SISTEM TEMU-KEMBALI INFORMASI DALAM DOKUMEN MENGGUNAKAN METODE LATENT SEMANTIC INDEXING

Muhammad[1], Sukmawati Nur Endah[2], dan Beta Noranita[3]

Program Studi Teknik Informatika Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Diponegoro
[1]muhammad.if@student.undip.ac.id, [2]sukma_ne@undip.ac.id, [3]betta@undip.ac.id

Abstract

Textual documents have been largely available digitally or electronically with increasing number since the invention of computer. The need of information makes a user to do a searching activity on document collections he or she has. Document searching method available on computers usually just look into the name of document (file) searched by user without noticing the content inside it, therefore it can not satisfy his or her information need. Latent Semantic Indexing (LSI) method can be a solution to retrieve relevant information from document collections inside a computer. LSI method has an ability to correlate semantically related terms that are latent in a collection of text using Singular Value Decomposition (SVD) technique. An information retrieval system using LSI method has been developed and tested on 40 Indonesian documents with 478 words per document in average. On testing process which has been done, with threshold of 0.6 the system could give search result with average precision score of 55.48% and average recall score of 96.67% in testing process.

Key words: information retrieval, latent semantic indexing, singular value decomposition, precision, recall.

1. Pendahuluan

Kebutuhan akan suatu informasi menuntut seseorang untuk menelusuri seluruh koleksi dokumen yang dimilikinya. Banyaknya dokumen yang tersimpan di dalam komputer akan membuat pengguna mengalami kesulitan untuk memperoleh informasi yang dibutuhkan. Pengguna tidak dapat melihat isi dokumen satu persatu untuk menda-patkan informasi yang tepat. Untuk mendapatkan informasi, pengguna biasa menggunakan fasilitas pencarian dokumen yang disediakan oleh sistem operasi dengan memasukkan kata kunci. Kata kunci yang dimasukkan pengguna kemudian menjadi kueri untuk program pencarian dokumen pada komputer tersebut. Pencarian dokumen di dalam komputer biasanya hanya melihat pada nama dokumen yang dicari tanpa memperhatikan isi yang terkandung di dalamnya sehingga seringkali tidak dapat memenuhi kebutuhan informasi pengguna.

Fasilitas pencarian menggunakan pencocokan kata kunci dengan kata di dalam dokumen yang juga disediakan oleh sistem operasi masih belum dapat memberikan hasil pencarian yang relevan. Metode pencocokan kata akan menghitung jumlah kata kunci yang muncul di dalam dokumen kemudian mengembalikan urutan dokumen dengan jumlah kemunculan terbanyak kepada pengguna. Karena pengguna bukan mencari dokumen yang mengandung banyak kata yang sama dengan kata kunci, metode pencocokan kata bukan merupakan solusi yang ideal untuk pencarian informasi. Selain itu, banyaknya kata yang memiliki kesamaan arti (sinonim) dan kata yang memiliki arti lebih dari satu (polisemi) dalam penggunaan kata kunci untuk mengekspresikan informasi yang dibutuhkan, akan membuat hasil pencarian dengan metode pencocokan kata semakin jauh dari relevan.

Metode Latent Semantic Indexing (LSI, kadang disebut dengan Latent Semantic Analysis) telah mencoba mengatasi masalah

pencocokan teks dalam bidang temu-kembali informasi (*information retrieval*). Metode ini dikembangkan pada tahun 1988 dan telah dipatenkan (U.S. Patent no. 4,839,853) oleh Deerwester, Susan Dumais, George Furnas, Richard Harshman, Thomas Landauer, Karen Lochbaum, dan Lynn Streeter. Metode ini dinamakan LSI karena memiliki kemampuan untuk menemukan hubungan tersembunyi (*latent*) antara semua *terms* (kata) yang memiliki kedekatan makna secara kontekstual [1]. Metode LSI memberikan solusi untuk masalah kata-kata sinonim dan polisemi yang sering terjadi dalam temu-kembali informasi.

