LAPORAN PENELITIAN

Rancang Bangun *Information Retrieval System (IRS)* Bahasa Jawa Ngoko pada Palintangan Penjebar Semangad dengan Metode *Vector Space Model (VSM)*

Oleh :

Fatkhul Amin, S.T.,M.Kom	0624097401 (Ketua)
Purwatiningtyas, SE, M.Kom	0617096601 (Anggota)
Agung Wicaksono	12.01.53.0062 (Anggota)
Dicky Setiawan	12.01.53.0057 (Anggota)

FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS STIKUBANK (UNISBANK) SEMARANG

AGUSTUS 2015

HALAMAN PENGESAHAN
LAPORAN HASIL PENELITIAN

1. Judul Penelitian	:	Rancang Bangun *Information Retrieval System (IRS)* Bahasa Jawa ngoko pada Palintangan Penjebar Semangad dengan Metode *Vector Space Model (VSM)*
2. Jenis Penelitian	:	Penelitian Terapan (Applied Research)
3. a. Bidang Penelitian	:	*Engineering and technology*
b. Kelompok	:	2 / 2.18 *Information Technology*
4. a. Tujuan Sosial Ekonomi	:	*Advancement of knowledge*
b. Kelompok	:	20 / 20.05 *Information, Computer and Communications technologies*
5. Ketua Peneliti		
a. Nama Lengkap	:	Fatkhul Amin, S.T.,M.Kom
b. Jenis Kelamin	:	Laki-laki
c. NIDN	:	0624097401
d. Disiplin Ilmu	:	Sistem Informasi
e. Pangkat / Golongan	:	Penata Muda / IIIB
f. Jabatan Fungsional	:	Asisten Ahli
g. Fakultas / Prodi	:	Teknologi Informasi / Teknik Informatika
h. Alamat Kampus	:	Jl. Tri Lomba Juang 1, Semarang
i. Telp/Faks/Email	:	0248311668/-/info@unisbank.ac.id
j. Alamat Rumah	:	Jl. Candi Pawon Timur VI / 15 , Semarang
k. Telp/Faks/Email	:	081215156265/-/ fatkhulamin@gmail.com
6. Jumlah Anggota Peneliti	:	3 orang
a. Nama Anggota I	:	Purwatiningtyas, SE, M.Kom / 0617096601
b. Mahasiswa yang terlibat	:	Agung Wicaksono /12.01.53.0062
c. Mahasiswa yang terlibat	:	Dicky Setiawan / 12.01.53.0057
7. Lokasi Penelitian	:	Universitas Stikubank (Unisbank)
8. Jangka Waktu Penelitian	:	28 Juli sd 30 September 2015
9. Jumlah Biaya yang diusulkan	:	Rp. 3.000.000,-

Semarang, 20 Agustus 2015
Ketua Peneliti,

(Fatkhul Amin, S.T.,M.Kom)
NIDN. 0624097401

Mengetahui,
Dekan Fakultas Teknologi Informasi

(Dr. Drs. Y. Suhari, M.MSI)
NIDN. 0620106502

Menyetujui,
Kepala LPPM Unisbank

(Dr. Endang Tjahjaningsih, S.E., M.Kom)
NIDN. 0622056601

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas segala karuniaNya yang Dia berikan kepada penulis sehingga laporan penelitian berjudul "Rancang Bangun *Information Retrieval System (IRS)* Bahasa Jawa Ngoko pada Palintangan Penjebar Semangad dengan Metode *Vector Space Model (VSM* " dapat diselesaikan. Penulisan penelitian ini dapat terselesaikan karena bantuan dari berbagai pihak, serta dorongan baik berupa pikiran, ide dan sumbang saran. Oleh karena itu, pada kesempatan yang berbahagia ini penulis menyampaikan terima kasih yang sebesar-besarnya kepada;

1. Bapak **Dr. H. Hasan Abdul Rozak, S.H., C.N.,M.M.** selaku Rektor Universitas Stikubank (Unisbank) Semarang.
2. Ibu **Dr. Endang Tjahjaningsih, S.E., M.Kom,** selaku Ketua Lembaga Penelitian dan pengabdian Masyarakat (LPPM) Universitas Stikubank (Unisbank) Semarang.
3. Bapak **Dr. Drs. Y. Suhari, M.MSI,** selaku Dekan fakultas Teknologi Informasi Universitas Stikubank (Unisbank) Semarang.
4. Rekan-rekan dosen yang telah memberikan masukan-masukan untuk perbaikan dan kesempurnaan penulisan laporan ini.

Semoga laporan ini dapat bermanfaat dan menambah ilmu bagi semua, serta dapat mendukung kemajuan ilmu pengetahuan khususnya di bidang Teknologi informasi.

Semarang, 20 Agustus 2015

Penulis

Rancang Bangun *Information Retrieval System (IRS)*
Bahasa *Jawa Ngoko* pada *Palintangan Penjebar Semangad*
dengan Metode *Vector Space Model (VSM)*

ABSTRAK

Bahasa Jawa adalah bahasa daerah yang paling banyak digunakan di Indonesia yang mulai ditinggalkan. Perlunya pelestarian bahasa jawa dalam bentuk online yang bisa diakses bagi penggunanya sehingga akanmemudahkan dalam pencarian dokumen teks khususnya dokumen bahasa jawa ngoko. *Information Retrieval System (IRS)* yang ada saat ini memberikan hasil pencarian dokumen dengan hasil perolehan dokumen dalam jumlah banyak (*recall* tinggi) dan akurasi yang rendah (*precision*rendah). Tujuan pembuatan rancang bangun *IRS* dengan metode *Vector Space Model (VSM)* agar *user* mudah dalam melakukan pencarian dokumen teks berbahasa Jawa ngoko. *Software IRS* dirancang untuk memberikan hasil pencarian dokumen dalam jumlah yang optimal (*recall* rendah) dan akurat (*precision* tinggi) menggunakan metode *VSM* , sehingga *user* akan mendapatkan hasil pencarian cepat dan akurat. Metode VSM akan melakukan pembobotan tiap dokumen yang ada pada database sehingga antar dokumen memiliki bobot yang berbeda untuk menentukan dokumen mana yang paling mirip (similar) dengan *query*, dokumen dengan bobot tertinggi menempati ranking teratas dalam hasil pencarian. Evaluasi hasil pencarian IRS dilakukan dengan uji *recall* dan *precision*. Studi kasus yang telah dilakukan menggunakan IRS ini didapatkan hasil sistem mampu melakukan proses preprosesing (tokenisasi, *filtering, dan stemming*) dengan waktu komputasi 18 detik. Sistem mampu melakukan pencarian dokumen dan menampilkan hasil pencarian dokumen dalam waktu komputasi rata-rata 2 detik, memiliki rata-rata *recall* 0,04 dan rata-rata *precision* 0,84. Sistem dilengkapi dengan bobot tiap dokumen dan letakknya yang akan memudahkan *user* dalam pencarian dokumen teks bahasa Indonesia.

Kata-Kunci: *IRS*, Basa Jawa Ngoko, *VSM*

DAFTAR ISI

Halaman :

DAFTAR GAMBAR

Halaman :

DAFTAR TABEL

Hal

BAB I
PENDAHULUAN

1.1. Latar Belakang

Bahasa Jawa sebagai bahasa yang paling banyak digunakan di wilayah Indonesia setelah bahasa indonesia, dewasa ini mulai banyak ditinggalkan oleh kebanyakan orang. Media offline dan media online juga kurang mengangkat bahasa jawa sehingga dikhawatirkan bahasa jawa lama-kelamaan akan ditinggalkan oleh bangsa kita. Beberapa media online berbahasa Jawa ada, namun belum menggunakan atau belum menyediakan pencarian informasi menggunakan mesin pencari khusus berbahasa jawa.

Implementasi *Vector Space Model* dapat dirasakan dan dinikmati pada berbagai bidang keilmuan seperti *Computational Linguistics* (Erk dkk, 2010), *Expert Systems* (Kim dkk, 2010), *Medical* (lopez dkk, 2010), *Knowledge-Based Systems* (Yu dkk, 2009), *Data and Knowledge Engineering* (Mao dkk, 2007), dan lain sebagainya. *Vector space model* dapat juga digunakan dalam sistem temu kembali informasi *(information retrieval)*. Sistem temu kembali informasi akan memberikan nilai tambah dalam pecarian informasi jika keinginan *user* bisa terpenuhi. Penelitian ini diharapakan dapat membuat sistem temu kembali informasi yang bernilai tambah yaitu menghasilkan pencarian informasi dengan cepat dan akurat.

Pencarian informasi saat ini dilakukan dengan menggunakan mesin pencari atau sistem temu kembali informasi, *user* menuliskan *query* dan mesin pencari akan menampilkan hasil pencarian. Mesin pencari yang sudah ada dan banyak digunakan saat ini memberikan hasil perolehan pencarian yang banyak (banyak dokumen yang terambil), sehingga diperlukan waktu untuk menentukan hasil pencarian yang relevan. Menentukan hasil yang relevan sesuai dengan keinginan user dengan jumlah hasil pencarian yang banyak akan menyulitkan *user*. Hal ini terjadi karena dokumen yang terambil oleh sistem jumlahnya banyak, maka sistem berkemungkinan menampilkan hasil pencarian yang tidak relevan. Banyaknya dokumen hasil pencarian ini membuat waktu yang dibutuhkan dalam pencarian menjadi lebih banyak dari yang diharapkan.

Perkembangan penelusuran informasi saat ini menghasilkan *recall* yang tinggi dan *precision* yang rendah. *Recall* yang tinggi diartikan bahwa dokumen yang dihasilkan dalam penelusuran dokumen adalah banyak, sedangkan *precision* rendah dapat diartikan bahwa dokumen yang diharapkan dapat ditemukan sedikit.

Solusi untuk mengatasi masalah ini adalah dengan membuat *software Information Retrieval System (IRS)* menggunakan metode *Vector Space Model (VSM)*. Metode VSM dipilih karena cara kerja model ini efisien, mudah dalam representasi dan dapat diimplementasikan pada *document-matching*. *Software IRS basa jawa ngoko* diharapkan menghasilkan *recall* rendah dan *precision* tinggi.

1.2. Perumusan Masalah

Bagaimana membuat Rancang Bangun *Information Retrieval System (IRS)* Bahasa Jawa Ngoko pada Palintangan Penjebar Semangad dengan Metode *Vector Space Model (VSM) ?*

BAB II
TUJUAN DAN MANFAAT PENELITIAN

2.1. Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian ini adalah;

a. Melestarikan bahasa Jawa agar tidak dilupakan oleh generasi penerus bangsa khususnya orang jawa.
b. Memberikan sumbangsih pemikiran tentang implementasi bahasa jawa di era teknologi informasi
c. Mengembangankan ide kreatif tentang perlunya mesin pencari bebahasa jawa yangbisa digunakan untuk pencarian bahasa jawa
d. Riset Seni Teater Membuat rancang bangun *Information Retrieval System (IRS)* Bahasa *Jawa Ngoko* dengan metode *Vector Space Model*.

2.2. Manfaat Penelitian

Manfaat yang dari penelitian ini adalah;

a. Memelihara Bahasa Jawa sebagai aset bangsa
b. Menanamkan kecintaan bahasa jawa kepada anak cucu
c. Mengembangkan mesin pencari dengan berbasis bahasa Jawa
d. Membuat sistem pencarian berupa Information Retrieval System berbahasa jawa ngoko yang mudah digunakan dan mampu menghasilkan informasi yang akurat.
e. Membuat alat bantu untuk pencarian dokumen teks bahasa jawa ngoko.
f. Menghemat waktu pencarian informasi untuk mendapatkan dokumen yang diinginkan.

BAB III
TELAAH PUSTAKA

3.1. Penelitian Terdahulu

Penggunaan *Information Retrieval System* saat ini banyak ditemukan di media online seperti internet, namun juga bisa ditemukan di beberapa bidang kerja nyata di perkantoran. Aplikasi *Information Retrieval System* sangat membantu dalam pencarian dokumen khususnya dokumen berupa teks. Mesin pencari sebagai salah satu bentuk aplikasi dari *Information Retrieval System* membantu penggunanya untuk mendapatkan informasi secara cepat. Perkembangan *Information Retrieval System* dari waktu ke waktu memberikan hasil yang maksimal bagi penggunanya, baik media offline maupun media online. Beberapa mesin pencari menghasilkan koleksi hasil pencarian yang banyak, dan beberapa mesin pencari menghasilkan sebaliknya. Hal ini dikarenakan algoritma yang dipergunakan berbeda-beda dan tujuan yang berbeda pula.

Information Retrieval System dikembangkan oleh *search engine* rumah sakit dengan *global search engine* memiliki perbedaan hasil dalam hal hasil temu kembali khususnya dibidang kesehatan (lopez, dkk, 2010). Informasi tentang kesehatan semakin diminati masyarakat pengguna internet. Studi ini bertujuan membandingkan penggunaan *local search engine* dengan *global search engine* dalam hal tingkat akurasinya. Studi ini mengevaluasi kinerja 4 generalis *Search Engine* (Google, Bing, Yahoo! dan Sapo) dan 3 khusus *Search Engine* (MedlinePlus, WebMD dan Sapo Sa'ude) dalam Information Retrieval System kesehatan. Berdasarkan kumpulan dokumen yang relevan, menghasilkan semua pengguna memilih Google sebagai salah satu dari empat *Search Engine*. *Search Engine* lain Sapo Sa'ude (27 pengguna), *Bing* (25 pengguna) dan *MedlinePlus* (23 pengguna).

Studi kasus pada bidang pengolahan bahasa alami (*natural language processing*) yang memfokuskan pada pencarian topik inti dari suatu paragraph (Haryono, 2005) menggunakan pemrograman Borland Builder C++ versi 6. Penelitian dilakukan terhadap sejumlah teks paragraf dari surat kabar lokal kedaulatan rakyat. Tema yang diambil adalah pendidikan dan budaya dengan

mengambil 20 sampel cerita dimana setiap sampel dipilah-pilah menurut paragrafnya. Tiap paragraf diberikan judgment terhadap intisarinya secara manual terlebih dahulu oleh penulis dan rekan penulis. Implemenasi dilakukan dengan mencocokkan hasil topik automatis oleh sistem dan hasil topik yang telah ditentukan oleh penulis untuk masing-masing paragrafnya. Prosesnya adalah sebagai berikut, teks suatu paragraf dibagi per kalimat, kemudian proses tokenisasi, kemudian pembobotan skor leksikal, perhitungan *similarity coefisient* antar blok dan penentuan batas segmen. Penentuan nilai segmen blok dihitung dengan menggunakan nilai *similarity coefisient* antar dua kalimat, dan hasilnya dirangking secara *descending* kemudian ditentukan n terbesar dari nilai paling atas.

Model mesin pencari yang ada memiliki kelemahan mencari relevansi antara *query* dan hasil. *Vector space model* diharapkan dapat membuat *query* dan hasil lebih *matching*. Model ruang vektor memiliki kelebihan *query* dapat berupa sekumpulan kata-kata / kalimat . Misalkan terdapat sejumlah *n* kata yang berbeda. Kata-kata ini akan membentuk ruang vektor yang memiliki dimensi sebesar *n*. Setiap kata *i* diberikan bobot sebesar w_i. Baik dokumen maupun *query* direpresentasikan sebagai vektor berdimensi *n*.

