PEMBUATAN APLIKASI PENCARIAN DOKUMEN BERBASIS GENERALIZED VECTOR SPACE MODEL DAN SEMANTIC RELATEDNESS

Oleh:

Andreas Handojo

Charistian Widjaja

Adi Wibowo

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS KRISTEN PETRA

SURABAYA

2012

LAPORAN PENELITIAN
NO: 131/Pen/Informatika/I/2012

PEMBUATAN APLIKASI PENCARIAN DOKUMEN BERBASIS GENERALIZED VECTOR SPACE MODEL DAN SEMANTIC RELATEDNESS

Oleh:

Andreas Handojo

Charistian Widjaja

Adi Wibowo

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS KRISTEN PETRA

SURABAYA

2012

LEMBAR IDENTITAS DAN PENGESAHAN

LAPORAN HASIL PENELITIAN

1.	a. Judul Penelitian	:	Pembuatan Aplikasi Pencarian Dokumen Berbasis Generalized Vector Space Model dan Semantic Relatedness
	b. Nomor Penelitian	:	131/Pen/Informatika /I/2012
	c. Jalur Penelitian	:	I / II / III / IV
2.	Ketua Peneliti		
	a. Nama lengkap dan Gelar	:	Andreas Handojo, S.T., M.MT.
	b. Jenis Kelamin	:	Laki-laki
	c. Pangkat/Golongan/NIP	:	Pembina Tingkat I/ IVB / 00016
	d. Bidang Ilmu yang diteliti	:	Teknologi Informasi
	e. Jabatan Akademik	:	Lektor
	f. Fakultas/Program studi	:	Fakultas Teknologi Industri / Teknik Informatika
	g. Universitas	:	Universitas Kristen Petra
3.	Anggota Tim Peneliti (I)	:	
	a. Nama lengkap dan Gelar	:	Charistian Widjaja, S.Kom
	b. Jenis Kelamin	:	Laki-laki
	c. Pangkat/Golongan/NIP	:	-
	d. Bidang Ilmu yang diteliti	:	Teknologi Informasi
	e. Jabatan Akademik	:	-
	f. Fakultas/Program studi	:	Fakultas Teknologi Industri / Teknik Informatika
	g. Universitas	:	Universitas Kristen Petra
	Anggota Tim Peneliti (II)	:	
	a. Nama lengkap dan Gelar	:	Adi Wibowo, S.T, M.T.
	b. Jenis Kelamin	:	Laki-laki
	c. Pangkat/Golongan/NIP	:	Penata / IIIC / 00003
	d. Bidang Ilmu yang diteliti	:	Sistem Informasi
	e. Jabatan Akademik	:	Asisten Ahli
	f. Fakultas/Program studi	:	Fakultas Teknologi Industri / Teknik Informatika
	g. Universitas	:	Universitas Kristen Petra
4.	Lokasi Penelitian	:	Surabaya
5.	Kerjasama dengan Instansi lain		
	Nama Instansi	:	-
	Alamat	:	-
6.	Tanggal Penelitian	:	Juli s/d Desember 2012
7.	Biaya	:	Rp. 425.000

Surabaya, 21 Desember 2012

Mengetahui,
Ketua Jurusan/ Ka. Unit

Yulia, M.Kom
NIP. 99-036

Ketua Peneliti

Andreas Handojo, M.MT.
NIP. 00-016

Menyetujui,
Dekan Fakultas Teknologi Industri

Djoni Haryadi Setiabudi, M.Eng
NIP. 85-009

ABSTRAK

Dengan pesatnya perkembangan dalam penggunaan teknologi komputer baik di perusahaan maupun di bidang pendidikan, maka semakin banyak pula dokumen-dokumen yang berbentuk digital. Untuk mencari dokumen-dokumen tersebut dibutuhkan waktu yang relatif lama apabila pencariannya dilakukan secara manual. Metode yang sering dipergunakan untuk mencari dokumen adalah *Vector Space Model* (VSM). Kelemahan utama dari VSM adalah tidak mampu menemukan dokumen yang walaupun relevan dengan kata kunci tetapi tidak mengandung kata kunci tersebut. Oleh karena itu dibutuhkan sebuah metode *search engine* yang dapat memanfaatkan kemiripan makna antar kata untuk mengatasi masalah diatas.

Salah satu metode yang dipergunakan dalam perancangan *search engine* adalah *Generalized Vector Space Model* (GVSM). Metode GVSM merupakan perkembangan dari metode VSM dengan menambahkan kemampuan untuk mempertimbangkan kedekatan *sense* antar *term* dalam merepresentasikan dokumen. George Tsatsaronis dan Vicky Panagiotopolou mengembangkan metode GVSM dengan melakukan pemberian nilai kedekatan antar *sense* didapatkan dengan metode *Semantic Relatedness* yang mempergunakan *database* leksikal *"WordNet"*.

Dari hasil pengujian yang dilakukan maka GVSM menghasilkan hasil pencarian dokumen-dokumen yang memiliki nilai *recall* yang sama atau lebih tinggi jika dibandingkan dengan VSM. Sedangkan nilai *precision* dari hasil pencarian GVSM memiliki nilai yang lebih rendah jika dibandingkan dengan nilai *precision* dari hasil pencarian VSM.

Kata kunci : *Vector Space Model, Generalized Vector Space Model, Semantic Relatedness, Search Engine*.

ABSTRACT

With the rapid developments in the use of computer technology in the corporate and education, so many documents in digital form. To search these documents, it takes a relatively long time when the quest is done manually. The common method for searching document is Vector Space Model (VSM). The main weaknesses from VSM is this method can't find the relevant document with the keyword, but the document not contain that keyword. Therefore needed a method of search engine that can take advantage of the similarity of meaning between words to solve the above problems.

One of the methods used in the design for search engine is the Generalized Vector Space Model (GVSM). GVSM method is the development of VSM by adding the ability to consider the sense of closeness between the term in representing the document. George Tsatsaronis and Vicky Panagiotopolou developed GVSM method to perform the proximity between the scores obtained by the the Semantic Relatedness lexical database "WordNet".

From the results of the tests performed GVSM generate results documents that have a higher recall value when compared to the VSM, but the precision of search results GVSM have a lower value when compared with the precision of search results.

Keywords : *Vector Space Model, Generalized Vector Space Model, Semantic Relatedness, Search Engine*.

KATA PENGANTAR

Penulis mengucapkan syukur kepada Tuhan Yang Maha Esa atas terselesaikannya penelitian ini. Penulis sadar bahwa hasil penelitian ini masih jauh dari sempurna, karena itu penulis mengharapkan saran dan kritik yang membangun dari berbagai pihak demi perbaikan dari penelitian ini.

Penulis berharap semoga penelitian ini dapat memberikan kontribusi bagi Jurusan Teknik Informatika dan juga bagi perkembangan bidang ilmu sistem informasi pada umumnya.

Akhir kata, penulis mengucapkan terima kasih kepada semua pihak yang tidak dapat disebutkan satu persatu yang telah membantu terselesaikannya penelitian ini

Surabaya, Desember 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

PENDAHULUAN

1.1 Latar Belakang

Dengan semakin pesatnya perkembangan dalam penggunaan teknologi komputer baik di perusahaan maupun di bidang pendidikan, maka semakin banyak pula dokumen-dokumen yang berbentuk digital. Untuk mencari dokumen-dokumen tersebut dibutuhkan waktu yang relatif lama apabila pencariannya dilakukan secara manual. Maka dari itu dibutuhkan sebuah search engine yang dapat mencari dokumen-dokumen yang relevan secara lebih mudah. Salah satu metode yang dipergunakan dalam perancangan search engine adalah Vector Space Model.

