BAB II

LANDASAN TEORI

2.1 Perpustakaan Digital

Menurut Putu Laxman Pendit dalam bukunya Perpustakaan Digital dari a-z, fenomena perpustakaan digital memperlihatkan perluasan upaya manusia di bidang informasi dan kepustakawanan dengan melibatkan berbagai disiplin, seperti manajemen data, *information retrieval*, manajemen dokumen, sistem informasi, teknologi web, pengolahan citra, kecerdasan buatan, interaksi manusia-komputer dan preservasi digital. Akibat dari sifatnya yang multi disipliner ini, konsep tentang perpustakaan digital juga disajikan dan dipahami secara beragam pula.

Definisi perpustakaan menurut DLF (*Digital Library Federation*) adalah organisasi-organisasi yang menyediakan sumber daya, termasuk pegawai yang terlatih khusus, untuk memilih, mengatur, menawarkan akses, memahami, menyebarkan, menjaga integritas, dan memastikan keutuhan karya digital, sedemikian rupa sehingga koleksi tersedia dan terjangkau secara ekonomis oleh sebuah atau sekumpulan komunitas yang membutuhkannya.

Association of Research Libraries (1995) telah mendefinisikan lima elemen berikut dari berbagai definisi perpustakaan digital:

1. Perpustakaan Digital buka merupakan sebuah entitas tunggal;
2. Perpustkaan Digital membutuhkan teknologi untuk menghubungkan sumber daya informasi yang banyak;
3. Keterkaitan antara banyak perpustakaan digital dan layanan informasi yang transparan kepada pengguna akhir;

4. Tujuan utama untuk akses universal untuk perpustakaan digital dn layanan informasi;
5. Koleksi dari perpustakaan digital tidak terbatas pada dokumen pengganti mereka memperpanjang artefak digital yang tidak dapat diwakili atau didistribusikan dalam format cetak.

2.2 Sistem Temu Kembali Informasi (*Information Retrieval*)

Sistem temu kembali informasi merupakan suatu sistem yang menemukan (*retrieve*) informasi yang sesuai dengan kebutuhan *user* dari kumpulan informasi secara otomatis. Prinsip kerja sistem temu kembali informasi jika ada sebuah kumpulan dokumen dan seorang *user* yang memformulasikan sebuah pertanyaan (*request* atau *query*). Jawaban dari pertanyaan tersebut adalah sekumpulan dokumen yang relevan dan membuang dokumen yang tidak relevan (Salton, 1989).

Menurut sebuah ensiklopedia, temu kembali informasi adalah seni dan ilmu dalam pencarian informasi di sekumpulan dokumen-dokumen, pencarian informasi di dokumen itu sendiri, pencarian metadata yang menjelaskan sekumpulan dokumen, atau pencarian di dalam basis data (WIKIPEDIA, 2005). Nantinya hasil akhir dari temu kembali informasi adalah sebuah sistem yang dapat melakukan penemu kembalian informasi atau disebut sistem temu kembali informasi.

Menurut Lancaster (1968) di dalam Rijsbergen (1979): "sebuah sistemtemu-kembali informasi tidak memberitahu (yakni tidak mengubah pengetahuan) pengguna mengenai masalah yang ditanyakannya. Sistem tersebut hanya memberitahukan keberadaan (atau ketidakberadaan) dan keterangan dokumen- dokumen yang berhubungan dengan permintaannya".

Sistem temu kembali informasi akan mengambil salah satu dari kemungkinan tersebut. Sistem temu kembali informasi dibagi dalam dua komponen utama yaitu sistem pengindeksan (*indexing*) menghasilkan basis data

sistem dan temu kembali merupakan gabungan dari *user interface* dan *look-up-table*. Sistem temu kembali informasi didesain untuk menemukan dokumen atau informasi yang diperlukan oleh *user*.

Sistem Temu Kembali Informasi bertujuan untuk menjawab kebutuhan informasi *user* dengan sumber informasi yang tersedia dalam kondisi seperti sebagai berikut (Salton, 1989);

a. Mempresentasikan sekumpulan ide dalam sebuah dokumen menggunakan sekumpulan konsep.
b. Terdapat beberapa pengguna yang memerlukan ide, tapi tidak dapat mengidentifikasikan dan menemukannya dengan baik.
c. Sistem temu kembali informasi bertujuan untuk mempertemukan ide yang dikemukakan oleh penulis. Dalam dokumen dengan kebutuhan informasi pengguna yang dinyatakan dalam bentuk *keyword query*/istilah penelusuran.

Fungsi utama sistem temu kembali informasi (Salton, 1989):

a. Mengidentifikasi sumber informasi yang relevan dengan minat masyarakat pengguna yang ditargetkan.
b. Menganalisis isi sumber informasi (dokumen).
c. Merepresentasikan isi sumber informasi dengan cara tertentu yang memungkinkan untuk dipertemukan dengan pertanyaan pengguna.
d. Merepresentasikan pertanyaan (*query*) *user* dengan cara tertentu yang memungkinkan untuk dipertemukan sumber informasi yang terdapat dalam basis data.
e. Mempertemukan pernyataan pencarian dengan data yang tersimpan dalam basis data.
f. Menemu kembalikan informasi yang relevan.
g. Menyempurnakan unjuk kerja sistem berdasarkan umpan balik yang diberikan oleh *user*.

