UNIVERSITAS INDONESIA

PENGINDEKSAN KONSEPTUAL SECARA DINAMIS DENGAN *SINGULAR VALUE DECOMPOSITION* PADA SISTEM TEMU KEMBALI INFORMASI

TESIS

Diajukan sebagai salah satu syarat untuk memperoleh gelar
Magister Ilmu Komputer

ADI WAHYU PRIBADI
7202000023

FAKULTAS ILMU KOMPUTER
PROGRAM MAGISTER ILMU KOMPUTER
JAKARTA
JANUARI 2006

UNIVERSITAS INDONESIA

PENGINDEKSAN KONSEPTUAL SECARA DINAMIS DENGAN *SINGULAR VALUE DECOMPOSITION* PADA SISTEM TEMU KEMBALI INFORMASI

TESIS

ADI WAHYU PRIBADI
7202000023

FAKULTAS ILMU KOMPUTER
PROGRAM MAGISTER ILMU KOMPUTER
JAKARTA
JANUARI 2006

HALAMAN PERNYATAAN ORISINALITAS

Tesis ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.

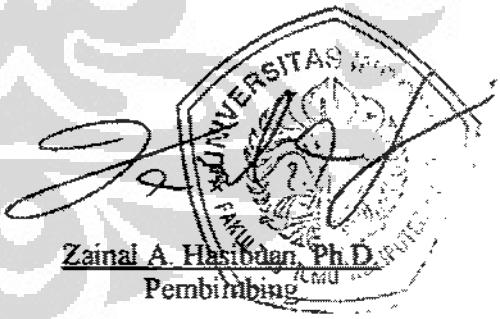
Nama	**: Adi Wahyu Pribadi**
NPM	**: 7202000023**
Tanda Tangan	**:**
Tanggal	**: 27 Januari 2006**

HALAMAN PENGESAHAN

Tesis	:	Pengindeksan Konseptual Secara Dinamis dengan *Singular Value Decomposition* pada Sistem Temu Kembali Informasi
Nama	:	Adi Wahyu Pribadi
NPM	:	7202000023

Tesis ini telah diperiksa dan disetujui,

Depok, Januari 2006



Zainal A. Hasibuan, Ph.D.
Pembimbing

KATA PENGANTAR

Puji syukur saya panjatkan kepada Allah SWT, karena atas berkat rahmat dan karunia-Nya, saya dapat menyelesaikan tesis ini. Penulisan tesis ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Magister Ilmu Komputer pada Fakultas Ilmu Komputer Universitas Indonesia. Selanjutnya saya mengucapkan terima kasih kepada semua pihak yang telah memberikan bimbingan, bantuan, moral, dan motivasi sehingga laporan tesis ini dapat diselesaikan, terutama kepada:

1. Ibunda tercinta yang senantiasa mendoakan penulis siang dan malam serta memberikan dorongan moril dan motivasi.
2. Bapak Zainal A. Hasibuan, Ph.D., selaku pembimbing tesis yang telah membimbing penulis selama penulisan tesis ini.
3. Indra Budi yang telah memberikan bantuan berupa saran dan sumber pustaka.
4. Hafidz yang telah membantu penulis dalam menyelesaikan beberapa *script* pemrograman.
5. Seluruh staf pengajar Fasilkom UI.
6. Rekan-rekan mahasiswa Program Magister Ilmu Komputer UI.
7. Pihak-pihak lainnya yang tidak dapat penulis sebutkan satu persatu

Akhir kata, saya berdoa agar Allah SWT berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga tesis ini membawa manfaat bagi pengembangan ilmu.

Depok, Januari 2006

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Adi Wahyu Pribadi
NPM : 7202000023
Program Studi : Magister Ilmu Komputer
Fakultas : Ilmu Komputer
Jenis karya : Tesis

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif** (***Non-exclusive Royalty-Free Right***) atas karya ilmiah saya yang berjudul:

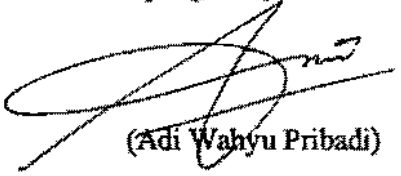
Pengindeksan Konseptual secara Dinamis dengan *Singular Value Decomposition* pada Sistem Temu Kembali Informasi

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia / formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada Tanggal : 27 Januari 2006

yang menyatakan



(Adi Wahyu Pribadi)

ABSTRAK

Nama : Adi Wahyu Pribadi

Program Studi : Magister Ilmu Komputer

Judul : Pengindeksan Konseptual secara Dinamis dengan Singular Value Decomposition pada Sistem Temu Kembali Informasi

Setiap dokumen pada koleksi menjelaskan suatu konsep berdasarkan topik yang dibahasnya. Konsep tersebut didapat dengan teknik pengindeksan konseptual atau *Latent Semantic Indexing*. Teknik tersebut mengakibatkan jumlah dokumen yang terambil lebih banyak karena adanya perluasan kueri (*query expansion*) secara konseptual. Seiring berjalannya waktu, terjadi penambahan dokumen sehingga indeks menjadi tidak lengkap. Digunakan metode penambahan dokumen secara dinamis pada indeks konseptual yang ada dengan metode *folding-in* dan *SVD-Update*. Ujicoba dilakukan pada kumpulan hasil penelitian lembaga BATAN sebanyak 1162 abstrak dokumen. Berdasarkan ujicoba, pada model pengindeksan konseptual dokumen yang terambil lebih banyak yaitu rata-rata 12,63% dibandingkan dengan penggunaan pengindeksan biasa sebanyak 10,37%. Pada ujicoba penambahan dokumen, terjadi penurunan kinerja yang tidak signifikan yaitu 0,5% hingga 2% saja.

Kata Kunci:

Temu Kembali Informasi, Pengindeksan Dinamis, *Singular Value Decomposition*, Ekspansi Kueri

ABSTRACT

Name : Adi Wahyu Pribadi

Studi Program : Magister of Computer Science

Judul : Conceptual Indexing Dinamically using Singular Value Decomposition in Informatin Retrieval System

Each document in the collection describes a concept based on particular topics. The concept is obtained with the technique of conceptual or latent Semantic Indexing. The technique resulting in the number of documents fetched more because of the conceptual query expansion. Over time, there was the addition of documents so that the indexes are not complete. Using Folding-in and SVD-Update to update the index of document collection conceptually. We use BATAN research collection of 1162 document abstracts. Based on testing, on the conceptual model of the document fetched more with the average of 12.63% compared with the normal indexing of 10.37%. On testing of adding documents, a decline of performance that is not significant, namely 0.5% to 2% only.

Key word :

Information Retrieval, dinamic indexing, singular value decomposition, query expantion

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Ledakan informasi dalam bentuk tulisan menyebabkan kesulitan bagi pengguna mendapatkan informasi yang cepat, padat, dan relevan terhadap kebutuhan informasinya. Untuk mengatasinya diperlukan suatu sistem pencarian yang dapat memberikan layanan terhadap kebutuhan informasi pengguna. Sistem pencarian seperti ini disebut sebagai Sistem Temu Kembali Informasi. Baeza mengatakan bahwasanya sistem temu kembali informasi berhubungan dengan representasi, media penyimpanan, organisasi, dan akses terhadap informasi (Baeza-Yates dan Ribeiro-Neto, 1999). Sistem temu kembali informasi adalah suatu sistem yang memproses berkas atau dokumen dan berusaha mengidentifikasikan serta menemukan kembali dokumen tertentu dari kumpulannya dalam rangka merespon kebutuhan informasi (kueri) pengguna. Penemuan kembali suatu dokumen bergantung pada kesamaan antara dokumen dan kueri yang diberikan (Salton, 1989).

Masing-masing dokumen pada koleksi menjelaskan suatu konsep berdasarkan topik yang dibahasnya (Berry, 1999). Cara jelas untuk mengerti isi atau konsep pada sebuah dokumen adalah dengan membaca keseluruhan dokumen tersebut, sehingga dapat diketahui dokumen-dokumen mana saja yang relevan dengan suatu kebutuhan informasi. Namun hal ini menjadi tidak praktis apabila dokumen yang disimpan pada koleksi berjumlah sangat banyak, setiap dokumen telah memiliki abstrak yang merupakan intisari dari dokumen. Oleh karena itu dilakukanlah proses identifikasi konsep sebuah dokumen. Proses identifikasi dilakukan dengan pemilihan istilah-istilah yang dianggap mewakili isi dari suatu dokumen. Proses identifikasi konsep dokumen disebut sebagai proses pengindeksan.

Proses pengindeksan dapat dilakukan dengan dua cara yaitu manual dan otomatis. Pengindeksan manual sering disebut juga sebagai *human indexing*. Pengindeks melakukan pengindeksan secara manual dengan memilih istilah-istilah yang dianggap mewakili dokumen. Sedangkan pengindeksan secara otomatis dilakukan

dengan menggunakan sistem berbasis komputer. Kedua cara sama-sama bertujuan menentukan istilah-istilah yang dianggap mewakili isi dari suatu dokumen.

Hasil proses pengindeksan yang baik adalah berupa istilah-istilah yang benar-benar dianggap mewakili isi suatu dokumen. Hal tersebut cukup sulit diwujudkan dikarenakan terdapatnya permasalahan yang muncul pada saat proses pengindeksan baik pengindeksan manual maupun otomatis. Pengindeksan manual bersifat subjektif sehingga hasil pengindeksan akan berbeda di antara para *indexer*. Juga terdapat ketidakkonsistenan istilah-istilah yang dipilih pada dokumen yang sama karena kekayaan bahasa sehingga ide atau topik dapat diungkapkan dengan cara yang berbeda-beda.

Lancester mengemukakan penggunaan kosa kata yang dikontrol (*vocabulary controller*) untuk mengatasi ketidakkonsistenan tersebut. Pengontrolan kosa kata memberikan sejumlah istilah yang sudah diotorisasi oleh seorang yang ahli yang dapat dijadikan indeks pada bidang-bidang tertentu. Pada pengindeksan otomatis terdapat beberapa proses yang ditempuh sehingga akhirnya didapat indeks istilah. Proses tersebut meliputi pemilihan kata atau *parsing*, pembuangan kata buang (*stoplist*), penghilangan atau pemotongan imbuhan sehingga didapat kata dasar (*stemming*), dan daftar istilah (*dictionary*) yang tidak perlu di-*stemming* (Indra Budi, 2002).

Proses temu-kembali nantinya tergantung pada istilah-istilah hasil proses pengindeksan. Istilah-istilah yang digunakan sebagai perwakilan isi dokumen kadang-kadang memiliki sinonim, yaitu kata-kata yang memiliki makna serupa. Bahkan suatu istilah yang sama memiliki konsep yang berbeda tergantung kalimat dan konteks yang menyertai istilah tersebut, disebut sebagai polisemi. Permasalahan tersebut dicoba untuk diatasi dengan menggunakan *thesaurus* yaitu daftar istilah-istilah terkelompok yang memiliki makna sama. *Thesaurus* dapat digunakan selama proses penyimpanan dokumen untuk mengontrol pembendaharaan kata. Di mana setiap variasi istilah-istilah diganti dengan istilah standar. Alternatif lain penggunaan *thesaurus* adalah selama proses kueri. Hal ini

memperluas istilah-istilah yang digunakan pada kueri sehingga dapat memastikan dokumen relevan tidak luput terambil akibat sempitnya istilah yang digunakan pada kueri (Korfhage, 1997).

Pada umumnya, pada proses *retrieval* istilah-istilah hasil pengindeksan pada dokumen dibandingkan dengan istilah-istilah yang terdapat pada kueri pemakai. Sedangkan seringkali terdapat lebih dari satu istilah untuk menerangkan sebuah konsep (sinonim) sehingga istilah yang digunakan pada kueri menjadi tidak optimal untuk menemukan kembali dokumen yang sebenarnya relevan. Begitu pula ketika istilah yang terdapat pada dokumen memiliki banyak konsep (polisemi). Akibatnya ketika kueri diberikan, dokumen yang terambil menjadi tidak relevan. Sehingga dilakukan pendekatan dengan membandingkan antara kueri dan dokumen berdasarkan konsepnya.

Proses temu kembali informasi berdasarkan konsep ini disebut sebagai metode *Latent Semantic Indexing* (LSI) (Deerwester, 1990). LSI menganggap terdapat hubungan konsep yang tersembunyi di antara istilah-istilah pada dokumen sehingga LSI disebut juga sebagai teknik pengindeksan konseptual (Tamara Kolda, 1998). Pengindeksan konseptual ini dilakukan dengan cara mendekomposisi matriks representasi istilah-dokumen. Langkah awal yang dilakukan untuk mendapatkan indeks konseptual tersebut adalah dengan membuat matriks istilah-dokumen yang merupakan representasi koleksi dokumen. Selanjutnya matriks tersebut didekomposisi dengan metode *Singular Value Decomposition* (SVD). Matriks hasil dekomposisi itulah yang kemudian digunakan sebagai indeks konseptual oleh kueri untuk menemukan kembali dokumen yang dicari. Secara intuisi kueri yang diberikan pada indeks konseptual merupakan perluasan kueri atau *query expansion* yang efeknya berupa jumlah dokumen yang terambil lebih banyak.

Sebagaimana matriks pada umumnya, matriks istilah-dokumen yang merupakan representasi koleksi dokumen memiliki nilai-nilai eigen (nilai-nilai karakteristik). Jumlah nilai eigen yang tak nol dari suatu matriks adalah *rank* dari matriks

tersebut (Steven J. Leon, 2001). Proses dekomposisi menghasilkan aproksimasi matriks istilah-dokumen baru yang memiliki *rank* lebih rendah dari matriks awal. Mengurangi *rank* suatu matriks berarti menghilangkan informasi ekstra atau *noise* dari basis data yang direpresentasikannya. Sedangkan pada LSI, menghilangkan *rank* berarti mengeluarkan hubungan konseptual yang tersembunyi antar istilah-istilah pada koleksi dokumen (Tamara Kolda, 1998).

Seiring berjalannya waktu, jarang sekali suatu informasi yang disimpan bersifat konstan sehingga menyebabkan indeks yang ada selalu berubah-ubah. Pada model LSI, hal yang paling jelas untuk mengakomodasi penambahan dokumen maupun istilah baru adalah dengan menghitung ulang indeks koleksi dokumen. O'Brien mengajukan metode penambahan koleksi dokumen baru tanpa harus menghitung ulang matriks representasi koleksi dokumen yang sudah dibuat (O'Brien, 1994). Sehingga secara intuisi pada koleksi dokumen berjumlah banyak metode tersebut lebih efisien dalam hal waktu dan ruang. Metode-metode untuk proses penambahan dokumen maupun istilah baru (*updating*) antara lain *folding-in* dan *SVD-Update* yang diajukan oleh O'Brien.

1.2 Perumusan Masalah

Berdasarkan latar belakang di atas maka permasalah yang dikaji dalam penelitian ini adalah:

- Bagaimana pengindeksan konseptual atau perluasan kueri (*query expansion*) dengan *Latent Semantic Indexing* dapat mengambil dokumen terambil lebih banyak?
- Bagaimana efisiensi dan efektivitas dalam mengindeks ulang matriks istilah-dokumen dengan metode *folding-in* dan *SVD-update*?

1.3 Tujuan

Adapun yang menjadi tujuan dalam penelitian ini adalah:

- Membandingkan efektivitas model temu kembali yang menggunakan model ruang vektor (pengindeksan leksikal) dengan model *LSI* (pengindeksan konseptual).
- Membandingkan efektivitas dan efisiensi penambahan koleksi dokumen baru antara mengindeks ulang keseluruhan matriks representasi koleksi dokumen dengan metode *folding-in* dan *SVD-Update*.

1.4 Ruang Lingkup

Uji coba dalam penelitian ini memanfaatkan dokumen-dokumen berbasis teks dalam koleksi dokumen berbahasa Indonesia. Adapun batasan-batasan dalam penelitian ini adalah:

- Koleksi yang digunakan adalah koleksi abstrak dokumen BATAN (Badan Tenaga Atom Nasional) dan dokumen yang digunakan adalah sebanyak 1162 dokumen.
- Kueri masukan mengacu kepada penelitian yang sudah dilakukan yaitu Mustangimah (1998), Ariwibowo (2001), dan Indra Budi (2002).
- Metode retrieval yang digunakan adalah model ruang vektor dan LSI dengan *Singular Value Decomposition*.
- Metode *updating* indeks koleksi dokumen berbasis LSI adalah dengan *folding-in* dan *SVD-Updating*.

1.5 Sistematika Penulisan

Bab 1 menjelaskan latar belakang dilakukannya penelitian ini beserta permasalahan dan ruang lingkupnya. Bab 2 memberikan penjelasan mengenai landasan teori dasar dalam bidang temu kembali informasi beserta model ruang vektor dan model pengindeksan konseptual yang digunakan pada penelitian ini. Bab ini juga membahas bagaimana menambah koleksi dokumen baru terhadap indeks koleksi dokumen yang sudah ada tanpa melakukan pengindeksan ulang.