2. Latent Semantic Indexing

Latent Semantic Indexing (LSI) adalah sebuah teknik berbasis bidang vektor (*vector space*) yang dapat menciptakan asosiasi antara dokumen-dokumen yang berhubungan secara konseptual [2]. Pada model bidang vektor, sekumpulan dokumen diindeks berdasarkan *terms* dan direpresentasikan dalam bentuk *m x n term-document matrix A* yang dinotasikan sebagai,

$$A = [a_{ij}] \quad \text{..... (1)}$$

Elemen-elemen di dalam matriks *A* merupakan bobot frekuensi *term i* yang muncul pada dokumen *j*. Kolom-kolom pada matriks *A* merepresentasikan *n* vektor dokumen dan baris-barisnya merepresentasikan *m* vektor *terms*. Fungsi temu-kembali mem-bandingkan vektor kueri *q* dengan tiap kolom yang merepresentasikan dokumen tertentu dalam bidang vektor. Sebuah derajat kesamaan antara vektor kueri dan semua dokumen yang direpresentasikan dalam bidang vektor kemudian diukur untuk perangkingan [3]. Contoh matriks *term-document* diperlihatkan pada gambar 1.

	C1	C2	C3	C4	C5	M1	M2	M3	M4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
eps	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Gambar 1. Contoh Matriks *Term-Document* [4]

LSI merupakan sebuah metode *automatic indexing* dan *retrieval* dengan memanfaatkan *semantic structure* (struktur asosiasi *terms* dengan dokumen) yang secara implisit terdapat dalam suatu dokumen untuk digunakan dalam pencarian dokumen yang relevan dengan terms dalam kueri [4]. Metode LSI mengasumsikan bahwa terdapat sebuah latent semantic structure, yaitu sebuah struktur semantik yang tersembuyi (latent) dalam setiap dokumen, yang kabur atau tidak jelas karena keberagaman pemakaian kata dalam penulisan dokumen tersebut (dikenal dengan istilah noise). LSI menggunakan teknik-teknik statistik untuk mendapatkan latent structure dan menghilangkan noise yang ada. Sebuah penggambaran atau deskripsi dari terms dan dokumen-dokumen berdasarkan latent semantic structure tersebut digunakan untuk proses pengindeksan (indexing) dan pencarian kembali (retrieval).

Dekomposisi nilai singular (singular value decom-*position*) atau biasa dikenal dengan SVD digunakan dalam LSI untuk melakukan dekomposisi terhadap matriks *term-document* menjadi tiga buah matriks. SVD dari suatu matriks *A* didefinisikan sebagai,

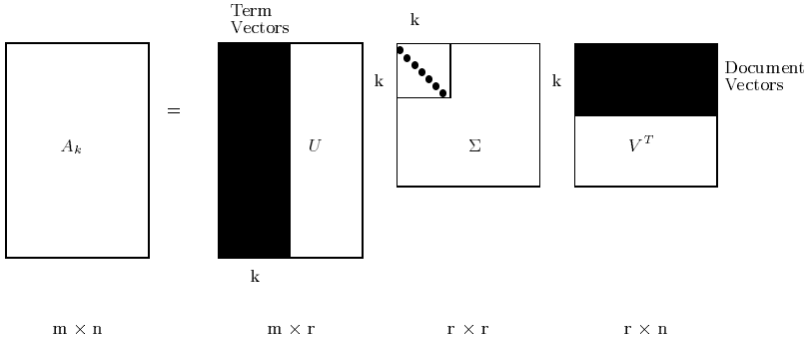
$$A = USV^T \quad \text{..... (2)}$$

Matriks *U* merupakan matriks yang berisi vektor eigen dari matriks AA^T (matriks vektor-vektor singular kiri), matriks *V* merupakan matriks yang berisi vektor eigen dari A^TA (matriks vektor-vektor singular kanan), dan akar kuadrat dari nilai eigen AA^T atau A^TA mengisi nilai singular matriks *S* [4]. Matriks *U* dan *V* dianggap sebagai matriks dari vektor-vektor *term* dan dokumen secara berturut-turut [5].