Informasi atau data yang dicari dapat berupa berupa teks, *image*, *audio*, video dan lain-lain. Koleksi data teks yang dapat dijadikan sumber pencarian juga

dapat berupa pesan teks, seperti *e-mail*, *fax*, dan dokumen berita, bahkan dokumen yang beredar di internet. Dengan jumlah dokumen koleksi yang besar sebagai sumber pencarian, maka dibutuhkan suatu sistem yang dapat membantu *user* menemukan dokumen yang relevan dalam waktu yang singkat dan tepat.

Dalam teknologi informasi terdapat istilah *data retrieval*, selain *information retrieval*. Dua hal ini sangatlah berbeda. *Data retrieval* secara umum menentukan dokumen yang tepat dari suatu koleksi data, yang isi dokumen tersebut mengandung *keyword* di dalam *query user*, tidak akan pernah cukup untuk memenuhi kebutuhan informasi *user*. Berbeda dengan *data retrieval*, *user* dari sistem *Information Retrieval* lebih memperhatikan dalam mendapatkan (*retrieve*) informasi melalui subyek, daripada *retrieve* data berdasarkan *query* yang diberikan, karena *user* tidak mau tahu bagaimana proses yang sedang berlangsung.

Tabel 2.1 Perbedaan *Information Retrieval* dan *Data Retrieval*

Information Retrieval	***Data Retrieval***
Berhubungan dengan teks bahasa umum yang tidak selalu terstruktur dan ada kemungkinan memiliki kerancuan arti	Berhubungan dengan data, yang mana semantik strukturnya sudah terdefinisikan
Informasi yang diambil mengenai subjek atau topik	Isi dokumen/data mengandung bagian dari *keyword*
Semantik sering kali hilang	Semantik terdefinisi dengan baik
Kesalahan kecil masih bisa ditorensi	Kesalahan kecil/tunggal dari suatu obyek menunjukkan kegagalan

Model yang terdapat dalam *Information Retrieval* terbagi dalam 3 model besar, yaitu:

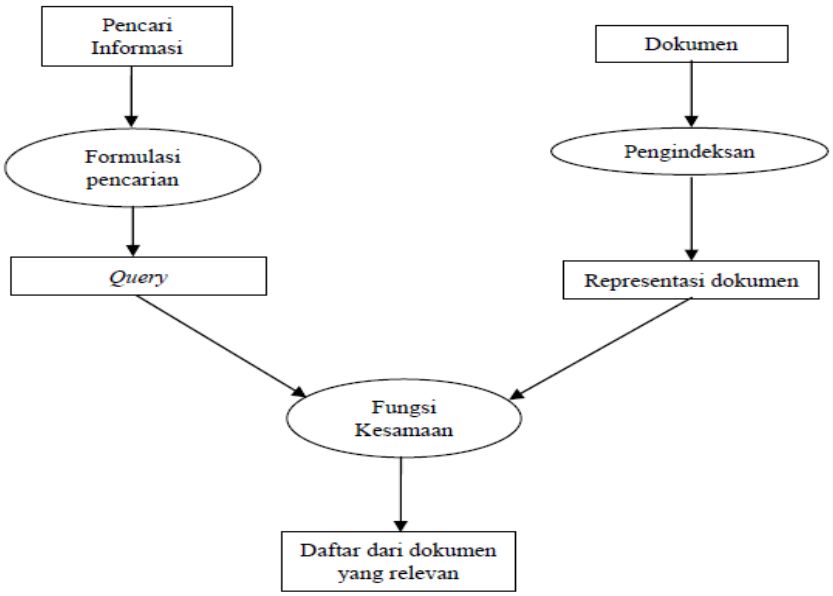
1. *Set-theoretic models*, model merepresentasikan dokumen sebagai himpunan kata atau frase. Contoh model ini ialah *standard Boolean model* dan *extended Boolean model*.
2. *Algebratic model*, model merepresentasikan dokumen dan *query* sebagai vektor atau matriks *similarity* antara vektor dokumen dan vektor *query* yang direpresentasikan sebagai sebuah nilai skalar. Contoh model ini ialah *vector space model* dan *latent semantic indexing* (LSI).
3. *Probabilistic model*, model memperlakukan proses pengembalian dokumen sebagai sebuah *probabilistic inference*. Contoh model ini ialah penerapan teorema bayes dalam model probabilistik.

Proses dalam *Information Retrieval* dapat digambarkan sebagai sebuah proses untuk mendapatkan *relevant documents* dari *collection documents* yang ada melalui pencarian *query* yang diinputkan *user*.

2.2.1 Arsitektur Sistem Temu Kembali Informasi

Sistem Temu Kembali Informasi terdapat proses yang berjalan yang terbagi dalam 2 proses. Proses pertama adalah pencarian informasi oleh *user* dengan memasukkan suatu kata kunci yang nantinya akan diterjemahkan oleh sistem dalam bentuk *query*. Proses kedua adalah proses pengindeksan dokumen di dalam *database*. Pada bagian ini dokumen akan direpresentasikan ke dalam bentuk indeks yang nantinya akan dibandingkan dengan *query* dari pengguna menggunakan fungsi kesamaan untuk mendapatkan dokumen yang relevan.

Berikut adalah arsitektur sederhana dari sistem temu kembali informasi dapat dilihat pada gambar 2.1:



Gambar 2.1 Arsitekstur Sederhana *Information Retrieval (Ingwersen, 1992)*

Perlu diingat bahwa pencarian sebuah informasi di dalam sistem temukembali informasi belum tentu mengembalikan seluruh dokumen yang relevan. Bisa hanya sebagian atau tidak sama sekali. Sistem temu kembali informasi mungkin tidak memberikan hasil apapun jika memang tidak ditemukan dokumen yang relevan.