Hasil perhitungan *Vector Space Model* diuji menggunakan nilai ketepatan (*precision*) dan kelengkapan/perolehan (*recall*). *Recall* dan *Precision* merupakan pengukuran yang sering digunakan untuk mengukur kualitas hasil proses dari hasil proses Information Retrieval System. *Precision* dapat dianggap sebagai ukuran ketepatan atau ketelitian, sedangkan *recall* adalah kesempurnaan. Nilai *precision* sama dengan 1 jika semua hasil pencarian yang didapatkan adalah relevan. Nilai *recall* sama dengan 1 jika semua dokumen yang relevan telah berhasil didapatkan (Salton, 1989).

3.2. *Information Retrieval System (IRS)*

Information Retrieval System menemukan informasi yang biasanya dalam bentuk dokumen dari sebuah data yang tidak terstruktur dalam bentuk teks untuk memenuhi kebutuhan informasi dari koleksi data yang sangat besar umumnya tersimpan dalam *database computer (Manning, 2008)*.

information retrieval (IRS) merupakan suatu sistem yang menemukan informasi yang sesuai dengan kebutuhan *user* dari kumpulan informasi secara otomatis. Aplikasi Information Retrieval System sudah digunakan dalam banyak bidang seperti dikedokteran, perusahaan dan lain sebagainya. Salah satu aplikasi dari Information Retrieval System adalah mesin pencari yang dapat diterapkan diberbagai bidang. Pada mesin pencari dengan Information Retrieval System user dapat memasukkan *query* yang bebas dalam arti kata *query* yang sesuai dengan bahasa manusia dan sistem dapat menemukan dokuen yang sesuai dengan *query* yang ditulis oleh *user*.

Prinsip kerja Information Retrieval System 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).

Information Retrieval System akan mengambil salah satu dari kemungkinan tersebut. Information Retrieval System 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*. Information Retrieval System didesain untuk menemukan dokumen atau informasi yang diperlukan oleh *user*.

Information Retrieval System bertujuan untuk menjawab kebutuhan informasi *user* dengan sumber informasi yang tersedia dalam kondisi seperti sebagai berikut (Salton, 1989);

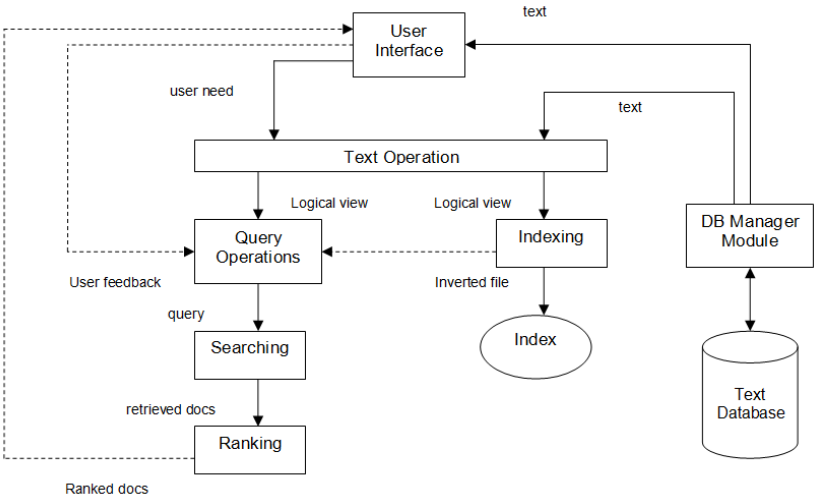
a. Mempresentasikan sekumpulan ide dalam sebuah dokumen menggunakan sekumpulan konsep.
b. Terdapat beberapa pengguna yang memerlukan ide, tapi tidak dapat mengidentifikasikan dan menemukannya dengan baik.
c. Information Retrieval System bertujuan untuk mempertemukan ide yang dikemukakan oleh penulis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

Fungsi utama Information Retrieval System (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.

3.3. Arsitektur *Information Retrieval System*

Proses *Information Retrieval System* seperti pada gambar 3.1 menggunakan arsitektur yang sederhana. Sebelum dilakukannya proses temu kembali diperlukan pendefinisian database. Selanjutnya mengikuti tahapan proses; Dokumen-dokumen yang akan digunakan, Operasi yang akan digunakan dalam pencarian, dan model pengolahan teks (Baeza, 1999,h.9).



Gambar 3.1 *The Process of Retrieving Information* (*Baeza*, 1999,h.10)

3.3.1. Korpus

Proses IRS dalam aplikasinya membutuhkan database yang didalamnya terdapat satu atau beberapa tabel yang digunakan sebagai tempat penyimpanan data yang akan diolah pada saat proses pencarian. Penelitian dengan menggunakan database pada aplikasinya biasanya memakai korpus untuk proses pembuatan tabel pendukungnya. Penelitian empiris dapat dilakukan dengan menggunakan teks tertulis atau lisan, seperti teks-teks dasar dari berbagai jenis sastra dan analisis linguistik. Tapi gagasan tentang korpus sebagai dasar untuk sebuah bentuk linguistic empiris berbeda dalam beberapa cara mendasar dari teks-teks tertentu. Pada prinsipnya, setiap koleksi lebih dari satu teks dapat disebut dengan *korpus* (McEnery dan Wilson, 2001): istilah *korpus* dalam bahasa latin berarti *body*, maka *korpus* dapat didefinisikan sebagai isi setiap teks. Tapi istilah korpus ketika digunakan dalam konteks *linguistic modern* memiliki konotasi yang lebih spesifik. Ada empat karakteristik dari korpus (McEnery dan Wilson, 2001):

a. *Sampling and Representativeness*

   Dalam membangun sebuah korpus dari berbagai bahasa, dapat ditarik dari sebuah sampel yang mewakili dari berbagai pengujian secara maksimal, yaitu menyediakan korpus seakurat mungkin dari kecenderungan yang beragam termasuk proporsi antara korpus dan informasi yang dicari. Jadi, tidak semata-mata berdasarkan pada teks sampel yang dipilih, akan tetapi sampel dari berbagai sumber yang diambil dari sumber dokumen aslinya, sehingga akan memberikan gambaran yang cukup akurat dari seluruh informasi yang akan didapatkan.

b. *Finite Size*

   Korpus juga cenderung menyiratkan suatu isi teks dengan urutan yang terbatas, misalkan 1.000.000 kata. Teks dapat terus ditambahkan ke dalamnya, sehingga semakin besar karena lebih banyak sampel yang ditambahkan. Keuntungan yang pertama adalah teks menjadi tidak statis karena teks yang baru akan selalu ditambahkan, keuntungan yang kedua adalah ruang lingkup akan lebih besar dan jauh lebih luas sehingga akan mencakup dari bahasa yang digunakan.

Kelemahannya, karena terus berubah dalam ukuran dan kurang ketatnya sampel, menjadi sumber yang kurang terpercaya dalam segi kuantitatif. Jadi sebaiknya pada awal pembangunan korpus, rencana riset ditetapkan secara rinci bagaimana berbagai bahasa yang digunakan diambil dari sampelnya, berapa banyak sampel dan kata harus dikumpulkan sehingga jumlah keseluruhan yang sudah ditetapkan ini dapat digunakan.

c. *Machine-Readable Form*

Corpora yang dapat dibaca oleh mesin memiliki beberapa keunggulan dibandingkan dengan format tertulis atau lisan. Pertama dan paling penting keuntungan dari corpora yang dapat dibaca oleh mesin adalah bahwa dimungkinkan untuk mencari dan memanipulasi dengan cara-cara yang tidak dilakukan dengan format lain. Sebagai contoh, sebuah korpus dalam format buku, akan perlu dibaca dari depan sampai belakang untuk mengambil semua contoh kata, dengan korpus yang dapat dibaca oleh mesin, tugas ini dapat dicapai dalam beberapa menit dengan menggunakan perangkat lunak, atau sedikit lebih lambat, dengan menggunakan fasilitas pencarian di pengolah kata. Keuntungan kedua corpora yang dapat dibaca oleh mesin adalah bahwa dapat dengan cepat dan mudah diperkaya dengan informasi tambahan.

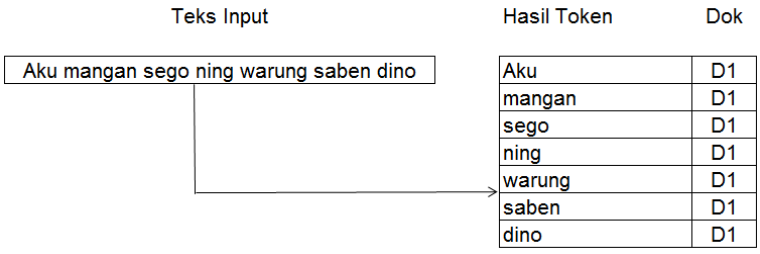
d. *A Standard Reference*

Meskipun tidak termasuk hal yang penting dari definisi suatu korpus, tetapi ada juga pemahaman bahwa korpus merupakan referensi standar untuk berbagai bahasa yang diwakilinya. Hal ini mengandaikan ketersediaan yang luas kepada peneliti lain, Keuntungan dari korpus yang tersedia luas adalah akan memberikan tolak ukur yang dapat digunakan sebagai pembanding dalam studi. Misalnya secara langsung dibandingkan dengan hasil yang dipublikasikan (selama metodologi sama) tanpa perlu perhitungan ulang. Korpus standar juga berarti penggunaan korpus yang sama digunakan untuk berbagai macam variasi studi dan yang membedakannya yaitu penggunaan data pengujiannya dan metodologi yang digunakan dalam studi.

3.3.2. Proses Tokenisasi

Proses pertama yang dilakukan dalam IRS adalah proses memisahkan kata yang ada pada dokumen berdasarkan spasi kemudian memproses kata yang telah dipisahkan tersebutk kedalam sebuah tabel untuk dilakukan proses berikutnya. Proses Tokenisasi merupakan proses pemisahan suatu rangkaian karakter berdasarkan karakter spasi, dan mungkin pada waktu yang bersamaan dilakukan juga proses penghapusan karakter tertentu, seperti tanda baca. Sebagai contoh, kata-kata "*computer*", "*computing*", dan "*compute*" semua berasal dari term yang sama yaitu "*comput*" , tanpa pengetahuan sebelumnya dari morfologi bahasa Inggris. 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). contoh tokenisasi bisa dilihat pada gambar 3.2. Input : Aku mangan sega ning warung saben dino

Output :



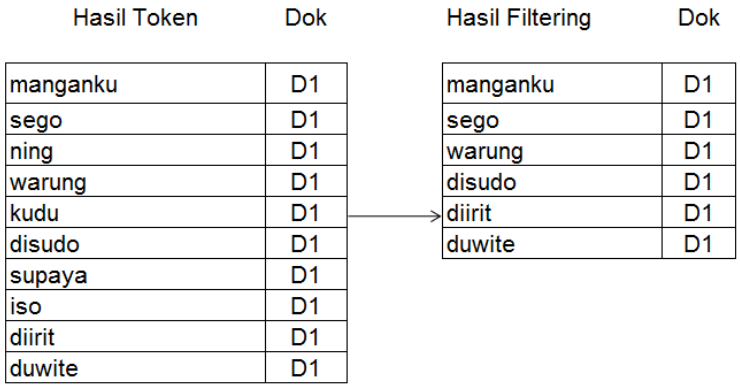
Gambar 3.2 Contoh hasil proses tokenisasi

3.3.3. *Filtering*

Proses selanjutnya setelah dilakukan pemisahan kata pada dokumen adalah proses *filtering*. *Filtering* akan memproses kata hasil tokenisasi menjadi lebih sedikit dengan cara mengurangi kata tersebut dengan kata yang termasuk dalam *stopwords*. Eliminasi *stopwords* memiliki banyak keuntungan, yaitu akan mengurangi *space* pada tabel *term index* hingga 40% atau lebih (Baeza, 1999,h.167). Proses *stopword removal* merupakan proses penghapusan *term* yang tidak memiliki arti atau tidak relevan. Proses ini dilakukan pada saat proses tokenisasi. Proses tokenisasi menghasilkan sebuah *term*, dan term tersebut

selanjutnya di periksa dalam daftar *stopword*. Apabila *term* tersebut terdapat dalam daftar *stopword* maka *term* tersebut tidak akan dimasukkan dalam tabel *term*. Sebaliknya *term* hasil tokenisasi apabila diperiksa ke dalam daftar *stopword* dan hasilnya nihil maka *term* tersebut akan dimasukkan ke dalam tabel *term*.

Ada beberapa cara dalam proses *stopword removal*, antara lain meletakkan proses stopword sebelum term hasil tokenisasi dimasukkan ke dalam tabel *term*, cara yang kedua menempatkan proses *stopword* setelah term hasil tokenisasi masuk kedalam tabel. Gambar 3.3 menunjukkan prosec filtering.



Hasil Token	Dok
manganku	D1
sego	D1
ning	D1
warung	D1
kudu	D1
disudo	D1
supaya	D1
iso	D1
diirit	D1
duwite	D1

Hasil Filtering	Dok
manganku	D1
sego	D1
warung	D1
disudo	D1
diirit	D1
duwite	D1

Gambar 3.3 Contoh hasil proses Filtering

Daftar *stopword* beberapa bahasa telah banyak digunakan pada proses *stopword removal*, antara lain *German Stopwords*, *French stopword*s, *English stowords*, Indonesia *stopwords* (Tala-*stop stopwords* dan Vega *stopwords*). Daftar *stopword* tersimpan dalam suatu tabel, dalam penelitian ini menggunakan daftar *stopword* yang digunakan oleh Tala (2003), yang merupakan *stopword* bahasa Indonesia yang berisi kata-kata seperti; adhep, ana, yaiku, anane, dll.

3.3.4. *Stemming*

Proses *Stemming* digunakan untuk mengubah *term* yang masih melekat dalam term tersebut awalan, sisipan, dan akhiran. Selanjutnya *term* tersebut diproses untuk dihilangkan awalan, sisipan dan akhiran sehingga menjadi term kata dasar. Proses membuat *term* dasar ini mengacu kepada bahasa jawa ngoko yang benar.

Proses stemming dilakukan dengan cara menghilangkan semua imbuhan (*affixes*) baik yang terdiri dari awalan *(prefixes*), sisipan *(infixes)*, akhiran *(suffixes)* dan *confixes* (kombinasi dari awalan dan akhiran) pada kata turunan. *Stemming* digunakan untuk mengganti bentuk dari suatu kata menjadi kata dasar dari kata tersebut yang sesuai dengan struktur morfologi bahasa jawa yang benar.