Proses penghitungan SVD ini akan menghasilkan perkiraan *low-rank k* terbaik yang dapat mengura-ngi *noise* dan menyingkapkan struktur nyata dari data yang digunakan [5, 6]. Proses menentukan *low-rank k* terbaik (*rank lowering*) dilakukan dengan cara menyimpan sebanyak *k* nilai singular pertama (terbesar) dan sisanya (nilai-nilai singular yang lebih kecil) diubah menjadi nol [4, 6]. Dengan melakukan *rank-lowering*, dimensi dari matriks singular *S* akan tereduksi, begitu juga dengan matriks singular kiri *U* dan matriks singular kanan *V*, sehingga persamaan SVD dituliskan sebagai,

$$A_k = U_k S_k V_k^T \quad \text{..... (3)}$$

U_k merupakan matriks *m x k* yang berisi *k* kolom pertama dari *U*, V_k merupakan matriks *n x k* yang berisi *k* kolom pertama dari *V*, dan S_k merupakan matriks diagonal *k x k* yang berisi *k* nilai singular terbesar [1, 7, 8]. Hasil dekomposisi matriks diperlihatkan pada gambar 2. Bagian dari matriks yang diarsir tebal adalah *rank-k approximation model* dari matriks *A*, dan *k* adalah jumlah nilai singular yang tidak diabaikan.



Gambar 2. Hasil Dekomposisi Matriks

Sumber: [5]

Low-rank (*reduction*) *model* ini kemudian digunakan sebagai *vector space model* dalam LSI untuk menentukan letak *terms* atau dokumen. Pada matriks *V*, untuk setiap *n* dokumen, matriks ini berisikan *n* baris yang yang merupakan vektor eigen. Maka setiap baris yang ada menyimpan koordinat dari tiap dokumen dalam model vektor.

Kueri (kata kunci) yang dimasukkan pengguna harus direpresentasikan dalam bentuk vektor untuk dibandingkan dengan vektor dokumen dalam proses pencarian [5]. Dalam model LSI, kueri yang dimasukkan dibentuk menjadi *pseudo-documents* yang akan menentukan lokasi kueri tersebut dalam bidang matriks *term-document* yang telah tereduksi [4, 9]. Vektor kueri $\hat{q}$ dari *pseudo-document q* dapat dituliskan sebagai,

$$\hat{q} = q^T U_k S_k^{-1} \quad \text{..... (4)}$$

Vektor kueri yang dihasilkan kemudian dibandingkan dengan seluruh vektor dokumen

dalam matriks V^T hasil dari dekomposisi matriks *term-document* dengan sebuah ukuran kesamaan (*similarity*). Sebuah ukuran kesamaan yang umum digunakan adalah *cosine similarity* yang mengukur sudut antara vektor kueri dengan vektor dokumen [5, 9]. *Cosine similarity* antara duah buah vektor A dan B dapat dituliskan,

$$similarity = \cos\theta = \frac{A \bullet B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2}\sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad (5)$$

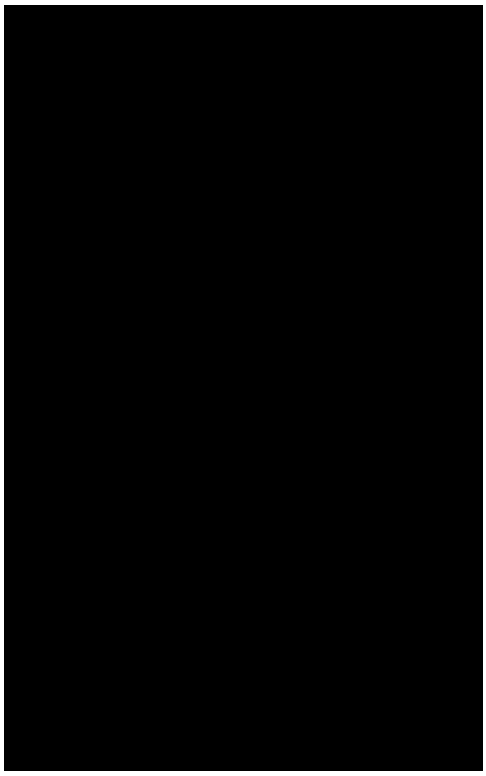
Dokumen-dokumen yang diperbandingkan dengan vektor kueri kemudian diurutkan dalam bentuk rangking berdasarkan nilai kesamaannya. Nilai kesamaan akan berada antara -1 hingga 1. Semakin nilai kesamaan mendekati 1 menunjukkan bahwa dokumen semakin cocok atau mirip dengan kueri.