2.2.2 Tahapan Proses dalam Sistem Temu Kembali Informasi

Adapun beberapa tahapan proses dari Sistem temu kembali informasi dalam menemukan dokumen yang relevan terhadap *query* pengguna adalah sebagai berikut:

1. Pengindeksan

Indeks adalah bahasa yang digunakan di dalam sebuah buku konvensional untuk mencari informasi berdasarkan kata atau istilah yang mengacu ke dalam suatu halaman. Dengan menggunakan indeks si pencari informasi dapat dengan mudah menemukan informasi yang diinginkannya. Pada sistem temu kembali

informasi, indeks ini nantinya yang digunakan untuk merepresentasikan informasi di dalam sebuah dokumen.

Elemen dari indeks adalah istilah indeks (*index term*) yang didapatkan dari teks yang dipecah di dalam sebuah dokumen. Elemen lainnya adalah bobot istilah (*term weighting*) sebagai penentuan rangking dari kriteria relevan sebuah dokumen yang memiliki istilah yang sama.

Adapun tahap-tahap dari proses pengindeksan dokumen adalah sebagai berikut:

a. *Tokenizing*

Pada tahap ini isi dokumen akan dipecah menjadi unit-unit yang lebih kecil berupa kata, frasa atau kalimat dengan cara menghilangkan seluruh tanda baca yang ada pada dokumen sehingga menghasilkan kata yang berdiri sendiri. Unit tersebut biasanya disebut sebagai token. Sedangkan algoritma untuk memecahkan kumpulan kalimat atau frasa menjadi token disebut *tokenizer*. Token seringkali disebut sebagai istilah (*term*) atau kata, sebagai contoh sebuah token merupakan suatu urutan karakter dari dokumen tertentu yang dikelompokkan sebagai unit semantik yang berguna untuk diproses (Salton, 1989).

b. *Filetring*

Tahap *filtering* adalah tahap pengambilan kata-kata yang penting dari hasil *tokenizing*. Pada tahap ini terdapat dua proses yaitu eliminasi *stopwords* dan pengambilan *wordlist*. Eliminasi *stopwords* yaitu penyaringan (*filtering*) terhadap kata-kata yang tidak memiliki arti atau tidak layak untuk dijadikan sebagai kata kunci dalam pencarian dokumen sehingga kata-kata tersebut dapat dihilangkan dari dokumen. Sedangkan *wordlist* adalah daftar kata yang mungkin digunakan sebagai kata kunci dalam pencarian dokumen. Eliminasi *stopwords* memiliki banyak keuntungan, yaitu akan mengurangi *space* pada tabel *term index* hingga 40% atau lebih (Baeza, 1999).

Proses *filtering* menggunakan daftar *stopword* yang digunakan oleh Tala (2003), merupakan *stopword* dalam Bahasa Indonesia yang berisi kata-kata seperti; ada, yang, ke, kepada, dan lain sebagainya.

c. *Stemming*

Stemming adalah proses mengubah kata menjadi kata dasarnya dengan menghilangkan imbuhan-imbuhan pada kata dalam dokumen. Dalam penelitian ini, proses *stemming* menggunakan Algoritma Porter.

Berikut ini adalah algoritma Porter untuk proses *stemming* (Agusta, 2009),

1) Hapus Partikel
2) Hapus *Possesive Pronoun*
3) Hapus awalan pertama. Jika tidak ada lanjutkan ke langkah 4a, jika ada maka lanjutkan ke langkah 4b.
4) a. Hapus awalan kedua, lanjutkan ke langkah 5a.
   b. Hapus akhiran. Jika tidak ditemukan maka kata tersebut diasumsikan sebagai *root word*. Jika ditemukan maka lanjutkan kelangkah 5b.
5) a. Hapus akhiran. Kemudian kata akhir diasumsikan sebagai *root word*.
   b. Hapus awalan kedua. Kemudian kata akhir diasumsikan sebagai *root word*.

Tabel 2.2 Aturan untuk *Infectional Particle*

Akhiran	*Replacement*	*Additional Condition*
-kah	*Null*	*Null*
-lah	*Null*	*Null*
-pun	*Null*	*Null*

Tabel 2.3 Aturan untuk *Infectional Possesive Pronoun*

Akhiran	*Replacement*	*Additional Condition*
-ku	*Null*	*Null*
-mu	*Null*	*Null*
-nya	*Null*	*Null*

Tabel 2.4 Aturan untuk *First Order Derivational Prefix*

Akhiran	*Replacement*	*Additional Condition*
-meng	*Null*	*Null*
-meny	S	V…*
-men	*Null*	*Null*
-mem	P	V…
-mem	*Null*	*Null*
-me	*Null*	*Null*
-peng	*Null*	*Null*
-peny	S	V…
-pen	*Null*	*Null*
-pem	P	V…
-pem	*Null*	*Null*
-di	*Null*	*Null*
-ter	*Null*	*Null*
-ke	*Null*	*Null*

Tabel 2.5 Aturan untuk *Second Order Derivational Prefix*

Akhiran	Replacement	Additional Condition
-ber	*Null*	*Null*
-bel	*Null*	*Ajar*
-be	*Null*	k*er
-per	*Null*	*Null*
-pel	*Null*	*Ajar*
-pe	*Null*	*Null*