Imbuhan *(affixes)* pada Bahasa Jawa ngoko lebih kompleks jika dibandingkan dengan imbuhan pada Bahasa Inggris. Imbuhan pada Bahasa Jawa ngoko terdiri dari awalan *(prefixes)*, sisipan *(infixes)*, akhiran *(suffixes)*, bentuk perulangan *(repeated forms)* dan kombinasi awalan akhiran *(confixes)*. Imbuhan-imbuhan yang melekat pada suatu kata harus dihilangkan untuk mengubah bentuk kata tersebut menjadi bentuk kata dasarnya. *Steming* teks berbahasa Jawa ngoko memiliki beberapa masalah yang sangat khusus terhadap bahasa. Salah satu masalah tersebut adalah perbedaan tipe dari imbuhan-imbuhan *(affixes),* bahwa awalan *(prefixes)* dapat berubah tergantung dari huruf pertama pada kata dasar. Sebagai contoh "me-" dapat berubah "mem-" ketika huruf pertama dari kata dasar tersebut adalah "n", misalkan "nulis"(kata dasar tulis), tetapi dapat berubah menjadi "ng-" ketika huruf pertama dari kata dasar melekat adalah "k", misalkan "ngethok" *(*kata dasar kethok*)*. Contoh proses *Stemming* bisa dilihat pada gambar 3.4.

Hasil Filtering	Dok
manganku	D1
sego	D1
warung	D1
disudo	D1
diirit	D1
duwite	D1

→

Hasil Stemming	Dok
mangan	D1
sego	D1
warung	D1
sudo	D1
irit	D1
duwit	D1

Gambar 3.4 Contoh hasil proses Stemming

Berikutnya jika ada lebih dari satu imbuhan *(affixes)* yang melekat pada suatu kata, maka urutan untuk menghilangkan imbuhan-imbuhan *(affixes)* pada kata tersebut menjadi sangat penting. Jika dalam proses menghilangkan imbuhan-imbuhan *(affixes)* tersebut kita tidak memperhatikan urutan penghilangan imbuhan-imbuhan *(affixes)* tersebut, maka kata dasar yang benar dari kata tersebut tidak akan ditemukan. Sebagai contoh kata "di-beri-kan" yang diturunkan dari

kata dasar "beri" *(to give)*. Jika menghilangkan akhiran *(suffixes)* "kan" terlebih dahulu sebelum menghilangkan awalan *(prefix)* "di-" maka pada proses stemming ini akan mendapatkan kata dasar yang benar yaitu "beri" *(to give),* namun jika algoritma stemming mencoba untuk menghilangkan awalan *(prefixes)* terlebih dahulu sebelum akhiran (suffixes) maka hasil kata dasar yang dihasilkan dari proses stemming dengan menggunakan algoritma tersebut adalah 'ikan" atau *fish* setelah menghilangkan awalan 'di" dan "ber") dimana "ikan" merupakan kata dasar yang benar untuk kata turunan 'diberikan".

Penelitian terhadap *stemming* untuk *retrieval, machine translation, document summarization* dan *text classification* sudah pernah dilakukan sebelumnya. Stemming yang dilakukan pada *text retrieval*, stemming ini meningkatkan kesensitifan *retrieval* dengan meningkatkan kemampuan untuk menemukan dokumen yang relevan, tetapi hal itu terkait dengan pengurangan pada pemilihan dimana pengelompokan menjadi kata dasar menyebabkan penghilangan makna kata.

3.3.4.1. *Stemmer* Bahasa *Jawa ngoko*

Stemmer Bahasa Jawa Ngoko adalah adopsi dari algoritma *stemmer* bahasa inggris *porter stemmer*. *Stemmer* menggunakan *rule base* analisis untuk mencari *root* sebuah kata. Stemmer ini sama sekali tidak menggunakan kamus sebagai acuan.

Struktur pembentukan kata dalam bahasa Jawa Ngoko adalah sebagai berikut:

[awalan-1]+[awalan-2]+[dasar]+[akhiran]+[kepunyaan]+[sandang]

Masing-masing bagian digabungkan dengan kata dasar membentuk kata berimbuhan. Algoritma Bahasa Jawa Ngoko menggunakan algoritma *rule based stemming* seperti halnya dengan algoritma *porter* pada *stemming* bahasa inggris.

Pada *stemmer* Bahasa Jawa ngoko Terdapat 5 langkah utama dengan 3 langkah awal dan 2 langkah pilihan, langkah-langkah tersebut adalah sebagai berikut:

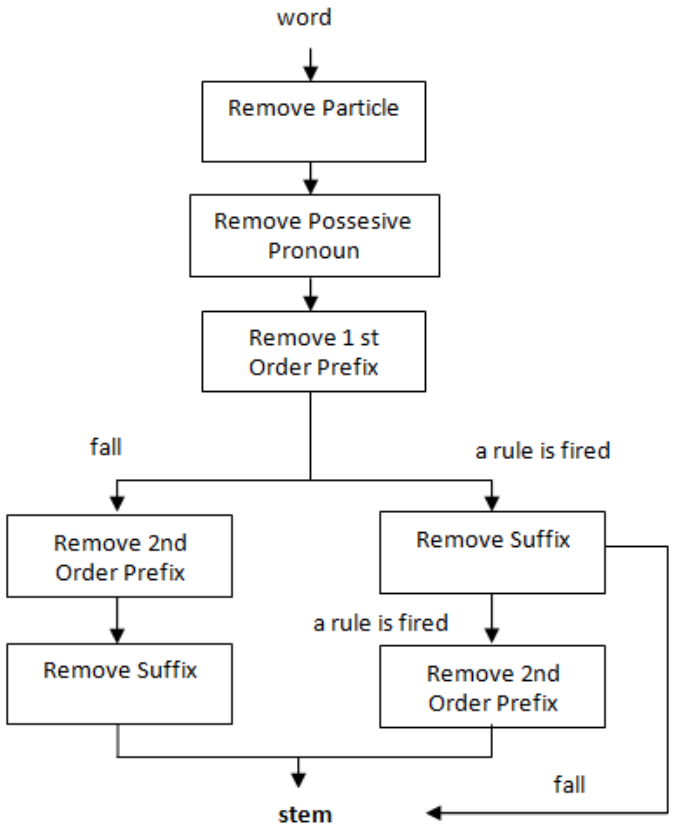
a. Menghilangkan partikel
b. Menghilangkan kata sandang dan kepunyaan
c. Menghilangkan awalan 1

d. Jika suatu aturan terpenuhi jalankan sbb :
   - Hilangkan akhiran
   - Jika suatu aturan terpenuhi, hilangkan awalan 2, jika tidak proses *stemming* selesai.

e. Jika tidak ada aturan yang terpenuhi jalankan sbb :
   a. Hilangkan awalan 2
   b. Hilangkan akhiran
   c. Proses *stemming* selesai.

Morfologi bahasa Indonesia dapat terdiri dari turunan dan imbuhan kata. Imbuhan yang sederhana digunakan akhiran dimana tidak akan merubah makna dari kata dasar.

Arsitektur proses *stemming* untuk bahasa Jawa Ngoko mengacu pada sporter stemmer untuk bahasa Indonesia Tala yang bisa dilihat pada gambar 3.5.



Gambar 3.5. *The basic design of a Porter stemmer for* Bahasa Indonesia (Tala, 2003)

3.3.4.1.1. Proses menghilangkan Awalan

Proses menghilangkan awalan dilakukan untuk mencari kata dasar dengan cara memisahkan awalan dengan kata dasarnya. Dalam bahasa Jawa, jumlah dan jenis prefiks *(ater-ater)* adalah sebagai berikut.

Jenis prefiks selanjutnya adalah sebagai berikut.

a. Awalan yang terdiri dari 4 huruf (4 digit): Kuma, Kapi

Kuma	+	Wani	=	Kumawani
Kapi	+	Lare	=	Kapilare
Kapi	+	Andreng	=	Kapiandreng

b. Awalan yang terdiri dari 3 huruf (3 digit) yaitu: Dak, Kok, Pan, Pra, Tar, Tak, Tok.

Dak	+	Gawa	=	Dakgawa
Dak	+	Tulis	=	Daktulis
Kok	+	Jupuk	=	Kokjupuk
Kok	+	Pangan	=	Kokpangan
Pan	+	gayuh	=	Panggayuh
Pra	+	karsa	=	Prakarsa
Pra	+	lambang	=	Pralambang
Tar	+	Buka	=	Tarbuka
Tak	+	Antem	=	Takantem
Tok	+	Simpen	=	Toksimpen

c. Awalan yang terdiri dari 2 huruf (2 digit) yaitu:Di, Ka, Ke, Ma, Pa, Pi, Sa.

Di	+	Pendhem	=	Dipendhem
DI	+	Pala	=	Dipala
Ka	+	Boyong	=	Kaboyong
Ke	+	Thutuk	=	Kethutuk
Ke	+	Banting	=	Kebanting
Ma	+	Guru	=	Maguru
Ma	+	Gawe	=	Magawe
Pa	+	warta	=	Pawarta
Pa	+	mirsa	=	Pamirsa
Pi	+	wulang	=	Piwulang
Pi	+	takon	=	Pitakon
Sa	+	wiji	=	sawiji
Sa	+	tunggal	=	satunggal
Sa	+	omah	=	saomah

d. Awalan yang terdiri dari 1 huruf (1 digit) yaitu: A

A	+	Gawe	=	Agawe
A	+	Wujud	=	Awujud

3.3.4.1.2. Menghilangkan akhiran

Proses menghilangkan akhiran dilakukan untuk mencari kata dasar dengan cara memisahkan akhiran dengan kata dasarnya. Wujud sufiks dalam bahasa Jawa beserta contoh pemakaiannya tampak dalam deret di bawah ini.

a. Akhiran yang terdiri dari 3 huruf (3 digit) yaitu: Ana, Ane.
   Silih + ana = Silihana
   Jupuk + ana = Jupukana

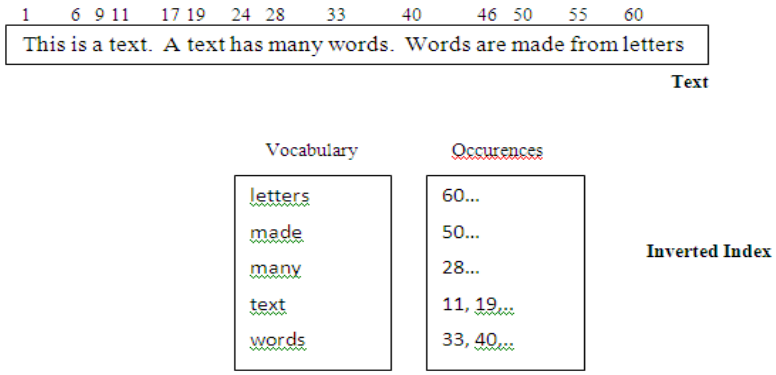
b. Akhiran yang terdiri dari 2 huruf (2 digit) yaitu: An, Na, Ne.
   Tulis + an = tulisan
   Suntik + an = Suntikan
   Gambar + na = Gambarna
   Tulis + na = Tulisna
   Rayi + ne = Rayine
   Sega + ne = Segane

c. Akhiran yang terdiri dari 1 huruf (1 digit) yaitu: A, e, i.
   Tuku + a = Tukua
   Lunga + a = Lungaa
   Mangan + a = mangana
   Omahe + e = Omahe
   Pitik + e = pitike
   Golek + I = Goleki

3.3.5. *Inverted Index*

Pada prinsipnya proses menemukan *records* adalah menjawab dari permintaan *(request)* informasi didasarkan pada kemiripan diantara *query* dan kumpulan *term* pada sistem (Salton, 1989). *Inverted file* atau *inverted index* merupakan mekanisme untuk pengindeksan kata dari koleksi teks yang digunakan untuk mempercepat proses pencarian. Elemen penting dalam struktur *inverted file* ada dua, yaitu: kata *(vocabulary)* dan kemunculan *(occurrences)*. Kata-kata tersebut adalah himpunan dari kata-kata yang ada pada teks, atau merupakan ekstraksi dari kumpulan teks yang ada.

Cara kerja *index inverted*, *term-term* dikonversikan menjadi karakter huruf kecil *(lower-case)*. Kolom *vocabulary* adalah kata-kata yang telah diekstraksi dari koleksi teks, sedangkan *occurrences* adalah posisi kemunculan pada teks (gambar 3.6).



Gambar 3.6 *A sample text and an inverted index built on it (Baeza*, 1999,h.193)

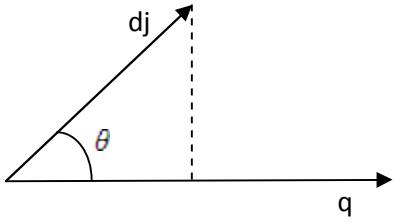
Nilai kemunculan dari kata-kata memerlukan ruangan *(space)* yang tidak sedikit, karena tiap kata muncul pada teks sekali pada struktur *occurrences*, sehingga ada ruangan extra atau dilambangkan dengan *O(n)*. Tidak semua kata diindekskan karena ada kata-kata *stopword* yang dibuang, overhead yang muncul akibat penambahan indeks ini mencapai 30% sampai dengan 40% dari ukuran besar koleksi teks (Baeza, 1999,h.193).

3.3.6. *Vector Space Model*

Vector Space Model (VSM) adalah metode untuk melihat tingkat kedekatan atau kesamaan *(similarity)* term dengan cara pembobotan *term*. Dokumen dipandang sebagi 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).

VSM memberikan sebuah kerangka pencocokan parsial adalah mungkin. Hal ini dicapai dengan menetapkan bobot non-biner untuk istilah indeks dalam query dan dokumen. 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 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*).

Sebuah dokumen dj dan sebuah *query* q direpresentasikan sebagai vektor t-dimensi seperti pada gambar 3.7.



Gambar 3.7 The Cosines of θ is adopted as sim d_j, q (Baeza, 1999)

Dalam VSM koleksi dokumen direpresentasikan sebagai sebuah matrik *term* document (atau matrik *term frequency*). Setiap sel dalam matrik bersesuaian dengan bobot yang diberikan dari suatu *term* dalam dokumen yang ditentukan. Nilai nol berarti bahwa term tersebut tidak ada dalam dokumen. Gambar 3.8 menunjukkan matrik *term document* dengan n dokumen dan t *term*.

	T_1	T_2	T_3	$T_{...}$	T_t
D_1	W_{11}	W_{21}	W_{31}	$\square_{...}$	T_{t1}
D_2	W_{12}	W_{22}	W_{32}	$\square_{...}$	T_{t2}
D_3	W_{13}	W_{23}	W_{33}	$\square_{...}$	T_{t3}
$D_{...}$	$\square_{...}$	$\square_{...}$	$\square_{...}$	$\square_{...}$	$\square_{\square}$
D_n	W_{1n}	W_{2n}	W_{3n}	$\square_{...}$	T_{tn}

Gambar 3.8 Matrik *term-document (Baeza, 1999)*

Proses perhitungan VSM melalui tahapan perhitungan *term frequency (tf), Inverse Document Frequency (idf), term frequency Inverse Document Frequency (tfidf),* Jarak query dan dokumen, pengukuran Similaritas *query document (inner product),* dan pengukuran *Cosine Similarity* (menghitung nilai kosinus sudut antara dua vector).