Ning Liu et al. (2004) mendefinisikan Vector Space Model (VSM) sebagai metode yang mengukur kemiripan antara suatu dokumen dengan suatu query user dengan menggunakan cosinus dari sudut antar vektor yang dibentuk oleh dokumen dengan vektor dari kata kunci yang diinputkan oleh user. Kristopher (2005) menyatakan salah satu kelemahan dari VSM adalah metode ini menganggap bahwa setiap term pada dokumen bersifat independen, yaitu metode ini tidak melihat hubungan makna dengan term lain. Sebagai contoh, apabila user melakukan pencarian dengan kata kunci "programming" maka hasil pencariannya adalah semua dokumen yang hanya memiliki kata "programming" saja, padahal masih banyak dokumen-dokumen yang masih berhubungan makna dengan kata "programming" seperti "PHP", "Java" , dan lain-lain. Dengan adanya kasus ini maka terjadi penurunan recall dari hasil pencarian. Karena itu dibutuhkan metode yang dapat mengembangkan VSM ini dengan menambahkan fungsi sense pada model ini yaitu GVSM (Generalized Vector Space Model).

Tsatsaronis dan Panagiotopoulou (2009) mendefinisikan Generalized Vector Space Model adalah model pencarian pengembangan dari Vector Space Model yang menambahkan fungsi sense dan penilaian terhadap hubungan makna antar term dalam dokumen. Generalized Vector Space Model (GVSM) adalah Vector Space Model yang mempertimbangkan kedekatan sense antar term dalam merepresentasikan dokumen. Dalam GVSM ini pemberian nilai kedekatan antar sense didapatkan dengan metode Semantic Relatedness. Dimana metode

Semantic Relatedness adalah metode yang menghitung nilai kedekatan sense dengan menggunakan kedalaman term dalam thesaurus dan banyaknya path yang dilalui antar dua term yaitu term yang ada di dokumen dan term pada kata kunci dari user. Dalam melakukan perhitungan dengan menggunakan metode Semantic Relatedness ini dibutuhkan thesaurus kata seperti "WordNet". Upaya penggunaan metode GVSM dan Semantic Relatedness ini dimaksudkan untuk meningkatkan recall dari hasil pencarian sehingga hasil pencariannya mencakup dokumen-dokumen yang relevan terhadap kata kunci dari user.

1.2 Permasalahan

Permasalahan yang dihadapi dan diharapkan dapat diselesaikan melalui penelitian ini adalah bagaimana mengimplementasikan sebuah searching engine guna mencari dokumen yang dibutuhkan dengan menggunakan metode GVSM (Generalized Vector Space Model) dan Semantic Relatedness.

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah membantu pencarian dokumen tertentu sesuai dengan kebutuhan pengguna.

1.4 Manfaat Penelitian

Manfaat dari penelitian ini adalah membantu pengguna dalam melakukan pencarian dokumen yang dibutuhkan oleh pengguna secara cepat dan akurat.

1.5 Ruang Lingkup Pembahasan

Penelitian yang dilakukan akan memperhatikan beberapa batasan sebagai berikut :

- Aplikasi ini melakukan pencarian pada dokumen-dokumen yang didapatkan dari "ClueWeb09_English_Sample.warc" pada *Web Track* TREC (*The Text Retrieval Conference*) yang seluruhnya menggunakan bahasa Inggris.
- Pencarian dilakukan berdasarkan isi dari dokumen *text*, dan jika terdapat gambar akan diabaikan.

- Proses pemisahan kata-kata yang ada pada dokumen dengan menggunakan proses *parsing*.
- Menghilangkan kata-kata yang tidak dibutuhkan menggunakan *stopword*. *Database stopword* yang digunakan bersumber dari internet (http://www.ranks.nl/resources/stopwords.html).
- Penentuan *similarity* antar kata didasarkan pada perhitungan dengan menggunakan metode *Semantic Relatedness*.
- *Thesaurus* yang dipergunakan dalam perhitungan *Semantic Relatedness* didapat dari *database "WordNet"* dalam bahasa Inggris.
- Penentuan kesamaan antara dokumen dengan kata kunci dari *user* yang didasarkan pada kedekatan makna, dihitung dengan menggunakan metode GVSM (*Generalized Vector Space Model*).
- *Output* dari aplikasi ini adalah dokumen-dokumen yang memiliki kemiripan dengan kata kunci dari *user*.
- Proses mencari informasi yang berkaitan dengan dokumen adalah *user* meng-klik tombol cari pada website, setelah memasukkan kata kunci dari *user* dan secara langsung akan tersambung ke sebuah halaman yang menampilkan *link-link* dari dokumen yang bersangkutan dengan kata kunci yang dimasukkan sebelumnya. *User* dapat meng-klik *link* tersebut dan langsung teralihkan pada dokumen yang dipilih.
- Pengujian dilakukan dengan menghitung *recall* dan *precision* dari hasil pencarian dan mengukur kecepatan pencarian dokumen.
- Aplikasi dibuat dengan menggunakan *PHP*, selain itu juga menggunakan *MySQL* untuk penyimpanan *database*.

1.6 Sistematika Penyusunan Laporan

Laporan penelitian ini secara keseluruhan terdiri dari lima bab dimana secara garis besar masing-masing bab membahas hal-hal sebagai berikut:

BAB 1 **Pendahuluan**: berisi latar belakang, permasalahan, tujuan penelitian, manfaat penelitian, ruang lingkup permasalahan, dan sistematika penyusunan laporan.

BAB 2 **Tinjauan Pustaka**: membahas tentang teori-teori dasar yang relevan dan metode yang digunakan untuk memecahkan persoalan yang dibahas pada penelitian ini.

BAB 3 **Metode Penelitian**: membahas tentang metode penelitian yang dilakukan serta perancangan aplikasi.

BAB 4 **Hasil Penelitian dan Pembahasan**: berisi tentang hasil dari penelitian, berupa aplikasi yang telah dikembangkan beserta dengan pengujian aplikasi tersebut.

BAB 5 **Kesimpulan dan Saran**: berisi kesimpulan yang mencakup beberapa hal penting pada hasil yang didapat dari penelitian dan saran-saran yang diajukan bagi penyempurnaannya.

BAB 2. TINJAUAN PUSTAKA

2.1 Vector Space Model

Menurut Turney, P.D. & Pantel, P. (2010), *Vector Space Model* adalah suatu model yang digunakan untuk mengukur kemiripan antara suatu dokumen dan suatu *query* dengan mewakili setiap dokumen dalam sebuah koleksi sebagai sebuah titik dalam ruang (vektor dalam ruang vektor). Poin yang berdekatan di ruang ini memiliki kesamaan semantik yang dekat dan titik yang terpisah jauh memiliki kesamaan semantik yang semakin jauh. Kesamaan antara vektor dokumen dengan vektor *query* tersebut dinyatakan dengan cosinus dari sudut antar keduanya (Ning Liu et al. 2004).