Bab 3 membahas metodologi dalam merancang sistem temu kembali informasi pada penelitian ini, yaitu perancangan indeks, pemberian bobot indeks koleksi dokumen, melakukan kueri, dan penambahan koleksi dokumen. Ujicoba dan analisa terhadap hasil ujicoba diberikan pada Bab 4. Kesimpulan dan saran disampaikan pada Bab 5.

BAB 2
TINJUAN PUSTAKA

Bab kedua ini menyampaikan beberapa konsep dasar yang digunakan dalam bidang Temu Kembali Informasi. Dimulai dari penjelasan umum tentang definisi Sistem Temu Kembali Informasi (STKI), bagaimana mengukur kinerja Sistem Temu Kembali Informasi, metode pengindeksan koleksi dokumen, dan teknik pengindeksan secara konseptual yaitu menggunakan *Singular Value Decomposition*. Selanjutnya adalah bagaimana menambah koleksi dokumen tanpa harus mengindeks ulang koleksi dokumen lama ditambah koleksi dokumen baru tetapi cukup mengindeks koleksi barunya saja. Metode yang digunakan untuk pengindeksan dinamis ini adalah *folding-in* dan *SVD-Update*.

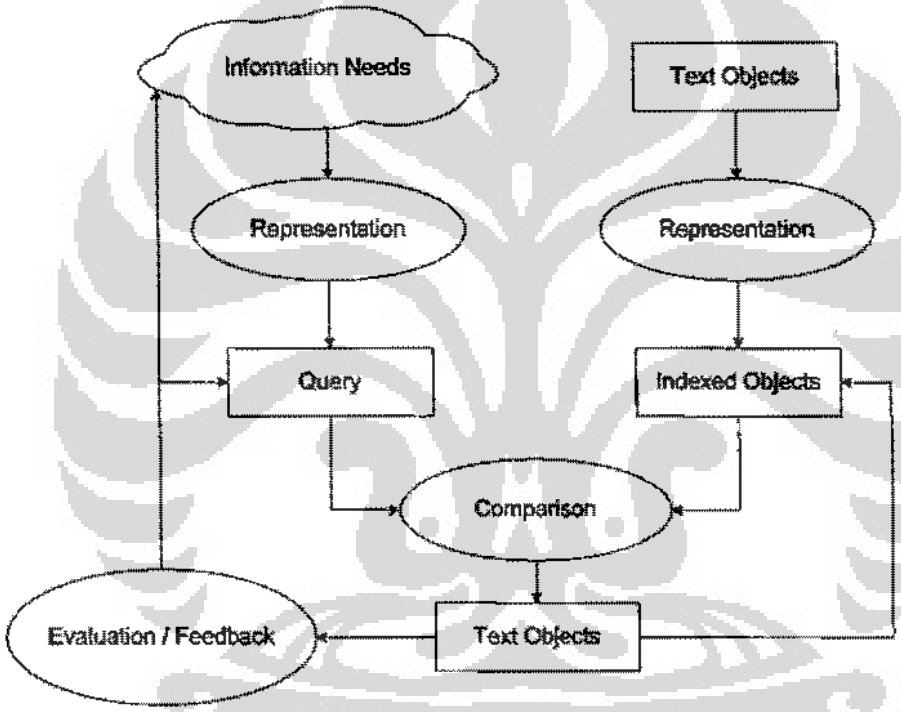
2.1 Sistem Temu Kembali Informasi

Sistem temu kembali informasi berhubungan erat dengan representasi, penyimpanan, organisasi, dan akses ke informasi. Representasi dan organisasi informasi yang baik akan memudahkan pengguna dalam mendapatkan akses informasi yang dibutuhkannya. Mengingat pesatnya perkembangan informasi maka diperlukan sistem temu kembali yang memiliki kinerja baik.

Tujuan utama dikembangkannya sistem temu kembali informasi adalah mendapatkan semua dokumen yang relevan sesuai dengan permintaan pengguna dan mendapatkan sedikit mungkin dokumen yang tidak relevan (Baeza-Yates dan Ribeiro-Neto, 1999).

James Allan (2004) merangkumkan proses-proses yang terdapat pada STKI. Proses awal dalam STKI adalah mengolah teks atau dokumen yang terdapat pada koleksi dokumen, yaitu dengan merepresentasikan atau mengidentifikasikan sejumlah istilah yang dianggap mewakili konsep dokumen sehingga didapatkan indeks istilah. Selanjutnya seorang pengguna menentukan kebutuhan informasinya dengan menuliskan kata kunci – kata kunci yang akan digunakan

sebagai kueri. Kemudian dilakukan proses pencocokan antara kata kunci pengguna dengan indeks istilah yang terdapat pada koleksi. Proses pencocokan ini dilakukan sesuai dengan model-model STKI tertentu dengan menentukan dan mengambil informasi atau dokumen yang sesuai dengan permintaan pengguna. Kadang kala hasil pencarian tidak memuaskan pengguna sehingga diterapkan mekanisme *relevance feedback* yaitu user memberikan umpan balik berupa perbaikan kueri sehingga didapatkan hasil pencarian yang lebih baik dari sebelumnya. Proses-proses tersebut dapat dilihat pada Gambar 2.1. di bawah ini.



Gambar 2.1. Alur Proses Sistem Temu Kembali Informasi (Allan, 2004)

Sistem temu kembali informasi (*Information Retrieval*) berbeda dengan sistem manajemen basis data (*Data Retrieval*). Perbedaannya, menurut Frakes (1992) antara lain terletak pada data objek dan informasi yang dihasilkan. Data objek yang digunakan pada sistem manajemen basis data adalah tabel-tabel terstruktur, sedangkan dalam sistem temu-kembali informasi, data objek yang digunakan adalah dokumen-dokumen berbasis teks yang tidak terstruktur.

Informasi yang dihasilkan sistem manajemen basis data bersifat deterministik. Pada sistem temu-kembali informasi, dokumen yang dihasilkan mengandung suatu probalibilitas, yaitu dokumen terambil tidak selalu sesuai dengan keinginan pengguna. Hal ini disebabkan karena dalam sistem manajemen basis data, berkas-berkas yang diolah terdiri dari *record-record* homogen yang bercirikan sejumlah atribut tertentu. Untuk memperoleh *record* yang dibutuhkan, kueri yang diberikan harus mengandung nilai atribut dari *record* tersebut (*exact match*). Sedangkan dalam sistem temu-kembali informasi, yang *record*-nya berupa teks atau dokumen, selain atribut formal juga digunakan atribut (identifikasi) informal yaitu istilah-istilah (*index term*). Untuk mengidentifikasikan suatu teks atau dokumen digunakan beberapa istilah yang berbeda. Istilah-istilah tersebut tidak dapat dijamin benar-benar mewakili keseluruhan isi dokumen.

2.2 Kinerja Temu Kembali Informasi

Keluaran STKI adalah dokumen terambil yang merupakan *subset* dari koleksi dokumen. Kinerja suatu STKI dilihat dari jumlah dokumen relevan yang terambil dari koleksi dokumen (Baeza-Yates dan Ribeiro-Neto, 1999). Perhitungan ini biasa disebut dengan *recall* dan *precision*. Dokumen yang relevan adalah dokumen yang terambil dari hasil penerapan sebuah kueri dan dokumen tersebut memenuhi kebutuhan informasi pengguna STKI.

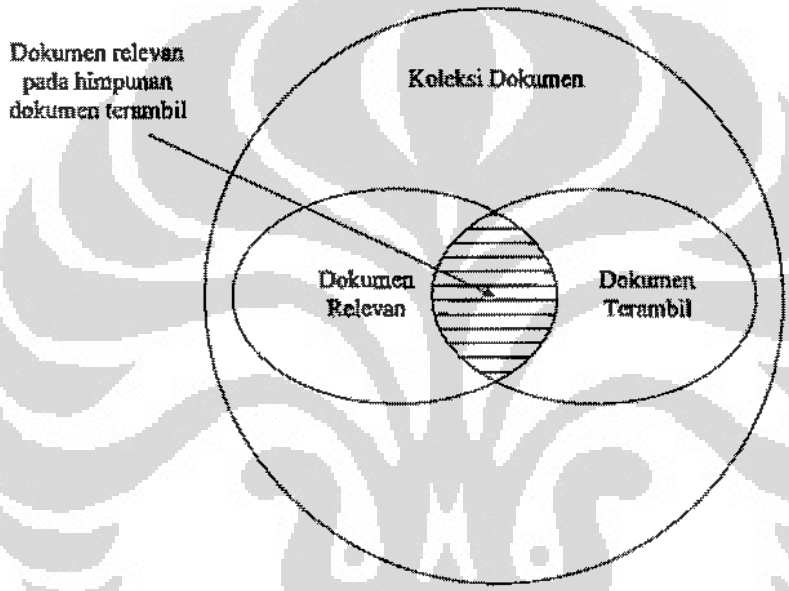
Recall menunjukkan berapa banyak dokumen terambil yang relevan sebagai hasil kueri dari sekumpulan dokumen yang relevan. Secara matematis, rumusan *recall* dapat ditulis dalam bentuk

$$recall = \frac{jumlah\ dokumen\ relevan\ terambil}{jumlah\ keseluruhan\ dokumen\ relevan\ pada\ koleksi\ dokumen}$$

Precision menunjukkan jumlah dokumen terambil yang relevan dari kumpulan dokumen hasil temu kembali. Bentuk rumus *precision* adalah seperti berikut

$$precision = \frac{jumlah\ dokumen\ relevan\ terambil}{jumlah\ keseluruhan\ dokumen\ yang\ terambil}$$

Recall dan *precision* dapat diilustrasikan berdasarkan diagram venn seperti yang tampak pada Gambar 2.2. di bawah ini. Pada gambar tersebut, irisan antara himpuan dokumen dan dokumen terambil adalah himpunan dokumen yang relevan dan juga terambil oleh STKI. Sehingga *recall* adalah banyaknya anggota himpunan irisan tersebut dibagi oleh banyaknya anggota himpunan dokumen relevan. Sedangkan *precision* adalah banyaknya anggota himpunan irisan tersebut dibagi oleh banyaknya anggota himpunan dokumen terambil.



Gambar 2.2. Diagram Venn himpunan dokumen relevan dan dokumen terambil serta irisan keduanya pada koleksi dokumen

Sebagai contoh, misalkan 40 buah dokumen berhasil diambil sebagai hasil dari suatu kueri dan 25 di antaranya relevan. Dari keseluruhan dokumen yang ada di koleksi, ternyata sebenarnya ada 50 dokumen yang relevan. Dari kedua persamaan tadi, maka nilai *precision* adalah $^{25}/_{40} = 0{,}625$ (62,5%). Sedangkan nilai *recall* adalah $^{25}/_{50} = 0{,}5$ (50%). Dari contoh di atas, jelas terlihat bahwa rumusan *precision* menghitung keakuratan dari hasil STKI. Sedangkan *recall* menghitung cakupan dari seluruh koleksi.

Dalam praktiknya, pengukuran kinerja melalui *recall* dan *precision* memiliki keterbatasan sebagai berikut:

- *Recall* tidak terdefinisi jika tidak ada dokumen relevan dalam koleksi dokumen
- *Precision* tidak terdefinisi jika tidak ada dokumen terambil
- Perhitungan jumlah dokumen relevan sulit dilakukan dalam koleksi dokumen yang besar. Selain harus ditentukan terlebih dahulu siapakah yang menilai dan apa saja yang berguna, manusia juga cenderung tidak konsisten dalam memberikan penilaian

2.3 Pengindeksan

Seperti dijelaskan sebelumnya, salah satu komponen penting yang terdapat pada proses STKI adalah menentukan istilah-istilah yang dianggap mewakili atau merepresentasikan koleksi dokumen. Proses tersebut disebut sebagai proses pengindeksan. Proses temu kembali dokumen yang relevan nantinya tergantung dari pemilihan istilah tersebut, sehingga jika istilah-istilah yang dipilih tidak cukup mewakili isi dokumen maka akan menurunkan efektivitas dari STKI.

Proses pengindeksan terbagi ke dalam dua bagian besar, yaitu pengindeksan manual dan pengindeksan otomatis (*manual and automatic indexing*). Pengindeksan manual adalah pengindeksan yang dilakukan secara manual, baik oleh penulis maupun oleh seorang ahli tertentu yang sesuai dengan topik pada dokumen yang diindeksnya. Pengindeksan manual juga dikenal dengan istilah *human indexing* (Lancaster, 1998). Sedangkan pengindeksan otomatis adalah pengindeksan yang dilakukan dengan menggunakan sistem berbasis komputer (*computer based system*). Pada penelitian ini, proses pengindeksan yang dilakukan adalah dengan pengindeksan otomatis.

Pada proses pengindeksan otomatis, sistem membaca dokumen kemudian menghasilkan sekumpulan istilah yang merupakan indeks. Hal-hal yang dilakukan dalam proses pengindeksan adalah:

- *Parsing* dokumen, merupakan proses pengambilan kata-kata untuk dijadikan istilah.
- *Stoplist*, merupakan kumpulan kata-kata yang tidak akan dijadikan istilah. Biasanya yang termasuk dalam *stoplist* adalah kata sambung, kata keterangan, dan sebagainya.
- *Stemming*, merupakan proses penghilangan atau pemotongan imbuhan dari suatu kata. Salah satu tujuan dari penghilangan imbuhan ini adalah untuk mengurangi penyimpanan istilah, dapat meningkatkan jumlah istilah terambil, dapat meningkatkan jumlah dokumen terambil, dan dapat memperluas arti dari suatu istilah (Salton, 1989).

Proses selanjutnya adalah membuat indeks pembalik dokumen (*inverted index*) yaitu teknik pengindeksan istilah dan dokumen secara terurut (Frakes, 1992). Misalkan terdapat sekumpulan *n* dokumen. Setiap dokumen mempunyai daftar istilah yang digunakan sebagai representasi dokumen. Daftar istilah dari masing-masing dokumen tersebut kemudian digabungkan dan diurutkan secara *ascending* tanpa menghapus hubungan antara suatu istilah dengan dokumennya. Hasilnya berupa istilah yang merujuk ke satu atau lebih dokumen, sehingga pencarian terhadap suatu istilah juga berarti mendapatkan dokumen-dokumen yang dirujuk oleh istilah tersebut.

Indeks pembalik dokumen itulah yang kemudian diberi bobot sesuai dengan model temu kembali yang digunakan.

2.4 Model Sistem Temu Kembali Informasi

Pada subbab berikut membahas mengenai model-model sistem temu kembali informasi yang berhubungan dengan penelitian ini, yaitu: Model Ruang Vektor, *Latent Semantic Indexing* berdasarkan *Singular Value Decomposition*, Kontribusi nilai singular, dan proyeksi kueri.

2.4.1 Model Ruang Vektor

Secara formal, sebuah ruang vektor didefinisikan oleh himpunan basis vektor yang saling bebas linier (Allan, 2004), dimana basis vektor berhubungan dengan besarnya dimensi atau arah pada ruang vektor. Pada model ruang vektor di STKI, setiap dokumen dan kueri direpresentasikan ke dalam ruang vektor. Setiap komponen vektor merefleksikan konsep tertentu, kata kunci, atau istilah yang berhubungan dengan dokumen yang diberikan (Berry, 1999).

Sebanyak m istilah hasil proses pengindeksan dari sejumlah n dokumen pada koleksi dokumen lalu direpresentasikan ke dalam model ruang vektor. Representasinya berupa matrik istilah-dokumen (*matrix term-document*). Elemen pada matrik tersebut merupakan bobot istilah dalam dokumen.

$$\begin{pmatrix} & d_1 & d_2 & \cdots & \cdots & d_n \\ t_1 & w_{11} & w_{21} & & & w_{1n} \\ t_2 & w_{12} & w_{22} & & & w_{2n} \\ \vdots & \vdots & \vdots & \cdots & & \vdots \\ \vdots & \vdots & \vdots & & \cdots & \vdots \\ t_m & w_{m1} & w_{m2} & \cdots & \cdots & w_{mn} \end{pmatrix}$$

Gambar 2.3. Representasi koleksi dokumen berupa matrik istilah-dokumen

Gambar 2.3. di atas adalah matrik istilah-dokumen yang merupakan representasi dari koleksi dokumen. Berdasarkan gambar tersebut, koleksi dokumen terdiri atas n dokumen yaitu d_1, d_2, ..., hingga d_n serta memiliki istilah-istilah yaitu t_1, t_2, t_3, ..., hingga t_m. Elemen w_{ij} adalah bobot istilah t_i pada dokumen d_j. Misalkan Elemen w_{49} adalah bobot istilah t_4 pada dokumen d_9.