3.3.6.1. Menghitung bobot dokumen dengan *term-frequency and inverse document frequency (tfidf)*

Langkah pertama dalam perhitungan VSM adalah Pembobotan *term frequency inverse document frequency (tfidf)* yaitu menggabungkan dua konsep untuk perhitungan bobot, frekuensi kemunculan sebuah kata didalam sebuah dokumen *(tf)* dan *inverse document frequency* yang mengandung kata tersebut *(idf)*.

Pembobotan dokumen dengan *tfidf* diawali dengan perhitungan *tf* dan perhitungan *idf*. perhitungan *tf* (*term frequency-tf*) dilakukan untuk memberikan bobot tiap token dalam tiap dokumen. Bobot token ditentukan dari jumlah kemunculan token tersebut di dalam dokumen (*term frequency-tf*). Perhitungan *term frequency* dilakukan untuk melihat frekuensi munculnya term dalam suatu dokumen (local).

Persamaan *term frequency (2)*

$$\boldsymbol{tf = tf_{i,j}} \qquad (2)$$

Dengan *tf* adalah *term frequency,* dan $tf_{i,j}$ adalah banyaknya kemunculan term t_i dalam dokumen d_j, *Term frequency (tf)* dihitung dengan menghitung banyaknya kemunculan term t_i dalam dokumen d_j . Kemunculan *term* t_i bisa dihitung bobotnya dan bisa digunakan untuk perangkingan dokumen. $tf_{i,j}$ menunjukkan setiap dokumen memiliki frekuensi kemunculan *term* yang berbeda-beda, namun jika dokumen yang diolah banyak, akan sulit dibuat perangkingan berdasarkan bobot *term* dalam sebuah dokumen karena akan dijumpai beberapa atau banyak dokumen yang memiliki jumlah *term* yang sama.

Setelah hasil perhitungan *tf* didapatkan, langkah selanjutnya dilakukan perhitungan *inverse document frequency (idf)* . Perhitungan *idf* dilakukan untuk melihat frekuensi kemunculan *term* tersebut dalam dokumen lainnya (global).

Perhitungan *inverse document frequency* dilakukan dengan persamaan (3).

$$\boldsymbol{idf_i = log\ \frac{N}{df_i}} \qquad (3)$$

Dengan idf_i adalah *inverse document frequency, N* adalah jumlah dokumen yang terambil oleh sistem, dan df_i adalah banyaknya dokumen dalam koleksi dimana term t_i muncul di dalamnya, maka Perhitungan idf_i digunakan untuk mengetahui banyaknya term yang dicari (df_i) yang muncul dalam dokumen lain yang ada pada database (korpus). Perhitungan ini didasarkan pada kemunculan term yang sedikit memiliki bobot yang tinggi. Term yang jarang muncul didokumen lain, berarti term tersebut unik dan memiliki bobot yang tinggi. Perhitungan ini berlawanan dengan *term* frekuensi yang melihat frekuensi kemunculan *term* sebagai dasar pembobotan *term*. Nilai idf_i menunjukkan hasil pembobotan, namun demikian hasil pembobotan menggunakan idf_i ini memiliki masalah dalam hal perangkingan dokumen karena banyak ditemukan bobot yang sama antar dokumen.

Selanjutnya, setelah nilai *tf* dan idf_i telah didapatkan, dilakukan normalisasi. Normalisasi digunakan untuk menormalkan vektor dokumen sehingga proses *retrieval* tidak terpengaruh oleh panjang dari dokumen. Normalisasi diperlukan karena dokumen panjang bisaanya mengandung perulangan term yang sama sehingga menaikkan *frekuensi term (tf)*. Dokumen panjang juga mengandung banyak term yang berbeda sehingga menaikkan ukuran kemiripan antar *query* dengan dokumen tersebut, meningkatkan peluang di-retrievnya dokumen yang lebih panjang (Salton, 1989). Perhitungan *tfidf weighting* menggunakan persamaan (4)

$$\boldsymbol{W_{ij} = tf_i \,.\, log\,(\frac{N}{df_i})} \qquad (4)$$

Dengan W_{ij} *adalah bobot dokumen, N adalah* Jumlah dokumen yang terambil oleh system, $tf_{i,j}$ adalah banyaknya kemunculan *term* t_i pada dokumen dj, dan df_i adalah banyaknya dokumen dalam koleksi dimana term t_i muncul di dalamnya. Bobot dokumen (W_{ij}) dihitung untuk didapatkannya suatu bobot hasil perkalian atau kombinasi antara term frequency ($tf_{i,j}$) dan Inverse Document Frequency (df_i). Perhitungan W_{ij} dimaksudkan bahwa pembobotan tidak saja

melihat frekuensi kemunculan suatu *term* pada satu dokumen namun juga memperhitungkan kemunculan *term* pada dokumen yang lain dikorpus. Perhitungan pembobotan dilakukan sejumlah N dokumen yang teambil oleh sistem.

3.3.6.2.Menghitung Jarak Query dan Dokumen

Langkah selanjutnya setelah perhitungan *tf-idf* dilakukan adalah menghitung jarak query dengan persamaan (5)

$$|\boldsymbol{q}| = \sqrt{\sum_{\boldsymbol{j=1}}^{\boldsymbol{t}} (\boldsymbol{W_{i,q}})^{\boldsymbol{2}}} \qquad (5)$$

Dengan |q| adalah Jarak query, dan W_{iq} adalah bobot query dokumen ke-i, maka Jarak *query* ($|q|$) dihitung untuk didapatkan jarak *query* dari bobot query dokumen (W_{iq}) yang terambil oleh sistem. Jarak *query* bisa dihitung dengan persamaan akar jumlah kuadrat dari *query*. Setelah jarak *query* didapatkan, selanjutnya dilakukan perhitungan jarak dokumen menggunakan persamaan (6)

$$\left|\boldsymbol{d_j}\right| = \sqrt{\sum_{\boldsymbol{i=1}}^{\boldsymbol{t}} (\boldsymbol{W_{ij}})^{\boldsymbol{2}}} \qquad (6)$$

Dengan $|d_j|$ adalah jarak dokumen, dan W_{ij} adalah bobot dokumen ke-i, maka Jarak dokumen ($|d_j|$) dihitung untuk didapatkan jarak *dokumen* dari bobot dokumen dokumen (W_{ij}) yang terambil oleh sistem. Jarak dokumen bisa dihitung dengan persamaan akar jumlah kuadrat dari dokumen.

3.3.6.3. Menghitung Similaritas Query dan Dokumen (*inner product)*

Selanjutnya setelah jarak dari tiap dokumen dan query didapatkan, dilakukan perhitungan similaritas antara *query* dengan dokumen (Salton:1989) menggunakan persamaan (7)

$$\boldsymbol{sim}(\boldsymbol{q}, \boldsymbol{d_j}) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{t}} \boldsymbol{W_{iq}} \ . \boldsymbol{W_{ij}} \qquad (7)$$

Dengan W_{ij} adalah bobot term dalam dokumen, W_{iq} adalah bobot query, dan Sim (q, dj) adalah Similaritas antara *query* dan dokumen. Similaritas antara *query* dan dokumen atau *inner product/Sim (q, dj)* digunakan untuk mendapatkan bobot dengan didasarkan pada bobot *term* dalam dokumen (W_{ij}) dan bobot query (W_{iq}) atau dengan cara menjumlah bobot *q* dikalikan dengan bobot dokumen. Pembobotan menggunakan ini didapatkan nilai bobot suatu dokumen dan dapat dibuat rangking, karena yang diinginkan adalah bobot dokumen yang mendekati nilai 1 maka dilakukan proses berikutnya yaitu dengan menghitung *Cosine Similarity*.

3.3.6.4. Menghitung *Cosine Similarity (cosine coefficient)*

Melalui VSM dan *tfidf weighting* akan didapatkan representasi nilai numerik dokumen sehingga dapat dihitung kedekatan antar dokumen. Semakin dekat dua vektor didalam suatu VSM, maka semakin mirip dua dokumen yang diwakili oleh dua vektor tersebut. Kemiripan antar dokumen dapat dihitung menggunakan suatu fungsi ukuran kemiripan *(similarity measure)*. Ukuran ini memungkinkan perankingan dokumen sesuai dengan kemiripannya atau relevansinya terhadap *query*.

Cosine Similarity atau *Sim(q,dj)* digunakan untuk mengevaluasi tingkat similaritas atau kemiripan dari dokumen (d_j) berkaitan dengan query (q) sebagai korelasi antara vektor dj dan q. Korelasi ini bisa diukur, dengan persamaan (1)

$$\boldsymbol{Sim(q, d_j)} = \frac{\boldsymbol{q}.\boldsymbol{d_j}}{|\boldsymbol{q}| * |\boldsymbol{d_j}|} = \frac{\sum_{i=1}^{t} \boldsymbol{W_{iq}} \ . \boldsymbol{W_{ij}}}{\sqrt{\sum_{j=1}^{t} (\boldsymbol{W_{iq}})^2} * \sqrt{\sum_{i=1}^{t} (\boldsymbol{W_{ij}})^2}} \qquad (1)$$

Dengan Sim(q,dj) adalah Similaritas antara *query* dan dokumen, q adalah Bobot *query*, dj adalah Bobot dokumen, |q| adalah Jarak query, dan |dj| adalah Jarak dokumen.

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*.

3.3.7. Uji *Recall* dan *Precision*

Information Retrieval System (IRS) mengembalikan satu set dokumen sebagai jawaban atas *user's query*. Terdapat dua kategori dokumen yang dihasilkan oleh Information Retrieval System terkait pemrosesan *query*, yaitu *relevant document* (dokumen yang relevan dengan *query*) dan *retrieved document* (dokumen yang diterima pengguna). Ukuran umum yang digunakan untuk mengukur kualitas dari text retrieval adalah kombinasi *recall* dan *precision*.

Tujuan uji *Recall* dan *Precision* adalah untuk mendapatkan informasi hasil pencarian yang didapatkan oleh IRS. Hasil pencarian IRS bisa dinilai tingkat *recall* dan *precision* nya. *Precision* dapat dianggap sebagai ukuran ketepatan atau ketelitian, sedangkan *recall* adalah kesempurnaan. Nilai *precision* adalah proporsi dokumen yang terambil oleh sistem adalah relevan. Nilai *recall* adalah proporsi dokumen relevan yang terambil oleh sistem (Salton, 1988).

Nilai *recall* dan *precision* bernilai antara 0 sd 1. Information Retrieval Sistem diharapkan untuk dapat memberikan nilai *recall* dan *precision* mendekati 1. Pengguna rata-rata ingin mencapai nilai *recall* tinggi dan *precision* tinggi, pada kenyataannya hal itu harus dikompromikan karena sulit dicapai (Salton, 1989,h.277). Hasil uji *recall* dan *precision* sistem pada umumnya, jika nilai *recall* tinggi maka nilai *precision* rendah atau jika nilai *recall* rendah maka nilai *precision* tinggi. Pada IRS ini Nilai *recall* dan *precision* dikatakan tinggi berdasarkan pada hasil uji *recall* dan *precision* yang mencapai diatas 50 persen. Jika hasil uji *recall* mencapai nilai diatas 50 persen maka nilai *recall* dikatakan tinggi dibandingkan nilai *precision*, begitu juga sebaliknya jika hasil uji *precision* menghasilkan nilai diatas 50 persen maka nilai *precision* dikatakan tinggi dibandingkan nilai *recall*.

Nilai *recall* dihitung menggunakan persamaan (8)

$$R = \frac{Number\ of\ relevant\ items\ retrieved}{Total\ number\ of\ relevant\ items\ in\ collection} \qquad (8)$$

Dengan R adalah *recall,*maka nilai *R* didapatkan dengan membandingkan *Number of relevant items retrieved* dengan *Total number of relevant items in collection. Recall* adalah dokumen yang terpanggil dari IRS sesuai dengan permintaan *user* yang mengikuti pola dari IRS. Nilai *recall* makin besar belum cukup untuk menilai suatu IRS baik atau tidak.

Nilai *precision* dihitung menggunakan persamaan (9)

$$P = \frac{Number\ of\ relevant\ items\ retrieved}{Total\ number\ of\ items\ retrieved} \qquad (9)$$

Dengan *P* adalah *Precision.* maka nilai *P* didapatkan dengan membandingkan *Number of relevant items retrieved* dengan *Total number of items retrieved.* Precision adalah jumlah dokumen yang terpanggil dari *database* relevan setelah dinilai user dengan informasi yang dibutuhkan. Semakin besar nilai *precision* suatu IRS, maka IRS dapat dikatakan baik.

Kesamaan pembilang yang dibandingkan dalam proses perhitungan *recall* dan *precision* (*number of relevant items retrieved*) memiliki arti bahwa hanya dokumen yang relevan (relevant document in answer set) hasil dari proses irisan antara dokumen (*relevant documents*) pada korpus dengan *query (answer set)* yang akan dihitung untuk didapatkan nilainya (Baeza, 1999).

BAB IV
METODE PENELITIAN

4.1. METODE PENELITIAN

Metodologi yang digunakan pada penelitian ini adalah:

4.1.1. Obyek Penelitian

Obyek penelitian dari penelitian ini adalah Bahasa Jawa Ngoko.

4.1.2. Teknik Pengumpulan Data

Pengumpulan data dimaksudkan agar mendapatkan bahan-bahan yang relevan, akurat dan *reliable.* Maka teknik pengumpulan data yang dilakukan dalam penelitian ini adalah sebagai berikut :

a. Observasi

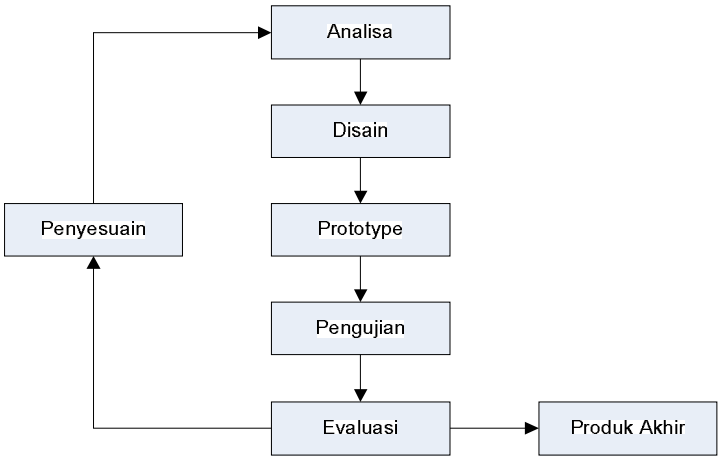
   Melakukan pengamatan dan pencatatan secara sistematis tentang hal-hal yang berhubungan dengan basis data dokumen teks dan kemampuan pencarian kemiripan dokumen.

b. Studi Pustaka

   Pengumpulan data dari bahan-bahan referensi, arsip, dan dokumen yang berhubungan dengan permasalan dalam penelitian ini.