Dalam metode *Vector Space Model* dihitung *weighted* dari setiap *term* yang didapat dalam semua dokumen dan *query* dari *user*. *Term* adalah suatu kata atau suatu kumpulan kata yang merupakan ekspresi verbal dari suatu pengertian. Perhitungan *weighted* tersebut dilakukan dengan persamaan yang didapat dari http://www.miislita.com/term-vector/term-vector-3.html dan *Document ranking and the vector-Space Model* (Dik, L. LEE. Et al. 1997) yaitu:

$$Term\ Weight: w_i = tf_i * \log \frac{D}{df_i} \quad (2.1)$$

tfi = frekuensi *term* atau banyak *term* i yang ada pada sebuah dokumen (*Term Frequency*)

dfi = frekuensi dokumen atau banyak dokumen yang mengandung *term* i (*Inverse Document Frequency*)

D = banyaknya dokumen yang terdapat pada *database*

Setelah mendapatkan *weighted* dari setiap *term* maka dihitung *magnitude* vektor untuk setiap dokumen (|Di|) dan vektor dari *query* (|Q|) dengan rumus:

$$|D_i| = \sqrt{\sum_j w_{i,j}^2} \quad (2.2)$$

$|D_i|$ = *magnitude* vektor dari dokumen i (D_1=Dokumen ke-1)

$|w_{i,j}|$ = bobot *term* j pada dokumen i ($w_{1,saya}$=bobot kata "saya" di dokumen 1)

i = indeks untuk menyatakan urutan dokumen

j = indeks untuk menyatakan *term*

$$|Q| = \sqrt{\sum_j w_{Q,j}^2} \quad (2.3)$$

|Q| = *magnitude* njang vektor dari *query* Q

$|w_{Q,j}|$ = bobot *term* j pada *query* ($w_{Q,saya}$=bobot kata "saya" di *query user*)

i = indeks untuk menyatakan urutan dokumen

j = indeks untuk menyatakan *term*

Kemudian dihitung *dot product* dari bobot *term* pada *query* dan dokumen.

$$Q \cdot D_i = \sum_j w_{Q,j} w_{i,j} \quad (2.4)$$

$w_{Q,j}$ = bobot *term* j pada *query* Q

$w_{i,j}$ = bobot *term* j pada dokumen i

i = indeks untuk menyatakan urutan dokumen

j = indeks untuk menyatakan *term*

Setelah itu dihitung cosinus dari sudut antar vektor dokumen dengan vektor *query* dengan rumus:

$$\cos \theta_{D_i} = \mathrm{Sim}(Q, D_i) \quad (2.5)$$

$$\mathrm{Sim}(Q, D_i) = \frac{\sum_j w_{Q,j} w_{i,j}}{\sqrt{\sum_j w_{Q,i}^2} \sqrt{\sum_j w_{i,j}^2}} \quad \text{Dim} \quad (2.6)$$

Sim (Q,D_i) = nilai kesamaan antara sebuah dokumen i dengan *query* Q

$w_{Q,j}$ = bobot *term* j pada *query* Q

$w_{i,j}$ = bobot *term* j pada dokumen i

i = indeks untuk menyatakan urutan dokumen

j = indeks untuk menyatakan *term*

Hasil cosinus tersebut diurutkan dan diranking dari hasil yang terbesar ke hasil yang terkecil, dimana hasil terbesar memiliki kedekatan yang lebih baik dengan *user query*. (http://www.miislita.com/term-vector/term-vector-3.html)

2.2 *Generalized Vector Space Model*

Generalized Vector Space Model (GVSM) adalah perkembangan dari *Vector Space Model* yang mempertimbangkan kedekatan *sense* antar *term* dengan lebih akurat, dalam merepresentasikan dokumen. Wong et al. (1987) membuat GVSM pertama, yang memperkenalkan korelasi antar *term*, yang menganggap

$$\cos(\vec{d_k}, \vec{q}) = \frac{\sum_{j=1}^{n} \sum_{i=1}^{n} a_{ki} q_j \vec{t_i} \vec{t_j}}{\sqrt{\sum_{i=1}^{n} a_{ki}^2 \sum_{j=1}^{n} q_j^2}} \quad (2.7)$$

bahwa setiap *term* dinyatakan sebagai kombinasi linier dari vektor 2 dimensi. Pengukuran *similiarity* antara sebuah dokumen dengan sebuah *query* dilakukan dengan rumus ini:

Dimana, t_i dan t_j adalah *term* vektor di sebuah ruang vektor 2 dimensi; d_k,q adalah vektor dokumen dan *query*; a_{ki} adalah bobot (*weight*) dari dokumen yang dihitung dengan rumus *Term Weight*; q_j adalah bobot (*weight*) dari *query* yang dihitung dengan rumus *Term Weight*; n adalah dimensi ruang. (Tsatsaronis, G. & Panagiotopoulou V.,2009,p.70)

2.3 *Semantic Relatedness*

Dalam *Semantic Relatedness* nilai dari t_i dan t_j dalam rumus GVSM Wong et al. dicari dengan rumus baru yang dikembangkan oleh George Tsatsaronis dan Vicky Panagiotopoulou dengan bantuan *database* leksikal "*WordNet*". Nilai t_i dan t_j dihitung dengan SCM (*semantic compactness*), SPE (*semantic path elaboration*), dan SR (*semantic relatedness*). Langkah-langkah mencari nilai t_i dan t_j adalah:

- Definisi pertama, sebuah *thesaurus* kata O, sebuah bagan berat (*weight*) untuk *edge* yang menentukan sebuah *weight* e ? (0,1) untuk setiap *edge*, sepasang

senses S=(s_1,s_2), dan sebuah panjang *path* l yang menyambungkan 2 *senses*. *Semantic compactness* dari S adalah :

$$SCM(S,O) = \prod_{i=1}^{l} e_i \quad (2.8)$$

dimana e_1, e_2, e_3 adalah *path's edges*

Jika $s_1 = s_2$ maka SCM(S,O) = 1 dan jika tidak ada *path* antar keduanya maka SCM(S,O) = 0.

- Definisi kedua, sebuah *thesaurus* kata O dan sepasang *senses* S=(s_1,s_2), dimana s_1,s_2 ? O dan s_1 ? s_2 dan sebuah panjang *path* l(L) yang menyambungkan 2 *senses*. *Semantic path elaboration* dari S adalah :

$$SPE(S,O) = \prod_{i=1}^{l} \frac{2 d_i d_{i+1}}{d_i + d_{i+1}} \cdot \frac{1}{d_{max}} \quad (2.9)$$

dimana d_i adalah kedalaman *sense* s_i yang didasarkan pada O dan d_{max} adalah kedalaman maksimum dari O.

Jika $s_1 = s_2$ dan d =d_1 = d_2 maka SPE(S,O) = d/d_{max} dan jika tidak ada *path* antar keduanya maka SPE(S,O) = 0.

- Definisi ketiga sebuah *thesaurus* kata O, sepasang *term* T=(t_1,t_2) dan semua pasang *senses* S=(s_{1i},s_{2j}), dimana s_{1i},s_{2j} yang merupakan *sense* dari t_1,t_2. *Semantic relatedness* dari T adalah :

$$SR(T,S,O) = \max\{SCM(S,O) \cdot SPE(S,O)\} \quad (2.10)$$

SR antar dua *terms* t_i,t_j dimana t_i = t_j = t dan t ? O didefinisikan dengan 1. Jika t_i ? O tapi t_j ? O atau t_i ? O tapi t_j ? O, SR=0.

(Tsatsaronis, G. & Panagiotopoulou V.,2009,p.70)

2.4 Precision dan Recall

Precision dan *Recall* adalah salah satu metode untuk melakukan evaluasi pada kinerja dari sistem informasi retrieval. *Precision* merupakan jumlah dokumen relevan yang ditemukan dengan total jumlah dokumen yang ditemukan oleh *search engine*.

