BAB II

LANDASAN TEORI

2.1 Tinjauan Studi

Perancangan prototipe sistem temu kembali informasi juga dilakukan oleh Ahmad Rifa'i dalam penelitian tesisnya. Dia menerapkan beberapa algoritma fuzzy clustering untuk meningkatkan kinerja dari sistem temu kembali informasi. Tujuannya mengevaluasi kinerja algoritma FCM, H-FCM. Hasil evaluasi pada dokumen berbahasa Indonesia menunjukkan bahwa algoritma H-FCM memiliki akurasi terbaik pada presentasi kata 10 % dengan akurasi 0,93 dan kolektifitas terbaik 0,92.

Penelitian lain dilakukan Heru Adi Darmawan, Tutut Wurijanto, Akh. Masturi dengan judul *"Rancang Bangun Aplikasi Search Engine Tafsir Al-Qur'an Menggunakan Teknik Text Mining Dengan Algoritma VSM (Vector Space Model)*". Dalam penelitiannya diuji coba dengan Dalam penelitian ini media atau sarana pembelajaran yang akan dikembangkan mengadaptasi cara kerja sebuah aplikasi search engine seperti, Google, Yahoo, bing, atau Alta Vista. Mengingat sisi interaksi antara user (pengguna) dengan aplikasi harus berjalan dengan baik, dimana informasi yang disajikan lebih tepat sasaran dan serelevan mungkin sesuai dengan keinginan user. Bahan atau materi yang menjadi informasi atau output dari aplikasi ini merujuk pada ayat-ayat Al-Qur'an yang merupakan tuntunan dan pedoman bagi penganut agama Islam, yang mana agama ini di peluk sekitar 86,1% penduduk Indonesia (bersasarkan data sensus penduduk tahun 2009). Untuk dapat memberikan informasi yang lebih berkualitas dan mudah dipahami oleh user, maka informasi yang diberikan tidak sebatas pada terjemahan ayat saja, namun juga lebih mendetail hingga tafsir per ayat bahkan per kata dari ayat-ayat tersebut. Berdasarkan pertimbangan bahwa aplikasi ini dikembangkan di Indonesia, materi atau bahan yang menjadi rujukan untuk output aplikasi ini adalah karya salah seorang ahli tafsir

Indonesia ternama, Bakry Omer yang berjudul "TAFSÎR RAHMAT dan "PERCETAKAN MUTIARA".

Sedangkan pada penelitian ini dilakukan penerapan dengan judul "Pencarian Ayat Al-Qur'an Dengan Terjemahan Bahasa Indonesia Dalam Temu Kembali Informasi Menggunakan Metode Vektor Space model untuk melakukan pencarian ayat yang relevan dan mirip dengan tema (query) yang diinputkan pada obyek Al-Qur'an.

2.2 Pengertian Al-Qur'an

Definisi Al-Qur'an seperti yang terdapat pada wikipedia Al-Qur'an menurut bahasa berarti bacaan atau yang dibaca. Menurut istilah, Al-Qur'an adalah wahyu Allah SWT yang diturunkan kepada Nabi Muhammad melalui malaikat Jibril sebagaipetunjuk bagi umat manusia. Al-Qur'an diturunkan untuk menjadi pegangan bagi mereka yang ingin mencapai kebahagian dunia dan akhirat. Al-Qur'an menggunakan bahasa Arab dan merupakan mu'zizat bagi rasul. Sebagian besar ayat-ayat Al-Qur'an diturunkan di kota Mekah dan kotaMadinah. Isi yang terkandung dalam Al-Qur'an 30 juz terdapat 114 surat, 6236 ayat, 1.027.000 huruf, 86 surat makiyah, dan 28 surat madiyah.(wikipedia)

2.3 Temu Kembali Informasi

Sistem Temu Kembali Informasi atau yang disebut dengan information retrieval system, memiliki definisi seperti yang disebutkan dalam Wikipedia, *"Sistem Temu-Balik Informasi (Information Retrieval) digunakan untuk menemukan kembali informasi-informasi yang relevan terhadap kebutuhan pengguna dari suatu kumpulan informasi secara otomatis."* Information Retrieval adalah bidang di persimpangan ilmu informasi dan ilmu komputer. Berkutat dengan pengindeksan dan pengambilan informasi dari sumber informasi heterogen dan sebagian besar-tekstual. Istilah ini diciptakan oleh Mooers pada tahun 1951, yang menganjurkan bahwa diterapkan ke *aspek intelektual* deskripsi informasi dan sistem untuk pencarian. Pada dasarnya sistem temu balik informasi

adalah suatu proses untuk mengidentifikasi, kemudian memanggil (retrieve) suatu dokumen dari suatu simpanan (file), sebagai jawaban atas permintaan informasi.