2.4.1.1 Pembobotan Elemen Matrik Istilah-Dokumen

Selanjutnya koefisien diberikan kepada vektor untuk menunjukkan kehadiran suatu istilah, nilai bobotnya, atau "*aboutness*" dari suatu vektor (Allan, 2004).

Terdapat bermacam-macam pilihan untuk memberikan bobot nilai suatu vektor. Diantaranya adalah:

- Bobot binari, elemen suatu vektor dokumen *d* diberi bobot "1" jika istilah *i* muncul pada dokumen tersebut. Sebaliknya jika tidak terdapat istilah *i* pada dokumen *j* tersebut maka bobotnya adalah "0".
- Bobot berdasarkan frekuensi kemunculan istilah *i* pada suatu dokumen *j*. Teknik pembobotan ini cukup sederhana dimana bobot suatu istilah pada sebuah dokumen didasarkan pada kemunculannya pada dokumen tersebut (*term frequency*).
- Teknik pembobotan local dan global. Teknik ini untuk menentukan seberapa pentingnya istilah *i* di dalam sebuah dokumen *j*.

Terdapat berbagai macam teknik berdasarkan pembobotan local dan global ini. Di antaranya adalah:

Pembobotan Lokal dan Global berdasarkan Rumus Savoy

Pembobotan ini dikembangkan oleh Jacques Savoy (1994). Pada teknik ini, setiap istilah *i*, yaitu $i = 1, 2, ..., m$ pada dokumen *j* diberikan bobot w_{ij} yang dihitung menggunakan rumusan:

$$w_{ij} = tf_{ij} * idf_i \qquad (2.1)$$

di mana

$$idf_i = \log\left(\frac{n}{df_i}\right) \qquad (2.2)$$

keterangan:

w_{ij} adalah bobot istilah *i* pada dokumen *j*.

tf_{ij} merupakan frekuensi dari istilah *i* pada dokumen *j*.

n adalah jumlah dokumen dalam koleksi dokumen.

df_i adalah jumlah dokumen yang mengandung istilah i.

Rumus di atas terdiri dari dua komponen yaitu bobot lokal tf_{ij} dan bobot global idf_i. Kedua komponen tersebut dinormalisasi dalam rentang [0, 1] menjadi:

$$w_{ij} = ntf_{ij} * nidf_i \quad (2.3)$$

$$ntf_{ij} = \frac{tf_{ij}}{Max_i tf_{ij}} \quad (2.4)$$

$$nidf_i = \frac{\log\left(\frac{n}{df_i}\right)}{\log(n)} \quad (2.5)$$

keterangan:

$Max_i tf_{ij}$ frekuensi istilah terbesar pada satu dokumen

Pada teknik pembobotan ini, bobot istilah dinormalisasikan dengan cara membagi frekuensi kemunculan istilah tersebut di suatu dokumen dan frekuensi terbesar suatu istilah yang dimiliki oleh dokumen yang bersangkutan. Contoh, diketahui istilah 'pandai' di dokumen 18 berjumlah 8, sedangkan di dokumen 18 tersebut ada istilah yang memiliki frekuensi kemunculan paling tinggi yaitu 'cerdik' sebanyak 16 kali. Maka untuk menghitung bobot istilah pandai pada dokumen 18 adalah membagi frekuensi kemunculan 'pandai' dengan frekuens kemunculan 'cerdik' yaitu $^{8}/_{16} = 0{,}5$.

Hal ini untuk menentukan posisi relatif bobot dari istilah tersebut dibandingkan dengan istilah-istilah lain di dokumen yang sama. Selain itu teknik ini juga memperhitungkan jumlah dokumen yang mengandung istilah yang bersangkutan dan jumlah keseluruhan dokumen. Sehingga berguna untuk mengetahui posisi relatif bobot istilah yang bersangkutan pada suatu dokumen dibandingkan dengan dokumen-dokumen lain yang memiliki istilah sama. Dengan demikian istilah yang sama pada dua dokumen yang berbeda belum tentu memiliki bobot yang sama.

Pembobotan Global lainnya

Berikut ini adalah empat macam pembobotan global yang umum digunakan seperti yang telah dilakukan oleh Dumais (1991), O'Brien (1994), dan Berry dan kawan-kawan (1995), yaitu:

- Normal $$G(i) = \frac{1}{\sqrt{\sum_j (tf_{ij})^2}} \quad (2.6)$$

- Gfidf $$G(i) = \frac{gf_i}{df_i} \quad (2.7)$$

- Idf $$G(i) = \log_2\left(\frac{ndocs}{df_i}\right) + 1 \quad (2.8)$$

- 1 – Entropi (noise) $$G(i) = 1 - \sum_j \frac{p_{ij}\log(p_{ij})}{\log(ndocs)} \quad (2.9)$$

di mana $p_{ij} = \frac{tf_{ij}}{df_i}$

keterangan:

tf_{ij} adalah frekuensi istilah i pada dokumen j

df_i adalah jumlah dokumen yang memiliki istilah i

gf_i frekuensi semua istilah i di dalam koleksi dokumen

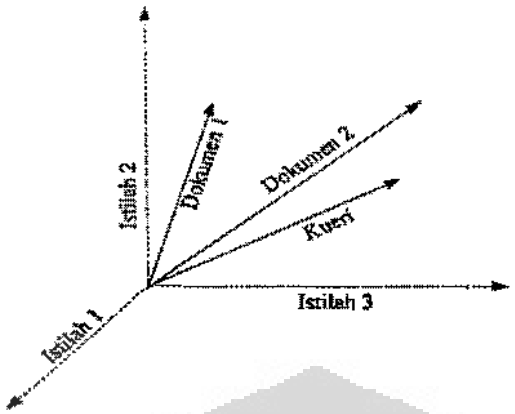
ndocs adalah jumlah semua dokumen

Berdasarkan penelitian yang telah dilakukan oleh Dumais (1991), dari keempat pembobotan global tersebut, maka pembobotan yang mengurangi entropi atau noise (2.9) memiliki kinerja paling baik. Oleh karena itu, metode pembobotan yang digunakan pada penelitian ini adalah metode pembobotan dengan pengurangan entropi (*noise*). Sehingga bobot istilah i di dalam dokumen j adalah:

$$w_{ij} = \log(tf_{ij} + 1) \times \left(1 - \sum_j \frac{p_{ij}\log(p_{ij})}{\log(nodes)}\right) \quad (2.10)$$

2.4.1.2 Dokumen Keluaran (*Output*)

Langkah selanjutnya pada model ruang vektor STKI adalah menentukan dokumen keluaran apabila diberikan kueri masukan dari pengguna. Metode ini dilakukan dengan menghitung kesamaan (*similarity*) antara vektor dokumen dengan vektor kueri.



Gambar 2.4. Similaritas pada ruang vektor

Similaritas berhubungan secara terbalik dengan sudut antara 2 vektor (Allan, 2004). Pada gambar 2.4. di atas terdapat 3 buah vektor yaitu vektor Dokumen 1, vektor Dokumen 2, dan vektor Kueri. Ketiga vektor tersebut berada pada ruangan 3 dimensi dengan sumbu istilah 1, istilah 2, dan istilah 3. Berdasarkan gambar, sudut antara vektor Kueri dengan vektor Dokumen 2 lebih dekat dibandingkan vektor Kueri terhadap vektor Dokumen1. Maka dapat dikatakan bahwa Dokumen 2 paling sama dibandingkan dengan Dokumen 1.

Dalam ruang vektor, kesamaan antara dua 2 vektor dapat dinyatakan dengan *dot product* yaitu hasil kali titik antara kedua vektor tersebut, atau dengan rumusan aturan kosinus. Semakin kecil sudut antara dua vektor berarti kedua vektor tersebut semakin mirip. Sebuah dokumen d_j dan sebuah kueri q direpresentasikan ke dalam vektor t-dimensi, dimana t adalah banyaknya indeks istilah pada koleksi dokumen. Maka derajat kesamaan antara vektor dokumen dengan vektor kueri adalah (Baeza-Yates dan Ribeiro-Neto, 1999).

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{\left|\vec{d_j}\right| \times \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \qquad (2.11)$$

keterangan:

$w_{i,j}$ adalah bobot istilah i pada dokumen j, dan

$w_{i,q}$ adalah bobot istilah i pada kueri q.

Semakin kecil sudut antara dua vektor berarti kedua vektor tersebut semakin mirip

2.4.2 *Latent Semantic Indexing*

Representasi koleksi dokumen berupa himpunan indeks istilah dapat menurunkan kinerja sistem temu kembali informasi. Seperti yang dibahas sebelumnya, seringkali terdapat lebih dari satu istilah hanya untuk menerangkan sebuah konsep (sinonim) sehingga istilah yang digunakan pada kueri menjadi tidak cocok terhadap dokumen yang sebenarnya relevan. Begitu pula ketika istilah yang terdapat pada dokumen memiliki banyak arti (polisemi). Akibatnya ketika kueri diberikan, dokumen yang terambil tidak relevan. Pendekatan yang lebih baik memungkinkan pengguna untuk mendapatkan kebutuhan informasinya berdasarkan topik atau konsep dari sebuah dokumen.

Hingga muncullah ide untuk mencocokkan dokumen dengan kueri berdasarkan konsep yang diberikan bukan berdasarkan pencocokkan indeks istilah, dengan menggunakan teknik *latent semantic indexing* (LSI) (Deerwester, 1990). Ide utama dari LSI adalah memetakan vektor dokumen dan vektor kueri ke dalam ruang dimensi yang lebih rendah dimana hal tersebut berhubungan dengan konsep (Furnas, 1988).

Terdapat beberapa metode dalam menerapkan LSI, diantaranya adalah dengan metode faktorisasi QR yang dilakukan oleh (Berry, 1999), dan *Singular Value Decomposition* (SVD) (dilakukan oleh Furnas, 1988; Deerwester, 1990; Berry, 1999; dan Yu Liao, 2001). Berry menunjukkan bahwa penggunaan metode QR dapat mengidentifikasikan dan sekaligus menghilangkan informasi redundan pada matriks representasi koleksi dokumen. Hal tersebut dapat dilakukan dengan cara mengurangi rank matriks koleksi dokumen dengan metode QR. Hal yang sama juga dilakukan pada metode SVD dimana matriks koleksi dokumen didekomposisi hingga membentuk matriks baru yang memiliki nilai eigen yang lebih sedikit. Tidak seperti pada metode QR, pada SVD terdapat representasi rank berdimensi rendah untuk ruang baris dan kolom dari matriks koleksi dokumen. Pada penelitian ini digunakan LSI dengan metode SVD untuk mengolah koleksi dokumen dan akan dilihat hasilnya berdasarkan percobaan yang telah dilakukan.

2.4.3 *Singular Value Decomposition*

Singular value decomposition adalah salah satu teknik ampuh dalam berbagai macam komputasi dan analisis matrik. Teknik SVD sudah banyak diterapkan pada berbagai macam aplikasi, antara lain pada masalah *least-square* untuk memecahkan sistem persamaan linier (Leach, 1995), *signal processing*, statistik, pengolahan citra, dan di bidang temu kembali informasi (Berry, 1999). Masing-masing aplikasi tersebut memanfaatkan kelebihan teknik SVD yaitu yang berhubungan jumlah *rank* yang digunakan untuk membangun matrik aproksimasi baru (Leach, 1995).

Jika A adalah matrik m x n, maka A mempunyai suatu dekomposisi nilai singular. Steven J. Leon (2001) menuliskan dekomposisi tersebut dituliskan sebagai:

$$A_{m\times n} = U_{m\times m} \times \Sigma_{m\times n} \times V_{n\times n}^{T} \qquad (2.12)$$

Dimana, U adalah matrik ortogonal $m \times m$ didapat dari eigenvektor dari AA^T; V adalah matrik ortogonal $n \times n$ didapat dari eigenvektor A^TA; sedangkan Σ adalah matrik $m \times n$ yang semua entri di luar diagonalnya adalah 0, dan elemen-elemen diagonalnya adalah akar kuadrat nilai eigen dari matrik orthogonal U dan V dan memenuhi persamaan $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$

$$\Sigma = \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_k \end{pmatrix}$$

Semua σ_i yang ditentukan dengan faktorisasi ini adalah tunggal dan disebut nilai-nilai singular dari matrik A. Faktorisasi U, Σ, dan V^T disebut dekomposisi nilai singular (*singular value decomposition*).

Teorema 2.1. SVD dari $A = U \times \Sigma \times V^T$ dan $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$. Kemudian $R(A)$ dan $N(A)$ merupakan perentang dan ruang null dari A, maka

- rank$(A) = r$, $N(A) \equiv$ rentang$\{v_{r+1},...,v_n\}$, dan $R(A) \equiv$ rentang$\{u_1,...,u_r\}$ dimana $U = [u_1 u_2 ... u_m]$ dan $V = [v_1 v_2 ... v_n]$
- *Dyadic decomposition*: $A \equiv \sum_{i=1}^{k} u_i . \sigma_i . v_i^T$
- Norms: $\|A\|_F^2 = \sigma_1^2 + ... + \sigma_r^2$, dan $\|A\|_2^2 = \sigma_1$

Teorema 2.2. Eckart dan Young mendefinisikan matrik A_k dibangun berdasarkan dari beberapa nilai singular pembagun matrik A (O'Brien 1994). Ambil SVD A dari persamaan (2.12.) dengan $r = rank(A) \leq p = \min(m,n)$ dan didefinisikan

$$A_k = \sum_{i=1}^{k} u_i . \sigma_i . v_i^T \qquad (2.13)$$

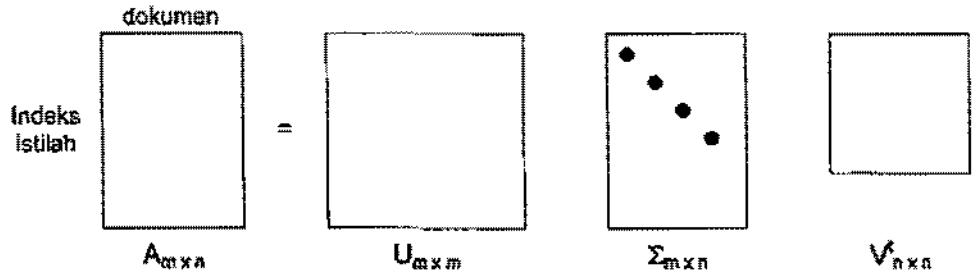
$$\min_{rank(B)=k} \|A - B\|_F^2 = \|A - A_k\|_F^2 = \sigma_{k+1}^2 + ... + \sigma_p^2$$

Dengan kata lain, A_k, yang dibangun berdasarkan nilai singular terbesar dari A, adalah matrik rank-k yang terdekat dengan matrik A. Jadi A_k merupakan matrik aproksimasi dari matrik A sehingga

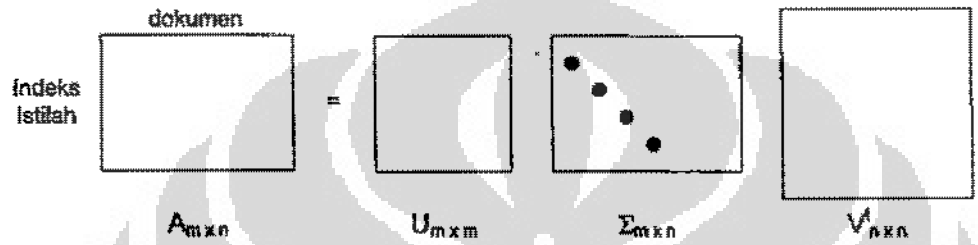
$$\min_{rank(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1} \qquad (2.14)$$

Jika matriks A merupakan representasi dari indeks koleksi dokumen maka matriks $A_{m \times n}$ merupakan matrik m istilah - n dokumen. Gambar 2.5 menjelaskan bagaimana faktorisasi matriks $A_{m \times n}$ menjadi hasilkali matriks-matriks $U_{m \times m}$, $V_{n \times n}$, dan $\sum_{m \times n}$ ketika $m > n$ dan $m < n$ (Berry, 1999). Gambar 2.6 merepresentasikan pengurangan jumlah nilai singular matriks A dari rank r ke rank k sehingga membentuk matriks A_k.