Precision mengindikasi kualitas himpunan jawaban, tetapi tidak memandang jumlah dokumen yang relevan dalam kumpulan dokumen. Maka dari itu diperlukan *Recall* yaitu rasio jumlah dokumen relevan yang ditemukan dengan total jumlah dokumen dalam kumpulan dokumen yang dianggap relevan.

$$precision = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ items\ retrieved} \quad (2.11)$$

$$recall = \frac{number\ of\ relevant\ items\ retrieved}{number\ of\ relevant\ items\ in\ collection} \quad (2.12)$$

Idealnya, *recall* dan *precision* harus memiliki nilai yang seimbang, yang berarti bahwa sistem mengembalikan semua dokumen yang relevan tanpa mengembalikan dokumen yang tidak relevan dalam hasil yang ditetapkan. Sayangnya, hal ini tidak mungkin untuk dicapai dalam praktek. Jika kita mencoba untuk meningkatkan *recall*, maka *precision* akan menurun. Selain itu kita hanya dapat meningkatkan *precision* dengan mengorbankan *recall*. Selain itu, sering kali ada *trade off* antara pengambilan efektivitas dan komputasi biaya. (Dik, L. LEE., Huei, C. & Kent E. S., 1997, p.68)

2.5 Parsing

Menurut Lusiana *et al*, 2008 *Parsing* adalah tahap mengambil *term-term* dari dokumen dan *query* dengan cara memotong *string input* berdasarkan tiap kata yang menyusunnya (dikutip oleh Zafikri A.). Elemen teks (*string input*) dipisahkan dengan teknik parsing menggunakan fungsi *split* dimana pemisahan *string* dilakukan berdasarkan *white space* (spasi atau tab) untuk kemudian diletakkan pada *array*. Selain itu, juga mencakup pemisahan atau *parsing* terhadap kata-kata majemuk menggunakan *database* kata majemuk yang diambil dari *database* "*WordNet*".

2.6 Stopword

Dalam proses ini digunakan sebuah daftar kata buang (*stoplist*) yaitu daftar kata-kata yang tidak digunakan (dibuang) karena tidak signifikan dalam

membedakan dokumen atau *query*. Stoplist ini terdiri atas kata tugas, kata hubung, kata bantu, yang mempunyai fungsi dalam kalimat penyusun dokumen tetapi tidak memiliki arti. Proses ini dilakukan dengan cara membandingkan hasil parsing dengan *database stopword* dan membuang semua hasil *parsing* yang sama dengan *stopword*. (Zafikri A., 2010)

2.7 WordNet

WordNet adalah *database* leksikal untuk bahasa Inggris. Kata benda, kata kerja, kata sifat dan kata keterangan dikelompokkan ke dalam *synsets* (synonym set). *Synsets* yaitu kumpulan kata yang merepresentasikan suatu makna. *WordNet* juga merupakan *database* yang bebas untuk didownload. Struktur *WordNet* yang membuatnya menjadi alat yang berguna bagi komputasi linguistik dan pengolahan bahasa alami (http://wordnet.princeton.edu/wordnet/).

WordNet menyerupai *thesaurus*, dimana *thesaurus* ini berisi kelompok kata yang didasarkan pada maknanya. Namun, ada beberapa perbedaan penting. Pertama, Interlinks *WordNet* bukan hanya menyambungkan bentuk katanya (string of letters) tetapi lebih spesifik yaitu makna dari kata tersebut. Akibatnya, kata-kata yang ditemukan memiliki kedekatan dengan kata lainnya yang disatukan secara *semantic*. Kedua, label-label *WordNet* yaitu hubungan semantik antara kata-kata, sedangkan kelompok kata dalam *thesaurus* tidak mengikuti pola yang eksplisit selain arti kesamaan. (http://wordnet.princeton.edu/wordnet/)

Perbedaan antara *WordNet* dengan kamus bahasa pada umumnya adalah kamus bahasa memfokuskan pada kata sedangkan *WordNet* memfokuskan diri kepada makna kata. Satu makna dalam *WordNet* dapat dinyatakan dengan *synsest.* Selain dari representasi makna, di dalam *WordNet* juga terdapat relasi/hubungan antar makna seperti *synonym, hypernym*, *hyponym*, *holonym*, *meronym,* dan lain-lain.

Synonym adalah hubungan simetris antara kata-kata (*synsets*). Jika Y merupakan *synonym* dari X, maka Y mempunyai arti yang sama dengan X.

Hypernym adalah hubungan transitif antara kata-kata (*synsets*) yang disebut dengan *super-name*. Jika Y merupakan *hypernym* dari X, maka setiap X merupakan anggota dari Y.

Hyponym hubungan transitif antara kata-kata (*synsets*) yang disebut dengan *sub-name*. Jika Y merupakan *hyponym* dari X, maka setiap Y merupakan anggota dari X.

Holonym adalah hubungan kompleks antara kata-kata (*synsets*) yang disebut dengan *whole-name*. Biasanya dibedakan menjadi komponen (*component parts*), substantif (*substantive parts*), dan anggota (*member parts*). Jika Y merupakan *holonym* dari X, maka X merupakan bagian dari Y.

Meronym adalah hubungan kompleks antara kata-kata (*synsets*) yang disebut dengan *part-name*. Biasanya dibedakan menjadi komponen (*component parts*), substantif (*substantive parts*), dan anggota (*member parts*). Jika Y merupakan *meronym* dari X, maka Y merupakan bagian dari X. (Miller, G. A., 1995, p.40)

2.8 TREC

TREC merupakan kependekan dari *The Text Retrieval Conference* yang disponsori oleh *National Institute of Standards and Technology* (NIST) dan *U.S. Department of Defense* yang dimulai pada tahun 1992 sebagai bagian dari TIPSTER *Text program*. Tujuannya adalah untuk mendukung komunitas peneliti *information retrieval* dengan menyediakan infrastruktur yang diperlukan untuk evaluasi metodologi pengambilan teks dalam skala besar.

TREC diawasi oleh suatu komite program yang terdiri dari perwakilan pemerintah, industri, dan akademisi. Untuk setiap TREC, NIST memberikan set dokumen dan pertanyaan. Peserta menjalankan *system* atau program mereka sendiri pada data, dan mengembalikan hasilnya berupa daftar peringkat atas dokumen yang diambil tersebut ke NIST. NIST mengumpulkan hasil individu, menilai kebenaran hasil peringkat, dan mengevaluasi hasilnya. Siklus TREC diakhiri dengan sebuah *workshop* yang merupakan forum bagi para peserta untuk berbagi pengalaman. (http://trec.nist.gov/overview.html)

BAB 3. METODE PENELITIAN

Guna pengaplikasian penelitian ini maka digunakan langkah-langkah penelitian sebagaimana akan dijelaskan pada bab ini.

3.1 Metodologi Penelitian

3.1.1 Studi Literatur

Studi literatur yang dilakukan meliputi beberapa hal, antara lain:

- Mempelajari metode GVSM (Generalized Vector Space Model) dan Semantic Relatedness.

3.1.2 Perancangan dan Pembuatan Sistem

Melakukan perancangan sistem dengan cara melakukan analisa dan pembuatan *flowchart*, desain program berorientasi objek serta perancangan *interface* yang dilanjutkan dengan pembuatan sistem. Pembuatan sistem meliputi beberapa bagian, antara lain:

- Melakukan perancangan desain GVSM (Generalized Vector Space Model) dan Semantic Relatedness.
- Melakukan perancangan desain *user interface*.
- Melakukan implementasi program.

3.1.3 Pengujian Sistem

Pengujian sistem yang dilakukan meliputi beberapa hal, antara lain:

- Melakukan pengujian semua fitur
- Melakukan pengujian terhadap data rill

3.1.4 Kesimpulan dan Saran

Melakukan penarikan kesimpulan berdasarkan hasil dari pengujian-pengujian yang telah dilakukan, serta memberikan saran-saran yang dapat dilakukan untuk pengembangan sistem lebih lanjut, dengan tujuan agar sistem yang dirancang dapat benar-benar dimanfaatkan

3.1.5 Pembuatan Laporan

Penulisan laporan hasil penelitian yang telah dilakukan.