3. Implementasi

Sistem temu-kembali informasi dalam dokumen menggunakan metode LSI dikembangkan berbasis desktop menggunakan bahasa pemrograman Java. Proses yang berjalan di dalam sistem ini dibagi menjadi proses pengindeksan dan proses pencarian seperti terlihat pada gambar 3 (gambaran umum sistem). Proses pengindeksan merupakan proses untuk membuat indeks dan representasi dokumen-dokumen di dalam komputer menjadi bentuk vektor-vektor dokumen. Indeks merupakan kumpulan kata-kata atau konsep yang telah dipilih dan digunakan sebagai petunjuk menuju informasi atau dokumen terkait [10].

Proses pengindeksan mencakup operasi teks, pembobotan, pembentukan matriks term-document, dan penghitungan SVD (Singular Value Decompo-sition). Operasi teks akan mengubah teks di dalam dokumen menjadi potongan-potongan kata dasar atau akar katanya (*stemming*) dalam huruf kecil dan menghapus kata sambung dan kata-kata lainnya yang tidak digunakan dalam temu-kembali informa-si (dikenal dengan *stop words*).

Pembobotan merupakan metode yang umum dan efektif dalam meningkatkan performa pencarian kembali dalam model vektor yang dilakukan dengan mentransformasi frekuensi kasar kemunculan suatu *term* di dalam dokumen menggunakan suatu fungsi [5].



Gambar 3. Gambaran Umum Sistem

Transformasi semacam itu memiliki dua komponen, *global weighting* dan *local weighting*. Setiap *term* diberikan sebuah *global weight* (bobot global) yang mengindikasikan tingkat pentingnya *term* tersebut di dalam keseluruhan koleksi dokumen, dan *local weight* (bobot lokal) untuk menunjukkan tingkat pentingnya *term i* yang muncul dalam dokumen *j*. Bobot dari matriks *A* dapat dituliskan sebagai,

$$a_{ij} = L(i,j) \times G(i) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (6)$$

$L(i,j)$ adalah fungsi bobot lokal untuk *term i* di dalam dokumen *j*, dan $G(i)$ adalah fungsi bobot global untuk *term i*. Bobot lokal dihitung dengan menggunakan *tf* (*term frequency*)

dengan norma-lisasi yang didefinisikan sebagai,

$$L(i,j) = tf_{ij} = \frac{n_{ij}}{m_j} \quad \text{..... (7)}$$

n_{ij} merupakan jumlah kemunculan dari *term i* di dalam dokumen *j*, dan m_j merupakan jumlah seluruh *terms* di dalam dokumen *j* [8]. Sedangkan bobot global dihitung dengan menggunakan *idf* yang didefinisikan sebagai [10],

$$G(i) = idf_i = \log\frac{N}{df_i} \quad \text{..... (8)}$$

N melambangkan jumlah total dari dokumen yang terdapat di dalam koleksi, dan df_i merupakan jumlah dokumen-dokumen di dalam koleksi yang mengandung *term i*.

Setelah bobot seluruh kata dihitung, matriks *term-document* dibentuk. Pembentukan matriks *term-document* digunakan untuk menyatakan hubungan antara *term*s dengan dokumen-dokumen dalam sistem temu kembali informasi [1]. Sebagaimana terlihat pada gambar 1, kolom pada matriks term-document mewakili dokumen, dan barisnya mewakili terms. Matriks yang dihasilkan kemudian digunakan dalam proses SVD. Akhir dari proses pengindeksan adalah sebuah vektor dokumen yang digunakan dalam proses pencarian.

Proses pencarian merupakan proses pencarian dokumen yang relevan terhadap kueri yang dimasukkan pengguna berupa suatu kata kunci. Dalam proses pencarian, operasi teks yang sama seperti dalam proses pengindeksan juga dilakukan terhadap kata kunci yang dimasukkan. Sesudah operasi teks dilakukan, operasi kueri kemudian memproses kata kunci menjadi *pseudo-document* dan menghitung vektor kueri. Vektor kueri yang dihasilkan dari operasi kueri kemudian digunakan untuk menghitung nilai kesamaan (*cosine similarity*) dengan vektor dokumen yang dihasilkan dari proses pengindeksan. Proses penghitungan nilai kesamaan ini menghasilkan nilai kesamaan tiap dokumen terhadap kata kunci.