Tabel 2.6 Aturan untuk *Derivational Suffix*

Akhiran	*Replacement*	*Additional Condition*

-kan	*Null*	*Prefix* bukan anggota {ke, peng}
-an	*Null*	*Prefix* bukan anggota {di, meng, ter}
-i	*Null*	*Prefix* bukan anggota {ber, ke, peng}

d. *Inverted Index*

Inverted index adalah salah satu mekanisme untuk pengindeksan sebuah koleksi teks yang digunakan untuk mempercepat proses pencarian. Struktur dari*inverted index* terdiri dari dua elemen yaitu kosakata dan posisinya di dalamsebuah dokumen (Baeza-Yates dan Ribeiro-Neto, 1999). Posisi dari sebuah istilahdi dalam indeks pada sebuah buku, diterjemahkan dalam bentuk nomor halaman.

Pada *inverted index*, setiap istilah di masukan ke dalam *inverted list* yang menyimpan daftar dari istilah yang menujuk ke sejumlah dokumen yang memiliki istilah tersebut. *Inverted list* juga kadang-kadang disebut *posting list* (Witten et all, 1999).

Penggunaan *inverted index* di dalam sistem temu-kembali informasi memiliki kelemahan yaitu lambat di dalam pengindeksan, tetapi cepat di dalam proses pencarian informasi.

2. *Term Weighting*

Pembobotan istilah (*term weighting*) adalah proses pembobotan pada istilah. Istilah di dalam suatu indeks harus bisa membedakan kepentingan darisebuah dokumen pada sebuah informasi. Caranya yaitu dengan pemberian bobot kepada sebuah istilah terhadap suatu dokumen. Semakin tinggi bobot dari sebuah istilah maka semakin penting istilah tersebut dibandingkan dengan istilah lainnyadi dalam sebuah dokumen. Bobot dari istilah ini dicantumkan pada *inverted index* untuk digunakan dalam proses penemu-kembalian dokumen.

Pembobotan dasar dilakukan dengan menghitung frekuensi kemunculan *term* dalam dokumen. Frekuensi kemunculan (*term frequency*) merupakan petunjuk sejauh mana *term* tersebut mewakili isi dokumen. Semakin besar kemunculan suatu *term* dalam dokumen akan memberikan nilai kesesuian yang semakin besar.

Metode TF-IDF (*Term FrequncyInverse Document Frequency*) merupakan suatu cara untuk memberikan bobot hubungan suatu kata (*term*) terhadap dokumen (Robertson, 2005). Metode ini menggabungkan dua konsep untuk perhitungan bobot yaitu, frekuensi kemunculan sebuah kata didalam sebuah dokumen tertentu dan *inverse* frekuensi dokumen yang mengandung kata tersebut.

Terdapat beberapa cara atau metode dalam melakukan pembobotan istilah pada metode TF-IDF, yaitu melalui skema pembobotan *query* dan dokumen. Pada teknik pembobotan ini, bobot istilah telah dinormalisasi. Dalam menentukan bobot suatu istilah tidak hanya berdasarkan frekuensi kemunculan istilah di satu dokumen, tetapi juga memperhatikan frekuensi terbesar pada suatu istilah yang dimiliki oleh dokumen bersangkutan. Hal ini untuk menentukan posisi relatif bobot dari istilah dibanding dengan istilah-istilah lain didokumen yang sama. Selain itu teknik ini juga memperhitungkan jumlah dokumen yang mengandung istilah yang bersangkutan dan jumlah keseluruhan dokumen.

Hal ini berguna untuk mengetahui posisi relatif bobot istilah bersangkutan pada suatu dokumen dibandingkan dengan dokumen-dokumen lain yang memiliki istilah yang sama. Sehingga jika sebuah istilah mempunyai frekuensi kemunculan yang sama pada dua dokumen belum tentu mempunyai bobot yang sama.

2.3 *Vektor Space Model*

Vector Space Model (VSM) adalah metode untuk melihat tingkat kedekatan atau kesamaan (*similarity*) *term* dengan cara pembobotan *term*. Dokumen dipandang sebagai sebuah vektor yang memiliki *magnitude* (jarak) dan *direction* (arah). Pada *Vector Space Model*, sebuah istilah direpresentasikan

dengan sebuah dimensi dari ruang vektor. Relevansi sebuah dokumen ke sebuah *query* didasarkan pada similaritas diantara vektor dokumen dan vektor *query (Baeza, 1999).*

Pada *Information Retrieval System* terdapat beberapa metode yang digunakan dalam *Searching* salah satunya adalah dengan merepresentasikan proses *Searching* menggunakan Model Ruang Vektor. Model Ruang Vektor dibuat berdasarkan pemikiran bahwa isi dari dokumen ditentukan oleh kata-kata yang digunakan dalam dokumen tersebut. Model ini menentukan kemiripan (*similarity*) antara dokumen dengan *query* dengan cara merepresentasikan dokumen dan *query* masing-masing ke dalam bentuk vektor. Tiap kata yang ditemukan pada dokumen dan *query* diberi bobot dan disimpan sebagai salah satu elemen vektor.