Temu kembali informasi sebagai suatu proses pencarian ayat dengan menggunakan istilah luas untuk mengidentifikasi ayat yang berhubungan dengan subjek tertentu. Artinya dalam proses penemuan informasi perlu digunakan istilah-istilah tertentu.

Sistem Temu Balik Informasi merupakan sistem yang mampu melakukan pencarian informasi pada kumpulan ayat, pencarian ayat itu sendiri, pencarian untuk ayat tersebut, atau pencarian teks, suara, gambar, atau data dalam basis data dan pengambilan dokumen yang relevan dari sebuah koleksi dokumen sesuai dengan query pengguna sistem. Input dari suatu sistem temu balik informasi adalah query dari pengguna dan koleksi dokumen, dan outputnya adalah dokumen yang dianggap relevan oleh sistem. Sistem temu balik informasi ini digunakan untuk mengurangi informasi yang terlalu banyak sehingga sulit untuk dikelola. Dengan adanya sistem temu balik informasi maka diharapkan pencarian informasi dapat dilakukan dengan efektif dan memberikan hasil pencarian yang tepat \

.

2.4 Pengertian Index

Jika kita membaca sebuah buku non-fiksi, terutama buku-buku ilmu pengetahuan yang ditulis secara serius, maka pada bagian belakang kita dapat menemukan daftar kata, istilah atau frasa diurut menurut abjad. Setiap kata, istilah atau frasa ini berisi informasi tentang nomor halaman. Jika seseorang ingin tahu di mana sebuah istilah muncul dalam pembahasan buku tersebut, maka orang itu tinggal melihat nomor halaman di sebelah istilah yang bersangkutan. Inilah yang kita namakan indeks, dan tugasnya juga amat sederhanan menunjukkan halaman di mana istilah itu dapat ditemukan. Kesederhanaan indeks di setiap buku menyebabkan buku itu ‘mudah digunakan’. Semakin sederhana, semakin memudahkan pembaca.Ketika komputer digunakan untuk pembuatan indeks, maka terjadi sebuah revolusi kecil-kecilan. Sewaktu belum ada komputer, pengindeksan mutlak urusan manusia. Mesin tidak ikut campur. Ketika komputer

hadir, pengindeksan dapat dilakukan dengan dua cara: intelektual atau mekanikal. Kegiatan menganalisis dan penerjemahan (translasi) dokumen yang akan diindeks merupakan kegiatan intelektual karena mencakup kegiatan mengidentifikasi dan memilih konsep-konsep penting yang tercakup dalam sebuah dokumen, sementara fase translasi sesungguhnya merupakan pekerjaan menjadikan konsep-konsep yang sudah dipilih itu menjadi indeks berdasarkan sebuah cara yang sudah ditetapkan sebelumnya.

Kegiatan mekanikal merupakan kegiatan yang mencakup pengurutan menurut abjad dan pembuatan format entri indeks. Kalau komputer digunakan baik untuk kegiatan intelektual maupun mekanikal, maka itu artinya automatic indexing (pengindeksan secara otomatis). Kalau komputer hanya digunakan untuk mekanikal sementara manusia untuk yang intelektual, maka namanya automated indexing (pengindeksan yang diotomatiskan). Automatic indexing biasanya bergantung kepada berbagai algoritma yang antara lain paling populer adalah keyword frequencies, yaitu cara mengindeks berdasarkan kata-kata yang sering muncul.

2.5 Text Mining

Text mining memiliki definisi menambang data yang berupa text dimana sumber data biasanya didapatkan dari dokumen, dan tujuannya adalah mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen, Dengan *text mining* tugas-tugas yang berhubungan dengan penganalisaan teks dengan jumlah yang besar, penemuan pola serta penggalian informasi yang mungkin berguna dari suatu teks dapat dilakukan.Berikut Tahapan dalam teks mining:

Terdapat 5 langkah pembangunan inverted index, yaitu:

1. **Penghapusan format dan *markup* dari dalam dokumen**

   Tahap ini menghapus semua *tag markup* dan format khusus dari dokumen, terutama pada dokumen yang mempunyai banyak *tag* dan format seperti

dokumen (X)HTML. Jika isi dokumen telah berada di dalam database maka tahapan ini sering dilewatkan.