Setelah nilai kesamaan dokumen terhadap kata kunci dihasilkan, dilakukan pengurutan nilai tersebut dengan memilih dokumen yang nilainya telah melebihi batas ambang (*threshold*) tertentu kemudian menampilkannya kepada pengguna dalam bentuk rangking yang diurutkan dari nilai yang terbesar hingga terkecil (*descending*).

4. Pengujian

Pengujian untuk sebuah sistem temu-kembali informasi dilakukan menggunakan *precision* dan *recall*. *Precision* mengukur kemampuan sistem untuk mengembalikan hanya dokumen-dokumen yang relevan, sedangkan *recall* mengukur kemam-puan sistem untuk mengembalikan semua dokumen yang relevan. Nilai *precision* (*P*) didefinisikan sebagai,

$$P = \frac{D_r}{D_t} \quad \text{..... (9)}$$

Sedangkan *recall* (*R*) didefinisikan dengan,

$$R = \frac{D_r}{N_r} \quad \text{..... (10)}$$

D_r merupakan jumlah dari dokumen relevan yang diperoleh, D_t adalah jumlah total dari dokumen yang diperoleh, dan N_r adalah jumlah total dari dokumen yang relevan di dalam koleksi dokumen.

Pengujian dilakukan dengan 40 dokumen berbahasa Indonesia. Sebelum dilakukan uji pencarian, terlebih dahulu dilakukan proses pengindeksan dokumen-dokumen mulai dari operasi teks, pembobotan, SVD, hingga vektor dokumen dihasil-kan. Proses pengindeksan berlangsung selama 135 menit 32 detik. Rata-rata jumlah kata dalam tiap dokumen setelah operasi teks adalah 478 kata dengan total seluruh kata yang berbeda (unik) adalah sebanyak 3888 kata sehingga ukuran matriks *term-document* yang terbentuk adalah 3888 x 40. Matriks tersebut kemudian digunakan untuk proses SVD dan penghitungan vektor dokumen.

Uji pencarian sistem temu-kembali informasi ini menggunakan 10 kueri (kata kunci). Tiap-tiap kueri memiliki kumpulan dokumen yang relevan terhadap kueri tersebut (N_r). Batas ambang skor dokumen yang digunakan dalam pengujian adalah 0.9, 0.8, 0.7, 0.6, dan 0.5. Hasil uji pencarian dengan kelima batas ambang tersebut kemudian digunakan untuk mencari batas ambang yang optimal untuk tiap kueri (batas ambang tertinggi yang paling banyak memperoleh dokumen-dokumen yang relevan). Proses pencarian dokumen membutuhkan waktu antara 9 sampai 10 detik untuk setiap kueri yang dimasukkan.

Himpunan batas ambang yang optimal untuk masing-masing kueri (10 kueri) diperoleh {0.9, 0.7, 0.9, 0.9, 0.8, 0.9, 0.9, 0.9, 0.6, 0.9}. Batas ambang yang optimal untuk kueri secara umum ditentukan dengan dua cara yaitu (1) dengan mengambil nilai batas ambang optimal yang paling rendah dari himpunan batas ambang optimal tiap kueri dan (2) dengan mengambil nilai rata-rata dari batas ambang optimal tiap kueri. Dengan cara pertama diperoleh batas ambang 0.6. Sedangkan dengan cara kedua, nilai batas ambang yang digunakan adalah 0.84. Tabel 1 memperlihatkan hasil uji pencarian menggunakan kedua batas ambang optimal yang telah ditentukan (0.6 dan 0.84). Nilai rata-rata *precision* dan *recall* untuk kedua batas ambang dapat dilihat pada tabel 2.