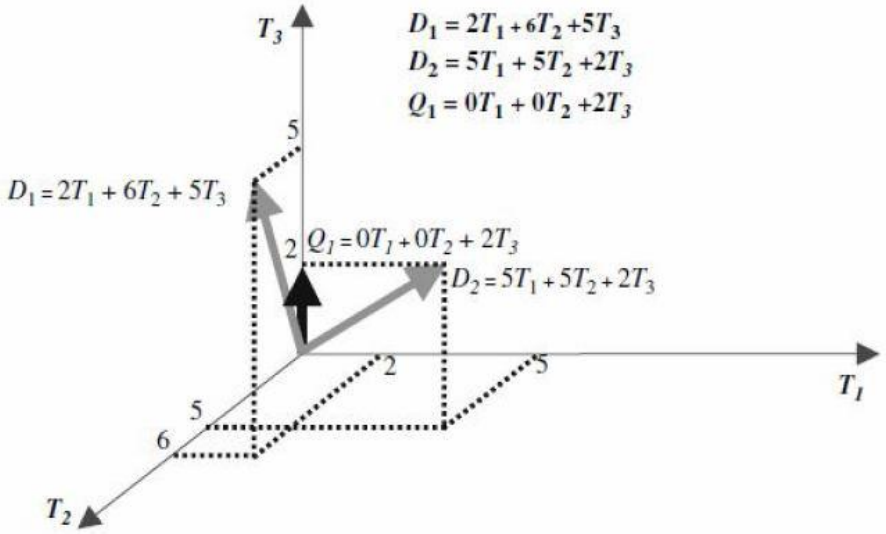
Bobot istilah yang akhirnya digunakan untuk menghitung tingkat kesamaan antara setiap dokumen yang tersimpan dalam sistem dan permintaan *user*. Dokumen yang terambil disortir dalam urutan yang memiliki kemiripan, model vektor dan memperhitungkan pertimbangan dokumen yang relevan dengan permintaan *user*. Hasilnya adalah himpunan dokumen yang terambil jauh lebih akurat (dalam arti sesuai dengan informasi yang dibutuhkan oleh *user*).

Pada sistem IR, kemiripan antar dokumen didefinisikan berdasarkan representasi *bag of words* dan dikonversi ke suatu model ruang vektor (*vector space model*, VSM). Model ini diperkenalkan oleh (Salton, 1983) dan telah digunakan secara luas. 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* wij.

c. Dokumen dan *query* diekspresikan sebagai vektor t dimensi dj = (w1, w2, ..., wtj) dan terdapat n dokumen di dalamkoleksi, yaitu j = 1, 2, ..., n.

Contoh dari model ruang vektor tiga dimensi untuk dua dokumen D1 dan D2, satu *query* pengguna Q1, dan tiga term T1, T2 dan T3 diperlihatkan pada Gambar 2.2 berikut ini :



Gambar 2.2 Representasi Dokumen dan *Query* pada Model Ruang Vektor

Dalam model ruang vektor, koleksi dokumen direpresentasikan oleh matriks term document (atau matriks *term-frequency*). Setiap sel dalam matriks bersesuaian dengan bobot yang diberikan dari suatu *term* dalam dokumen yang ditentukan. Nilai nol berarti bahwa *term* tersebut tidak hadir di dalam dokumen.

Keberhasilan dari model VSM ini ditentukan oleh skema pembobotan terhadap suatu term baik untuk cakupan local 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. Pendekatan dalam pembobotan lokal yang paling banyak diterapkan adalah *term frequency* (tf) meskipun terdapat skema lain

seperti pembobotan biner, *augmented normalized tf,* logaritmik tf dan logaritmik alternatif.

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. Pemanfaatan pembobotan ini dapat menghilangkan kebutuhan *stopword removal* karena *stopword* mempunyai bobot global yang sangat kecil.

Namun pada prakteknya lebih baik menghilangkan *stopword* di dalam fase *pre processing* sehingga semakin sedikit *term* yang harus ditangani. Pendekatan terhadap pembobotan global mencakup *inverse document frequency (idf), squared idf, probabilistic idf,* TF-IDF, *entropy*. Pendekatan IDF merupakan pembobotan yang paling banyak digunakan saat ini.

Faktor normalisasi digunakan untuk menormalkan vektor dokumen sehingga proses *retrieval* tidak terpengaruh oleh panjang dari dokumen. Normalisasi ini diperlukan karena dokumen panjang biasanya mengandung perulangan *term* yang sama sehingga menaikkan frekuensi *term* (tf). Dokumen panjang juga mengandung banyak *term* yang berbeda sehingga menaikkan ukuran kemiripan antara *query* dengan dokumen tersebut, meningkatkan peluang di-*retrieve*nya dokumen yang lebih panjang. Beberapa pendekatan normalisasi adalah normalisasi cosinus, penjumlahan bobot, normalisasi ke-4, normalisasi bobot maksimal dan normalisasi *pivoted unique*.

Proses perhitungan VSM melalui tahapan pembobotan kata *perhitungan term frequency (tf)* menggunakan persamaan (1):

$$\text{tfi}_j = \frac{f_{ij}}{max_i(f_{ij})} \qquad (2.1)$$

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. Bobot global dari suatu *term* i pada pendekatan *inverse document frequency* (idfi) dapat didefinisikan sebagai,

$$\text{idf}_i = \log \frac{N}{df_i} \qquad (2.2)$$

Dengan idf_i adalah *inverse document frequency*, Nadalah jumlah dokumen yang terambil oleh sistem, dan dfi adalah banyaknya dokumen dalam koleksi dimana *term* t_i muncul di dalamnya, maka Perhitungan idfi digunakan untuk mengetahui banyaknya *term* yang dicari (dfi) yang muncul dalam dokumen lain yang ada pada database (korpus).