2. **Pemisahan rangkaian kata (*tokenization*)**

   Tokenization adalah tugas memisahkan deretan kata di dalam kalimat, paragraf atau halaman menjadi *token* atau potongan kata tunggal atau *termmed word*. Tahapan ini juga menghilangkan karakter-karakter tertentu seperti tanda baca dan mengubah semua *token* ke bentuk huruf kecil (*lower case*).

3. **Penyaringan (*filtration*)**

   Pada tahapan ini ditentukan *term* mana yang akan digunakan untuk merepresentasikan ayat sehingga dapat mendeskripsikan isi ayat dan membedakan ayat tersebut dari ayat lain di dalam koleksi. *Term* yang sering dipakai tidak dapat digunakan untuk tujuan ini, setidaknya karena dua hal. Pertama, jumlah ayat yang relevan terhadap suatu query kemungkinan besar merupakan bagian kecil dari koleksi. *Term* yang efektif dalam pemisahan ayat yang relevan dari ayat tidak relevan kemungkinan besar adalah *term* yang muncul pada sedikit ayat. Ini berarti bahwa *term* dengan frekuensi kemunculan tinggi bersifat *poor descriminator*. Kedua, *term* yang muncul dalam banyak ayat tidak mencerminkan definisi dari topik atau sub-topik ayat. Karena itu, *term* yang sering digunakan dianggap sebagai *stop-word* dan dihapus. Penghapusan *stop-word* dari dalam suatu koleksi ayat pada satu waktu membutuhkan banyak waktu. Solusinya adalah dengan menyusun suatu pustaka *stop-word* atau *stop-list* dari *term* yang akan dihapus. Strategi umum penentuan *stop-list* adalah mengurutkan *term* berdasarkan frekuensi koleksi (jumlah total kemunculan setiap *term* di dalam koleksi dokumen) dan memasukkan *term* yang paling sering muncul sebagai *stop-word* [Manning dan Hinrich, 2008].

4. **Konversi *term* ke bentuk akar (*stemming*)**

   Stemming adalah proses konversi term ke bentuk umumnya, sebagaimana dijelaskan sebelumnya. Dokumen dapat pula diekspansi dengan mencarikan

sinonim bagi *term-term* tertentu di dalamnya. Sinonim adalah kata-kata yang mempunyai pengertian serupa tetapi berbeda dari sudut pandang morfologis.

2.6 Stemming Bahasa Indonesia

Stemmning dapat dikatakan sebagai proses membentuk suatu kata menjadi kata dasar. Misalnya:

berkata → kata

mengatakan → kata

perkata → kata

Bebarapa algoritma dasar dalam stemming antar lain: (Wikipedia)

a) Brute force stemming. Algoritma ini adalah algoritma yang paling sederhana. Bermodalkan database kata dengan kata dasarnya, komputer dengan mudah mencari kata dasar. Namun metode ini mempunyai kelemahan yaitu jumlah database kata dan kata dasarnya harus besar. Kesalahan terjadi bila kata tidak ditemukan di database dan kemudian dianggap kata dasar, padahal bukan.

b) Menghilangkan imbuan (awalan, akhiran, sisipan). Untuk menggunakan metode ini harus tahu terlebih dahulu aturan bahasanya. Kata aka dipotong imbuhannya berdasarkan aturan bahasanya. Kesalahan terjadi bila kata tersebut adalah kata dasar yang dipotong, misalnya: perawan → awan

c) Dan masih banyak algoritma-algoritma dasar lainnya, seperti gabungan algoritma di atas, stokastik, lematasi, dll.

Untuk bahasa Indonesai beberapa algoritma yang biasanya digunakan Antara lain :

a. *Porter Stemmer*. Algoritma ini terkenal digunakan sebagai stemmer untuk bahasa inggris. Porter Stemmerdalam bahasa Indonesia akan menghasilkan keambiguan karena aturan morfologi bahasa Indonesia (Tala, 2003).

b. *Nazief &Andrian Stemmer*. Algoritma ini paling sering dibicarakan dalam *stemming* bahasa Indonesia. Algoritma ini merupakan hasil penelitianinternal UI (Universitas Indonesia) dan tidak diplublish secara umum.Algoritma ini merupakan gabungan antara algoritma menghilangkan imbuhan dan *brute force stemming.* Namun algoritma ini mempunyai dua masalah, yaitu (1) kemampuannya tergantung dari besarnya database kata dasar, (2) hasil *stemming* tidak selalu optimal untuk aplikasi information retrival.