3.1.6 Teknik Pengambilan *Sample*

Teknik pengambilan sampel dilakukan terhadap proses yang terjadi, kemudian diolah sesuai dengan ruang lingkup penelitian.

3.2 Variabel Penelitian

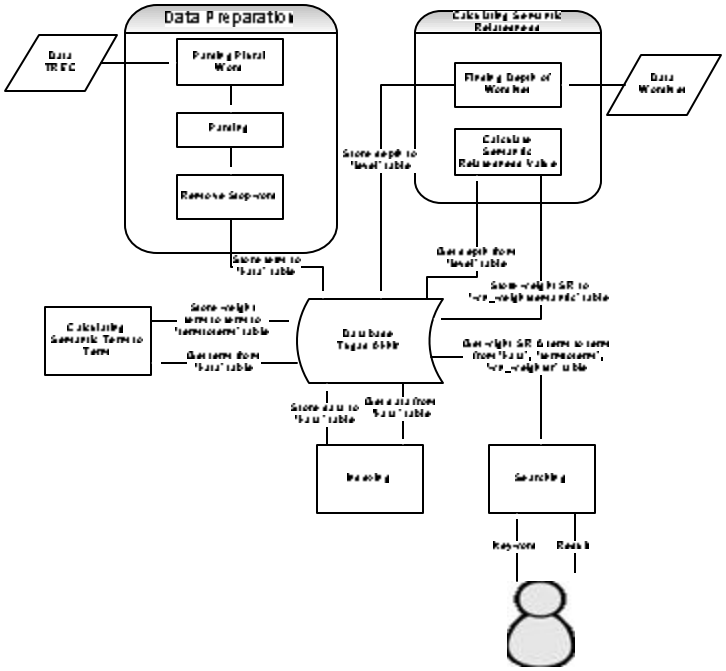
Variabel penelitian ditujukan langsung pada data dokumen yang ada dengan menggunakan *database* TREC

3.3 Metode Analisa Data

Analisa dilakukan pada perubahan jumlah dokumen yang diproses terhadap waktu proses yang dibutuhkannya.

3.4 Perencanaan Sistem

Berikut ini, merupakan block diagram dari aplikasi pencarian dokumen berbasis metode GVSM dan *Semantic Relatedness* (SR), seperti terlihat pada Gambar 3.1.



a. *Data Preparation*

Proses ini melakukan perubahan terhadap file yang dipergunakan sebagai obyek pencarian yaitu "ClueWeb09_English_Sample.warc" yang berisi kumpulan file HTML menjadi beberapa file HTML yang terpisah. Setelah selesai akan dilakukan proses merubah HTML ke teks, yang kemudian diteruskan dengan proses *parsing* pada teks tersebut.

b. *Indexing*

Proses ini melakukan perhitungan *weight* pada setiap kata yang merupakan hasil *parsing* dari proses *data preparation* dengan menggunakan metode *Term Frequency* dan *Inverse Document Frequency* (TF-IDF) yang juga terdapat pada metode *Vector Space Model* (VSM). Hasil perhitungan *weight* untuk setiap kata/*term* ini nantinya dipergunakan dalam proses *Generalized Vector Space Model* (GVSM), yang nilainya dapat berpengaruh terhadap kemunculan dokumen yang diwakili oleh kata/*term* tersebut pada hasil pencarian.

c. *Calculating Semantic Relatedness*

Proses ini melakukan perhitungan *semantic relatedness* dari kata/*term* dengan *database* "*WordNet*" yang nilainya nanti dijadikan sebagai nilai kedekatan makna antara dua kata/*term*, yang dapat meningkatkan *recall* dari hasil pencarian. Nilai kedekatan makna ini nantinya dipergunakan dalam proses *Generalized Vector Space Model* (GVSM).

d. *Calculating Semantic Term to Term*

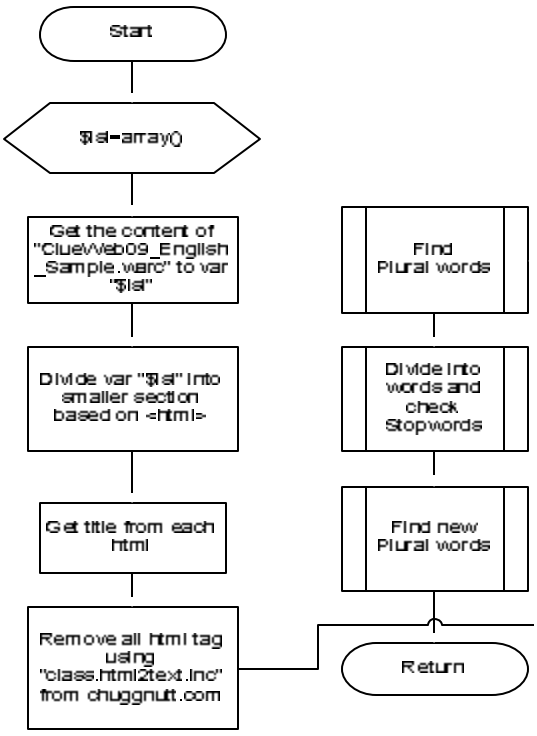
Proses ini melakukan perhitungan terhadap nilai kedekatan makna dengan menghitung jumlah kemunculan bersama antara dua *term* yang berbeda. Jumlah kemunculan tersebut nantinya dinormalisasikan dengan mambagi setiap jumlah tersebut dengan jumlah terbesar. Nilai kedekatan makna dari *semantic term to term* nantinya dipergunakan sebagai nilai kedekatan makna yang menggantikan nilai *semantic relatedness* apabila kata/*term* tersebut tidak terdapat pada *database* "*WordNet*" atau nilai *semantic relatedness* menghasilkan nilai 0.

e. *Searching*

Proses ini berguna untuk mencari dokumen yang dicari oleh *user* sesuai dengan kata kunci yang dimasukkan oleh *user*. Pada proses ini menggabungkan nilai *weight* hasil dari proses *indexing* dengan nilai kedekatan makna, baik dari *semantic relatedness* ataupun dari *semantic term to term* dengan metode *Generalized Vector Space Model* (GVSM) perhitungan cosinus, untuk melakukan perankingan terhadap hasil pencarian.