$m > n$

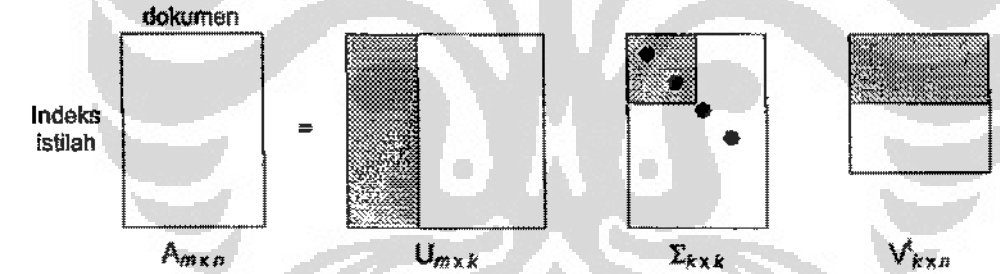


$m < n$

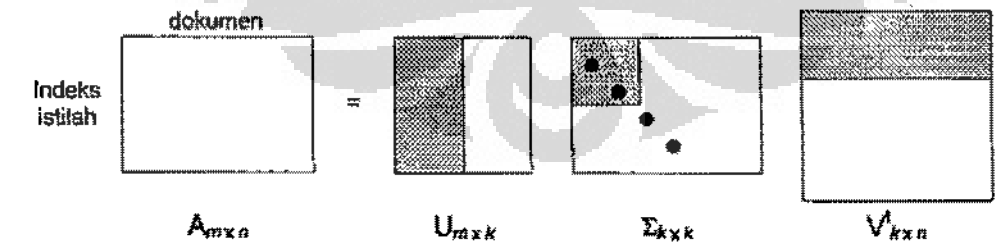


Gambar 2.5. Faktorisasi matriks A menjadi hasilkali $U \sum V^T$

$m > n$



$m < n$



Gambar 2.6. Membangun matriks aproksimasi A dengan rank k tertinggi

Sebagai contoh, matriks $A = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 2 & 1 \\ 1 & 0 & 4 \\ 3 & 2 & 1 \end{pmatrix}$ memiliki faktorisasi SVD

$$U = \begin{pmatrix} -0.5975 & -0.0328 & -0.3673 & -0.7121 \\ -0.2709 & 0.2097 & -0.7284 & 0.5934 \\ -0.5840 & -0.6563 & 0.3183 & 0.3560 \\ -0.4780 & 0.7240 & 0.4829 & 0.1187 \end{pmatrix},$$

$$\Sigma = \begin{pmatrix} 6.1550 & 0 & 0 \\ 0 & 2.9410 & 0 \\ 0 & 0 & 1.8619 \\ 0 & 0 & 0 \end{pmatrix}, \text{ dan}$$

$$V = \begin{pmatrix} -0.4250 & 0.5042 & 0.7518 \\ -0.4375 & 0.6127 & -0.6582 \\ -0.7925 & -0.6086 & -0.0398 \end{pmatrix}$$

Dari contoh di atas, diketahui rank(A) = 3, selanjutnya dipilih dua nilai singular terbaik yaitu nilai singular 6.1550 dan 2.9410 atau $k = 2$. Maka didapatkan matriks aproksimasi:

$$A_k = \underbrace{\begin{pmatrix} -0.5975 & -0.0328 \\ -0.2709 & 0.2097 \\ -0.5840 & -0.6563 \\ -0.4780 & 0.7240 \end{pmatrix}}_{U_k} \underbrace{\begin{pmatrix} 6.1550 & 0 \\ 0 & 2.9410 \end{pmatrix}}_{\Sigma_k} \underbrace{\begin{pmatrix} -0.4250 & -0.4375 & -0.7925 \\ 0.5042 & 0.6127 & -0.6086 \end{pmatrix}}_{V_k^T}$$

$$A_k = \begin{pmatrix} 1.5142 & 1.5498 & 2.9728 \\ 1.0196 & 1.1073 & 0.9461 \\ 0.5544 & 0.3901 & 4.0236 \\ 2.3240 & 2.5918 & 1.0358 \end{pmatrix}$$

Jadi A_k merupakan matriks aproksimasi A hasil dekomposisi SVD dengan menggunakan rank $k = 2$.

2.4.4 Kontribusi Nilai Singular

Hal yang sulit dilakukan adalah menentukan berapa banyak nilai singlar yang dibutuhkan untuk mendapatkan matriks aproksimasi A_k yang optimal sehingga kinerja temu kembali informasi meningkat. Skillicorn (2003) menjelaskan kontribusi nilai singular terhadap matriks aproksimasi dapat diterangkan dengan fungsi aljabar berikut ini.

$$f_k = \frac{s_k^2}{\sum_{i=1}^{r} s_i^2} \qquad (2.15)$$

Dimana:

s_k adalah nilai singular yang didapatkan dari dekomposisi matriks, dan

r adalah banyak nilai singular yang ada pada suatu matriks.

Kemudian entropinya dirumuskan sebagai:

$$entropy = \frac{-1}{\log r} \sum_{i=1}^{r} f_k \log(f_k) \qquad (2.16)$$

yang bernilai antara 0 dan 1. Untuk menentukan sebarapa banyak nilai singular yang dibutuhkan sehingga kinerja STKI optimal diperlukan penelitian lebih lanjut.

2.4.5 Proyeksi Kueri

Sebuah kueri dapat dianggap sebagai sebuah dokumen juga, sehingga dapat direpresentasikan sebagai sebuah vektor di dalam ruang k-dimensi. Sebagai sebuah *pseudo-document*, kueri merupakan jumlah dari bobot komponen vektor istilah, sehingga dapat dibandingkan dengan seluruh dokumen dalam koleksi. Sebuah kueri q_v didefinisikan sebagai sebuah vektor dari bobot istilahnya. Vektor kueri q_v dapat direpresentasikan secara matematik sebagai *pseudo-document* dengan (O'Brien, 1994).

$$q = q_v^t U_k \Sigma_k^{-1} \qquad (2.17)$$

di mana,

q = vektor *pseudo-document*

q_v = vektor kueri

Secara konsep vektor dokumen yang terdekat dengan *pseudo-document* dapat diperoleh (O'Brien, 1994). Salah satu metode pengukuran dokumen yang terdekat dengan vektor *pseudo-document* adalah dengan kosinus. Sama dengan metode peringkat pada model ruang vektor, semakin dekat vektor kueri dengan suatu vektor dokumen berarti keduanya semakin memiliki kesamaan. Suatu nilai ambang (*threshold*) dapat ditetapkan sehingga dokumen dengan nilai kosinus yang berada di atas ambang tersebut dapat digunakan sebagai dokumen keluaran terhadap kueri pengguna (Deerwester, 1990).

2.5 Penambahan Koleksi Dokumen

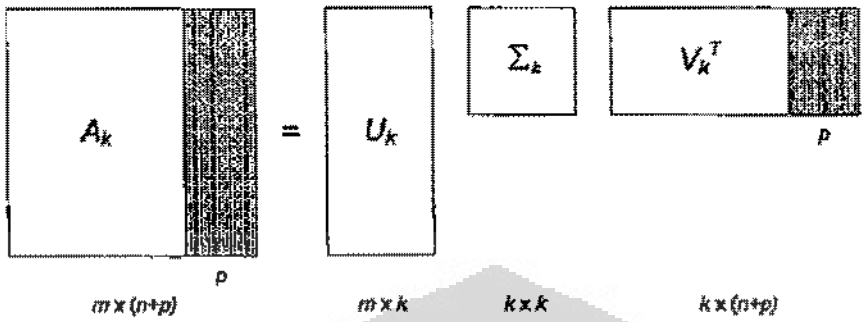
Seiring berjalannya waktu, jarang sekali suatu informasi yang disimpan bersifat konstan. Informasi selalu bertambah atau berkurang dan hal ini menyebabkan indeks yang ada selalu berubah-ubah. Pada model LSI, pendekatan yang paling jelas dalam mengakomodasi penambahan istilah atau dokumen baru adalah dengan menghitung ulang SVD dari matriks istilah-dokumennya. Sayangnya, untuk database berskala besar prosedur ini menjadi sangat mahal baik dalam hal waktu dan ruang. Pada penelitian ini, digunakan 2 metode dalam memperbaharui (*updating*) matriks istilah-dokumen tanpa harus menghitung ulang SVD, yaitu *folding-in* dan *SVD-updating*.

2.5.1 *Folding – In*

Folding-in merupakan teknik menambah dokumen atau istilah baru ke dalam matriks istilah-dokumen yang sudah diolah berdasarkan LSI. Vektor-vektor dokumen atau istilah cukup diproyeksikan ke dalam ruang *k*-dimensi (Berry, 1999).

Folding-in dokumen baru pada dasarnya sama dengan proses proyeksi kueri menjadi *pseudo-document*. Setiap dokumen baru direpresentasikan sebagai jumlah bobot dari komponen vektor istilah. Vektor yang dihasilkan tersebut kemudian dilampirkan ke dalam vektor dokumen yang sudah ada (lihat Gambar 2.7). Sehingga formula untuk *folding-in* dokumen baru adalah:

$$D' = D^T U_k \Sigma_k^{-1} \qquad (2.18)$$

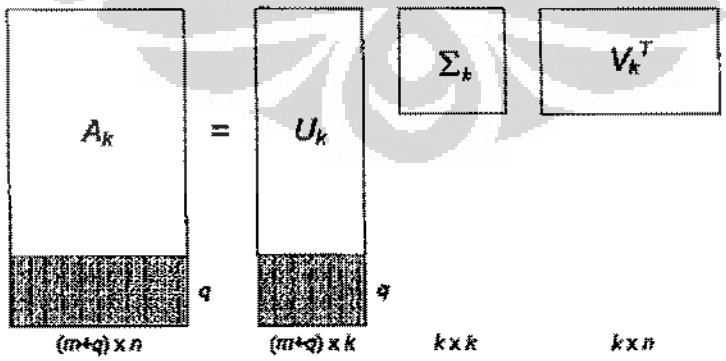


Gambar 2.7. Representasi *folding-in* p dokumen baru

Gambar 2.7. menjelaskan bahwa matriks D yang merupakan representasi koleksi dokumen baru cukup diproyeksikan ke dalam ruang k-dimensi menggunakan persamaan (2.18.). Hasilnya kemudian menggantikan vektor singular kanan V_k yang akan digunakan untuk membangun matriks aproksimasi baru. Matriks aproksimasi baru tersebut sudah mencakup penambahan p dokumen baru.

Begitu juga dengan *folding-in* istilah baru. Istilah baru dapat direpresentasikan sebagai jumlah bobot dari komponen vektor dokumen. Vektor yang dihasilkan tersebut kemudian dilampirkan ke dalam vektor istilah yang sudah ada (lihat Gambar 2.8.). Maka formula untuk *folding-in* istilah baru adalah:

$$T' = T^T V_k \Sigma_k^{-1} \qquad (2.19)$$



Gambar 2.8. Representasi *folding-in* q istilah baru

Gambar 2.8. menjelaskan bahwa matriks T yang merupakan representasi istilah-istilah baru pada koleksi dokumen cukup diproyeksikan ke dalam ruang k-dimensi menggunakan persamaan (2.19.). Hasilnya kemudian menggantikan vektor singular kiri U_k yang akan digunakan untuk membangun matriks aproksimasi baru. Matriks aproksimasi baru tersebut sudah mencakup penambahan q istilah baru.

Misal (menggunakan contoh pada matrik di halaman 22), matrik $B = \begin{pmatrix} 2 & 1 \\ 0 & 2 \\ 0 & 1 \\ 3 & 0 \end{pmatrix}$

adalah matriks yang akan ditambahkan pada matriks aproksimasi matriks A_k maka didapatkan matriks U, Σ, dan V baru, yaitu:

$$U' = \begin{pmatrix} -0.59661 & -0.045767 \\ -0.27542 & 0.20373 \\ -0.56962 & -0.6689 \\ -0.49369 & 0.71342 \end{pmatrix}$$

$$\Sigma' = \begin{pmatrix} 6.2395 & 0 & 0 & 0 \\ 0 & 3.0145 & 0 & 0 \end{pmatrix}, \text{ dan}$$

$$V' = \begin{pmatrix} -0.42428 & 0.47292 & -0.16977 & -0.048309 \\ -0.43777 & 0.57814 & -0.19514 & -0.052886 \\ -0.77529 & -0.62887 & 0.020491 & -0.037924 \\ -0.14729 & 0.2136 & 0.96571 & -0.0090921 \\ -0.074638 & 0.03162 & -0.0089941 & 0.99667 \end{pmatrix}$$

Berdasarkan ketiga matriks di atas maka didapatkan matriks aproksimasi A_k baru dengan adanya penambahan matriks yaitu

$$A_k' = \begin{pmatrix} 1.5142 & 1.5498 & 2.9728 & 0.51883 & 0.27348 \\ 1.0196 & 1.1073 & 0.94608 & 0.3843 & 0.14768 \\ 0.55439 & 0.39015 & 4.0236 & 0.092798 & 0.20152 \\ 2.324 & 2.5918 & 1.0358 & 0.91309 & 0.29792 \end{pmatrix}$$

Dengan *folding-in* dokumen atau istilah baru, ortogonalitas vektor istilah baru dan vektor dokumen baru tidak dapat dipertahankan. Bahkan jika p dokumen baru

atau q istilah baru mendekati ortogonal terhadap matriks U_k maupun V_k, banyak informasi yang berada dalam dokumen tersebut hilang akibat adanya proses proyeksi. Hal ini mengakibatkan kinerja sistem temu kembali menurun berdasarkan penelitian yang telah dilakukan oleh O'Brien (1994), Berry (1999), dan Yu Liao (2000).

2.5.2 SVD - Update

SVD-Updating adalah metode alternatif *folding-in* dengan usaha mempertahankan ortogonalitas hasil SVD. Penelitian dengan metode SVD-Update telah dilakukan oleh O'Brien (1994), Dian I Witter (1998), Michael W. Berry (1999), dan Yu Liao (2000). Untuk menambah d vektor dokumen ke dalam matriks LSI representasi koleksi dokumen yang sudah ada, maka D adalah matriks vektor dokumen (m x d) dimana m adalah banyaknya istilah. Kemudian D dilampirkan pada kolom dari matriks LSI A_k sehingga menjadi,

$$B = (A_k \mid D) \tag{2.20}$$

Dimana B adalah matriks dengan ordo m x (n + d). Dan SVD dari matriks tersebut adalah $B = U_B \Sigma_B V_B^T$.

$$U_k^T B \begin{pmatrix} V_k & \\ & I_d \end{pmatrix} = \left(\Sigma_k \mid U_k^T D\right),$$

karena $A_k = U_k \Sigma_k V_k^T$, dan jika $F = \left(\Sigma_k \mid U_k^T D\right) = U_F \Sigma_F V_F^T$, sehingga

$$B = (U_k U_F)\Sigma_F \left(\begin{pmatrix} V_k & \\ & I_d \end{pmatrix} V_F \right)^T$$

didapat,

$$U_B = U_k U_F \tag{2.21}$$

$$\Sigma_B = \Sigma_F \tag{2.22}$$

$$V_B = \begin{pmatrix} V_k & \\ & I_d \end{pmatrix} V_F \tag{2.23}$$

Jadi U_B, Σ_B, dan V_B berturut-turut adalah vektor singular kiri baru, matriks diagonal baru, dan vektor singular kanan baru. Ketiganya digunakan untuk

membangun matriks aproksimasi representasi koleksi dokumen baru yang sudah mengakomodir penambahan d dokumen baru tanpa harus melakukan proses pengindeksan ulang.

Sedangkan untuk menambah t vektor istilah ke dalam model LSI yang sudah ada, maka T adalah matriks vektor istilah (t x n) dimana n adalah banyaknya dokumen. Lalu T dilampirkan pada baris dari matriks LSI A_k sehingga menjadi,

$$C = \begin{pmatrix} A_k \\ T \end{pmatrix} \tag{2.24}$$

Dimana C adalah matriks dengan ordo $(m + t)$ x n. Dan SVD dari matriks tersebut adalah $C = U_C \Sigma_C V_C^T$.

$$\begin{pmatrix} U_k^T & \\ & I_t \end{pmatrix} C V_k = \begin{pmatrix} \Sigma_k \\ T V_k \end{pmatrix},$$

karena $A_k = U_k \Sigma_k V_k^T$, dan jika $H = \begin{pmatrix} \Sigma_k \\ T V_k \end{pmatrix} = U_H \Sigma_H V_H^T$, sehingga

$$C = \left(\begin{pmatrix} U_k & \\ & I_t \end{pmatrix} U_H \right) \Sigma_C (V_k V_H)^T$$

didapat,

$$U_C = \begin{pmatrix} U_k & \\ & I_t \end{pmatrix} U_H \tag{2.25}$$

$$\Sigma_C = \Sigma_H \tag{2.26}$$

$$V_C = V_k V_H \tag{2.28}$$

Jadi U_C, Σ_C, dan V_C berturut-turut adalah vektor singular kiri baru, matriks diagonal baru, dan vektor singular kanan baru. Ketiganya digunakan untuk membangun matriks aproksimasi representasi koleksi dokumen baru yang sudah mengakomodir penambahan q istilah baru tanpa harus melakukan proses pengindeksan ulang.