Bila dibandingkan, untuk teks berbahasa Indonesia, Porter stemmer lebih cepat prosesnya dari pada Nazief & Adriani stemmer namun algoritma Nazief & Adriani memiliki tingkat keakuratan lebih tinggi dari pada Porter stemmer.

2.6.1 Struktur Morfologi Bahasa Indonesia

Morfologi adalah bagian dari ilmu bahasa yang menyelidiki peristiwa-peristiwa umum mengenai seluk-beluk kata terhadap fungsi dan arti kata. Morfologi kata bahasa Indonesia bisa berdiri dari struktur infleksional dan derivasional. Infleksional adalah struktur yang paling sederhana yang dinyataka dengan sufiks dimana tidak mempengarui arti sebenarnya dari kata dasar yang dilekati. Sufiks infleksional dapat dibagi menjadi 2 jenis :

1. Sufiks lah, kah, pun, tah. Sufiks ini sebenarnya adalah partikel yang tidak mempunyai arti, keberadaannya pada suatu kata adalah untuk penekanan.
   Contoh :
   dia + kah → diakah
   duduk + lah → duduklah
2. Sufiks ku, mu, nya. Sufiks ini berfungsi sebagai kata ganti kepunyaan.
   Contoh :
   komputer + ku → komputerku
   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. Penambahan sufiks infleksional tidak akan merubah

bentuk dasar dari kata berimbuan. Dengan kata lain, tidak ada penghilangan atau peleburan kata dasar pada kata berimbuan. Kata dasar dapat ditentukan dengan mudah pada struktur infleksional. Struktur derivasional dalambahasa Indonesia terdiri dari previks, sufiks, dan kombinasi dari keduanya. Prefiks yang sering dipakai adalah *ber, di, ke, meng, peng, ter.*

Contoh penggunaan prefix adalah :

ber + lari → berlari

di + ketik → diketik

ke + kasih → kekasih

meng + antar → mengantar

peng + atur → pengatur

per + tebal → pertebal

ter + baca →terbaca

Beberapa prefiks seperti ber, meng, peng, 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 infleksional pengucapan kata mungkin berubah setelah adanya penambahan prefiks. Seperti contoh menyapu yang terdiri dari prefiks meng dan kata dasar sapu. Prefiks meng berubah 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 sebelumnya, struktur derivasional juga terdiri dari konfliks, yaitu gabungan sebelumnya, struktur derivasional juga

terdiri dari konfliks, yaitu gabungan dari prefiks dan sufiks yang melekat secara bersama-sama pada suatu kata. Contoh :

per + main + an →permainan

ke + kalah + an →kekalahan

meng + ambil + i →mengambili

ber + jatuh + an →berjatuhan

Tidak semua prefiks dan sufiks bisa bekerjasama membentuk konfliks. Ada beberapa kombinasi prefiks dan sufiks yang tidak diperbolehkan. Kombinasi tersebut dapat dilihat di **Tabel 2.1**

Tabel 2.1 Pembentukan konfiks yang tidak diperbolehkan

Prefiks	Sufiks
Per	I
Di	An
Ke	i|kan
Meng	An
Peng	i|kan
Ter	An

Tabel 2.2 Aturan Pembentukan prefiks ganda

Prefiks 1	Prefiks 2
Meng	**Per**
Ke	**Ber**
Di	
Ter	

Prefiks atau konfliks dapat ditambahkan pada suatu kata yang telah terdapat konfliks atau prefiks, yang menghasilkan struktur prefiks ganda. Seperti pada pembentukan sebuah konfiks, pada pembentukan prefiks ganda, tidak semua

prefiks atau konfiks dapat ditambahkan pada kata yang telah mendapatakan prefiks atau konfiks. Ada beberapa aturan dalam urutan pembentukan prefiks ganda. Aturan-aturan tersebut dapat dilihat di **Tabel 2.2**