3.4.2 Alur Kerja Proses *Data Preparation*

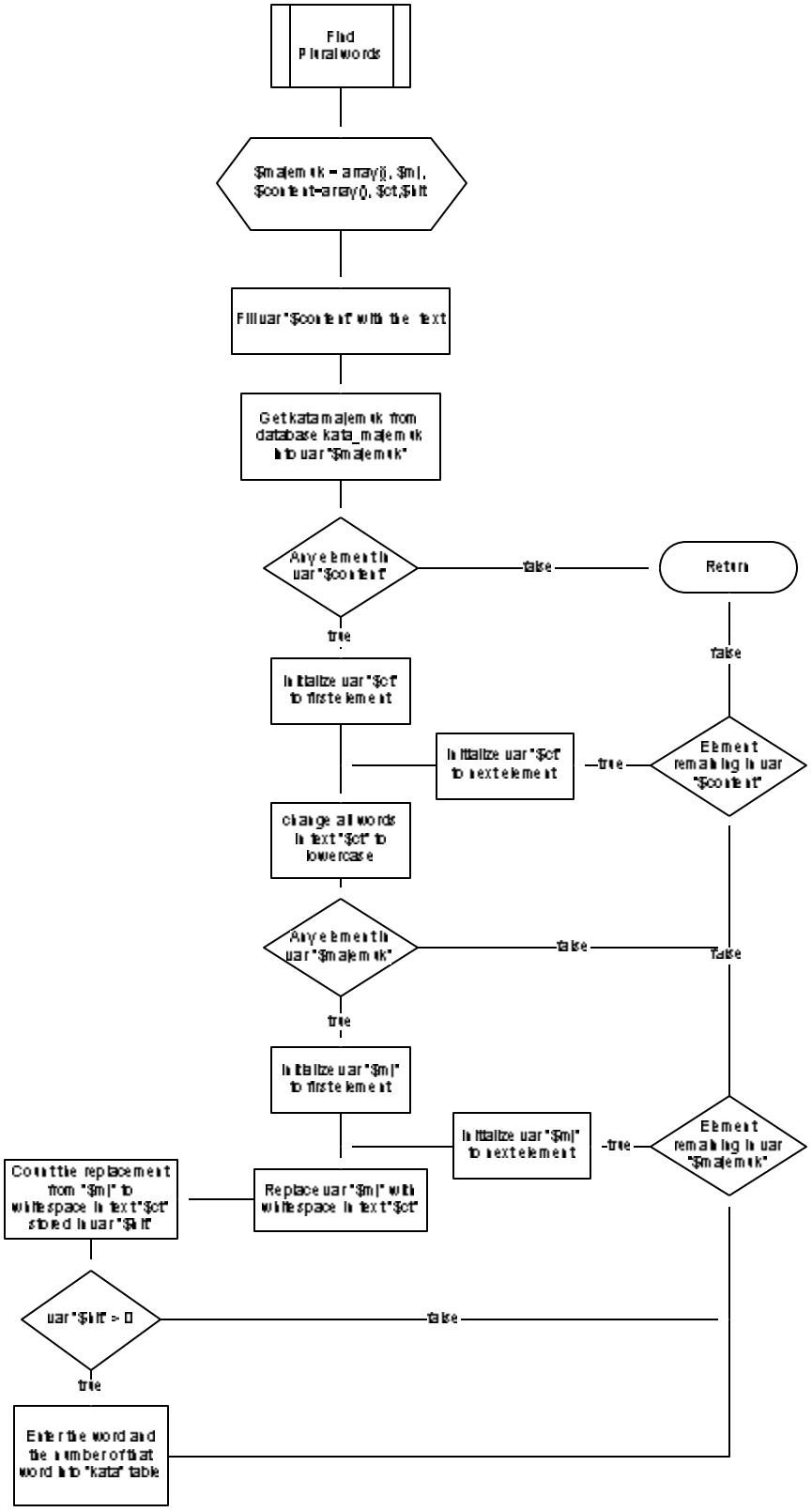
Proses *data preparation* ini adalah proses awal yang harus dilakukan untuk mempersiapkan obyek pencarian supaya dapat diproses pada *indexing*, *semantic relatedness, semantic term to term,* dan GVSM. Alur Kerja *Data Preparation* dapat dilihat pada Gambar 3.2.



b. Memisahkan setiap file HTML dan disimpan ke *database*.
c. Ambil title pada setiap file HTML dan disimpan.
d. Menghilangkan semua tag HTML dengan menggunakan *class* PHP dari chuggnutt.com yaitu "class.html2text.inc".
e. Semua file HTML yang telah diubah menjadi teks tersebut dicari kata majemuknya/ *plural word* dan jumlah kata tersebut pada setiap file kemudian dihapus kata majemuk tersebut dari file.
f. Kemudian teks yang telah dihilangkan kata majemuknya dipisahkan menjadi kata-kata yang nantinya di-*filter stopword*-nya dengan database *stopword* yang telah tersedia.
g. Yang terakhir, merupakan proses untuk mencari kata majemuk baru dari file teks yang ada, yang nantinya digunakan untuk memperbaharui *database* kata majemuk. Sebelum dimasukkan ke dalam *database* kata majemuk, administrator diberikan hak khusus untuk melakukan filter terhadap kata-kata majemuk baru yang ditemukan.

3.4.3 Alur Kerja Proses *Find Plural Words*

Proses *find plural words* merupakan proses untuk mencari kata majemuk dan jumlah kata majemuk tersebut pada setiap file. Selain itu, semua kata majemuk yang ditemukan beserta jumlahnya dimasukkan ke dalam *database*, lalu kata tersebut dihapus pada teks yang diproses pada proses selanjutnya. Alur kerja *find plural words* (Gambar 3.3) sebagai berikut :


Find
Plural words
$majemuk = array(), $mj,
$content=array(), $ct,$hit
Fill uar "$content" with the text
Get kata majemuk from
database kata_majemuk
into uar "$majemuk"
Any element in
uar "$content"
take
Return
true
take
Initialize uar "$ct"
to first element
Initialize uar "$ct"
to next element
true
Element
remaining in uar
"$content"
change all words
in text "$ct" to
lowercase
Any element in
uar "$majemuk"
take
Take
true
Initialize uar "$mj"
to first element
Initialize uar "$mj"
to next element
true
Element
remaining in uar
"$majemuk"
Count the replacement
from "$mj" to
whitespace in text "$ct"
stored in uar "$hit"
Replace uar "$mj" with
whitespace in text "$ct"
uar "$hit" > 0
take
true
Enter the word and
the number of that
word into "kata" table

BAB 4. HASIL PENELITIAN DAN PEMBAHASAN

4.1 Implementasi Aplikasi

Untuk implementasi dan pengujian digunakan dua buah laboratorium yang terkoneksi dengan jaringan komputer (Lokal Area Network) dengan spesifikasi sebagai berikut:

Spesifikasi komputer:

- *Processor* : Intel Core 2 Duo 2.6 GHz
- *Memory* : 2 GB RAM
- *Hard Disk* : 200 GB
- *Operating System* : Windows XP Service Pack 3

4.2 Pengujian Aplikasi

Pengujian dilakukan pada halaman atau menu yang ada untuk mengetahui kelayakan program untuk digunakan.

4.3 Menu Utama

Pada saat program aplikasi pertama kali dijalankan akan membuka halaman utama seperti pada Gambar 4.1. Dimana, pengguna dapat melakukan proses *query* terhadap dokumen yang diinginkan dengan memasukkan keyword.

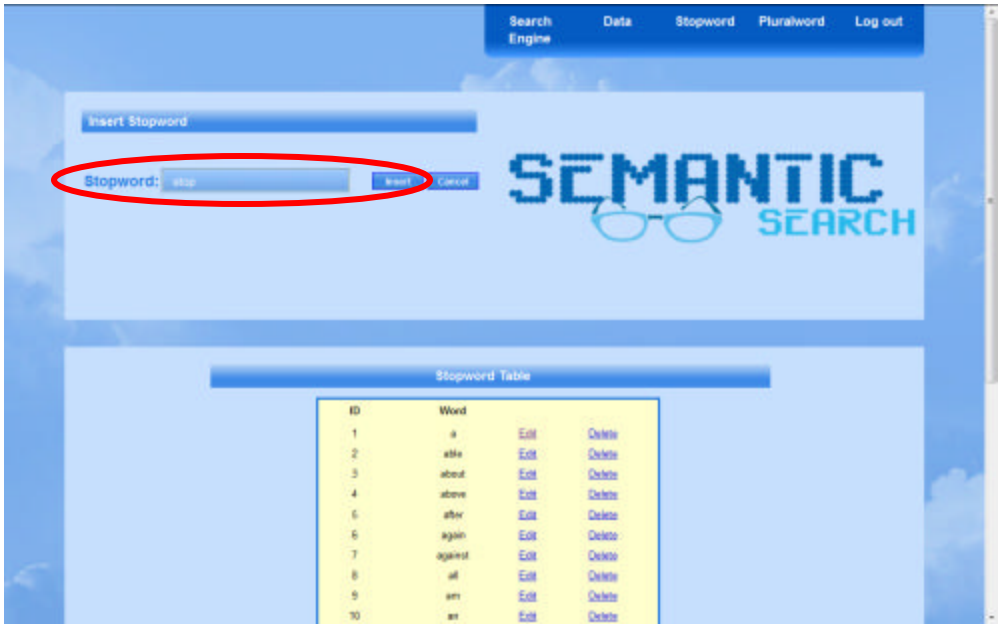
Gambar 4.1 Tampilan Halaman Utama Searching Pengguna