Perhitungan *Term Frequency Inverse Document Frequency* (TFIDF), menggunakan persamaan (3):

$$\text{Wij} = tf_{ij}\ xidf_i \qquad (2.3)$$

Dengan W_{ij} adalah bobot dokumen, N adalah Jumlah dokumen yang terambil oleh sistem, $tf_{i,j}$ adalah banyaknya kemunculan *term* ti pada dokumen dj, dan dfi adalah banyaknya dokumen dalam koleksi dimana *term* ti muncul di dalamnya. Bobot dokumen (Wij) dihitung untuk didapatkannya suatu bobot hasil perkalian atau kombinasi antara *term frequency* (tfi, j) dan *Inverse Document Frequency* (dfi).

Pengukuran *Cosine Similarity* (menghitung nilai kosinus sudut antara dua vector) menggunakan persamaan (7):

$$sim(q, dj) = \frac{\sum_{i=1}^{t} (W_{iq}.W_{ij})}{\sqrt{\sum_{j=1}^{t} (W_{iq})^2}.\sqrt{\sum_{i=1}^{t} (W_{ij})^2}} \qquad (2.4)$$

Similaritas antara *query* dan dokumen atau Sim(q,dj) berbanding lurus terhadap jumlah bobot *query* (q) dikali bobot dokumen (dj) dan berbanding terbalik terhadap akar jumlah kuadrat q (|q|) dikali akar jumlah kuadrat dokumen (|dj|). Perhitungan similaritas menghasilkan bobot dokumen yang mendekati nilai 1 atau menghasilkan bobot dokumen yang lebih besar dibandingkan dengan nilai yang dihasilkan dari perhitungan *inner product*.

2.4 *Hierarchical Agglomerative Clustering*

Metode pembentukan klaster biasanya dikategorikan menurut tipe dari struktur klaster yang dihasilkan. Secara umum metode klaster terbagi menjadi dua, yaitu metode *Non-Hierarchical Clustering* (klastering non-hirarkhis) dan metode *Hierarchical Clustering* (klastering hirarkhis). Metode non-hirarkhis disebut juga metode partisi, yaitu membagi serangkaian data yang terdiri dari n obyek ke dalam k klaster (k<n) yang tidak saling tumpang tindih (*overlap*), dimana nilai k telah ditentukan sebelumnya. Salah satu prosedur pengelompokkan pada non hirarkhis adalah dengan menggunakan metode k-means. Metode ini merupakan metode pengelompokkan yang bertujuan untuk mengelompokkan objek sedemikian hingga jarak tiap-tiap objek ke pusat kelompok di dalam suatu kelompok adalah minimum.

Metode klaster yang kedua adalah metode *Hierarchical Clustering* (klastering hirarkhis). Metode pengelompokkan hirarkhis biasanya digunakan apabila belum ada informasi jumlah kelompok yang akan dipilih. Arah pengelompokkan bisa bersifat *divisive* (*top to down*) artinya dari 1 klaster sampai menjadi k buah klaster atau bersifat *agglomerative* (*bottom up*) artinya dari n klaster (dari n-buah data yang ada) menjadi k buah klaster. Teknik hirarkhis (*hierarchical methods*) adalah teknik klastering membentuk kontruksi hirarki atau berdasarkan tingkatan tertentu seperti struktur pohon. Dengan demikian proses pengelompokkannya dilakukan secara bertingkat atau bertahap.

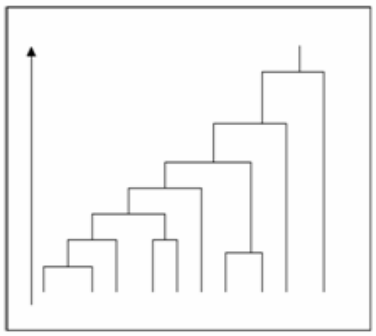
Hierarchical Agglomerative Clusstering adalah salah satu algoritma klastering yang dapat digunakan untuk mengklaster dokumen *(document clustering*). Dari teknik *Hierarchical Agglomerative Clustering*, dapat dihasilkan suatu kumpulan partisi yang berurutan, dimana dalam kumpulan tersebut terdapat:

1. Klaster-klaster yang mempunyai poin-poin individu. Klaster-klaster iniberada di level yang paling bawah.
2. Sebuah klaster yang didalamnya terdapat poin-poin yang dipunyai semua klaster didalamnya. Single klaster ini berada di level yang paling atas.

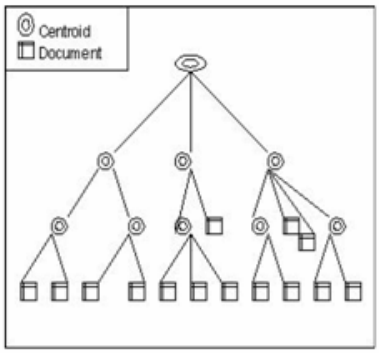
Pembentukan klaster dokumen dalam sistem temu kembali informasi dengan metode hirarkhis adalah sebagi berikut:

1. Mengidentifikasi dua dokumen yang paling mirip dan menggabungkannya menjadi sebuah klaster.
2. Mengidentifikasi dan menggabungkan dua dokumen yang paling mirip berikutnya menjadi sebuah klaster sampai semua dokumen tergabung dalam klaster-klaster yang terbentuk.
3. Proses penelusuran dokumen dilakukan dengan cara mencocokkan *query* dengan *centroid*. *Centroid* merupakan dokumen *parent* pada masing-masing klaster dokumen Berikutnya dokumen yang beradadalam satu klaster dengan centroid akan ditampilkan sebagai hasil *query*.