Struktur lain yang mungkin terjadi dalam morfologi bahasa Indonesia adalah penambahan sufiks infleksional pada struktur derivasional, yang-dinamakan multiple sufiks. Sehingga dapat disimpulkan secara umum struktur morfologi kata bahasa Indonesia adalah :

Struktur morfologi = [*prefiks* 1] + [*prefiks* 2] + kata dasar + [*sufiks*] + [kata ganti milik] + [partikel]

2.6.2 *Stemming* Bahasa Indonesia Algoritma Nazief & Andriani

Adapun langka-langka yang digunakan oleh algoritma Nazief dan Adriani yaitu sebagai berikut :

1. Kata dicari di dalam daftar kamus. Bila kata tersebut ditemukan di dalam kamus maka dapat diasumsikan kata tersebut adalah kata dasar sehingga algoritma dihentikan.
2. Bila kata di dalam langka pertama tidak ditemukan di dalam kamus, maka diperiksa apakah sufiks tersebut yaitu sebuah artikel(“-lah” atau “-kah”). Bila ditemukan maka partikel tersebut dihilangkan.
3. Pemeriksaan dilanjutkan pada kata ganti milik (“-ku”, “-mu”, “-nya”). Bila ditemukan maka kata tersebut dihilangkan.
4. Memeriksa akhiran (“-i”, “-an”). Bila ditemukan akhiran tersebut dihilangkan.

   Hingga langka ke-4 dibutuhkan ketelitian memeriksa apakah akhiran “-an” merupakan hanya bagian dari akhiran “-kan” dan memeriksa lagi apakah partikel (“-lah”, “-kah”) dan kata ganti milik (“-ku”, “-mu”, “-nya”) yang telah dihilangkan pada langka 2 dan 3 merupakan bagian dari kata dasar.
5. Memeriksa awalan (“se-”, “ke-”, “di-“, “te-“, “be”, “pe-“, “me-“). Bila ditemukan, maka awalan tersebut dihilangkan. Pemeriksaan dilakukan dengan berulang mengingat adanya kemungkinan *multi-prefix*. Langka ke-5 ini juga

membutuhkan ketelitian untuk memriksa kemungkinan peluluhan awalan, perubahan *prefix* yang disesuaikan dengan huruf awal kata dan aturan kombinasi prefix-suffix yang diperbolehkan.

6. Setelah menyelesaikan semua langka dengan sukses, maka algoritma akan mengembalikan kata dasar yang ditemukan.

Tabel 2.3 Daftar prefik yang meluluh

Jenis prefiks	**Huruf**	**Hasil peluluhan**
Pe-/me-	K	-ng-
Pe-/me-	P	-m-
Pe-/me-	S	-ny-
Pe-/me-	T	-n-

Tabel 2.4 Daftar kemungkinan perubahan prefiks

Prefiks	Perubahan
Se-	Tidak Berubah
Ke-	Tidak Berubah
di-	Tidak Berubah
Be-	Ber-
Te-	Ter-
Pe-	Pe-,pen-,pem-,peng-
Me-	Men-, Mem-,Meng-

Tabel 2.5 Daftar kombinasi prefiks dan sufiks yang tidak diperbolehkan

Prefiks	Sufiks yang tidak Diperbolehkan
Be-	-i
di-	-an
Ke-	-I, -kan
Me-	-an
Se-	-I, -kan
Te-	-an
Pe-	-kan

2.7 Pembobotan tf-idf

Pembobotan global digunakan untuk memberikan tekanan terhadap term yang mengakibatkan perbedaan dan berdasarkan pada penyebaran dari term tertentu di seluruh dokumen. Banyak skema didasarkan pada pertimbangan bahwa semakin jarang suatu term muncul di dalam total koleksi maka term tersebut menjadi semakin berbeda.

Pendekatan terhadap pembobotan global mencakup *inverse document Frequency* (*idf*), *squared idf, probabilistic idf, GF-idf, entropy.* Pendekatan *idf* merupakan pembobotan yang paling banyak digunakan saat ini. Beberapa aplikasi tidak melibatkan bobot global, hanya memperhatikan *tf,* yaitu ketika *tf* sangat kecil atau saat diperlukan penekanan terhadap frekuensi *term* di dalam suatu dokumen.