Misal (menggunakan contoh pada matrik di halaman 22), matrik $B = \begin{pmatrix} 2 & 1 \\ 0 & 2 \\ 0 & 1 \\ 3 & 0 \end{pmatrix}$,

sehingga didapat berturut-turut:

$$U_{up} = \begin{pmatrix} 0.60456 & 0.027 \\ 0.27207 & -0.18049 \\ 0.56512 & 0.6773 \\ 0.49105 & -0.71271 \end{pmatrix}$$

$$S_{up} = \begin{pmatrix} 6.2398 & 0 \\ 0 & 3.0159 \end{pmatrix}, \text{ dan}$$

$$V_{up} = \begin{pmatrix} 0.42426 & -0.47216 \\ 0.43774 & -0.57727 \\ 0.77519 & 0.62883 \\ 0.14784 & -0.21796 \\ 0.074892 & -0.029681 \end{pmatrix}$$

Berdasarkan ketiga matriks di atas maka didapatkan matriks aproksimasi A_k baru dengan adanya penambahan matriks yaitu

$$A_{kup} = \begin{pmatrix} 1.562 & 1.6043 & 2.9755 & 0.53995 & 0.92149 \\ 0.97728 & 1.0574 & 0.97373 & 0.36963 & 0.1433 \\ 0.53155 & 0.3644 & 4.018 & 0.076089 & 0.20346 \\ 2.3148 & 2.5821 & 1.0236 & 0.92149 & 0.29327 \end{pmatrix}$$

BAB 3
METODELOGI

Pada bab ketiga ini dijelaskan cara-cara yang dilakukan mulai dari pengolahan koleksi data, menyusun matriks representasi istilah-dokumen, pembobotan, membangun aproksimasi matriks SVD, hingga mekanisme penambahan koleksi dokumen. Juga dibahas mengenai kueri yang digunakan dalam percobaan.

3.1 Koleksi Data

Data yang digunakan sebagai ujicoba dalam penelitian ini adalah dokumen-dokumen hasil penelitian yang dilakukan dalam lingkungan Badan Tenaga Atom Nasional (BATAN) sebanyak 1162 dokumen. Penelitian menggunakan data BATAN karena data tersebut sudah digunakan pada penelitian-penelitian sebelumnya oleh Fitriyanti (1997), Mustangimah (1998), Ariwibowo (2001), dan Indra Budi (2002). Gambar 3.1. merupakan format dokumen masukan.

```
DOC: 0001
TIT: Judul dokumen
AUT: Nama pengarang
AFF: Nama lembaga
JOU: Nama jurnal
ABS: Abstrak dokumen
BIB: Bibliografi
1.
2.
.
.
N.
```

Gambar 3.1. Format masukan dokumen

Pada penelitian yang dilakukan oleh Indra Budi (2002) diketahui bahwa perbedaan kinerja antara sistem temu kembali yang mengindeks seluruh isi dokumen dibandingkan dengan abstrak dokumen tidak beda jauh. Oleh karena itu pada penelitian ini, proses pengindeksan hanya dilakukan pada bagian abstrak dari dokumen saja untuk efisiensi waktu dan ruang penyimpanan.

Untuk lebih memahami bagaimana penerapan STKI dengan model ruang vektor dan LSI, pada bab ini, diberikan contoh dua koleksi dokumen. Koleksi dokumen pertama akan diindeks, diberi bobot, lalu menentukan dokumen keluaran ketika diberikan kueri dengan model ruang vektor. Selanjutnya indeks koleksi dokumen tersebut didekomposisi sehingga menghasilkan indeks konseptual. Indeks konseptual tersebut kembali diberikan kueri untuk melihat dokumen keluarannya. Sedangkan koleksi dokumen kedua digunakan untuk menjelaskan bagaimana metode penambahan koleksi dokumen baru diterapkan pada penelitian ini.

Tabel 3.1. dan tabel 3.2. merupakan koleksi dokumen awal dan koleksi dokumen baru yang akan dikomputasi menggunakan model ruang vektor dan LSI. Jumlah dokumen pada koleksi awal adalah sembilan dan jumlah dokumen yang baru yang ditambahkan adalah tujuh. Kata yang dicetak tebal berarti kata tersebut digunakan sebagai indeks istilah.

Tabel 3.1. Koleksi 9 Dokumen Awal

ID	Judul
01	***Human*** *Machine* ***Interface*** *for Lab ABC Computer Applications*
02	***A Survey*** *of User Opinion of* ***Computer System Response Time***
03	*The* ***EPS USER Interface Management System***
04	***System*** *and* ***Human Systems Engineering*** *Testing of EPS-2*
05	*Relation of* ***User***-*Perceived* ***Response*** *Time to Error Measurement*
06	*The Generation of Random, Binary, Unordered* ***Tree***
07	*Intersection* ***Graph*** *of Paths in a* ***Tree***
08	***Graph Minors*** *IV: Tree-width and Well-Quasi-Ordering*
09	***Graph Minors***–*A* ***Survey***

Tabel 3.2. Koleksi 7 Dokumen yang akan ditambahkan

ID	Judul
10	*System Time to Traverse a B-Tree Graph*
11	*Interface Graph Tools*
12	*Graph Minors implemented on Computer Systems*
13	*System Tree*
14	*Computer Graph*
15	*Survey of Computer Time*
16	*A Survey of Human Interface Computer Systems*

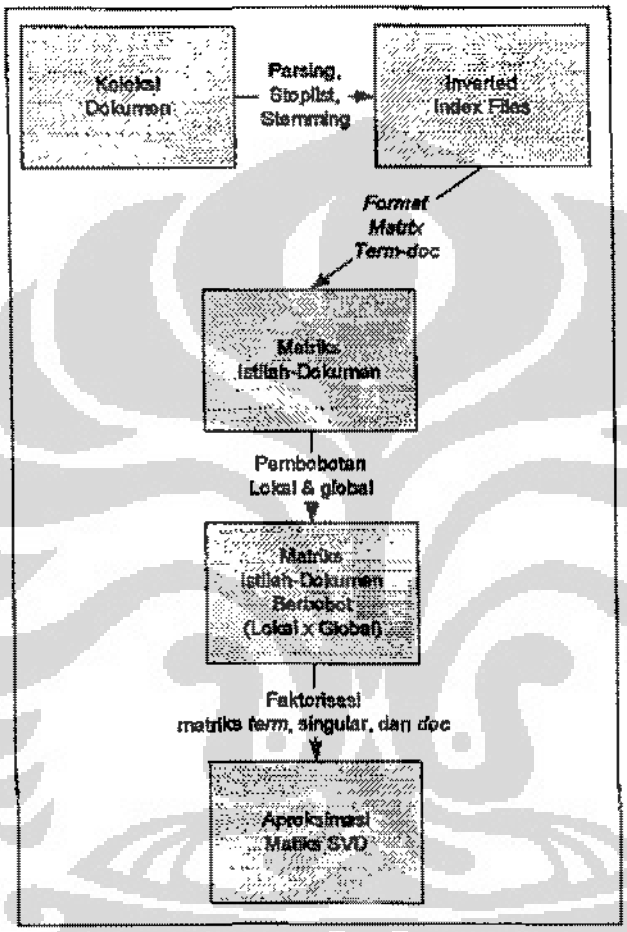
Koleksi dokumen tersebut kemudian diindeks berdasarkan kata-kata yang dicetak tebal. Tabel 3.3. merupakan matriks istilah-dokumen representasi koleksi dokumen awal. Kolom tabel tersebut merupakan representasi dokumen sedangkan barisnya adalah istilah-istilah dokumen yang ada di koleksi dokumen. Bobot suatu istilah di suatu dokumen bernilai satu (1) artinya istilah tersebut muncul satu kali dan nol (0) berarti istilah tersebut tidak ada.

Tabel 3.3. Matriks representasi koleksi dokumen

	01	02	03	04	05	06	07	08	09
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	0	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	0	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
tree	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	0	1	1
minors	0	0	0	0	0	0	0	1	1

3.2 Alur Proses Pengindeksan hingga Aproksimas Matrik SVD

Dalam mengembangkan sistem pada penelitian ini, langkah pertama yang dilakukan adalah pembuatan matriks-matriks representasi istilah-dokumen. Proses tersebut dapat terlihat pada gambar 3.2.



Gambar 3.2. Alur proses pengindeksan hingga aproksimasi matriks SVD

Langkah awal yang dilakukan adalah melakukan pengindeksan bagian abstrak dari koleksi dokumen melalui proses (*parsing*), pembuangan kata buang (*stoplist*), dan pencarian akar kata (*stemming*) menjadi format pembalikan dokumen (*Inverted-Index*). Format tersebutlah yang dijadikan masukan dalam menyusun matriks representasi istilah-dokumen lalu kemudian dibobot dengan teknik pembobotan lokal dan global pada rumus (2.10.). Untuk sistem temu kembali berbasis ruang vektor, matriks istilah-dokumen yang sudah dibobot sudah dapat

digunakan untuk proses pencarian dengan memasukkan kueri. Namun pada sistem temu kembali berbasis indeks konseptual, matriks tersebut kembali diolah dengan didekomposisi berdasarkan matriks ortogonal kiri, matriks singular, dan matriks ortogonal kanan sambil menentukan nilai-nilai singular *k* yang terbaik saja seperti pada persamaan (2.13.). Setelah didapatkan matriks-matriks tersebut maka matriks aproksimasi dapat diperoleh dari matriks awal dengan beberapa nilai rank *k* yang terbaik.

3.3 Pengindeksan

Sebagaimana yang telah dijelaskan pada bagian sebelumnya, bagian abstrak dari dokumen diindeks melalui proses *parsing*, pembuangan kata buang (*stoplist*), dan pencarian akar kata (*stemming*). Pengindeksan dilakukan dengan membalikkan dokumen yaitu pengindeksan istilah dan dokumen secara terurut (*lexicographical index*) (Frakes, 1992). Dengan demikian suatu istilah akan merujuk ke satu atau lebih dokumen, sehingga temu kembali terhadap suatu istilah juga berarti temu kembali dokumen-dokumen yang dirujuk oleh istilah tersebut. Proses pembalikan ini mengikuti apa yang telah dilakukan pada penelitian Indra Budi (2002). Adapun format pembalikan dokumen dapat terlihat pada gambar 3.3 berikut ini.

<istilah 1>
<id dokumen 1> <jumlah kemunculan istilah 1 dalam dokumen 1>
<id dokumen 2> <jumlah kemunculan istilah 1 dalam dokumen 2>
.
.
<istilah 2>
<id dokumen 1> <jumlah kemunculan istilah 2 dalam dokumen 1>
.
.

Gambar 3.3. Format pembalikan dokumen

Format pembalikan dokumen tersebut kemudian diubah ke dalam format yang dapat dibaca oleh matlab karena sistem temu kembali informasi dalam penelitian ini dibangun menggunakan matlab. Adapun contoh format baru tersebut dapat terlihat pada gambar 3.4. berikut ini.

```
4       3       12
        1       2       2.00000000
        1       3       1.00000000
        2       1       1.00000000
        3       3       2.00000000
        4       1       1.00000000
```

Gambar 3.4. Format matriks masukan program matlab

Pada baris pertama terdapat informasi berturut-turut berupa 4, 3, dan 12. 4 berarti jumlah istilah yang terdapat pada koleksi data adalah sebanyak 4. Sedangkan 3 menunjukkan jumlah dokumen yang ada pada koleksi. Kemudian 12 menunjukkan jumlah elemen yang terdapat pada matriks karena didapatkan dari 4 baris istilah dikalikan dengan 3 kolom dokumen. Pada matlab, setelah dibaca format tersebut akan membentuk matriks seperti berikut ini.

$$\begin{pmatrix} 0 & 2 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \\ 1 & 0 & 0 \end{pmatrix}$$

Kolom-kolom matriks di atas adalah representasi vektor dokumen sedangkan baris-barisnya merupakan vektor istilah. Elemen matriks tersebut berupa frekuensi kemunculan istilah di dalam dokumen.

3.4 Pembobotan

Pembobot relasi istilah-dokumen menggunakan bobot lokal $L(i, j) = \log(tf_{ij} + 1)$ dan bobot global $G(i) = 1 - \sum_j \frac{p_{ij} \log(p_{ij})}{\log(nodcs)}$, $p_{ij} = \frac{tf_{ij}}{gf_i}$. Kemudian untuk bobot setiap elemen dari matriks tersebut didapat dengan mengalikan bobot lokal dengan bobot global $a_{ij} = L(i, j) * G(i)$ seperti pada persamaan (2.10.)

dimana:

tf_{ij} merupakan frekuensi istilah i di dalam dokumen j.

gf_i adalah frekuensi global dari istilah *i*.

df_i adalah jumlah dokumen dimana istilah *i* muncul.

ndocs adalah jumlah dokumen di dalam koleksi.

Tujuan pembobotan adalah untuk menaikkan atau menurunkan kepentingan relatif suatu istilah baik di dalam sebuah dokumen maupun dokumen lainnya di dalam koleksi dokumen yaitu dengan adanya bobot lokal dan global (Yu Liao, 2000).

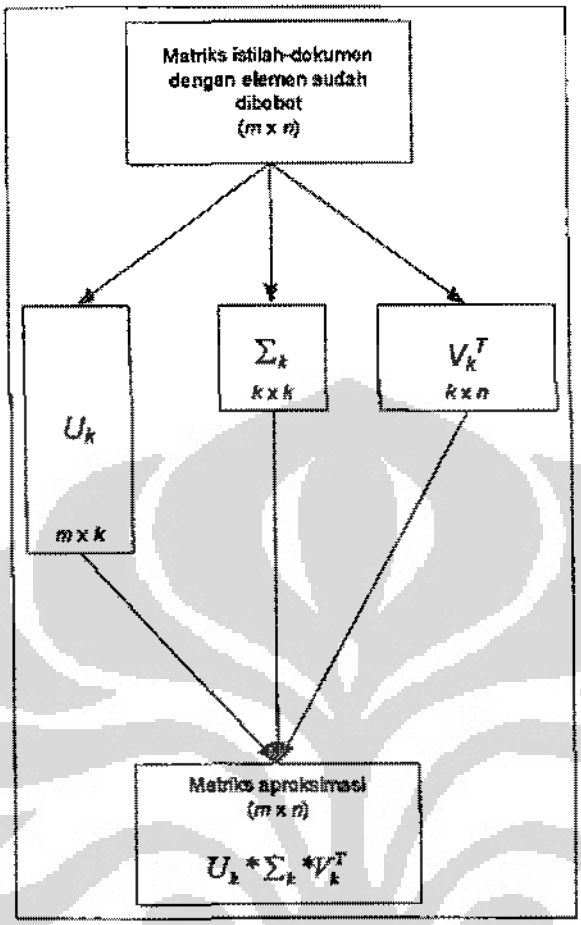
Tabel 3.4. adalah matriks istilah-dokumen koleksi dokumen awal yang telah diboboti dengan persamaan (2.10.). Untuk STKI model ruang vektor, matriks ini sudah dapat digunakan dengan memasukkan vektor kueri pemakai.

Tabel 3.4. Matriks istilah-dokumen yang telah diberi bobot

	01	02	03	04	05	06	07	08	09
human	0.396	0	0	0.396	0	0	0	0	0
interface	0.396	0	0.396	0	0	0	0	0	0
computer	0.30	0	0	0	0	0	0	0	0
user	0	0.452	0.452	0	0.452	0	0	0	0
system	0	0.396	0.396	0	0	0	0	0	0
response	0	0.396	0	0	0.396	0	0	0	0
time	0	0.396	0	0	0.396	0	0	0	0
eps	0	0	0.396	0.396	0	0	0	0	0
survey	0	0.396	0	0	0	0	0	0	0.396
tree	0	0	0	0	0	0.452	0.452	0.452	0
graph	0	0	0	0	0	0	0	0.396	0.396
minors	0	0	0	0	0	0	0	0.396	0.396

3.5 Membangun Aproksimasi Matrik SVD

Matriks istilah-dokumen yang elemennya sudah diboboti berdasarkan perkalian antara bobot lokal dan global kemudian didekomposisi dengan menggunakan *Singular Value Decomposition* (SVD). Hasil dekomposisi berupa matriks ortogonal kiri atau matriks representasi istilah, matriks singular diagonal, dan matriks ortogonal kanan atau matriks representasi dokumen.



Gambar 3.5. Proses membangun aproksimasi matriks SVD

Ambil matriks istilah-dokumen yang sudah diboboti tersebut sebagai matriks A. Kemudian jika r adalah rank dari matriks A maka kemudian dipilih rank sebanyak k terbaik. Akibatnya matriks ortogonal kiri yang pada awalnya memiliki ordo m istilah x r rank menjadi m istilah x k rank. Sedangkan matriks singular yang tadinya berdimensi m x n yang diagonalnya berupa nilai singular dari matriks A menjadi matriks dengan ordo k x k. Matriks ortogonal kanan atau matriks representasi dokumen yang semula memiliki ordo m x n menjadi k x n. Dari matriks-matriks ortogonal kiri dan kanan serta matriks singular didapatkan matriks aproksimasi A_k yang didapat dengan $A_k = U_k * \Sigma_k * V_k^T$. Gambar 3.5. menjelaskan proses pembentukan matriks aproksimasi A_k dari matriks A.