Setelah *user* memasukkan kata kuncinya dan menekan tombol "Search" maka sistem melakukan proses perhitungan dan kemudian menampilkan hasilnya pada halaman *searching* kedua seperti pada gambar 4.2.

Gambar 4.2 Tampilan Hasil *Searching User*

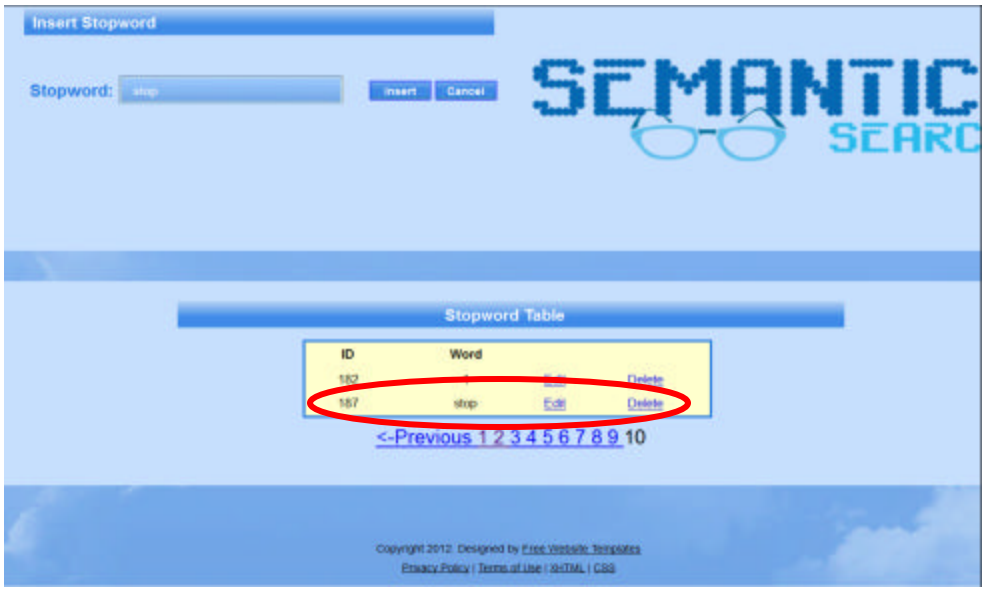
Pada halaman *administrator* terdapat fitur tambahan selain fitur *searching*, yaitu fitur *insert*, *update* dan *delete* untuk *database stopword* dan *database* kata majemuk dan *view database files*.

Pada halaman *database stopword* ini, administrator dapat melakukan *insert*, *update* dan *delete* pada *database stopword*. *Database stopword* ini berguna pada proses *data preparation*, yang dipergunakan untuk menghapus kata/*term* yang tidak signifikan dalam membedakan dokumen atau *query*. Dengan adanya halaman ini administrator dapat mengontrol *stopword* yang ada, karena *stopword* sangat berpengaruh pada perhitungan dan hasil pencarian. Pada gambar 4.3. ditampilkan halaman untuk melakukan *insert*, *update* dan *delete* pada *database stopword*.



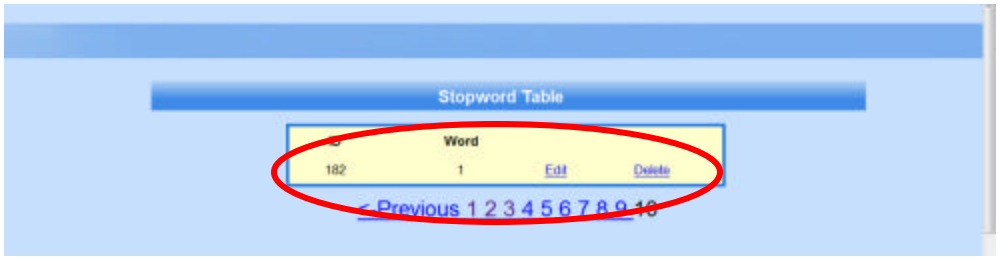
Gambar 4.3 Tampilan Halaman Insert Dan Delete Database Stopword

Pada halaman di atas, administrator dapat memasukkan kata *stopword* baru dan men-*delete stopword* yang ada. Untuk melakukan *insert stopword*, admin hanya perlu memasukkan kata yang akan dijadikan *stopword* ke *textbox* yang tersedia lalu menekan tombol "Insert", maka kata tersebut akan langsung masuk ke *database stopword* (Gambar 4.4).



Gambar 4.4 Tampilan kata *stopword* hasil *insert* masuk ke *database*

Untuk men-*delete* *stopword* yang sudah ada, maka hanya tinggal menekan "Delete" yang ada pada bagian kanan kata yang akan dihapus, seperti pada Gambar 4.5.



Gambar 4.5 Hasil delete database stopword

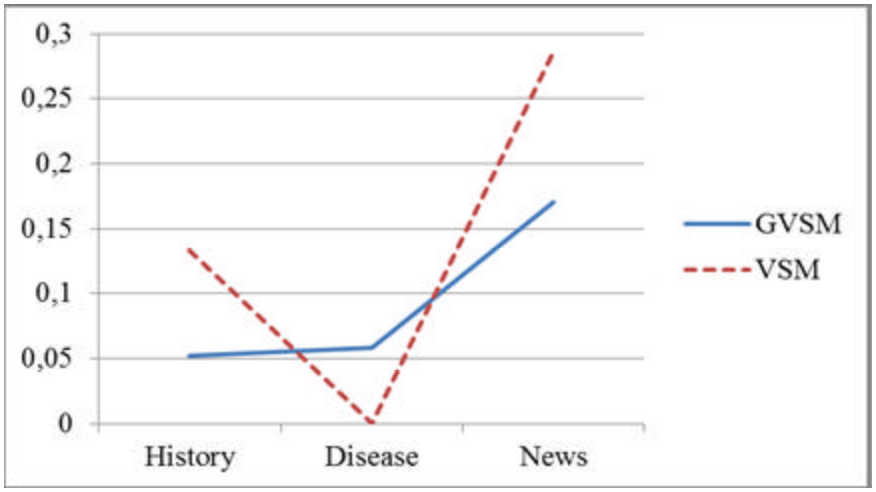
Sementara untuk melakukan *edit stopword* yang sudah ada, maka hanya tinggal menekan "Edit" yang ada pada bagian kanan kata yang akan di-*edit*, lalu sistem akan menampilkan halaman *edit stopword* seperti pada Gambar 4.6. Lalu mengisi ulang *textbox* yang tersedia, lalu menekan tombol "Update" dan kata yang lama akan diganti dengan yang baru di *database stopword*.