4.1.3. Metode Pengembangan

Penelitian ini menggunakan model *prototype.* Di dalam model ini sistem dirancang dan dibangun secara bertahap dan untuk setiap tahap pengembangan dilakukan percobaan-percobaan untuk melihat apakah sistem sudah bekerja sesuai dengan yang diinginkan. Sistematika model *prototype* terdapat pada Gambar 4.1 memperlihatkan tahapan pada *prototype.*



Gambar 4.1. Tahapan *Prototype (Pressman, 2001)*

Berikut adalah tahapan yang dilakukan pada penelitian ini dengan metode pengembangan *prototype*

a. **Analisa**

   Pada tahap ini dilakukan analisa tentang masalah morfologi bahasa *jawa ngoko* dan menentukan pemecahan masalah yang tepat untuk menyelesaikannya.

b. **Desain**

   Pada tahap ini dibangun rancangan sistem dengan beberapa diagram bantu seperti *Data Flow Diagram*. Pembuatan *Flowcart* dilakukan untuk membuat proses menjadi runtut dan membantu proses desain.

c. ***Prototype***

   Pada tahap ini dibangun aplikasi *Information Retrieval System (IRS)* berbasis web yang sesuai dengan desain dan kebutuhan sistem.

d. **Pengujian**

   Pada tahap ini dilakukan pengujian *software IRS* dengan uji *Recall* dan *Precission*.

e. **Evaluasi**

Pada tahap ini dilakukan evaluasi apakah performa aplikasi sudah sesuai dengan yang diharapkan, apabila belum maka dilakukan penyesuaian-penyesuaian secukupnya.

f. **Penyesuaian**

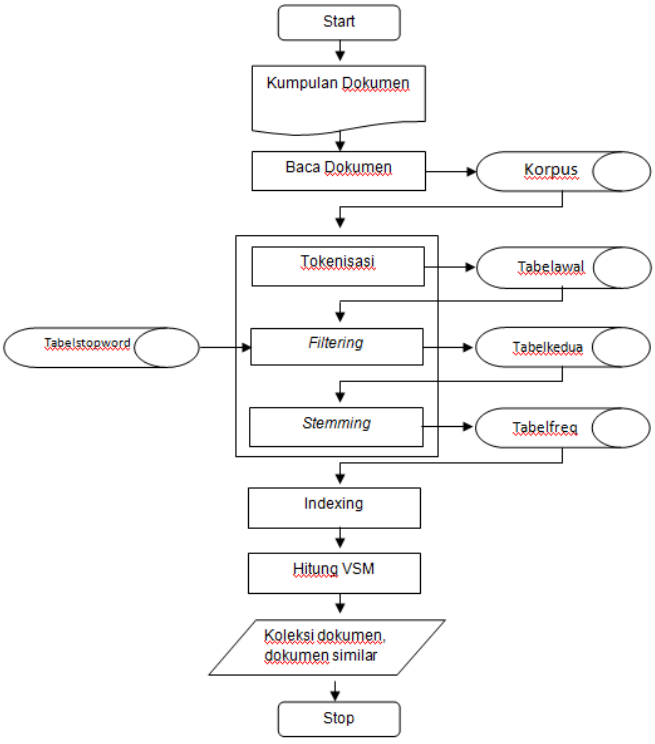
Tahap ini dilakukan apabila pada evaluasi performa aplikasi kurang memadai dan dibutuhkan perbaikan, tahap ini melakukan penyesuaian dan perbaikan pada aplikasi sesuia dengan kebutuhan

BAB V
HASIL DAN PEMBAHASAN

5.1. Rancang Bangun *Information Retrieval System (IRS)*

5.1.1. Flowchart IRS

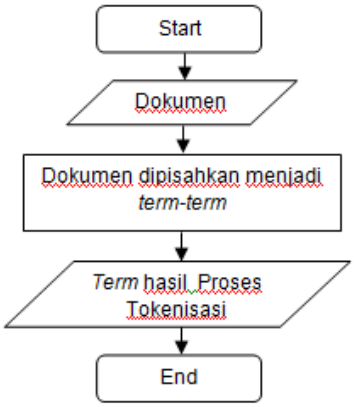
IRS dirancang bisa mengembalikan informasi yang diminta oleh *user* berupa hasil pencarian yang relevan. Proses temu kembali oleh sistem melalui proses-proses seperti gambar 5.1. *Flowchart* diawali dengan *input* dokumen-dokumen kedalam korpus. Selanjutnya dokumen melalui proses preprosesing, dihitung bobotnya dan dibuat rankingnya berdasarkan bobot dokumen yang tertinggi. Hasil IRS adalah dokumen yang relevan dengan permintaan *user*.



Gambar 5.1. *flowchart Information Retrieval System.*

5.1.1.1.Flowchart tokenisasi

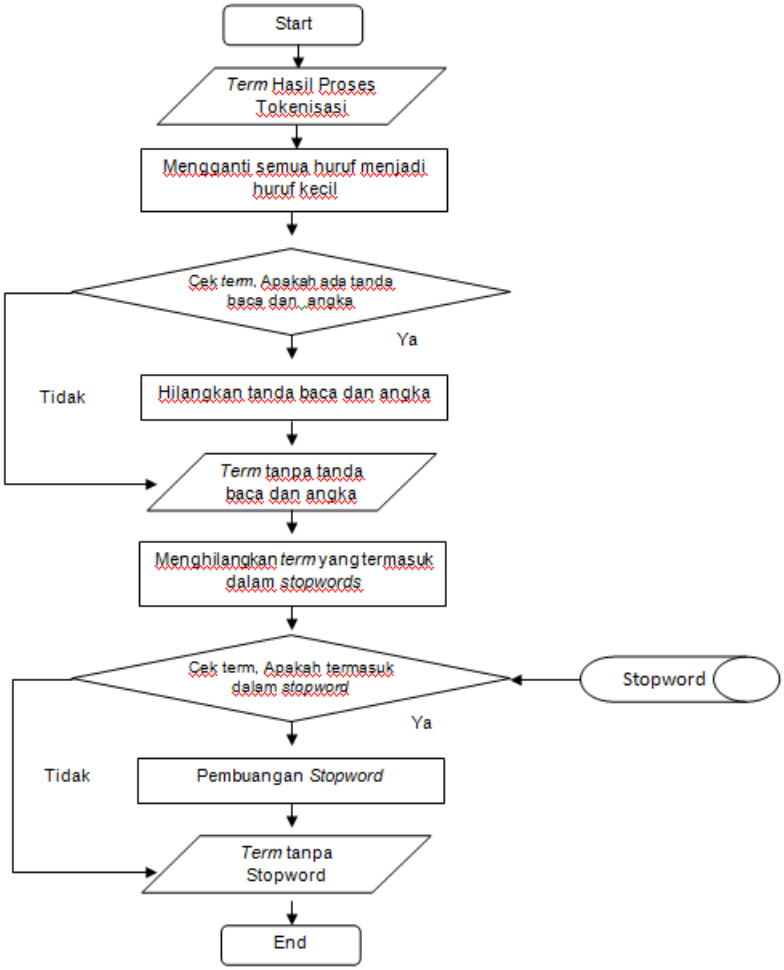
Proses Tokenisasi dirancang untuk dapat memisahkan dokumen menjadi *term-term* yang akan diproses pada tahap *filtering*. Proses tokenisasi diawali dengan *scanner* dokumen yang ada pada korpus kemudian diproses menjadi *term*. *Flowchart* tokenisasi bisa dilihat pada gambar 5.2.



Gambar 5.2 *Flowchart* Proses Tokenisasi

5.1.1.2. Flowchart Filtering

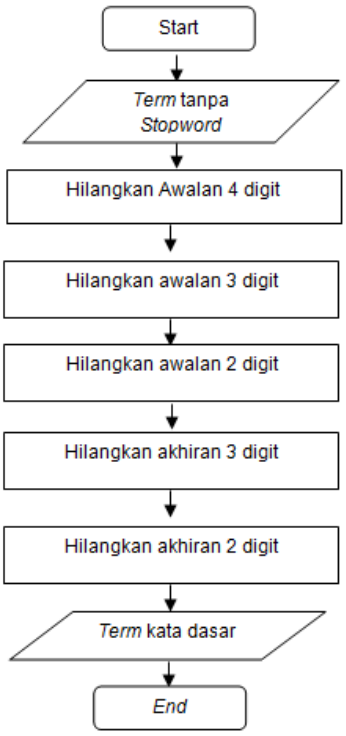
Proses *Filtering* dirancang untuk menghasilkan *term* tanpa *stopwords*. *Flowchart filtering* dimulai dengan mengganti huruf kapital menjadi huruf kecil, menghilangkan tanda baca dan angka, dan menghilangkan *term* yang termasuk dalam *stopwords*. Gambar 5.3. menunjukkan *flowchart* proses *filtering*.



Gambar 5.3. *Flowchart* Proses *Filtering*

5.1.1.3. Flowchart stemming

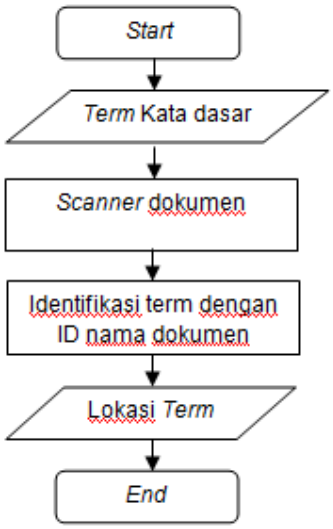
Proses *stemming* dirancang agar *term* hasil *filtering* diubah menjadi *term* kata dasar. Proses *stemming* dimulai dengan menghilangkan awalan dan akhiran. Proses ini juga dirancang dapat melakukan *replace* ketika awalan dihilangkan dan menggantinya dengan huruf yang sesuai. Proses menghilangkan awalan, akhiran, dan *replace* sisipan dilakukan dalam satu tahap proses. Gambar 5.4 menunjukkan *flowchart stemming*.



Gambar 5.4. *Flowchart* Proses *Stemming*

5.1.1.4. Flowchart Indexing

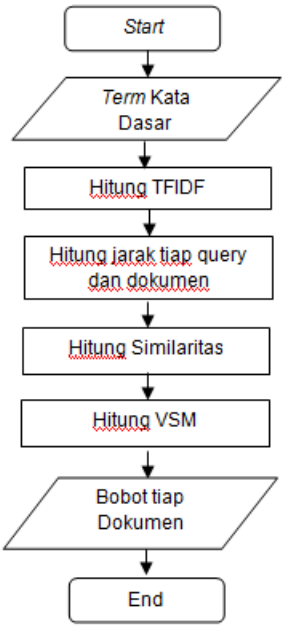
Term kata dasar hasil proses *stemming* selanjutnya dimasukkan dalam tabel untuk diproses pada perhitungan *Vector Space Model*. Proses *indexing* menggunakan metode *inverted indexing*, yaitu dengan membedakan letak tiap term dalam dokumen. Gambar 5.5. menunjukkan *flowchart indexing*.



Gambar 5.5. *Flowchart* Proses *Indexing*

5.1.1.5. Flowchart Hitung VSM

Proses selanjutnya adalah proses perhitungan pembobotan menggunakan metode VSM. Proses ini dimulai dengan perhitungan tf, idf, tfidf, jarak dokumen dan query, similaritas dan *Cosine Similarity* (gambar 5.6). Proses hitung VSM dirancang menghasilkan dokumen hasil pencarian disertai dengan letak dokumen dan bobot dokumen.



Gambar 5.6. *Flowchart* Proses Hitung *Vector Space Model*

5.1.1.6. Rancangan Tabel

Pada Information Retrieval System ini menggunakan beberapa tabel untuk tempat meletakkan kumpulan data pada korpus, *term-term* hasil proses Tokenisasi, *Filtering* dan *Stemming*. Selanjutnya untuk proses perhitungan *vector space model* digunakan tabel *freq* yaitu tabel yang berisi kumpulan *term-term* yang telah menjadi kata dasar. Berikut ini Rancangan tabel yang akan digunakan dalam *Information Retrieval System* pada penelitian ini;

5.1.1.6.1. Rancangan Tabel Korpus

Tabel korpus digunakan untuk meletakkan dokumen-dokumen dengan *field-field* id, judul, isi dan dokumen (tabel 5.1).

Tabel 5.1 Rancangan Tabel Korpus

Field	Type	Collation	Attributs	Null	Default	Extra
id	int(11)	utf8_general_ci		No		auto_increment
judul	varchar(500)	utf8_general_ci		No		
isi	varchar(3000)	utf8_general_ci		No		
dokumen	varchar(100)	utf8_general_ci		No		

5.1.1.6.2. Rancangan Tabel Tabelawal

Tabelawal digunakan untuk meletakkan term hasil tokenisasi dengan *field-field* judul, *term* dan dokumen (tabel 5.2).

Tabel 5.2 Rancangan Tabel Tabelawal

Field	Type	Collation	Attributs	Null	Default	Extra
judul	varchar(250)	utf8_general_ci		No		
term	varchar(500)	utf8_general_ci		No		
dokumen	varchar(100)	utf8_general_ci		No		

5.1.1.6.3. Rancangan Tabel TabelKedua

Tabelkedua digunakan untuk meletakkan term hasil proses *filtering* dengan *field-field* judul, term dan dokumen (tabel 5.3).

Tabel 5.3 Rancangan Tabel Tabelkedua

Field	Type	Collation	Attributs	Null	Default	Extra
judul	varchar(250)	utf8_general_ci		No		
term	varchar(250)	utf8_general_ci		No		
dokumen	varchar(100)	utf8_general_ci		No		

5.1.1.6.4. Rancangan Tabel Tabelfreq

Tabelfreq digunakan untuk meletakkan term hasil proses *stemming* dengan *field-field* judul, *term, freq* dan freqpangkat (tabel 5.4).

Tabel 5.4 Rancangan Tabel Tabelfreq

Field	Type	Collation	Attributs	Null	Default	Extra
judul	varchar(250)	utf8_general_ci		No		
term	varchar(250)	utf8_general_ci		No		
freq	int(11)	utf8_general_ci		No		
frekpangkat	int(11)	utf8_general_ci		No		

5.1.1.6.5. Rancangan Tabel Tabelstopword

Tabelstopword digunakan untuk meletakkan *term stopword* dengan *field-field* term (tabel 5.5).