Hasil keseluruhan dari algoritma *Hierarchical Clustering* secara grafik dapat digambarkan sebagai *tree*, yang disebut dengan dendogram. *Tree* ini secara grafik menggambarkan proses penggabungan dari klaster-klaster yang ada, sehingga menghasilkan klaster dengan level yang lebih tinggi. Cabang-cabang dalam pohon menyajikan *cluster*. Kemudian cabang-cabang bergabung pada *node* yang posisinya sepanjang sumbu jarak (similaritas) menyatakan tingkat di mana penggabungan terjadi. Gambar 2.3 dan 2.4 memperlihatkan struktur dendogram dan diagram pohon untuk klastering hirarkhis.



Gambar 2.3 Struktur Dendogram dan Diagram Pohon UntukKlastering Hirarkhis



Gambar 2.4 Dendogram dan Struktur Pohon dari *Hierarchical Clustering* (Salton, 1989)

Kemiripan antar dokumen ditentukan dengan mengukur jarak antar dokumen. Dua dokumen yang mempunyai jarak paling kecil dikatakan mempunyai kemiripan paling tinggi dan dikelompokkan ke dalam satu klaster yang sama. Sebaliknya dua dokumen yang mempunyai jarak paling besar dikatakan mempunyai kemiripan paling rendah, dan dimasukkan ke dalam klaster yang berbeda.

2.4.1 Metode *Hierarchical Agglomerative Clustering*

Metode *Hierarchical Agglomerative Clustering* adalah metode yang menggunakan strategi desain *Bottom-Up* yang dimulai dengan meletakkan setiap obyek sebagai sebuah klaster tersendiri (atomic klaster) dan selanjutnya

menggabungkan *atomic* klaster-*atomic* klaster tersebut menjadi klaster yang lebih besar dan lebih besar lagi sampai akhirnya semua obyek menyatu dalam sebuah klaster atau proses dapat pula berhenti jika telah mencapai batasan kondisi tertentu. Langkah-langkah dalam algoritma *Hierarchical Agglomerative Clustering*:

1. Mulai dengan N klaster, setiap klaster mengandung entiti tunggal dan sebuah matriks simetrik dari jarak (similarities) D = {dik} dengan tipe matrik adalah NxN.
2. Cari matriks jarak untuk pasangan klaster yang terdekat (paling mirip), yaitu dengan mencari similaritas terbesar. Misalkan jarak antara klaster U dan V yang paling mirip adalah duv.
3. Gabungkan klaster U dan V. Label klaster yang baru dibentuk dengan (UV). Update entries pada matrik jarak dengan cara :
   a. Hapus baris dan kolom yang bersesuaian dengan klaster U dan V
   b. Tambahkan baris dan kolom yang memberikan jarak-jarak antara klaster (UV) dan klaster-klaster yang tersisa

2.4.2 Metode *Single Linkage*

Input untuk algoritma single linkage bisa berwujud jarak atau similarities antara pasangan-pasangan dari objek-objek. Kelompok-kelompok dibentuk dari entities individu dengan menggabungkan jarak paling pendek atau similarities (kemiripan) yang paling besar. Pada awalnya, kita harus menemukan jarak terpendek dalam *D* = {*duv*} dan menggabungkan objek-objek yang bersesuaian misalnya *U* dan *V*, untuk mendapatkan (*UV*). Untuk langkah (3) dari algoritma di atas jarak-jarak antara (*UV*) dan *cluster W* yang lain dihitung dengan cara,

$$d_{(UV)W} = \min\{ d_{UW}, d_{VW} \} \qquad (2.5)$$

Di sini besaran-besaran *dUW* dan *dVW* berturut-turut adalah jarak terpendek antara *cluster-cluster U* dan *W* dan juga *cluster-cluster V* dan *W*.

2.5 *Precision & Recall*

Evaluasi dari sistem temu-kembali informasi dipengaruhi oleh duaparameter utama yaitu *recall* dan *precision*. *Recall* adalah rasio antara dokumen relevan yang berhasil ditemu kembalikan dari seluruh dokumen relevan yang ada di dalam sistem, sedangkan *precision* adalah rasio dokumen relevan yang berhasilditemukembalikan dari seluruh dokumen yang berhasil ditemu-kembalikan (Grossman, 2002).

Dengan menggunakan nilai dari parameter *recall* dan *precision* akan dicari nilai dari *average precision* untuk menghitung keefektifan dan keakuratan darialgoritma sistem temu-kembali informasi. *Average precision* adalah suatu ukuran evaluasi sistem temu-kembali informasi yang diperoleh dengan cara menghitung rata-rata *precision* pada seluruh tingkat *recall* (Grossman, 2002).

Sedangkan untuk menentukan nilai dari *recall* dan *precision* harus didapatkan jumlah dokumen yang relevan terhadap suatu topik informasi. Satu-satunya cara untuk mendapatkannya yaitu dengan membaca dokumen itu satu persatu.

Menurut Rijsbergen (1979) relevansi merupakan sesuatu yang sifatnya subyektif. Setiap orang mempunyai perbedaan untuk mengartikan sesuatu dokumen tersebut relevan terhadap sebuah topik informasi.