Faktor normalisasi digunakan untuk menormalkan vektor dokumen sehingga proses *similarity* tidak terpengaruh oleh panjang dari dokumen. Normalisasi ini diperlukan karena dokumen panjang biasanya mengandung perulangan *term* yang sama sehingga menaikkan *frekuensiterm* (*tf*). Dokumen panjang juga mengandung banyak *term* yang berbeda sehingga menaikkan ukuran

Kemiripan antara query dengan dokumen tersebut. Beberapa pendekatan normalisasi adalah normalisasi cosinus, penjumlahan bobot, normalisasi ke-4, normalisasi bobot maksimal dan normalisasi *pivoted unique.*

Bobot lokal suatu *term* i di dalam dokumen j (tf_{ij}) dapat didefinisikan sebagai :

$$\text{tf}_{\text{ij}} = \frac{\text{fij}}{\text{Max(fij)}} \quad (2.1)$$

Dimana f*ij* adalah jumlah berapa kali *term* i muncul di dalam dokumen j. Frekuensi tersebut dinormalisasi dengan frekuensi dari *most common term* di dalam dokumen tersebut. Atau dapat dilakukan tanpa normalisasi, sehingga nilai tf merupakan jumlah kemunculan *term* i dalam dokumen j.

$$\text{tf}_{\text{ij}} = \text{f}_{\text{ij}} \quad (2.2)$$

Bobot global dari suatu *term* i pada pendekatan *inverse document frekuency* (*idf*) dapat didefinisikan sebagai :

$$\text{id} = \frac{\text{Log10(D)}}{\text{df}}$$

Normalisasi untuk nilai Idf adalah sebgai berikut:

$$idf = \log 10(\frac{N}{Df}) + 1 \quad (2.3)$$

Dimana *df*adalah frekuensi dokumen dari *term* i dan sama dengan jumlah dokumen yang mengandung *term* i. Log_{10} digunakan untuk memperkecil pengaruhnya relatif terhadap $tf_{ij.}$

Bobot dari *term* i (w_{ij}) dihitung menggunakan ukuran *tf-idf* yang didefinisikan sebagai berikut :

$$W_{d,t} = tf_{d,t} \times idf_t \quad (2.4)$$

Dimana : i=dokumen ke i

J=kata ke-j dari kata kunci

W=bobot dokumen ke-i terhadap kata ke-j

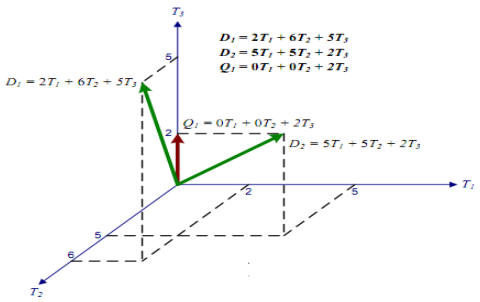
D=data

N=jumlah data

2.8 *Vector space model* (VSM)

Pada *vector space model*, setiap dokumen di dalam database dan query pengguna di presentasikan oleh suatu vektor multi-dimensi (Polettini, 2004; Cios 2007). Dimensi sesuai dengan jumlah *term* dalam dokumen yang terlibat. Pada model ini :

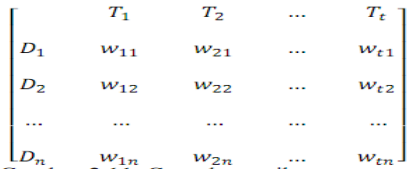
a. *Vocabulary* merupakan kumpulan semua *term* berbeda yang tersisa dari dokumen setelah *preprocessing* dan mengandung t *term* index. Term-term ini membentuk suatu ruang vektor.
b. Setiap *term* i di dalam dokumen atau *query* j, diberikan suatu bobot (*weight*) bernilai *real* w_{ij}
c. Dokumen dan *query* diekspresikan sebagai vektor t dimensi d_j = (w1, w2,....w_{tj}) dan terdapat n dokumen di dalam koleksi, yaitu j = 1, 2,....n.
   Contoh dari model ruang vektor tiga dimensi untuk dua dokumen D_1 dan D_2, satu *query* pengguna Q_1, dan tiga *term* T_1 , T_2, dan T_3 di perhatikan pada **Gambar 2.1**



Gambar 2.1 Contoh dari model ruang vektor tiga dimensi untuk dua dokumen D_1 dan D_2, satu *query* pengguna Q_1.