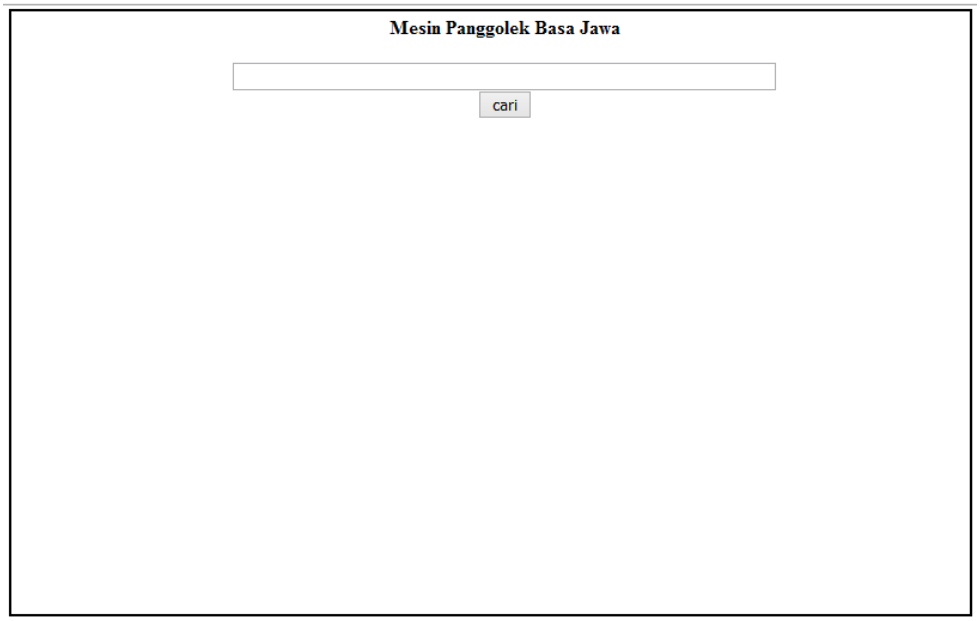
Tabel 5.5 Rancangan Tabel Tabelstopword Jawa Ngoko

Field	Type	Collation	Attributs	Null	Default	Extra
term	varchar(50)	utf8_general_ci		No		

5.1.1.7. Rancangan *Interface*

5.1.1.7.1. Rancangan *Interface Home Page Information Retrieval System*

Pada rancangan *interface* ini akan ditampilkan kolom *query* yang bisa digunakan untuk memasukkan *query* oleh pengguna. Kotak *button* dengan label cari digunakan untuk memproses setelah *query* di *input*. Tombol *button* cari jika sudah diklik akan menampilkan abstraksi hasil pencarian. Gambar rancangan *Interface* dapat dilihat pada gambar 5.7.



Gambar 5.7. Rancangan *Interface Home Page*

5.1.1.7.2. Rancangan Menu Hasil Pencarian

Rancangan menu Hasil Pencarian ini akan tampil setelah Information Retrieval System melakukan pencarian. Sistem akan menampilkan sejumlah halaman yang relevan dengan *query* setelah melalui proses perangkingan. Tata letak hasil pencarian menggunakan bobot yang dimiliki oleh tiap dokumen berdasarkan proses perhitungan *Vector Space Model*. Hasil pencarian dengan bobot terbesar akan diletakkan pada posisi pertama diikuti oleh bobot yang lebih kecil (*descending*).

Rancangan *Interface* hasil pencarian akan menampilkan informasi-informasi tentang dokumen-dokumen yang relevan dengan query yang diinput oleh user. *Interface* hasil pencarian akan menampilkan informasi berupa dokumen-dokumen yang relevan dengan hasil pencarian, dilengkapi dengan nilai similaritas dokumen dan posisi dokumen tersebut.

5.1.1.8. Implementasi Perhitungan

Implementasi *vector space model* agar lebih jelas dibuat contoh *query* (Q) dan dokumen (N = 3), seperti di bawah ini:

Contoh :

Query (Q) = Sistem
Dokumen 1 (d1) = Sistem adalah kumpulan elemen
Dokumen 2 (d2) = adalah kumpulan elemen yang saling berinteraksi
Dokumen 3 (d3) = Sistem berinteraksi untuk mencapai tujuan

Melalui proses tokenizing selanjutnya masuk pada proses *filtering (stopword removal)*, maka kata adalah pada d1, kata adalah dan yang pada d2, serta kata untuk pada d3 dihapus. Selanjutnya, kumpulan kata dasar yang telah terpilih dilakukan proses pembobotan dokumen melalui beberapa perhitungan di bawah ini.

5.1.1.8.1. Menghitung *term frequency (tf)*

Perhitungan bobot *term frequensi* atau *tf* menggunakan persamaan (2) dan perhitungan *Invers Document Frequency* atau *idf* menggunakan persamaan (3). Hasil perhitungan bisa dilihat pada tabel 5.6

Tabel 5.6 Hasil Perhitungan tf

Token	tf				df
	q	d1	d2	d3	
sistem	1	1	0	1	2
kumpul	0	1	1	0	2
elemen	0	1	1	0	2
saling	0	0	1	0	1
interaksi	0	0	1	1	2
capai	0	0	0	1	1
tuju	0	0	0	1	1

5.1.1.8.2. Menghitung *inverse document frequency (idf)*

Setelah hasil perhitungan *tf* didapatkan, langkah selanjutnya dilakukan perhitungan *inverse document frequency (idf)* tiap token untuk menghitung bobot token Hasil perhitungan *inverse document frequency (idf)* bisa dilihat pada tabel 5.7

Tabel 5.7 Hasi Perhitungan idf

Token	tf				df	N/df	idf log(N/df)
	q	d1	d2	d3			
sistem	1	1	0	1	2	1.5	0.176
kumpul	0	1	1	0	2	1.5	0.176
elemen	0	1	1	0	2	1.5	0.176
saling	0	0	1	0	1	3	0.477
interaksi	0	0	1	1	2	1.5	0.176
capai	0	0	0	1	1	3	0.477
tuju	0	0	0	1	1	3	0.477

5.1.1.8.3. Menghitung *term frequency. inverse document frequency (tfidf)*

Selanjutnya, setelah nilai *tf* dan *idf* telah didapatkan, kemudian dimasukkan dalam perhitungan *tf-idf weighting* untuk menghitung bobot hubungan suatu *term* di dalam dokumen. Perhitungan *tfidf* menggunakan persamaan (4) dan hasil perhitungan bisa dilihat pada tabel 5.8

Tabel 5.8 Hasil Perhitungan tfidf

Token	tf				df	N/df	idf log(N/df)	W			
	q	d1	d2	d3				q	d1	d2	d3
sistem	1	1	0	1	2	1.5	0.176	0.176	0.176	0	0.176
kumpul	0	1	1	0	2	1.5	0.176	0	0.176	0.176	0
elemen	0	1	1	0	2	1.5	0.176	0	0.176	0.176	0
saling	0	0	1	0	1	3	0.477	0	0	0.477	0
interaksi	0	0	1	1	2	1.5	0.176	0	0	0.176	0.176
capai	0	0	0	1	1	3	0.477	0	0	0	0.477
tuju	0	0	0	1	1	3	0.477	0	0	0	0.477

5.1.1.8.4. Menghitung jarak query

Perhitungan jarak *query* menggunakan persamaan (5). Hasil perhitungan jarak query bisa dilihat pada tabel 5.9

$= \sqrt{(0.176)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2}$
$= \sqrt{(0.031 + 0 + 0 + 0 + 0 + 0 + 0)}$
$= \sqrt{(0.001)}$
$=$ **0.031**

5.1.1.8.5. Menghitung jarak dokumen

Perhitungan jarak dokumen menggunakan persamaan (6). Hasil perhitungan jarak dokumen bisa dilihat pada tabel 5.9

Dokumen 1 (d1)

$= \sqrt{(0.176)^2 + (0.176)^2 + (0.176)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2}$

$= \sqrt{(0.031 + 0.031 + 0.031 + 0 + 0 + 0 + 0)}$

$= \sqrt{(0.003)}$

= **0.054**

Dokumen 2 (d2)

$= \sqrt{(0)^2 + (0.176)^2 + (0.176)^2 + (0.477)^2 + (0.176)^2 + (0)^2 + (0)^2}$

$= \sqrt{(0 + 0.031 + 0.031 + 0.228 + 0.031 + 0 + 0)}$

$= \sqrt{(0.055)}$

= **0.234**

Dokumen 3 (d3)

$= \sqrt{(0.176)^2 + (0)^2 + (0)^2 + (0)^2 + (0.176)^2 + (0.477)^2 + (0.477)^2}$

$= \sqrt{(0.031 + 0 + 0 + 0 + 0.031 + 0.228 + 0.228)}$

$= \sqrt{(0.105)}$

= **0.325**

Tabel 5.9 Hasil Perhitungan jarak query dan dokumen

Token	q^2	$d1^2$	$d2^2$	$d3^2$
sistem	0.031	0.031	0	0.031
kumpul	0	0.031	0.031	0
elemen	0	0.031	0.031	0
saling	0	0	0.228	0
interaksi	0	0	0.031	0.031
capai	0	0	0	0.228
tuju	0	0	0	0.228
	Sqrt(q)	**Sqrt(di)**		
	0.031	0.054	0.234	0.325

5.1.1.8.6. Menghitung Similaritas Query dan Dokumen

Perhitungan Similaritas Query dan Dokumen menggunakan persamaan (7). Hasil perhitungan similaritas bisa dilihat pada tabel 5.10

Dokumen 1
= 0.001 + 0.001 + 0.001 + 0 + 0 + 0 + 0
= **0.003**
Dokumen 2
= 0 + 0.001 + 0.001 + 0.007 + 0.001 + 0 + 0
= **0.010**
Dokumen 3
= 0.001 + 0 + 0 + 0 + 0.001 + 0.007 + 0.007
= **0.016**

Tabel 5.10 Hasil Perhitungan Similaritas Query dan Dokumen

Token	q^2	$d1^2$	$d2^2$	$d3^2$	**q*d1**	**q*d2**	**q*d3**
sistem	0.031	0.031	0	0.031	0.001	0	0.001
kumpul	0	0.031	0.031	0	0.001	0.001	0
elemen	0	0.031	0.031	0	0.001	0.001	0
saling	0	0	0.228	0	0	0.007	0
interaksi	0	0	0.031	0.031	0	0.001	0.001
capai	0	0	0	0.228	0	0	0.007
tuju	0	0	0	0.228	0	0	0.007
	Sqrt(q)	**Sqrt(di)**			**Sum(q* di)**		
	0.031	0.054	0.234	0.325	0.003	0.010	0.016

5.1.1.8.7. Menghitung *Cosines Similarity*

Perhitungan nilai *Cosinus* sudut antara vector kata kunci dengan tiap dokumen dengan menggunakan persamaan (1). Hasil perhitungan bisa dilihat pada tabel 5.11

$$Sim\ (q, d_1) = \frac{q \cdot d_1}{|q| * |d_1|} = \frac{0.003}{0.031 * 0.054} = \mathbf{0.005}$$

$$Sim\ (q, d_2) = \frac{q \cdot d_2}{|q| * |d_2|} = \frac{0.010}{0.031 * 0.234} = \mathbf{0.075}$$

$$Sim\ (q, d_3) = \frac{q \cdot d_3}{|q| * |d_3|} = \frac{0.016}{0.031 * 0.325} = \mathbf{0.168}$$

Tabel 5.11 Hasil Perhitungan *Cosines Similarity*

Similarity (q, di)	d1	d2	d3
bobot	0.005	0.075	0.168

5.1.1.8.8. Membuat Ranking

Hasil perhitungan *Cosinus similarity* (tabel 3.11) menunjukkan hasil bahwa Dokumen 3 (d3) memiliki tingkat similaritas tertinggi kemudian disusul dengan d2 dan d1. Sehingga pada Hasil pencarian akan dibuat urutan dengan bobot tertinggi akan menempati urutan teratas.

5.2. IRS Bahasa Jawa Ngoko

IRS bekerja melalui beberapa tahapan proses sebagai berikut; Proses Tokenisasi dilakukan dengan mekanisme jika dokumen pada korpus ditemukan spasi, maka *term* yang ada diantara spasi akan di *retrieved* (akan diambil oleh sistem) kemudian *term* ditempatkan dalam tabel tabelawal. Hasil proses berupa *term* asli (term yang masih memiliki imbuhan, tanda baca yang melekat, dan angka). Keunggulan proses ini waktu komputasi cepat.

Proses *Filtering* dilakukan dengan mekanisme jika *term* pada tabel tabelawal ditemukan tanda baca, huruf kapital, dan angka. Maka program akan menghilangkan (tanda baca dan angka) dan mengganti (huruf kapital menjadi huruf kecil), kemudian memeriksa *term* dengan *stopwords*. Hasil proses berupa *term* pilihan (tanpa tanda baca, tanpa huruf kapital, dan bukan termasuk *stopwords*). Keunggulan proses ini sistem mampu mereduksi tanda baca, angka, merubah term menjadi huruf kecil, dan memeriksa term stopwords dengan waktu komputasi yang cepat.

Proses Tokenisasi dan *Filtering* diproses dalam 1 (satu) tahap dengan waktu komputasi 5 detik (jumlah dokumen 252).

Proses *Stemming* dilakukan program dengan cara menghilangkan imbuhan yang terdapat pada term hasil *filtering*. Proses menghilangkan dilakukan dengan menghilangkan awalan, sisipan, dan akhiran. Hasil proses ini dimasukkan dalam tabel

tabelfreq. Hasil proses berupa *term* kata dasar. Keunggulan proses ini waktu komputasi 13 detik (jumlah dokumen 252)

Proses Pembobotan dokumen dengan metode VSM dilakukan dalam proses pencarian dokumen. Program akan bekerja ketika *user* melakukan *query*, selanjutnya program akan memproses *query* tersebut dengan perhitungan-perhitungan *tf*, *idf, tfidf*, jarak *query* dan dokumen, similaritas dan *cosine similarity*. Hasil proses pembobotan dengan metode VSM berupa dokumen hasil pencarian disertai dengan bobot dokumen, letak dokumen dan disusun *descending* (dokumen dengan bobot terbesar diletakkkan di atas). Keunggulan Proses waktu komputasi rata-rata 1,5 detik dengan hasil pencarian yang akurat (*precision* tinggi).

Selengkapnya proses atau cara kerja IRS mulai dari persiapan dokumen hingga pencarian dokumen beserta hasilnya akan dibahas pada proses IRS.

5.3. Implementasi *IRS* Bahasa *Jawa Ngoko*

5.3.1. Memasukkan Dokumen kedalam tabel Korpus

Semua dokumen abstrak di *input* secara manual dengan format dokumen teks. Proses ini dilakukan dengan cara memasukkan abstrak-abstrak skripsi bahan kajian penelitian kedalam tabel korpus. Sebelum dimasukkan kedalam tabel, dibuat satu tabel dengan nama tabel korpus yang digunakan sebagai tempat data. Tabel korpus ini memiliki *fiel-field* id, judul, isi dan dokumen. *Field* id berisi urutan data penelitian didalam korpus yang tersusun sesuai dengan urutan input data. Field judul berisi judul skripsi. Field isi berisi abstrak skripsi dan field dokumen berisi nama dokumen dengan kode tertentu. Proses memasukkan dokumen ke dalam tabel korpus ini memerlukan waktu relative lama bergantung pada jumlah data yang akan di *input* kedalam tabel korpus. Bentuk tabel korpus seperti terlihat pada tabel 5.12.

Pada tabel korpus terdapat 252 Dokumen yang sudah diketahui no urut dan nama dokumen. Dokumen sejumlah 252 tersebut memiliki kapasitas 104.3 kb.