Gambar 4.6 Tampilan halaman edit dan delete database stopword

Pada halaman ini administrator dapat melihat data yang menjadi obyek pencarian atau isi dari setiap file dari obyek pencarian. Gambar 4. 7. merupakan tampilan halaman administrator untuk melihat obyek pencarian.

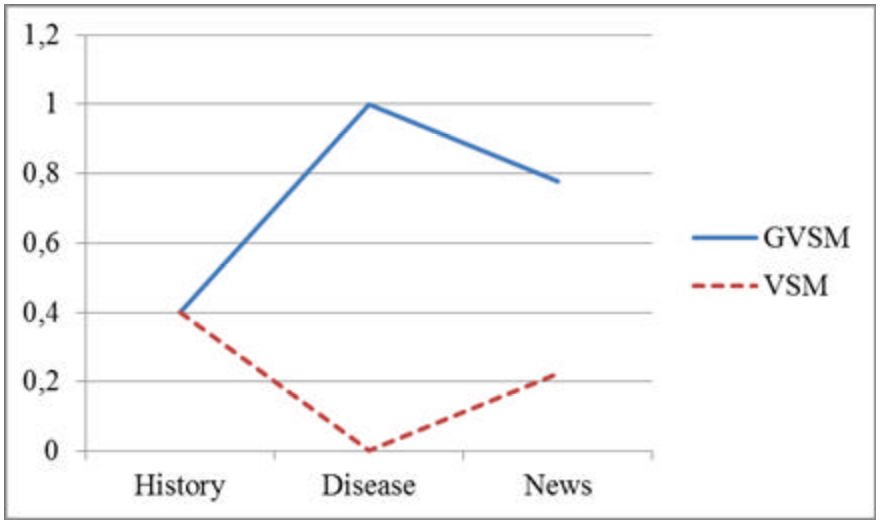
Gambar 4.7 Tampilan halaman *view* data obyek pencarian

Perbandingan *precision* dan *recall* dari GVSM dan VSM dapat digambarkan dengan grafik seperti pada gambar 4.8. dan 4.9.



Gambar 4.8 Grafik perbandingan nilai *Precision* antara GVSM dan VSM

Dapat dilihat pada Gambar 4.8. bahwa GVSM memiliki nilai *precision* yang lebih kecil jika dibandingkan dengan VSM. Hanya pada kata kunci "Disease" saja yang nilai *precision* GVSM-nya lebih tinggi jika dibanding dengan nilai *precision* VSM, dikarenakan tidak ditemukan sama sekali dokumen yang relevan pada hasil pencarian VSM. Dapat dilihat pada gambar 4.9 bahwa GVSM memiliki nilai *recall* yang selalu lebih besar atau sama jika dibandingkan dengan VSM.



Gambar 4.9 Grafik perbandingan nilai *Recall* antara GVSM dan VSM

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan dan pembuatan sistem perhitungan harga pokok produksi ini, dapat diambil beberapa kesimpulan antara lain:

1. Dengan melakukan perbandingan antara *Generalized Vector Space Model* (GVSM) dan *Vector Space Model* (VSM), maka dapat dilihat bahwa *Generalized Vector Space Model* dapat membantu dalam meningkatkan *recall*. Nilai *recall* yang dihasilkan oleh GVSM adalah 0,4 ; 1 ; 0,7778 , sedangkan nilai *recall* yang dihasilkan oleh VSM adalah 0,4 ; 0 ; 0,2222 . Peningkatan *recall* terjadi karena *Generalized Vector Space Model* tidak hanya menampilkan dokumen yang mengandung *keyword* yang dimasukkan *user* saja, tetapi juga menampilkan dokumen yang mengandung *keyword* lain yang memiliki *similiarity* makna dengan *keyword user*.
2. Kelemahan dari *Generalized Vector Space Model* adalah kecilnya *precision* dari hasil pencarian jika dibandingkan dengan *Vector Space Model.* Nilai *precision* yang dihasilkan oleh GVSM adalah 0,0526 ; 0,0588 ; 0,1707 , sedangkan nilai *precision* yang dihasilkan oleh VSM adalah 0,1333 ; 0 ; 0,2857 . Tetapi hal ini dapat diminimalisasi dengan penampilan hasil pencarian yang didasarkan pada urutan *similiarity* antara *keyword* dengan obyek pencarian.
3. Berdasarkan pegujian lama waktu pencarian nilai SR, dapat dilihat bahwa rata-rata waktu proses terus meningkat secara linear terhadap jumlah hasil pencarian. Jadi semakin banyak hasil pencarian yang dibutuhkan, maka semakin banyak pula rata-rata waktu untuk melakukan proses tersebut, sehingga semakin banyak waktu yang dibutuhkan untuk melakukan proses untuk mendapatkan hasil pencarian nilai SR tersebut.
4. Berdasarkan pembandingan waktu *searching* antara *Generalized Vector Space Model* (GVSM) dan *Vector Space Model* (VSM), maka dapat dilihat bahwa lama proses *searching* dengan GVSM jauh lebih lama jika dibandingkan dengan lama proses *searching* dengan VSM. Dikarenakan

proses *searching* dengan GVSM membutuhkan waktu untuk pencarian kedekatan makna antar *term*.

5.2 Saran

Setelah melakukan evaluasi terhadap sistem secara keseluruhan, diharapkan penelitian ini dapat dikembangkan lebih lanjut dengan saran-saran pengembangan sebagai berikut:

1. Diperlukan metode pendamping lain untuk mengatasi kelemahan VSM dalam hal penghitung relasi antar kata satu per satu, karena hal tersebut memakan waktu yang cukup lama.

DAFTAR PUSTAKA

Dik, L. LEE., Huei, C. & Kent E. S. (1997). *Document ranking and the vector-Space Model*, 67-75.

Kristopher David Harjono (2005). Perluasan Vektor pada Metode Search Vector Space. Integral, Vol. 10 No. 2, Juli 2005.

Miller, G. A. (1995). *WordNet : A Lexical Database for English.*

Ning Liu et al. (2004). Learning Similarity Measures in Non-orthogonal Space. CIKM'04, November 813, 2004, Washington D.C., U.S.A.

Overview of TREC. Retrieved Maret 15, 2012 from http://trec.nist.gov/overview.html.

The Classic Vector Space Model. Retrieved Maret 15, 2012 from http://www.miislita.com/term-vector/term-vector-3.html.

Tsatsaronis, G. & Panagiotopoulou V. (2009). *A Generalized Vector Space Model for Text Retrieval Based on Semantic Relatedness*. *The EACL 2009 Student Research Workshop*, 70–78.

Turney, P.D. & Pantel, P. (2010). *From Frequency to Meaning: Vector Space Models of Semantics. Journal of Artificial Intelligence Research,* 37 (2010) 141-188.

What is WordNet. Retrieved Maret 15, 2012 from http://wordnet.princeton.edu/wordnet/

Zafikri A. (2010). Implementasi Metode Term Frequency Inverse Document Frequency (TF-IDF) pada Sistem Temu Kembali Informasi.

Anggaran Kegiatan Penelitian

No	Uraian	Penggunaan	Jumlah	Harga Satuan (Rp)	Total (Rp)
1	CD	Back up aplikasi, laporan, dan user manual	1 CD	25.000	25.000
2	Tinta printer	Cetak laporan	1 buah	175.000	175.000
3	Kertas	Cetak laporan	1 rim	35.000	35.000
4	Fotocopy dan penjilidan	Penggandaan laporan	4 eks	35.000	140.000
5	Biaya Internet	Koneksi internet untuk uji coba	1 pax	100.000	100.000
Total					475.000