Menurut Mizzaro (1998), evaluasi pada sebuah sistem temu-kembali informasi dengan menggunakan *recall* dan *precision* sudah cukup baik untuk menjadi ukuran dari sistem tersebut.

Adapun rumus dari *Precision* dan *Recall* yang akan digunakan akan dijelaskan pada tabel 2.7 berikut:

Tabel 2.7 Precision dan Recall (Manning dkk, 2009)

	Relevant	***Not Relevant***
Retrieved	*true positif (tp)*	*false positif (fp)*
Not Retrieved	*false negative (fn)*	*true negative (tn)*

Maka: *Precision (P) = tp/(tp+fp)* (2.6)

Recall (R) = tp/(tp+fn) (2.7)

Berdasarkan pada Tabel 2.7, ketika kondisi dokumen ditemukembalikan (*retrieved*), maka terdapat dua jenis kesesuaian yang muncul yaitu apakah dokumen tersebut relevan atau tidak relevan dengan *query* pengguna. Adapun keterangan penjelasan pada tabel di atas yaitu:

true positif (tp): jumlah dokumen yang ditemu kembalikan sesuai dengan *query* pengguna

false negative (fn): jumlah dokumen yang relevan dengan *query* namun tidak ditemukembalikan oleh sistem

false positif (fp): jumlah dokumen yang ditemu kembalikan tidak sesuai dengan *query*

true negative (tn): jumlah dokumen tidak relevan dengan *query* yang tidak ditemu kembalikan oleh sistem

2.6 Penelitian Terkait

Berdasarkan penelitian sebelumnya tentang perbandingan metode *single linkage clustering* dan *k-means clustering* (Rendy et all, 2014) dengan menggunakan parameter perhitungan *Silhouette Coefficient* dan *Purity* sebagai pembanding performasi kedua metode tersebut, didapatkan bahwa metode perhitungan *Single Linkage* memiliki performansi temu kembali informasi yang lebih baik dibandingan dengan algoritma *k-means.* Hal ini dikarenakan proses saat pembentukan *cluster* awal pada algoritma *k-means* dilakukan secara acak. Pada penelitian tersebut data yang diujicobakan adalah dokumen berita *online* yang disimpan di dalam *notepad.* Proses pengujiannya adalah menguji performansi *clustering* yang dibangun dengan *dataset* 4 kategori (25, 50, 75 dan 100 *dataset*) dengan jumlah *cluster* adalah 3, 4, 5, 6, 7, 8, 9, dan 10 yang selanjutnya akan dicari masing-masing nilai *Silhouette Coefficient* dan *Purity* dari 4 bagian dataset yang diujicobakan. Hasil penelitian yang didapat dalam penelitian tersebut bahwa

metode *K-means* membutuhkan jumlah *cluster* yang lebih banyak untuk menyamai nilai *Silhouette Coefficient* metode *single linkage.*

Pada penelitian lainnya dengan judul Temu Kembali Informasi Berbasis Kluster untuk Sistem Temu Kembali Informasi Teks Bahasa Indonesia (Hamzah, 2009) tentang melakukan *retrieval* berbasis kluster (*cluster-based retrieval*) dengan model ruang vektor. Dataset yang diujicobakan adalah 500 dokumen berita, 1009 dokumen berita dan dokumen abstrak. Adapun metode *clustering* yang dipakai yaitu *partitional clustering* dengan perhitungan algoritma *bisecting kmeans, k-means* dan *buckshot* serta *hierarchical clustering* dengan perhitungan UPGMA dan *complete linkage*. Penelitian ini fokus pada perbandingan performa metode partitional dan *hierarchical clustering* dalam menemukembalikan informasi dari 3 jenis dokumen yang diujicobakan. Hasil dan kesimpulan yang didapat adalah bahwa metode *clustering hierarchical* memiliki tingkat *F-measure* yang tinggi dalam temu kembali informasi berbasis kluster untuk dokumen berbahasa Indonesia serta temu kembali informasi berbasis kluster mengalami peningkatan kinerja perolehan informasi berbasis kluster sebesar berturut-turut 12.3% dan 9.5% dibandingkan dengan perolehan linear berbasis *word –matching*.

Penelitian lain yang mendasari penelitian ini adalah analisa mengenai perancangan dan implementasi *automated document integration* dengan menggunakan algoritma *complete linkage agglomerative hierarchical clustering* (Pradnyanaet all, 2012). Pengujian *white box* yang digunakan dalam pembuatan automated document integration system ini adalah pengujian basis *path*. Pengujian tersebut ditujukan untuk mencari *path-path* yang dilalui saat program dijalankan. Pengujian basis *path* pada penelitian ini difokuskan pada proses perhitungan bobot relasi antar kalimat dan proses integrasi dengan algoritma *complete linkage agglomerative hierarchical clustering*. Alur logika dari sistem diujicoba dengan menyediakan kasus ujicoba yang melakukan semua kondisi atau perulangan yang ada pada sistem. Dimana setelah

melakukan pengujian, setiap *path* yang ada dalam tiap proses dapat dijalan dengan baik.

Pengujian terhadap proses integrasi dilakukan dengan membandingkan proses integrasi dengan perhitungan manual dan dengan sistem. Dokumen yang dipergunakan adalah dokumen dengan judul Data Mining 1 dan Data Mining 2. Pada akhir pengujian didapatkan hasil bahwa hasil integrasi secara manual menghasilkan dokumen yang sama dengan dokumen yang diintegrasikan oleh sistem. *Agglomerative*