Tabel 5.12. Tabel Korpus

id	judul	isi	dokumen
1	CAPRICORN Sasi Mei Minggu kapapat	Kumawani Kapilare Kapiandreng Dakgawa Daktulis Kokjupuk Kokpangan Panggayuh Prakarsa Pralambang Tarbuka Takantem Toksimpen Dipendhem Dipala Kaboyong Kethutuk Kebanting Maguru Magawe Pawarta Pamirsa Piwulang Pitakon sawiji Agawe Awujud Silihana Jupukana tulisan Suntikan Gambarna Tulisna Rayine Segane Tukua Lungaa mangana Omahe pitike Goleki Becike tansah konsentrasi marang rancangan kang lagi ditindakake. Sabar lan wening iku kuncine bisa kasil Aja kabotan mikirake saingan lan aja gampang peercaya gosip Kasarasan dijaga, utamane yen duwe lara mag Kulawarga harmonis.	CAP54
2	AQUARIUS Sasi Mei Minggu kapapat	Gaweyane isih akeh, durung wayahe leyeh-leyeh. Wong-wong kang bisa menehi bebathen diubungi maneh. Butuhe akeh, nanging pametu uga mundhak, ora perlu kuwatir Samubarang dijaga sing satimbang amrih saras lan bregas Kulawarga butuh dingerteni.	AQU54
3	PISCES Sasi Mei Minggu kapapat	Aja rangu-rangu anggone nggawe putusan Kudu teges lan mantep Sing wicaksana ngecakake dhuwit, aja sembrono anggone tetuku. Nalika ana wektu longgar, dinggo ngaso, amrih awak tetep seger Marang pasangan aja egois, supaya ora paduflon	PIS54
4	ARIES Sasi Mei Minggu kapapat	Yen arep kongsi becike ditliti luwih dhisik, aja mung modhal kekancan. Pengalaman sing kepungkur kena kanggo pancadan Asil lan butuhe isi selaras. Mitra lawas sajak nuwuhake kabecikan Kulawarga dijak muji syukur, ora perlu mikir aneh-aneh.	ARI54
5	TAURUS Sasi Mei Minggu kapapat	Obyekan sing entheng kena digarap, saengga bisa rada sela Wayahe introspeksi, nalika emosi stabil. Yen ana kalodhangan aja disepelekake. Aja gampang kepencut duweke liyan Pasemon saka kiwa tengen ora susah digagas, amrih tentrem atine.	TAU54
6	GEMINI Sasi Mei Minggu kapapat	Ora saben sing dikandhakake wong hya iku bener. Sing ana tetep dilakoni, apamaneh kalodhangan jumedhul. Asile tambah, nanging kepriye bisa nyelengi yen yen blanjane uga mundhak Anggone mamangan pedhes dilongi Sanadyan bener, olehe kandha sing alus.	GEM54

5.3.2. Proses Tokenisasi

Proses *scanner* dokumen korpus menggunakan format teks dilakukan dengan cara masuk kedalam dokumen korpus melalui perantara program php ke dalam database mysql. Proses *scanner* data dilakukan dengan cara *scanner* baris per baris, untuk tiap-tiap file naskah yang ada di dokumen. Tokenisasi dimulai dengan memisahkan *term-term* yang ada pada dokumen korpus menjadi kumpulan term melalui proses *scanner* dengan dasar spasi. Selanjutnya term hasil proses tokenisasi di masukkan kedalam tabelawal dengan menyertakan *field-field* judul, *term* dan dokumen.

Proses tokenisasi dilakukan dengan dua tahap yaitu tahap scanner *term* pada korpus kemudian term hasil *scanner* dimasukkan ke tabelawal dan tahap berikutnya adalah *scanner term* pada tabel awal dan menempatkan *term* hasil scanner di tabel kedua.

Hasil *scanner term* pada korpus diletakkan pada tabel awal. Hasil *scanner term* yang ada pada tabel awal adalah 8.203 term dengan space 418.3 kb. Hasil *scanner file* pada proses tokenisasi bisa dilihat pada tabel 5.13.

Tabel 5.13. Implementasi Proses Tokenisasi pada Tabel Awal

judul	term	dokumen
CAPRICORN Sasi Mei Minggu kapapat	kumawani	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kapilare	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kapiandreng	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dakgawa	CAP54
CAPRICORN Sasi Mei Minggu kapapat	daktulis	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kokjupuk	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kokpangan	CAP54
CAPRICORN Sasi Mei Minggu kapapat	panggayuh	CAP54
CAPRICORN Sasi Mei Minggu kapapat	prakarsa	CAP54
CAPRICORN Sasi Mei Minggu kapapat	pralambang	CAP54
CAPRICORN Sasi Mei Minggu kapapat	tarbuka	CAP54
CAPRICORN Sasi Mei Minggu kapapat	takantem	CAP54
CAPRICORN Sasi Mei Minggu kapapat	toksimpen	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dipendhem	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dipala	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kaboyong	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kethutuk	CAP54

Selanjutnya proses scanner file dari tabel awal ke tabel kedua dilakukan dengan cara penscanneran term yang ada pada tabel awal. Hasil scanner *term* selanjutnya diletakkan pada tabel kedua. Hasil penscanneran term pada tabel kedua adalah 4.689 term dengan space 245.5 Kb bisa dilihat pada tabel 5.14.

Tabel 5.14. Implementasi Proses Tokenisasi pada Tabel Kedua

judul	term	dokumen
CAPRICORN Sasi Mei Minggu kapapat	kumawani	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kapilare	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kapiandreng	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dakgawa	CAP54
CAPRICORN Sasi Mei Minggu kapapat	daktulis	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kokjupuk	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kokpangan	CAP54
CAPRICORN Sasi Mei Minggu kapapat	panggayuh	CAP54
CAPRICORN Sasi Mei Minggu kapapat	prakarsa	CAP54
CAPRICORN Sasi Mei Minggu kapapat	pralambang	CAP54
CAPRICORN Sasi Mei Minggu kapapat	tarbuka	CAP54
CAPRICORN Sasi Mei Minggu kapapat	takantem	CAP54
CAPRICORN Sasi Mei Minggu kapapat	toksimpen	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dipendhem	CAP54
CAPRICORN Sasi Mei Minggu kapapat	dipala	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kaboyong	CAP54
CAPRICORN Sasi Mei Minggu kapapat	kethutuk	CAP54

5.3.3. Proses *Filtering*

Proses selanjutnya setelah proses tokenisasi adalah proses *filtering*. Proses filtering dilakukan untuk menghilangkan *term-term* yang tidak memiliki arti dengan menggunakan *stopword list* tala. Proses filtering adalah proses baca tabel kedua untuk diperiksa apakah semua term memiliki term-term yang termasuk dalam stopword list menurut tala. Jika dalam tabel kedua terdapat *term-term* yang termasuk dalam *stopword*, maka akan dilakukan penghilangan *term-term* tersebut. Hasil dari proses *filtering* menjadikan jumlah *term* berkurang menjadi 4.641 dari jumlah *term* semula 4.689. Pengurangan ini membuat space yang dibutuhkan untuk tabel term menjadi 251.5 kb. Hasil proses filtering selanjutnya dimasukkan dalam tabel freq (tabel 5.15).

Tabel 5.15 Hasil Proses Filtering pada Tabel Frekuensi

judul	term
AQUARIUS Sasi Januari Minggu Kapindho	tam
AQUARIUS Sasi Januari Minggu Kapindho	lara.
AQUARIUS Sasi Januari Minggu Kapindho	lara
AQUARIUS Sasi Januari Minggu Kapindho	ati
AQUARIUS Sasi Januari Minggu Kapindho	ruweting
AQUARIUS Sasi Januari Minggu Kapindho	pikir
AQUARIUS Sasi Januari Minggu Kapindho	gampang
AQUARIUS Sasi Januari Minggu Kapindho	percaya
AQUARIUS Sasi Januari Minggu Kapindho	tenag
AQUARIUS Sasi Januari Minggu Kapindho	forsir
AQUARIUS Sasi Januari Minggu Kapindho	dhuwit bisa
AQUARIUS Sasi Januari Minggu Kapindho	saras
AQUARIUS Sasi Januari Minggu Kapindho	utama.
AQUARIUS Sasi Januari Minggu Ketelu	wiwit
AQUARIUS Sasi Januari Minggu Ketelu	susu
AQUARIUS Sasi Januari Minggu Ketelu	garep-arep
AQUARIUS Sasi Januari Minggu Ketelu	gedhen. sing
AQUARIUS Sasi Januari Minggu Ketelu	baku

5.3.4. Proses *stopword Removal*

Proses membuang ***stopword (stopword removal)*** dilakukan untuk menghilangkan *term-term* yang tidak memiliki arti dengan menggunakan ***stopword jawa ngoko***. Proses ini dilakukan dengan cara *scanner* dan scanner tabel kedua. Tabel *stopword* jawa ngoko bisa dilihat pada tabel 5.16 dan lebih lengkapnya bisa dilihat pada lampiran 1.

Tabel 5.16 Tabel Stopword Jawa Ngoko

term
adhep
adhepe
adoh
aja
ajaa
akeh
akhir
akhire
akhiri
aku
amarga
amerga
amergane
ana
anadene

5.3.5. Proses *Stemming*

Proses *stemming* yang digunakan adalah proses *stemmer* menggunakan *stemmer* untuk bahasa Jawa ngoko berdasarkan stemmer bahasa Indonesia yang dibuat Tala. Proses stemming dengan menggunakan *stemmer* jawa ngoko melalui beberapa tahapan seperti terlihat pada gambar 5.4 dan untuk mendukung proses ini juga digunakan *stopword list jawa ngoko*. Hasil akhir dari proses stemming adalah kumpulan *term* yang sudah menjadi kata dasar yang diinput dalam tabel *freq*.

Proses *stemming* menghasilkan kumpulan term berupa kata dasar hasil scanner *term* pada tabel kedua. Proses *stemming* didukung stopword jawa ngoko yang digunakan untuk mengurangi term yang ada pada tabel kedua. Selanjutnya *term* hasil *stemming* di letakkan pada tabel freq (tabel 5.17). Jumlah *stopword jawa ngoko* adalah 573 term dengan space 12.2 KB

Tabel 5.17 Tabel Frekuensi

judul	term	freq	freqpangkat
AQUARIUS Sasi Januari Minggu Kapindho	tam	9	81
AQUARIUS Sasi Januari Minggu Kapindho	lara.	5	25
AQUARIUS Sasi Januari Minggu Kapindho	lara	9	81
AQUARIUS Sasi Januari Minggu Kapindho	ati	16	256
AQUARIUS Sasi Januari Minggu Kapindho	ruweting	2	4
AQUARIUS Sasi Januari Minggu Kapindho	pikir	2	4
AQUARIUS Sasi Januari Minggu Kapindho	gampang	21	441
AQUARIUS Sasi Januari Minggu Kapindho	percaya	12	144
AQUARIUS Sasi Januari Minggu Kapindho	tenag	3	9
AQUARIUS Sasi Januari Minggu Kapindho	forsir	1	1
AQUARIUS Sasi Januari Minggu Kapindho	dhuwit bisa	1	1
AQUARIUS Sasi Januari Minggu Kapindho	saras	23	529
AQUARIUS Sasi Januari Minggu Kapindho	utama.	1	1
AQUARIUS Sasi Januari Minggu Ketelu	wiwit	17	289
AQUARIUS Sasi Januari Minggu Ketelu	susu	4	16

5.3.6. Proses *Indexing*

Proses *indexing* dilakukan untuk mengambil atau meretrieve *term-term* yang ada pada tabelfreq untuk selanjutnya diproses pada saat pencarian dilakukan oleh Information Retrieval System. Proses perhitungan dilakukan langsung pada Information Retrieval System saat *query* diproses oleh sistem.

User memasukkan Kata Kunci (*query*) pada mesin pencari, kemudian setelah kata kunci ditulis mesin pencari akan melakukan pencarian *query* pada *database* dengan mengolahnya terlebih dahulu sesuai dengan arsitektur mesin pencari menggunakan metode *vector space model* dan memberikan hasil pencarian.

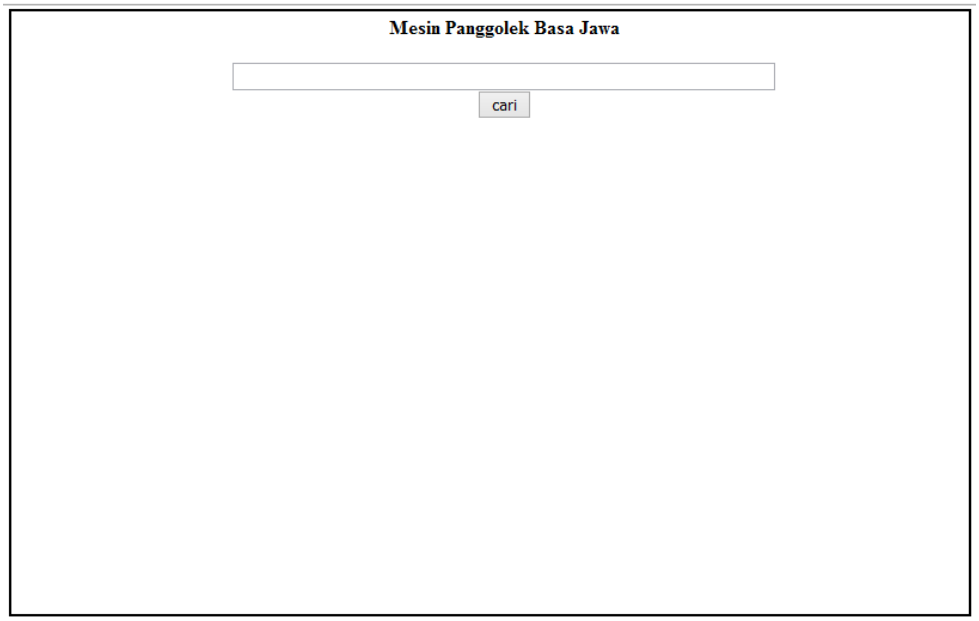
5.3.7. Proses *Vector Space Model*

Information Retrieval System akan melakukan proses perhitungan dimulai dari menghitung tfidf, menghitung jarak *query* dan jarak dokumen, menghitung similaritas produk, dan menghitung bobot dokumen. Information Retrieval System akan mengeksekusi *query* dari *user* dan akan mengolah *query* tersebut (gambar 5.18).

Query yang di *input* oleh user selanjutnya akan dilakukan pencarian pada tabel freq kemudian dilakukan perhitungan pembobotan menggunakan metode *Vector Space Model*. Perhitungan dilakukan dalam sistem pencarian, sistem pencarian akan melakukan perhitungan kemudian akan menampilkan hasilnya seperti pada gambar 5.9. Hasil pencarian akan menampilkan nama dokumen di korpus, kemudian bobot similaritas dan disusun berdasarkan perankingan. Bobot terbesar akan menempati ranking teratas pada hasi pencarian.