Dalam model ruang vektor, koleksi dokumen di presentasikan oleh *matriks term-dokumen* (atau matriks *term-frekuensi*) setiap sel dalam matriks bersesuaian dengan bobot yang diberikan dari suatu *term* dalam dokumen yang di

tentukan. Nilai nol berarti bahwa *term* tersebut tidak hadir dalam dokumen. (Cios, 2007). Pada **gambar2.2** ditunjukkan contoh *matriks term-dokument* untuk database dengan n dokumen dan t *term.*



Gambar 2.2 Ditunjukkan contoh *matriks term-dokument* untuk database dengan *n* dokumen dan t *term.*

Keberhasilan dari model VSM ini ditentukan oleh skema pembobotan terhadap suatu *term* baik untuk cakupan lokal maupun global, dan faktor normalisasi. Pembobotan lokal hanya berpedoman pada frekuensi munculnya *term* dalam suatu dokumen dan tidak melihat frekuensi kemunculan *term* tersebut di dalam dokumen lainnya.

2.9 Perhitungan Tingkat Relevansi

Penentuan relevansi dokumen dengan query ataupun antar dokumen dipandang sebagai pengukuran kesamaan (similiarity measure) antara vektor dokumen dengan vektor query. *Semakin "sama" suatu vektor dokumen dengan vektor query maka dokumen dapat dipandang semakin relevan dengan query.* Salah satu pengukuran derajat kemiripan antar dokumen dengan query ataupun antar dua dokumen yang paling populer adalah dengan cosine similiarity().Pada metode ini bobot term yang terkandung pada setiap dokumen direpresentasikan dalam sebuah vektor. Misalnya $\bar{x} = \{x_1, x_2, ..., x_t\}$ dan dokumen y direpresentasikan oleh vektor $\bar{y} = \{y_1, y_2, ...y_t\}$.

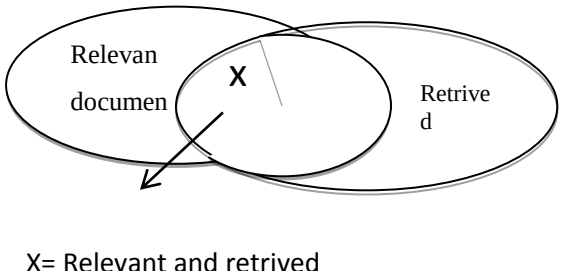
Korelasi ini dapat dikuantifikasi dengan sudut cosine antar dua vektor pada persamaan (Cios, 2007):

$$sim(x, y) = \frac{\bar{x} \bullet \bar{y}}{|\bar{x}| \times |\bar{y}|} = \frac{\sum_{i=1}^{t} W_{i,x} \times W_{i,y}}{\sqrt{\sum_{i=1}^{t} W^2{}_{i,x}} \times \sqrt{\sum_{i=1}^{t} W^2{}_{i,y}}} \qquad (2.5)$$

Dengan $|\vec{x}|$ dan $|\vec{y}|$ merupakan norms x dan y. Nilai sim(x,y) bervariasi dari 0 sampai 1. Nilai ini menunjukkan bahwa semakin tinggi nilai sim(x,y), maka makin besar kemiripan dua vektor tersebut.

2.10 Recall & Precision

Ada beberapa alasan yang berbeda mengapa tahap evaluasi IR (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 pencarian 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.Terdapat dua kategori dokumen yang dihasilkan oleh sistem IR terkait pemrosesan query, yaitu relevant ayat (ayat yang relevan dengan query) dan retrieved document (ayat yang diterima pengguna). Hubungan antara kedua kategori ini digambarkan menggunakan diagram Venn pada **gambar 2.3**



Gambar 2.3 Relasi antara relevant dan retrieved dokumen.

Nilai precission, recall dan akurasi dapat dihitung dengan menggunakan tabel ketergantungan(Tabel 2.7)(Manning, 2008: 155)

Tabel 2.6 Tabel Ketergantungan

	Relevant	nonRelevant
Retrieved	True positive (tp)	False positive (fp)
Non retrieved	False negative (fn)	True negative (tn)

Rumus menentukan precission :

$$\Pr ecission = \frac{tp}{(tp + fp)} \tag{2.6}$$

Rumus menentukan nilai Recall :

$$\mathrm{Re} call = \frac{tp}{(tp + fn)} \tag{2.7}$$

Rumus menentukan nilai Akurasi :

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{2.8}$$