Dengan contoh koleksi dokumen yang ada, maka tabel 3.4. kemudian didekomposisi dengan SVD, sehingga didapat matriks U_k, V_k, dan Σ_k dengan rank $k = 2$. Kemudian dibentuk matriks aproksimasi baru yang merupakan hasil kali $U_k \Sigma_k V_k^T$. Matriks aproksimasi baru inilah yang disebut sebagai indeks konseptual yang digunakan pada STKI model LSI. Contoh gambar 3.5 tersebut memperlihatkan matriks-matrik U_k, V_k, Σ_k, yang membangun matriks aproksimasi A_k. Berikut adalah matrik U_k, V_k, Σ_k:

$$U_k = \begin{pmatrix} -0.052 & -0.043 \\ -0.184 & -0.077 \\ -0.021 & -0.017 \\ -0.626 & -0.093 \\ -0.388 & -0.054 \\ -0.391 & -0.028 \\ -0.391 & -0.028 \\ -0.182 & -0.074 \\ -0.272 & 0.216 \\ -0.024 & 0.625 \\ -0.056 & 0.518 \\ -0.056 & 0.518 \end{pmatrix}$$

Matriks U_k hasil dekomposisi SVD dengan rank $k = 2$

$$\Sigma_k = \begin{pmatrix} 1.210 & 0 \\ 0 & 0.953 \end{pmatrix}$$

Matriks Σ_k hasil dekomposisi SVD dengan rank $k = 2$

$$V_k = \begin{pmatrix} -0.083 & -0.706 & -0.481 & -0.077 & -0.490 & -0.009 & -0.009 & -0.045 & -0.126 \\ -0.055 & 0.000 & -0.129 & -0.049 & -0.068 & 0.296 & 0.296 & 0.726 & 0.520 \end{pmatrix}$$

Matriks V_k^T hasil dekomposisi SVD dengan rank $k = 2$

Tabel 3.5. Matriks istilah-dokumen hasil dekomposisi SVD

	01	02	03	04	05	06	07	08	09
human	0.007	0.045	0.036	0.007	0.034	-0.012	-0.012	-0.027	-0.013
interface	0.022	0.157	0.117	0.021	0.114	-0.020	-0.020	-0.043	-0.010
computer	0.003	0.018	0.014	0.003	0.013	-0.005	-0.005	-0.011	-0.006
user	0.067	0.534	0.375	0.062	0.377	-0.020	-0.020	-0.030	0.049
system	0.042	0.332	0.233	0.039	0.234	-0.011	-0.011	-0.016	0.032
response	0.041	0.334	0.231	0.038	0.234	-0.004	-0.004	0.002	0.046
time	0.041	0.334	0.231	0.038	0.234	-0.004	-0.004	0.002	0.046
eps	0.022	0.156	0.115	0.020	0.113	-0.019	-0.019	-0.041	-0.009
survey	0.016	0.232	0.132	0.015	0.147	0.064	0.064	0.165	0.149
tree	-0.031	0.020	-0.063	-0.027	-0.026	0.176	0.176	0.434	0.313
graph	-0.022	0.048	-0.031	-0.019	0	0.147	0.147	0.362	0.265
minors	-0.022	0.048	-0.031	-0.019	0	0.147	0.147	0.362	0.265

Tabel 3.5. adalah indeks konseptual yang digunakan oleh kueri pemakai untuk mencari dokumen yang dicarinya. Tabel 3.4. dan tabel 3.5. memiliki bobot setiap elemennya yang berbeda. Tabel 3.4 adakag matri sparse yang mayoritas elemennya adalah nol (0). Sedangkan pada tabel 3.5. setidaknya hampir semua elemennya memiliki bobot. Hal ini mengakibatkan jumlah dokumen yang terambil pada pengindeksan konseptual lebih banyak dibandingkan dengan pengindeksan biasa.

3.6 Peringkat Dokumen Terambil

Sebelum proses memberikan peringkat pada dokumen yang teretrieve dikerjakan, kueri pemakai diproses terlebih dahulu agar didapatkan akar kata dari setiap kata-kata yang digunakan. Kemudian dibuat sebuah vektor yang mewakili kueri pemakai tersebut. Vektor kueri disusun berdasarkan istilah-istilah yang terdapat pada koleksi dokumen yaitu pada matriks istilah-dokumen. Sebagai contoh kueri pemakai adalah "*human*", "*computer*", dan "*interaction*". Istilah "*interaction*" tidak terdapat pada indeks. Maka representasi vektor kueri q menjadi $q = (1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)$.

Baik sistem temu kembali informasi yang menggunakan model ruang vektor maupun model LSI, vektor kueri tersebut diukur kesamaannya dengan setiap dokumen pada koleksi dokumen dengan menggunakan rumus cosine seperti pada rumus (2.13.). Hasilnya kemudian diurut berdasarkan nilai terbesar hingga yang terkecil.

Tabel 3.4. adalah matriks representasi koleksi dokumen dengan bobot local x global sedangkan pada tabel 3.5. adalah matriks dengan indeks konseptual, maka keduanya akan dimasukkan kueri pemakai *q* yaitu "*human computer interaction*". Untuk STKI dengan model ruang vektor maka dokumen terambil dengan kueri *q* adalah dokumen 01 dan dokumen 04 dengan masing-masing bobotnya adalah 0.775 dan 0.500. Sedangkan untuk STKI dengan pengindeksan konseptual maka dokumen terambil dengan kueri *q* adalah dokumen 01, dokumen 04, dokumen 03, dokumen 05, dan dokumen 02 dengan masing-masing bobotnya adalah 0.0655, 0.0651, 0.0591, 0.0557, dan 0.0514. Berdasarkan dokumen terambil terlihat bahwa jumlah dokumen terambil pengindeksan konseptual lebih banyak dibandingkan dengan model ruang vektor.

3.7 Penambahan Koleksi Dokumen

Pada penelitian ini, matriks representasi koleksi dokumen yang sudah dibangun menggunakan SVD akan ditambah dengan beberapa koleksi dokumen baru. Selain langkah yang paling jelas berupa mengindeks ulang seluruh matriks istilah dokumen juga digunakan teknik *folding-in* dan *SVD-Update*.

Misalkan A adalah matriks awal yang terdiri dari m istilah dan n dokumen. Kemudian terdapat matriks B baru yang terdiri dari h istilah dan j dokumen. Untuk menambahkan matriks B pada matriks A adalah dengan membuat dua matriks baru B yang merupakan matriks dokumen baru dan matriks istilah baru. Karena dengan menambahkan matriks B pada matriks A berarti terdapat j dokumen baru dan terdapat i istilah baru yang tidak terdapat pada matriks A. Jika

Bdoc adalah matriks dokumen baru dan *Bterm* adalah matriks istilah baru maka matriks *A* ditambah matriks *B* adalah $\left(\frac{A \mid Bdoc}{Bterm}\right)$.

Matriks *Bdoc* dan matriks *Bterm* inilah yang digunakan pada setiap proses penambahan dokumen baik dengan metode *folding-in* maupun metode *SVD-Update*.

Berdasarkan tabel 3.2. maka didapatkan matriks representasi koleksi dokumen baru seperti pada tabel 3.6. berikut ini.

Tabel 3.6. Matriks koleksi dokumen baru berdasarkan tabel 3.2

	10	11	12	13	14	15	16
human	0	0	0	0	0	0	0
interface	0	0	1	0	0	0	0
computer	0	0	0	1	0	1	1
user	0	0	0	0	0	0	0
system	1	1	0	1	1	0	0
response	0	0	0	0	0	0	0
time	1	1	0	0	0	0	1
eps	0	0	0	0	0	0	0
survey	0	0	0	0	0	0	1
tree	1	1	0	0	1	0	0
graph	1	1	1	1	0	1	0
minors	0	0	0	1	0	0	0

3.7.1 *Folding-in*

Adapun *updating* menggunakan metode folding-in terdiri dari dua langkah, pertama adalah menyusun matriks representasi dokumen baru dari matriks representasi dokumen matriks awal. Kedua adalah menyusun matriks representasi istilah dari matriks representasi istilah matriks awal.

Formula *folding-in* dokumen baru sesuai dengan persamaan (2.18.) yaitu $D' = D^T U_k \Sigma_k^{-1}$ dan untuk *folding-in* istilah baru sesuai dengan persamaan (2.20.) yaitu $T' = T^T V_k \Sigma_k^{-1}$. Input dari proses *folding-in* dokumen baru adalah matriks dokumen baru (*Bdoc*), bobot global matriks lama, matriks representasi istilah (U_k), dan matriks singular ($\sum$). Output dari proses *folding-in* adalah matriks representasi dokumen baru *D*. Sedangkan pada proses *folding-in* istilah baru inputnya berupa matriks istilah baru (*Bterm*), matriks representasi dokumen baru *D* yang baru saja diolah, dan matriks singular ($\sum$). Outputnya berupa matriks representasi istilah baru *T*.

Oleh karena itu didapat matriks representasi baru dari penambahan koleksi dokumen baru dengan formula sebagai berikut:

$$A_f = T * \Sigma * D'^T \qquad (3.1.)$$

Matrik *D'* berikut adalah matrik baru menggantikan matrik singular lama.

$$D' = \begin{pmatrix}
0.05 & -0.08 & 0 & 0 & 0.01 & 0 & 0.01 & -0.01 & -0.02 \\
0.53 & -0.35 & -0.15 & -0.04 & -0.08 & -0.07 & 0 & -0.14 & -0.19 \\
0.32 & -0.34 & -0.06 & -0.01 & -0.02 & -0.02 & 0.02 & -0.09 & -0.13 \\
0.04 & -0.08 & 0 & 0 & 0.01 & 0 & 0.01 & -0.01 & -0.02 \\
0.35 & -0.30 & -0.08 & -0.02 & -0.04 & -0.03 & 0.01 & -0.09 & -0.13 \\
0.10 & 0.22 & -0.11 & -0.04 & -0.09 & -0.06 & -0.05 & -0.01 & 0 \\
0.10 & 0.22 & -0.11 & -0.04 & -0.09 & -0.06 & -0.05 & -0.01 & 0 \\
0.26 & 0.53 & -0.27 & -0.11 & -0.23 & -0.16 & -0.12 & -0.04 & 0 \\
0.25 & 0.33 & -0.22 & -0.08 & -0.18 & -0.12 & -0.09 & -0.04 & -0.02 \\
0.36 & 0.22 & 0.90 & -0.04 & -0.08 & -0.06 & -0.03 & -0.03 & -0.03 \\
0.12 & 0.10 & -0.04 & 0.99 & -0.03 & -0.02 & -0.01 & -0.01 & -0.01 \\
0.25 & 0.22 & -0.08 & -0.03 & 0.94 & -0.04 & -0.03 & -0.02 & -0.02 \\
0.19 & 0.14 & -0.06 & -0.02 & -0.04 & 0.97 & -0.02 & -0.02 & -0.01 \\
0.08 & 0.15 & -0.04 & -0.01 & -0.03 & -0.02 & 0.98 & -0.01 & 0 \\
0.19 & -0.06 & -0.03 & -0.01 & -0.02 & -0.01 & 0 & 0.98 & -0.03 \\
0.23 & -0.14 & -0.03 & -0.01 & -0.01 & -0.01 & 0 & -0.03 & 0.96
\end{pmatrix}$$

Tabel 3.5. yang merupakan matriks representasi koleksi dokumen kemudian ditambahkan 6 dokumen baru seperti yang terlihat pada tabel 3.6. Karena tidak terdapat istilah baru pada koleksi dokumen baru tersebut, maka langkah yang dilakukan adalah dengan memproyeksikan matriks koleksi dokumen baru tersebut ke ruang *k*-dimensi dengan persamaan (2.18.).

Selanjutnya dibangun matriks aproksimasi baru dari gabungan antara koleksi dokumen lama dengan koleksi dokumen baru melalui hasil kali $A_f = U_k * \Sigma_k * D^T$. Matriks A_f dengan ordo 12 x 16 inilah yang kemudian digunakan sebagai representasi koleksi dokumen baru. Tabel 3.7. adalah representasi koleksi dokumen baru tersebut dengan metode *folding-in*.

Tabel 3.7. Matriks representasi koleksi dokumen hasil *folding-in*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
human	0,01	0,04	0,04	0,01	0,03	-0,01	-0,01	-0,03	-0,01	0	0	-0,01	0	-0,01	0,01	0,02
interface	0,02	0,16	0,12	0,02	0,11	-0,02	-0,02	-0,04	-0,01	0,03	0	0,01	0,01	-0,01	0,04	0,07
computer	0	0,02	0,01	0	0,01	0	0	-0,01	-0,01	0	0	0	0	0	0	0,01
user	0,07	0,53	0,38	0,06	0,38	-0,02	-0,02	-0,03	0,05	0,17	0,04	0,09	0,08	0	0,16	0,23
system	0,04	0,33	0,23	0,04	0,23	-0,01	-0,01	-0,02	0,03	0,11	0,03	0,06	0,05	0	0,1	0,14
response	0,04	0,33	0,23	0,04	0,23	0	0	0	0,05	0,12	0,03	0,07	0,06	0,01	0,1	0,14
time	0,04	0,33	0,23	0,04	0,23	0	0	0	0,05	0,12	0,03	0,07	0,06	0,01	0,1	0,14
eps	0,02	0,16	0,12	0,02	0,11	-0,02	-0,02	-0,04	-0,01	0,03	0	0,01	0,01	-0,01	0,04	0,07
survey	0,02	0,23	0,13	0,02	0,15	0,06	0,06	0,16	0,15	0,19	0,06	0,14	0,1	0,05	0,09	0,1
tree	-0,03	0,02	-0,06	-0,03	-0,03	0,18	0,18	0,43	0,31	0,29	0,11	0,24	0,17	0,13	0,05	0,02
graph	-0,02	0,05	-0,03	-0,02	0	0,15	0,15	0,36	0,27	0,25	0,1	0,21	0,14	0,11	0,05	0,03
minors	-0,02	0,05	-0,03	-0,02	0	0,15	0,15	0,36	0,27	0,25	0,1	0,21	0,14	0,11	0,05	0,03

3.7.2 *SVD-Update*

Sama dengan proses *updating* metode *folding-in*, metode *SVD-Update* terdiri dari dua langkah yaitu menyusun matriks representasi dokumen baru kemudian menyusun matriks representasi istilah baru. Formula *SVD-Update* dokumen baru sesuai dengan persamaan (2.20. – 2.23.) dan untuk *SVD-Update* istilah baru sesuai dengan persamaan (2.24. – 2.27.).

Input dari proses *SVD-Update* dokumen baru adalah matriks dokumen baru (Bdoc), bobot global matriks lama, matriks representasi istilah (Uk), dan matriks singular ($\sum$). Output dari proses *SVD-Update* adalah matriks repersentasi dokumen baru (*Dbaru*), nilai singular baru ($\sum baru$), dan matriks repersentasi istilah baru (*Tbaru*). Sedangkan pada proses *SVD-Update* istilah baru inputnya berupa matriks dokumen baru (*Dbaru*), nilai singular baru ($\overline{\sum}$), dan matriks representasi istilah baru (*Tbaru*) yang kesemuanya merupakan output dari proses *SVD-Update* dokumen baru. Output proses *SVD-Update* adalah matriks representasi dokumen baru ($\overline{D}$), nilai singular baru ($\overline{\sum}$), dan matriks representasi istilah baru ($\overline{T}$).

Oleh karena itu didapat pula matriks representasi baru dari penambahan koleksi dokumen baru dengan formula sebagai berikut:

$$A_2 = \overline{T} * \overline{\sum} * \overline{D}^T \qquad (3.2.)$$

Sama seperti pada metode penambahan *folding-in*, tabel 3.5. yang merupakan matriks representasi koleksi dokumen kemudian ditambahkan 6 dokumen baru (tabel 3.6.). Langkah yang dilakukan adalah dengan menggunakan persamaan (2.21. – 2.23.). Jika matriks singular kiri baru adalah U_B berordo 12 x 2, matriks singular kanan baru adalah V_B berordo 16 x 9, dan matriks nilai singular diagonal baru adalah Σ_B berordo 2 x 9. Maka hasilkali representasi matriks koleksi dokumen yang baru adalah matriks A_B berordo 12 x 16 dimana $A_B = U_B \Sigma_B V_B^T$. Tabel 3.8 adalah representasi koleksi dokumen baru dengan metode *SVD-Update*.