Tabel 1. Hasil Uji Pencarian Menggunakan Batas Ambang 0.6 dan 0.84

No.	Kueri	Batas Ambang	D_r	D_t	N_r	*P*	*R*
1.	Pertanian	0.60	4	16	4	0.25	1
		0.84	4	10	4	0.4	1
2.	Keuangan Negara	0.60	5	13	5	0.385	1
		0.84	3	9	5	0.333	0.6
3.	Statistika	0.60	2	2	2	1	1
		0.84	2	2	2	1	1
4.	Matriks dan vektor	0.60	2	2	3	1	0.667
		0.84	2	2	3	1	0.667
5.	Sistem temu-kembali informasi	0.60	6	22	6	0.273	1
		0.84	6	7	6	0.857	1
6.	Jenis-jenis ikan	0.60	5	5	5	1	1
		0.84	5	5	5	1	1
7.	Filsafat	0.60	4	20	4	0.2	1
		0.84	4	14	4	0.286	1
8.	Efek globalisasi	0.60	4	21	4	0.19	1
		0.84	4	12	4	0.333	1
9.	Sistem operasi windows	0.60	4	4	4	1	1
		0.84	2	2	4	1	0.5
10.	Perdagangan anak	0.60	3	12	3	0.25	1
		0.84	3	9	3	0.333	1

Keterangan:
D_r = Jumlah dokumen relevan yang diperoleh
D_t = Jumlah dokumen yang diperoleh
N_r = Jumlah dokumen relevan dalan koleksi dokumen
P = *Precision*
R = *Recall*

Tabel 2. Rata-rata Nilai *Precision* dan *Recall* untuk Batas Ambang 0.6 dan 0.84

No.	Batas Ambang	Rata-rata *Precision*	Rata-rata *Recall*
1.	0.60	0.5548	0.9667
2.	0.84	0.6542	0.8767

5. Analisis Hasil Pengujian

Hasil pengujian menunjukkan bahwa batas ambang yang optimal untuk suatu kueri belum tentu optimal untuk kueri yang lain. Oleh karena itu, dilakukan percobaan agar didapatkan nilai batas ambang yang optimal untuk kueri secara umum dengan dua cara yang berbeda. Setelah dilakukan percobaan, batas ambang 0.6 memiliki rata-rata nilai *recall* yang lebih tinggi untuk seluruh kueri (0.9667) dibandingkan dengan rata-rata nilai *recall* pada batas ambang 0.84 (0.8767). Hal tersebut menunjukkan bahwa dengan batas ambang 0.6 sistem temu-kembali informasi mampu mempero-leh 96.67% dokumen relevan dari seluruh dokumen relevan dalam kumpulan dokumen yang digunakan. Sedangkan dengan batas ambang 0.84 dokumen relevan yang dapat dikembalikan oleh sistem adalah sebesar 87.67%.

Sebaliknya, batas ambang 0.84 memiliki rata-rata nilai *precision* yang lebih tinggi (0.6542) diban-dingkan rata-rata nilai precision pada batas ambang 0.6 (0.5548). Hal tersebut menunjukkan bahwa dengan batas ambang 0.84 kemampuan sistem dalam mengembalikan hanya dokumen yang relevan adalah sebesar 65.42% yang berarti perban-dingan dokumen relevan yang diperoleh dengan seluruh dokumen yang diperoleh lebih tinggi atau ketat dibandingkan dengan batas ambang 0.6 sebesar 55.48%.

Hasil pencarian menggunakan batas ambang 0.6 lebih komprehensif karena meskipun dokumen-dokumen lain yang kurang relevan juga akan ikut muncul, pengguna tetap dapat memperoleh dokumen yang sesuai dengan kueri yang dimasukkan. Dokumen-dokumen yang relevan dengan kueri (dengan nilai kesamaan 0.83, misalnya) dapat tidak diperoleh karena berada di bawah batas ambang jika batas ambang 0.84 digunakan.

6. Kesimpulan

Setelah dilakukan penelitian pada sistem temu-kembali informasi dalam dokumen menggunakan metode LSI, maka dapat disimpulkan beberapa hal sebagai berikut:

1. Berdasarkan pengujian yang dilakukan, sistem temu-kembali informasi menggunakan metode LSI dapat memberikan hasil pencarian yang relevan dengan nilai *recall* 96.67% dan precision 55.48% pada batas ambang 0.6.
2. Operasi teks sangat mempengaruhi ukuran matriks dan basis data yang digunakan oleh sistem. Jumlah total seluruh kata yang berbeda (unik) yang tersimpan dalam basis data dan matriks *term-documet* setelah operasi teks adalah sebanyak 3888 kata (dalam bentuk kata dasar, tanpa *stopwords*) pada pengujian sistem. Jumlah tersebut menunjukkan bahwa operasi teks menggunakan bahasa yang sesuai dengan bahasa dokumen akan menghemat penggunaan basis data dan memperkecil ukuran matriks, sehingga memori yang digunakan untuk penghitungan matriks menjadi lebih efisien.
3. Proses pengindeksan yang dilakukan dalam pengujian sistem memakan waktu yang relatif lama yaitu 135 menit 32 detik untuk jumlah dokumen sebanyak 40 dengan rata-rata jumlah kata hasil operasi teks adalah 478 kata di setiap dokumen. Akan tetapi setelah pengindeksan dokumen selesai dilakukan, proses pencarian dapat dilakukan dalam waktu yang relatif singkat.

