval dilakukan dengan menghitung nilai precision dan recall berdasarkan kerelevanan sistem menampilkan dokumen sesuai dengan query. Nilai precision adalah keakurasian atau kecocokan (antara permintaan informasi dengan jawaban terhadap perrmintaan itu) jika seseorang mencari informasi di sebuah sistem, dan sistem menawarkan beberapa dokumen maka keakurasian ini sebenarnya juga adalah relevansi. Artinya, seberapa persis atau cocok dokumen tersebut untuk keperluan pencari informasi, bergantung kepada seberapa relevan dokumen tersebut bagi si pencari. Kemudian recall adalah proporsi jumlah dokumen yang dapat ditemu kembalikan oleh sebuah proses pencarian sistem Information retrieval. Berikut Tabel 2.1 parameter untuk menghitung precision dan recall.

Tabel 2.1 Parameter menghitung precision dan recall

Keterangan	**Relevan**	**Tidak Relevan**
Terambil	True positive (tp)	False positive (fp)
Tidak terambil	False negative (fn)	True negative (tn)

Rumus untuk menghitung Precision :

$$Precision = \frac{tp}{tp+fp} \quad (2.1)$$

Rumus untuk menghitung Recall :

$$Recall = \frac{tp}{tp+ fn} \quad (2.2)$$

Rumus untuk menghitung Accuracy :

$$Accuracy = \frac{tp+tn}{tp+fp+tn+fn} \quad (2.3)$$

Rumus untuk menghitung F-Measure :

$$F - measure = \frac{2 \times precision \times recall}{precision+recall} \quad (2.4)$$

2.8 Analisa Optimasi

Analisa optimasi dilakukan untuk mengetahui perhitungan manual dari metode atau algoritma yang akan digunakan untuk menentukan kemiripan deskripsi. 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 2.2 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 :

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

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

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

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

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

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

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

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

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

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

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

= **0,9779**

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

2.9 Penelitian Sebelumnya

Penelitian sebelumnya yang dilakukan oleh Azhar Firdaus, Ernawati, dan Arie Vatresia 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 2.3 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 2.4 Vektor berdasarkan Aturan metode *cosine similarity*

No	Kalimat	Vektor yang dibentuk
D1	*Komputer bantu pekerjaan manusia jadi lebih mudah*	A= (1 0 1 1 1 1 1 0 1)
D2	*Pekerjaan manusia jadi lebih mudah bila bantu oleh komputer*	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}

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

$$\vec{V}(D_2) = \frac{1\,1\,1\,1\,1\,0\,1\,1\,1}{\sqrt{8}}$$

$$\vec{V}(D_2) = \frac{1\,1\,1\,1\,1\,0\,1\,1\,1}{2{,}82842}$$

D2 = {0,35355} {0,35355 }{0,35355}{0,35355}{0,35355}{0} {0,35355} {0,35355} {0,35355}

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

Sim (D1,D2) = **0,8018**

Table 2.5 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.