5.4. Prosedur

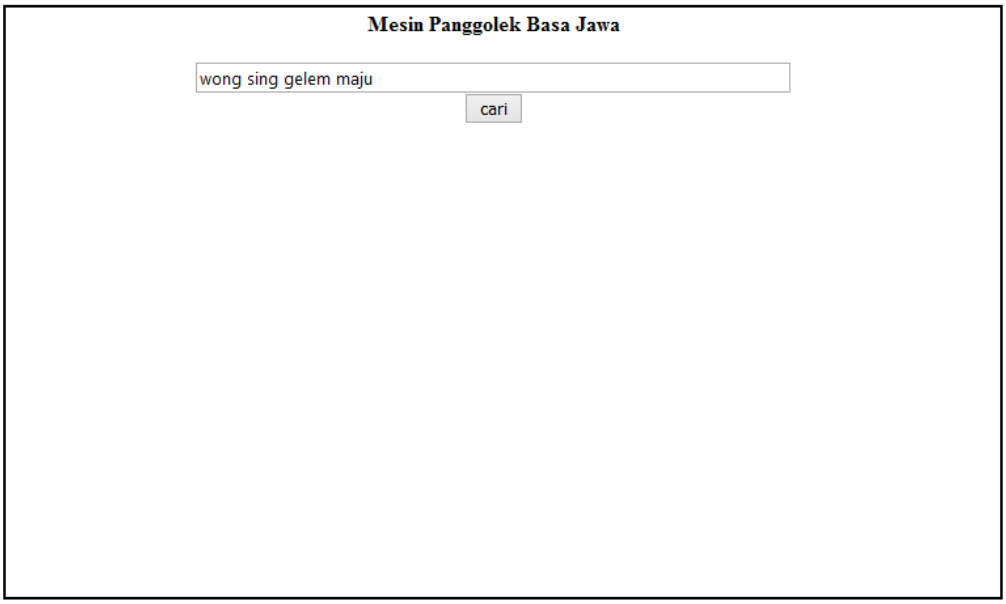
Information Retrieval System ini dirancang agar *user* mudah menggunakan dalam mencari dokumen yang relevan. Tampilan *interface* juga dirancang seperti mesin pencari pada umumnya, sehingga siapapun usernya akan langsung mudah beradaptasi dalam menggunakan mesin pencari (gambar 5.8). Prosedur menggunakan Information Retrieval System ini sangat mudah, yaitu *user* hanya perlu menuliskan *query* atau kata kunci yang akan di cari pada kotak dialog kemudian setelah *query* di masukkan *user* tinggal mengklik tombol cari atau tekan *enter*.



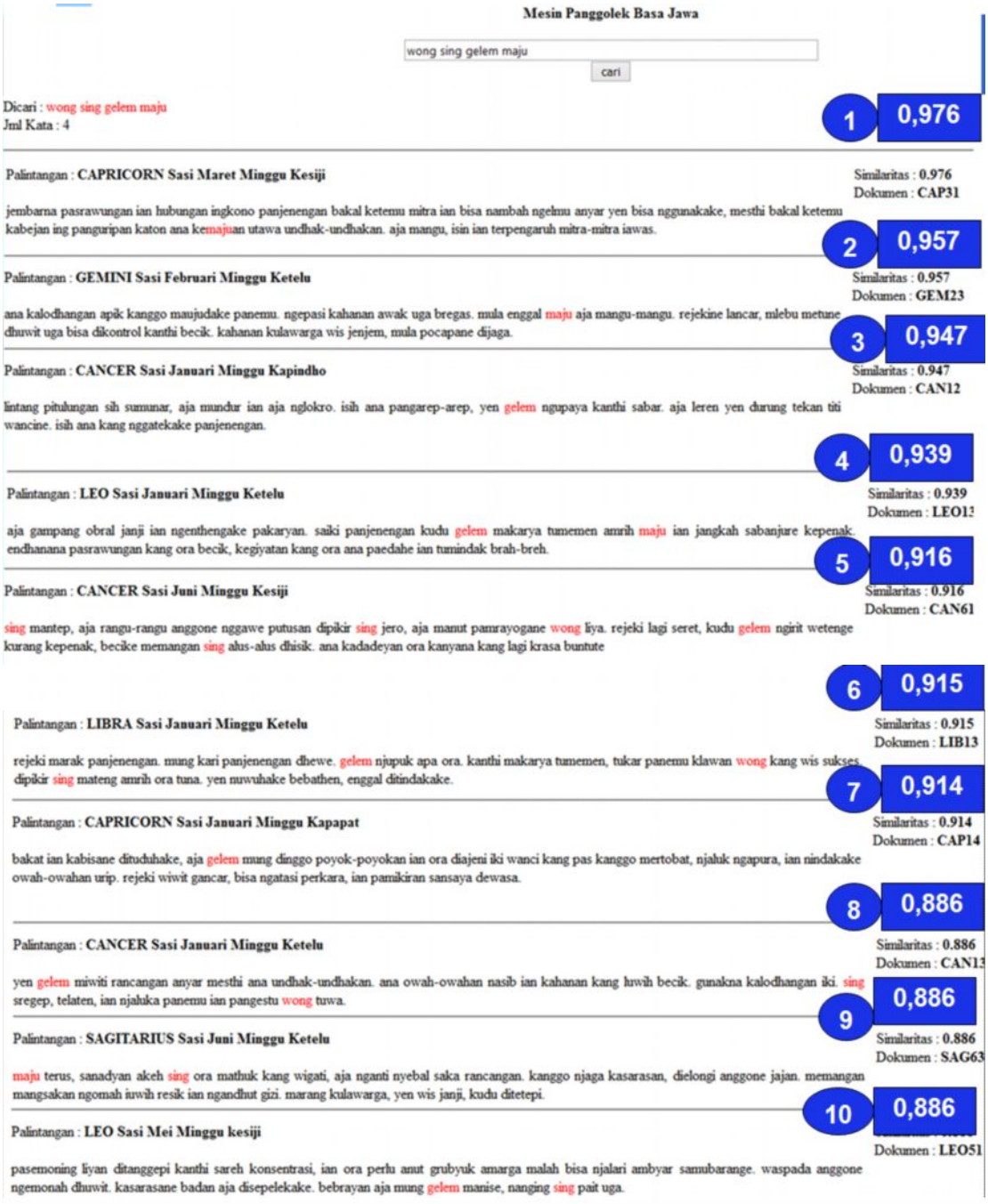
Gambar 5.8. *Interface* Mesin *Panggolek Basa Jawa*

5.5. Studi Kasus *Keyword* Bahasa Jawa Ngoko

Studi kasus pada aplikasi Information Retrieval System ini menggunakan dokumen-dokumen Palintangan Basa Jawa pada Majalah Online Penjebar Semangad yang terdapat pada 12 Palintangan (zodiak) yaitu; Capricorn, Aquarius, Pisces, Aries, Taurus, Gemini, Cancer, Leo, Virgo, Libra, Scorpio, dan Sagitarius. *Query* yang dimasukkan pada Information Retrieval System adalah *keyword* dengan 2 *term* yaitu "*seneng ngalah*" , 3 *term* "*wong angel sukses*". 4 *term* "*wong sing gelem maju*". 5 *term* "*wong sing sabar lan waspada*". 6 *term* "*wong sing angel sukses lan mutungan*". Dan 7 *term* "*wong sing angel sukses lan angel maju*".



Gambar 5.9 Aplikasi Mesin Panggolek Basa Jawa



Gambar 5.10 Hasil Pencarian *keyword*

Hasil pencarian dokumen dengan keyword "*Wong sing gelem maju*", menunjukkan dokumen dengan bobot tertinggi adalah dokumen letak dokumen CAP31 (bobot 0,976). Dokumen CAP31 memiliki frekuensi kemunculan *term* tertinggi dan tidak banyak muncul didokumen lain untuk *term* "*Wong sing gelem maju*". Dokumen CAP31 (dokumen Palintangan Capricorn Sasi Maret Minggu Kesiji nomer 31) memiliki bobot tertinggi atau memiliki tingkat kemiripan tertinggi dibandingkan dengan dokumen lain yang ada pada korpus.

Dari Hasil pencarian *Term* "*Wong sing gelem maju*" dengan hasil didapatkan 10 peringkat teratas paling banyak hasil didapatkan oleh palintangan CANCER (3 kali muncul).

5.6. Pengujian *recall* dan *precision*

Pengujian *recall (P)* dan *precision (R)* dilakukan dengan cara *input query* ke dalam Information Retrieval System *input* 1 *term*, 2 *term* dan 3 *term, 4 term, 5 term, 6 term dan 7 term*. Perhitungan *recall* dan *precision* menggunakan persamaan (8) dan persamaan (9). Hasil pengujian *recall* dan *precision* dengan menguji 1 *term*, 2 *term* dan 3 *term* sampai dengan 7 term menunjukkan bahwa jika *recall* rendah maka *precision* akan tinggi, selengkapnya terlihat pada tabel 5.18.

Tabel 5.18 Hasil Pengujian *Recall* dan *Precision*

No	Query	Recall	Precision
1	Sabar	0.05	1.00
2	Sukses	0.02	1.00
3	seneng ngalah	0.04	0.82
4	wong angel sukses	0.03	0.80
5	wong sing susah asile	0.07	0.77
6	wong sing gelem maju	0.06	0.89
7	wong sing ora gelem maju	0.06	0.78
8	wong sing sabar lan waspada	0.04	0.77
9	wong sing angel sukses lan mutungan	0.04	0.90
10	wong sing angel sukses lan angel maju	0.01	0.67

5.6.1. Pengujian program berdasarkan waktu

Pengujian program dengan didasarkan pada waktu komputasi dengan menggunakan Komputer *processor Intel Core i3*, 1,8 GHz, RAM 2 GB, *Harddisk* 250 GB. Sistem operasi Windows 8.

5.6.1.1. Waktu *Preprosesing*

Preprosesing yang dididalamnya terdapat proses Tokenisasi, *filtering* dan *Stemming*. Proses preprosesing pada penelitian ini dibagi menjadi dua tahap, yaitu pada tahap awal adalah memproses data yang ada di korpus kemudian memasukkannya pada tabel tabelawal dan tabel tabelkedua. Selanjutnya dengan menggunakan tabelkedua diolah dan data berupa *term* yang sudah menjadi kata dasar dimasukkan pada tabel tabelfrek.

Waktu yang dibutuhkan dalam proses komputasi proses tokenisasi dengan jumlah dokumen 252 adalah 3 detik. Sedangkan waktu komputasi untuk proses *filtering* adalah 13 detik. Jadi total waktu yang dibutuhkan dalam proses preprosesing dengan jumlah dokumen 252 adalah 18 detik (perhitungan waktu komputasi dilakukan menggunakan *stopwatch*).

5.6.1.2. Waktu Pencarian

Waktu komputasi preprosesing bergantung pada jumlah data yang akan diolahnya. Semakin besar data yang akan diproses membutuhkan waktu yang lebih lama. Waktu preposesing untuk *scanner file* dari tabelkedua menjadi tabelfrek dipilih dengan jumlah dokumen 252. Waktu komputasi proses pembobotan bergantung pada jumlah *term* yang terambil untuk diolah. Semakin besar jumlah *term* yang harus diolah akan membutuhkan waktu yang lebih lama. Waktu pencarian rata-rata untuk pencarian 1, 2, 3, 4, 5, 6 dan 7 *keyword* adalah 5 detik.

BAB VI
SIMPULAN DAN SARAN

6.1. Kesimpulan

a. IRS mampu melakukan pencarian dokumen teks dan menampilkan hasil pencarian dokumen teks berbahasa Jawa Ngoko dengan disertai bobot tiap dokumen beserta letak dokumen
b. Hasil Uji *recall* dan *precision* STKI menunjukkan hasil pencarian dokumen teks memiliki rata-rata *recall* = 0,04 dan rata-rata *precision* = 0,84.
c. IRS yang dibangun memiliki keunggulan mampu melakukan pencarian dokumen teks bahasa jawa ngoko dan hasil pencarian yang akurat (*precision* = 0,84), serta dilengkapi dengan bobot dan letak dokumen pada database.

6.2. Saran

a. Penulisan bahasa jawa yang benar perlu dilakukan karena berkemungkinan tidak akan terambil oleh sistem jika penulisannya salah.
b. Proses *stemming* yang ada masih belum bisa sepenuhnya membuat semua *term* kedalam bentuk *term* kata dasar dengan benar. Proses ini akan mempengaruhi hasil untuk proses indexing, sehingga akan mempengaruhi hasil akhir perhitungan.

DAFTAR PUSTAKA

Budi, I., Aji, R.F., 2006. Efektifitas Seleksi Fitur dalam Sistem Temu Kembali Informasi. Seminar Nasional Aplikasi Teknologi Informasi (SNATI), ISSN : 1907-5022.

Bum, K.Y., 2010. *An autonomous assessment system based on combined latent semantic kernels. Expert Systems with Applications: An International Journal , Volume 37 Issue 4.*

Bunyamin., 2008. Aplikasi IR CATA dengan Metode *Generalized Vector Space Model*, Jurnal Informatika, Vol. 4. No. 1

Bo, Y., 2009. Combining neural networks and semantic feature space for email classification. Knowledge-Based *Systems* , Volume 22 Issue 5.

Erk, K., 2008. *A Structured Vector Space Model for Word Meaning in Context. Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics*. 897-906.

Erk, K., 2010. *A flexible, corpus-driven model of regular and inverse selectional preferences . Computational Linguistics* , Volume 36 Issue 4.

Haryono, M.E.A., 2005. Pembentukan Intisari Topik secara Otomatis dalam suatu Paragraf dengan *Vector Space Model.* Seminar Nasional Aplikasi Teknologi Informasi, SNATI 2005.

Haryono, M.EA., Wahyudi., 2005. *Customer Information Gathering* menggunakan Metode Temu Kembali Informasi dengan Model Ruang Vektor. Seminar Nasional Aplikasi Teknologi Informasi, SNATI 2005.

Hasugian, J., 2006. Penggunaan Bahasa Alamiah dan Kosa Kata Terkendali dalam Sistem Temu Balik Informasi berbasis Teks. Jurnal Studi Perpustakaan dan Informasi, Vol 2, No 2 Desember.

Kadir, A., 2001. Dasar Pemrograman Web Dinamis menggunakan PHP. Penerbit Andi. Yogyakarta.

Lopez, C., Ribeiro, C., 2010. *Using Local Precision to Compare Search Engines in Consumer Health Information Retrieval.* **SIGIR '10:** *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*.

Manning, C., Raghavan, P., 2007. An Introduction to Information Retrieval, Stanford. USA.

Mao, W., 2007. *The phrase-based vector space model for automatic retrieval of free-text medical documents*. *Data & Knowledge Engineering* , Volume 61 Issue 1.

Meadow, C.T., 1997. *Text Information Retrieval Systems*. Academic Press.New York.

Mcenery,1998.**W3Corporaproject**.*1998*http://www.essex.ac.uk/linguistics/external/clmt/w3c/corpus_ling/content/introduction2.html diakses tanggal 14 juli 2012.

Mondal, D., Gangopadhyay, A., 2010. *Medical Decision Making Using Vector Space Model.* IHI '10**:** *Proceedings of the 1st ACM International Health Informatics Symposium*.

Peranginangin, K., 2006. Aplikasi Web dengan PHP dan MySQL. Penerbit Andi, Yogyakarta.

Tala, F.Z., 2003*, A Study of Stemming Effects on Information Retrieval in bahasa Indonesia*. Institut for logic, Language and Computation Universiteit van Amsterdam The Netherlands.

Rijsbergen, C.J.V., 1979. Information *Retrieval, Second Edition. Butterworths*. London.

Salton, G., 1989, *Automatic Text Processing, The Transformation, Analysis, and Retrieval of information by computer*. Addison – Wesly Publishing Company, Inc. USA.

Yates, R.B, 1999. *Modern Information Retrieval*, *Addison Wesley*-Pearson international edition, Boston. USA.