7. Saran

Beberapa hal yang dapat dijadikan saran untuk penelitian dan pengembangan sistem

temu-kembali informasi lebih lanjut adalah sebagai berikut:

1. Metode LSI menggunakan SVD (*Singular Value Decomposition*) memiliki kekurangan yaitu membutuhkan memori untuk penyimpan-an matriks yang lebih besar dibandingkan model bidang vektor biasa dan waktu yang lama dalam penghitungannya. Diperlukan alternatif untuk mengatasinya. Alternatif lain yang dapat digunakan diantaranya adalah SDD (*Semi Dis-crete Decomposition*) dan *Eigenvalue Analysis*.
2. Setiap kali terdapat perubahan (penambahan / pengurangan dokumen ataupun teksnya) dalam kumpulan dokumen pada lokasi (direktori) yang sama, perlu dilakukan pengindeksan dan penghitungan ulang dari awal untuk mendapat-kan nilai SVD yang baru. Algoritma untuk memperbaharui (*update*) SVD diperlukan untuk menghemat waktu dan memori daripada meng-hitung keseluruhan nilai SVD dari awal.
3. Sistem temu-kembali informasi menggunakan LSI ini belum diujicobakan untuk dokumen dalam jumlah yang besar (ratusan atau ribuan dokumen), sehingga kemungkinan munculnya dokumen-dokumen yang tidak relevan akan semakin banyak jika menggunakan batas ambang yang sama. Oleh karena itu penelitian lebih lanjut perlu dilakukan demi dihasilkannya sebuah sistem temu-kembali informasi yang baik untuk jumlah dokumen yang besar.

Daftar Pustaka

[1] Basuki, T. A., Penggunaan Semi Discrete Decomposition pada Latent Semantic Indexing untuk Temu-Kembali Informasi, *INTEGRAL*, vol. 6 no. 1, April 2001, 5-13.

[2] Chen, C., Stoffel, N., Post, M., Basu, C., Bassu, D., Behrens, C., Telcordia LSI Engine: Implementation and Scalability Issues, *Proc. of the 11th Int. Workshop on Research Issues in Data Engineering (RIDE 2001): Document Management for Data Intensive Business and Scientific Applications*, Heidelberg, Germany, Apr. 1-2, 2001.

[3] Alhensiri, A. A., Web Information Retrieval and Search Engine Techniques, *Al-Satil Journal*, Libya, 2003, 55-92.

[4] Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W., Harshman, R. A., Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science*, 41(6), 1990, 391-407.

[5] Berry, M. W., Dumais, S. T., O'Brien, G. W., Using Linear Algebra for Intelligent Information Retrieval, *SIAM Review*, 37(4), 1995, 573-595.

[6] Dumais, S. T., LSI meets TREC: A Status Report, In: D. Harman (Ed.), *The First Text REtrieval Conference* (*TREC1*), *National Institute of Standards and Technology Special Publication 500-207*, 1993, pp. 137-152.

[7] Berry, M. W., Browne, M., *Understanding Search Engines: Mathematical Modelling and Text Retrieval Second Edition*, Philadelphia: Society for Industrial and Applied Mathematics, 2005.

[8] Blom, K., *Information Retrieval using the Singular Value Decomposition and Krylov subspaces*, Sweden: Department of Mathematics, Chalmers University of Technology, 1999.

[9] Letsche, T. A., *Toward Large-Scale Information Retrieval Using Latent Semantic Indexing*, Master of Science Thesis, Knoxville: University of Tennessee, Knoxville, 1996.

[10] Baeza-Yates, R., Ribeiro-Neto, B., *Modern Information Retrieval*, England: Pearson Education Limited, 1999.