Tabel 3.8. Matriks representasi koleksi dokumen hasil *SVD-Update*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
human	0,01	0,04	0,04	0,01	0,03	-0,01	-0,01	-0,03	-0,01	0	0	-0,01	0	-0,01	0,01	0,02
interface	0,02	0,16	0,12	0,02	0,11	-0,02	-0,02	-0,04	-0,01	0,03	0	0,01	0,01	-0,01	0,04	0,07
computer	0	0,02	0,01	0	0,01	0	0	-0,01	-0,01	0	0	0	0	0	0	0,01
user	0,07	0,53	0,38	0,06	0,38	-0,02	-0,02	-0,03	0,05	0,17	0,04	0,09	0,08	0	0,16	0,23
system	0,04	0,33	0,23	0,04	0,23	-0,01	-0,01	-0,02	0,03	0,11	0,03	0,06	0,05	0	0,1	0,14
response	0,04	0,33	0,23	0,04	0,23	0	0	0	0,05	0,12	0,03	0,07	0,06	0,01	0,1	0,14
time	0,04	0,33	0,23	0,04	0,23	0	0	0	0,05	0,12	0,03	0,07	0,06	0,01	0,1	0,14
eps	0,02	0,16	0,12	0,02	0,11	-0,02	-0,02	-0,04	-0,01	0,03	0	0,01	0,01	-0,01	0,04	0,07
survey	0,02	0,23	0,13	0,02	0,15	0,06	0,06	0,16	0,15	0,19	0,06	0,14	0,1	0,05	0,09	0,1
tree	-0,03	0,02	-0,06	-0,03	-0,03	0,18	0,18	0,43	0,31	0,29	0,11	0,24	0,17	0,13	0,05	0,02
graph	-0,02	0,05	-0,03	-0,02	0	0,15	0,15	0,36	0,27	0,25	0,1	0,21	0,14	0,11	0,05	0,03
minors	-0,02	0,05	-0,03	-0,02	0	0,15	0,15	0,36	0,27	0,25	0,1	0,21	0,14	0,11	0,05	0,03

3.8 Penilaian Relevansi

Penelitian ini memanfaatkan penilaian relevansi yang telah diterapkan oleh Mustangimah (1998). Mustangimah memakai jasa pemakai sistem *real user* untuk mengevaluasi sistem temu kembali informasi. *Real user* adalah orang yang memiliki pertanyaan atau kebutuhan informasi yaitu para peneliti yang sedang melakukan penelitian di bidang sains dan teknologi nuklir di lingkungan BATAN. Pertanyaan yang diberikan pemakai adalah masalah seputar topik yang sedang diteliti. Pencarian berdasarkan istilah pada sistem temu kembali informasi dimanfaatkan untuk mencari dokumen atau makalah yang berkaitan dengan topik penelitian.

Tingkat relevansi yang digunakan meliputi tiga kategori yaitu relevan (R), relevan marginal (RM), dan tidak relevan (TR). Tabel 3.11. menjelaskan definisi dan interpretasi dari tiga tingkat relevansi tersebut (Budi, 2002).

Tabel 3.9. Definisi dan interpretasi terhadap tingkat relevansi

Tingkat Relevansi	Definisi	Interpretasi
Relevan	Makalah/dokumen berhubungan langsung dengan subyek atau permasalahan penelitian	Saya sangat kecewa apabila sistem tidak dapat menemukan makalah/dokumen ini
Relevan Marjinal	Makalah/dokumen berhubungan dengan penelitian tetapi tidak berhubungan langsung dengan subyek atau permasalahan penelitian	Saya tidak merasa senang jika dokumen/makalah ini ditemukan atau tidak oleh sistem
Tidak Relevan	Makalah/dokumen tidak berhubungan dengan penelitian	Saya kecewa karena sistem menemukan makalah/dokumen ini

Di samping menggunakan penilaian relevansi dari penelitian Mustangimah (1998), penelitian ini juga menggunakan penilaian relevansi subyektif yang dilakukan dalam penelitian Ariwibowo (2001). Kueri yang diberikan tetap sama seperti yang dilakukan pada penelitian Mustangimah (1998). Tingkat relevansi

juga tetap menggunakan 3 kategori yaitu relevan, relevan marginal, dan tidak relevan. Hanya saja untuk menentukan suatu dokumen masuk ke dalam salah satu kategori tersebut ditentukan secara subyektif oleh Ariwibowo (2001).

BAB 4
UJICOBA DAN PEMBAHASAN

Pada bab ini dijelaskan hasil pengujian terhadap sistem dan interpretasinya dengan penilaian mengacu kepada tulisan Mustangimah (1998) dan Ariwibowo (2001) seperti yang telah dijelaskan pada subbab 3.8. Penilaian relevansi ini juga dipakai sebagai acuan pada penelitian yang dikerjakan oleh Indra Budi (2002).

Pada bagian awal bab ini dijelaskan mengenai dokumen masukan dan kueri yang digunakan serta penilaian relevansi yang digunakan. Selanjutnya dijelaskan hasil ujicoba sistem dengan menggunakan model ruang vektor dan SVD. Setelah didapatkan hasil dari setiap model, koleksi dokumen ditambah dengan 2 metode yaitu folding-in dan SVD-Update. Dari hasil ujicoba masing-masing updating tersebut kemudian dibandingkan dengan metode ruang vektor dan SVD yang diindeks secara keseluruhan. Sehingga akan diketahui kinerja dari metode folding-in dan SVD-Update jika dibandingkan dengan menghitung ulang matriks representasi istilah-dokumen baik dengan model ruang vektor maupun SVD.

Kinerja temu kembali diukur berdasarkan jumlah dokumen teraktifasi, jumlah dokumen teraktifasi yang relevan, jumlah dokumen relevan pada 20 dokumen peringkat teratas.

Pada penelitian ini, ujicoba dilakukan dengan menggunakan PC dengan spesifikasi Pentium IV 2.4 GHz dan Memory 1 GB pada sistem operasi Microsoft Windows XP Professional dan program Matlab 6.5.

4.1 Dokumen Masukan dan Kueri

Dokumen yang digunakan sebagai masukan dalam ujicoba adalah kumpulan dokumen bidang sains dan teknologi dari Badan Tenaga Atom Nasional (BATAN).

Pada penelitian ini akan dilihat bagaimana kinerja sistem temu kembali informasi pada koleksi dokumen yang terus bertambah secara progresif, sebagaimana yang dilakukan dalam penelitian O'Brien (1994). Koleksi dokumen BATAN tersebut dibagi menjadi beberapa koleksi dokumen yang berukuran lebih kecil. Terdapat dua kelompok utama dalam pembagian koleksi dokumen.

Kelompok pertama adalah kelompok yang diawali dari koleksi dokumen yang terdiri dari 300 dokumen. Koleksi dokumen tersebut kemudian ditambah dengan koleksi dokumen kecil yang terdiri dari 10 dokumen. Hasil dari penjumlahan kedua koleksi dokumen tersebut kemudian kembali ditambah dengan koleksi dokumen baru sebanyak 20 dokumen. Hasil penjumlahannya pun kembali ditambah koleksi dokumen baru secara berturut-turut yang terdiri dari 40 dokumen, 80 dokumen, dan terakhir adalah sebanyak 100 dokumen.

Sedangkan untuk kelompok kedua, koleksi dokumen awal adalah sebanyak 500 dokumen yang kemudian ditambahkan secara berturut-turut koleksi dokumen baru sebanyak 200 dokumen, 300 dokumen, dan terakhir sebanyak 162 dokumen sehingga total dokumen yang diolah sebanyak 1162 dokumen. Dokumen sebanyak itu adalah total dokumen yang terdapat pada koleksi dokumen BATAN.

Dengan penambahan koleksi dokumen secara progresif akan dilihat bagaimana efektivitas dokumen yang te-*retrieve* antara koleksi dokumen yang diindeks ulang keseluruhan dengan koleksi dokumen yang diindeks dengan metode penambahan *folding-in* dan *SVD-update*. Terlebih lagi, jumlah koleksi dokumen baru yang ditambahkan lebih dari satu kali. Dari percobaan ini akan dilihat apakah akan terjadi penurunan efektivitas yang masih dapat ditolerir atau tidak jika dibandingkan dengan penambahan koleksi dokumen dengan diindeks ulang.

Koleksi dokumen BATAN dibagi menjadi dua bagian utama, yaitu koleksi dokumen kecil (300 dokumen) dengan penambahan dokumen yang sedikit (10, 20, 40, 60, 80, dan 100 dokumen) dan koleksi dokumen besar (500 dokumen) dengan penambahan dokumen yang banyak (200, 300, dan 162 dokumen) karena

disini akan dilihat bagaimana kinerja STKI jika metode penambahan koleksi dokumen diterapkan pada koleksi dokumen berukuran kecil dan pada koleksi dokumen berukuran besar.

Tabel 4.1. dan tabel 4.2. memperlihatkan dua kelompok utama pembagian koleksi dokumen yang sudah didapatkan dalam format pembalikan dokumen. Untuk mempermudah, setiap koleksi dokumen diberi nama yaitu koleksi dokumen AA, A1, A2, A3, A4, A5, dan A6 untuk pembagian kelompok pertama serta BB, B1, B2, dan B3 untuk pembagian kelompok kedua.

Tabel 4.1. Kelompok pertama koleksi BATAN

no	Koleksi dokumen	#dokumen	#istilah	Rentang ID
1.	AA	300	3194	0001 – 0300
2.	A1	10	327	0301 – 0310
3.	A2	20	480	0311 – 0330
4.	A3	40	669	0331 – 0370
5.	A4	60	1000	0371 – 0430
6.	A5	80	1299	0431 – 0510
7.	A6	100	1496	0511 – 0610

Tabel 4.2. Kelompok kedua koleksi BATAN

no	Koleksi dokumen	#dokumen	#istilah	Rentang ID
1.	BB	500	4190	0001 – 0500
2.	B1	200	2328	0501 – 0700
3.	B2	300	3089	0701 – 1000
4.	B3	162	2136	1000 – 1162

Koleksi dokumen AA dan BB merupakan koleksi dokumen awal yang diolah menggunakan model ruang vektor dan *Latent Semantic Indexing* dengan metode

Singular Value Decomposition yang menggunakan rank $k = 50$, 75, dan 100. Dari ketiga rank tersebut akan dilihat mana yang memiliki kinerja yang paling baik. Kemudian juga dilihat perbandingan kinerja antara model ruang vektor dengan SVD.

Untuk sistem temu kembali informasi model ruang vektor, tidak ada metode penambahan khusus yang diterapkan untuk menambah koleksi dokumen. Penambahan koleksi-koleksi dokumen baru dilakukan dengan cara mengindeks ulang keseluruhan dokumen yang ada. Sedangkan untuk model LSI, penambahan koleksi dokumen dilakukan dengan dua metode yaitu *folding-in* dan *SVD-Update*.

Tabel 4.3. dan tabel 4.4. memperlihatkan nama koleksi dokumen baru yang merupakan penambahan koleksi dokumen dengan koleksi dokumen lainnya secara berturut-turut (progresif). Untuk kelompok pertama (tabel 4.3) koleksi awal berjumlah 300 dokumen kemudian koleksi bertambah berturut-turut sebanyak 10, 20, 40, 60, 80, dan 100 dokumen. Koleksi akhir kelompok pertama berjumlah 610 dokumen. Sedangkan untuk kelompok kedua (tabel 4.4) koleksi awal berjumlah 500 dokumen kemudian koleksi bertambah berturut-turut sebanyak 200, 300, dan 162 dokumen. Dari kedua kelompok tersebut akan dilihat kinerja STKI pada indeks dinamis menggunakan *folding-in* dan *SVD-Update* pada dua kelompok koleksi yang berbeda yaitu pada kelompok koleksi kecil dan kelompok koleksi besar.

Tabel 4.3. Penambahan koleksi dokumen kelompok pertama

no	Penambahan Koleksi	Koleksi dokumen	#dokumen	#istilah	Rentang ID
1.	AA + A1	AAA1	310	3262	0001 – 0310
2.	AAA1 + A2	AAA2	330	3376	0001 – 0330
3.	AAA2 + A3	AAA3	370	3532	0001 – 0370
4.	AAA3 + A4	AAA4	430	3804	0001 – 0430
5.	AAA4 + A5	AAA5	510	4235	0001 – 0510
6.	AAA5 + A6	AAA6	610	4662	0001 – 0610

Tabel 4.4. Penambahan koleksi dokumen kelompok kedua

no	Penambahan Koleksi	Koleksi dokumen	#dokumen	#istilah	Rentang ID
1.	BB + B1	BBB1	700	5069	0001 – 0700
2.	BBB1 + B2	BBB2	1000	6353	0001 – 1000
3.	BBB2 + B3	BBB3	1162	6969	0001 – 1162

Kueri yang diujicobakan merupakan kueri yang digunakan dalam penelitian Mustangimah (1998), Ariwibowo (2001), dan Indra Budi (2002). Terdapat sepuluh kueri seperti yang terlihat pada tabel 4.5. berikut ini.

Tabel 4.5. Daftar kueri yang diujicobakan

Kode Kueri	Istilah Kueri
s1	Nuklir unsur kelumit air
s2	Kristal tunggal silikon
s3	Lapisan tipis paduan logam Ti-Ni
s4	Lapisan tipis semikonduktor
s5	Pemungutan uranium pelat elemen bakar elektronis
s6	Tomografi radiografi neutron
s7	Limbah industri analisa aktivasi neutron instrumental
s8	Pengerasan permukaan bahan laser
s9	Penilaian sistem pengendalian reaktor daya
s10	Pemisahan isotop uranium

4.2 Penilaian Relevansi

Dari 1162 dokumen, 630 dokumen di antaranya telah dinilai relevansinya baik oleh *real user* yang dilakukan oleh Mustangimah (1998) maupun yang dilakukan secara subyektif oleh Ariwibowo (2001). Penilaian relevansi akan diberikan pada dokumen keluaran dari masing-masing model STKI yang diterapkan pada koleksi dokumen seperti yang tertera pada tabel 4.3. dan tabel 4.4.

Masing-masing koleksi dokumen akan diujicobakan dengan model ruang vektor, SVD, dan penambahan koleksi dokumen secara progresif. Kemudian dilihat relevansi masing-masing dokumen keluaran tersebut berdasarkan 630 dokumen yang telah dinilai relevansinya sehingga kinerja masing-masing model dapat diperbandingkan.

Penilaian relevansi dokumen secara subjektif dilakukan dengan melihat kesesuaian antara judul dan abstrak dokumen dengan konsep dan pertanyaan kueri. Penilaian kesesuaian isi dokumen dengan konsep kueri dilakukan tanpa melibatkan pakar dalam bidang nuklir. Karena tidak melibatkan orang yang ahli dalam bidang nuklir, penilaian kesesuaian isi dokumen dilakukan dengan mencari kata kunci yang diperkirakan menjadi bahasan pokok dari penelitian. Kata kunci ini dibandingkan dengan konsep dan pertanyaan yang mewakili tiap kueri (Ari Wibowo, 2001).

Apabila isi dokumen (yang diwakili oleh kata kunci) memiliki kesesuaian dengan konsep kueri, maka dokumen tersebut dinilai Relevan secara Subjektif (S-R). Bila kata kunci dokumen tidak memiliki kaitan langsung dengan konsep kueri, maka dokumen tersebut dinilai Relevan Marjinal secara Subjektif (S-RM). Selain itu, dokumen dinilai Tidak Relevan secara Subjektif (S-TR) (Ari Wibowo, 2001).

Pada penelitian ini, dokumen yang dinilai sebagai relevan (R), relevan marginal (RM), subjektif-relevan (S-R), maupun subjektif-relevan marginal (S-RM) dianggap sebagai relevan.

Sebagaimana telah disebutkan pada awal subbab ini, baru sebagian saja yaitu sebanyak 630 dokumen dari 1162 dokumen yang telah dinilai relevansinya baik yang dilakukan Mustangimah maupun Ariwibowo maka penilaian performa sistem temu kembali informasi dengan *recall/precission* tidak dapat dilakukan secara komplit. Sebagai alternatif, penulis mengajukan istilah penduga presisi yaitu banyak dokumen relevan dibandingkan dengan jumlah dokumen yang

terambil. Hal ini dari semua dokumen yang terambil tidak semuanya dapat dinilai relevansinya diakibatkan belum komplitnya penilaian relevansi terhadap seluruh koleksi dokumen.

4.3 Tahapan Ujicoba Koleksi Dokumen

Permasalahan yang diajukan pada penelitian ini adalah mengenai istilah yang karena konsepnya kadang memiliki sinonim dan polisemi. Dengan pengindeksan konseptual maka sebuah dokumen yang awalnya tidak memiliki bobot terhadap suatu istilah akan memiliki bobot hal ini berakibat dokumen tersebut dapat terambil jika suatu kueri diberikan. Dampaknya adalah jumlah dokumen yang terambil relatif lebih banyak dibandingkan dengan pengindeksan biasa. Sehingga secara instituisi dengan adanya pengindeksan konseptual maka terjadi perluasan kueri (*query expansion*).

Ujicoba tahap awal yang dilakukan adalah dengan membandingkan ujicoba model ruang vektor (VSM) dengan model *Latent Semantic Indexing* menggunakan *Singular Value Decomposition*. Akan dilihat kinerja kedua model tersebut berdasarkan penduga presisi peringkat 20 teratas dokumen yang terambil, penduga presisi semua dokumen yang terambil, dan jumlah dokumen yang teraktivasi.

Pada model LSI dengan SVD, digunakan rank k = 50, 75, dan 100 sebagai percobaan untuk melihat mana yang menghasilkan kinerja terbaik dari ketiga rank tersebut. Sebagaimana telah disebutkan pada Bab II Subbab 2.4.4., perlu dilakukan lagi penelitian lebih mendalam guna menentukan rank optimal dalam mendekomposisi matriks menggunakan SVD (killicorn, 2003).

Tahapan selanjutnya adalah ujicoba pada koleksi dokumen dinamis. Artinya terjadi penambahan koleksi dokumen secara progresif sebagaimana yang telah dilakukan pada penelitian O'Brien (1994). Uji coba ini akan dilihat bagaimana efektivitas dan efisiensi penambahan koleksi dokumen menggunakan metode *folding-in* dan *SVD-Update*. Efektivitas sistem dilihat dari penduga presisi

peringkat 20 teratas dokumen yang terambil, penduga presisi semua dokumen yang terambil, dan jumlah dokumen yang teraktivasi. Perbandingan dilakukan terhadap koleksi dokumen yang diindeks ulang secara keseluruhan dan koleksi dokumen yang diindeks menggunakan metode *folding-in* dan *SVD-Update*.

4.4 Ujicoba Model VSM dan SVD

Pada ujicoba ini, jumlah dokumen yang digunakan adalah sebanyak 1162 dokumen atau keseluruhan dokumen yang ada pada koleksi BATAN. Hal ini dilakukan guna melihat secara keseluruhan kinerja model VSM dan SVD pada keseluruhan dokumen.

Sebagai contoh awal diberikan kueri s1 terhadap koleksi BATAN menggunakan model VSM maka didapatkan dokumen-dokumen:

- Dokumen 219 dengan bobot 0.28136
- Dokumen 47 dengan bobot 0.23799
- Dokumen 128 dengan bobot 0.23468
- Dokumen 183 dengan bobot 0.21593
- Dokumen 239 dengan bobot 0.19535
- Dokumen 159 dengan bobot 0.15752

Dan seterusnya hingga Dokumen 220 dengan bobot 0.044697 yang merupakan dokumen ke-75 atau dokumen terakhir yang terambil. Sehingga kueri s1 terhadap 1162 dokumen BATAN terambil 75 dokumen di mana berdasarkan dataset jumlah dokumen relevannya adalah sebanyak 2 dokumen.

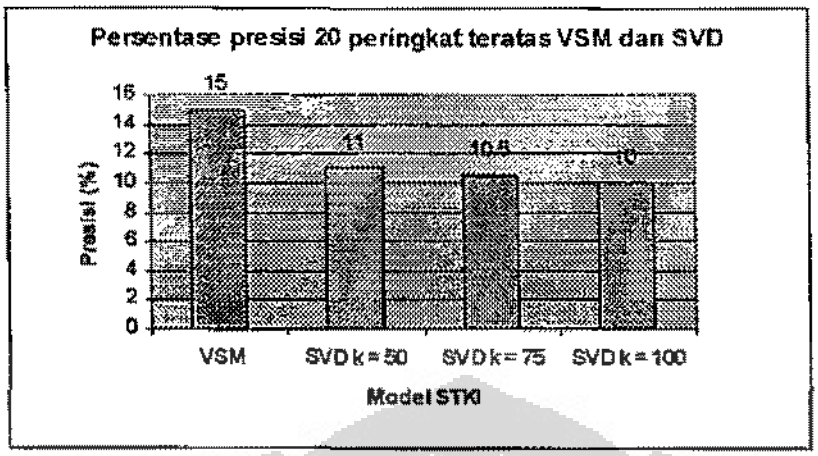
Tabel 4.6. adalah penduga presisi peringkat 20 teratas ujicoba pada 1162 dokumen koleksi BATAN menggunakan model VSM dan SVD.

Tabel 4.6. Penduga presisi peringkat 20 teratas model VSM dan SVD

metode	VSM	SVD		
		k=50	k=75	k=100
#dokumen	1162			
kueri	p20 (%)	p20 (%)	p20 (%)	p20 (%)
s1	15	15	10	10
s2	5	0	0	0
s3	15	5	10	5
s4	20	20	20	20
s5	5	10	10	10
s6	25	25	25	25
s7	15	0	0	0
s8	5	10	5	5
s9	40	20	20	20
s10	5	5	5	5
rata-rata	15	11	10.5	10

Berdasarkan tabel tersebut model VSM mengungguli model SVD baik yang menggunakan rank $k = 50$, 75, maupun 100. Kueri s1, s2, s3, s4, s5, s6, s7, s8, s9, dan s10 diujicobakan terhadap 1162 dokumen BATAN. Kueri s6 memiliki penduga presisi tertinggi yaitu sebesar 25% baik untuk ujicoba VSM maupun LSI. Rata-rata penduga presisi peringkat 20 teratas model VSM sebesar 15%. Sedangkan rata-rata penduga presisi peringkat 20 teratas model SVD untuk $k = 50$ adalah sebesar 11%, $k = 75$ sebesar 10.5%, dan $k = 100$ sebesar 10%. Artinya pengindeksan konseptual untuk data BATAN pada percobaan ini optimal ketika digunakan nilai singular sebanyak 50 atau $k = 50$.

Berdasarkan percobaan 1162 dokumen BATAN pemakai STKI cenderung lebih senang menggunakan metode VSM dikarenakan pada pemakai akan menemukan informasi yang dicarinya pada dokumen-dokumen awal yang terambil. Sedangkan pada SVD, hasil yang didapat dari 20 dokumen terambil hanya terdapat 2 dokumen yang relevan. Gambar 4.1 adalah grafik persentase presisi 20 peringkat teratas model VSM dan SVD pada 1162 dokumen koleksi BATAN.



Gambar 4.1. Grafik presisi 20 peringkat teratas VSM dan SVD

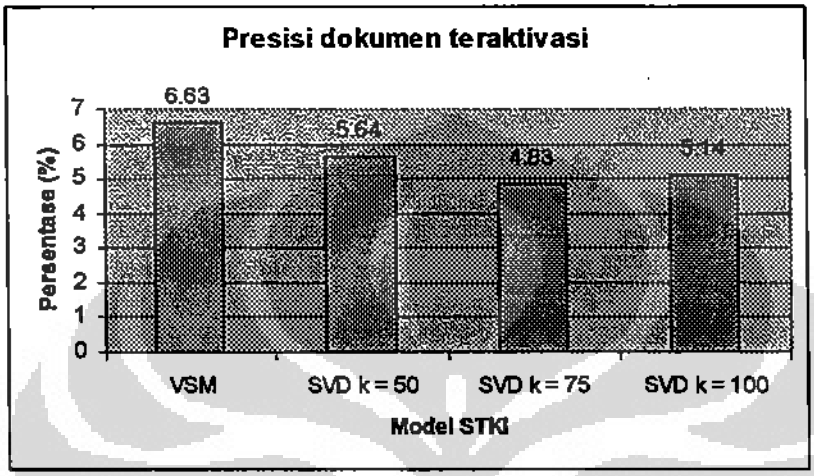
Tabel 4.7 adalah presisi dokumen terambil ujicoba pada 1162 dokumen koleksi BATAN menggunakan model VSM dan SVD.

Tabel 4.7. Penduga presisi dokumen terambil model VSM dan SVD

metode	VSM	SVD		
		k=50	k=75	k=100
#dokumen	1162			
kueri	p (%)	p (%)	p (%)	p (%)
s1	4.00	3.57	3.26	3.40
s2	7.69	0.00	0.00	0.00
s3	7.55	7.41	4.76	7.84
s4	15.79	17.86	12.77	12.50
s5	3.50	2.58	2.72	2.91
s6	5.56	5.15	5.26	5.21
s7	9.60	9.94	9.43	9.21
s8	2.68	2.03	2.02	2.11
s9	8.08	6.33	6.56	6.62
s10	1.89	1.55	1.54	1.56
rata-rata	6.63	5.64	4.83	5.14

Berdasarkan tabel 4.7 terlihat bahwa model VSM kembali mengungguli model SVD. Dari semua dokumen yang terambil model VSM memiliki rata-rata 6.63% dokumen yang relevan. Bahkan rata-rata penduga presisi model SVD berada di bawah 6% yaitu 5.63% untuk rank $k = 50$, 4.48% untuk rank $k = 75$, dan 5.14% untuk rank $k = 100$. Hal ini berarti walau dokumen yang terambil dengan metode SVD jumlahnya banyak namun jumlah dokumen yang relevan tidak bertambah

banyak dibandingkan dengan metode VSM. Akibatnya adalah terjadi penurunan presisi sebagaimana terlihat pada Gambar 4.2. Gambar 4.2. adalah grafik persentase presisi dokumen teraktivasi 1162 dokumen koleksi BATAN model VSM dan SVD.



Gambar 4.2. Grafik presisi dokumen terambil model VSM dan SVD

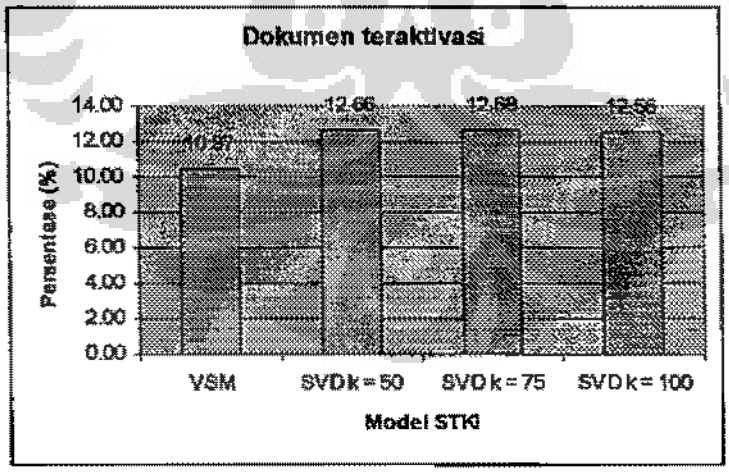
Dengan jumlah dokumen BATAN yang masih terbatas penilaian relevansinya maka jika dilihat dari tabel 4.6 dan tabel 4.7 kinerja STKI berdasarkan penduga presisi peringkat 20 teratas dan penduga presisi dokumen terambil, kinerja VSM lebih baik dibandingkan dengan SVD. Artinya pemakai STKI cenderung menyukai model VSM karena mereka akan menemukan informasi yang dicarinya pada dokumen-dokumen awal yang terambil dibandingkan dengan model SVD. Hal ini disebabkan jumlah dokumen yang teraktivasi pada model SVD lebih banyak dibandingkan model VSM sedangkan jumlah dokumen relevan berdasarkan data BATAN tidak bertambah.

Tabel 4.8 berikut ini adalah persentase dokumen teraktivasi pada 1162 dokumen koleksi BATAN menggunakan model VSM dan SVD.

Tabel 4.8. Persentase dokumen teraktivasi model VSM dan SVD

metode	VSM	SVD		
		k=50	k=75	k=100
#dokumen	1162			
kueri	ak (%)	ak (%)	ak (%)	ak (%)
s1	15.06	19.28	18.50	17.73
s2	1.12	1.46	1.55	3.01
s3	4.56	2.32	3.61	4.39
s4	3.27	2.41	4.04	4.13
s5	12.31	16.70	15.83	14.80
s6	7.75	8.35	8.18	8.26
s7	10.76	13.86	13.68	13.08
s8	12.82	16.95	17.04	16.35
s9	22.38	28.57	27.54	27.28
s10	13.68	16.70	16.78	16.52
rata-rata	10.37	12.66	12.68	12.56

Berdasarkan tabel 4.8, persentase rata-rata dokumen teraktivasi SVD mengungguli model VSM. Rata-rata dokumen teraktivas SVD dengan rank $k = 50$ adalah 12.66%, 12.68% untuk rank $k = 75$, dan 12.56% untuk rank $k = 100$. Artinya jumlah dokumen yang terambil model SVD lebih banyak dibandingkan dengan model VSM.



Gambar 4.3. Grafik dokumen teraktivasi model VSM dan SVD

Berdasarkan ujicoba yang dilakukan pada 1162 dokumen koleksi BATAN menggunakan model VSM dan SVD, pada penelitian ini, ternyata kinerja VSM

lebih baik dibandingkan dengan model SVD. Dengan VSM pemakai akan menemukan lebih banyak informasi yang dicarinya pada dokumen yang terambil dibandingkan dengan model SVD. Lebih jauh lagi, jumlah dokumen yang terambil model VSM lebih sedikit dibandingkan model SVD yang mengakibatkan pemakai tidak perlu mencari terlalu banyak dokumen yang terambil.

Namun, pernyataan di atas kemungkinan dapat berubah seandainya 1162 dokumen koleksi BATAN telah dinilai relevansinya terhadap keseluruhan dokumen pada setiap kueri.

Pada penelitian ini, jumlah dokumen teraktivasi pada model SVD lebih banyak dibanding model VSM dikarenakan adanya persamaan konseptual antara kueri yang dimasukkan dengan istilah-istilah pada koleksi dokumen. Hal ini berbeda dengan model VSM yang memcocokkan keyword pada kueri dengan istilah pada indeks secara leksikal. Akibatnya dengan model SVD terjadi perluasan kueri (*query expansion*).

4.5 Ujicoba Pada Koleksi Dokumen Dinamis

Ujicoba ini akan membandingkan antara pengindeksan ulang keseluruhan koleksi dokumen dengan pengindeksan menggunakan metode *folding-in* dan *SVD-Update*. Ujicoba dilakukan terhadap dua kelompok koleksi dokumen seperti yang diperlihatkan pada tabel 4.3 dan tabel 4.4. Model VSM dan SVD dengan pengindeksan total dijadikan sebagai acuan bagi metode *folding-in* dan *SVD-Update* terhadap kelompok koleksi dokumen yang ada. Pada percobaan ini akan dilihat bagaimana pengaruh metode *folding-in* dan *SVD-Update* terhadap kelompok dokumen yang kecil dan kelompok dokumen yang besar.

4.5.1 Ujicoba Kelompok Pertama

Pada ujicoba kelompok pertama, 300 dokumen koleksi BATAN, koleksi AA, dengan ID 0001 hingga ID 0300 diindeks menggunakan VSM dan SVD masing-masing dengan rank k = 50, 75, dan 100 (lihat tabel 4.3). Kemudian dilakukan

penambahan dokumen-dokumen baru terhadap koleksi AA tersebut. Koleksi yang akan ditambahkan adalah koleksi A1, A2, A3, A4, A5, dan A6. Langkah pertama adalah menambah koleksi A1 ke koleksi AA, hasil penambahan koleksi baru tersebut adalah koleksi AAA1. Selanjutnya penambahan koleksi A2. Koleksi A2 tidak ditambahkan ke koleksi AA melainkan ke koleksi AAA1 yang merupakan koleksi baru hasil komputasi sebelumnya. Dan seterusnya hingga penambahan koleksi A6 ke koleksi AAA5. Oleh karena itu, penambahan koleksi ini merupakan penambahan progresif. Ditujukan untuk melihat sejauh mana efektivitas STKI dalam menemukan dokumen relevan tanpa harus melakukan pengindeksan ulang dan dilakukan pada jumlah dokumen yang relatif kecil.

Hasil ujicoba yang dicatat adalah persentase penduga presisi 20 peringkat teratas, persentase penduga presisi dokumen teraktivasi, dan persentase dokumen teraktivasi. Ujicoba SVD hanya dilakukan pada rank $k = 50$. Karena pada ujicoba perbandingan antara VSM dan SVD (subbab 4.4), terlihat bahwa dekomposisi SVD terhadap matriks istilah dokumen kinerja terbaiknya diperoleh ketika rank $k = 50$.

Tabel 4.9 adalah penduga presisi 20 peringkat teratas hasil ujicoba pada koleksi AA dengan menggunakan VSM dan SVD. Koleksi AA inilah yang dijadikan indeks acuan sebelum indeks ditambah dengan koleksi dokumen lainnya.

Tabel 4.9. Penduga presisi peringkat 20 teratas koleksi AA

metode	VSM			SVD		
#dokumen	300					
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	2.86	11.67	5	2.70	12.33
s2	0	0.00	1.33	5	5.88	5.67
s3	5	8.33	4.00	5	4.00	8.33
s4	0	0.00	2.66	0	0.00	3.67
s5	0	0.00	11.67	0	0.00	15.33
s6	10	11.76	5.67	10	11.76	5.67
s7	20	16.67	10.00	15	14.71	11.33
s8	10	7.69	13.00	10	6.25	16.00
s9	15	6.98	14.33	15	5.77	17.33
s10	0	0.00	15.33	0	0.00	19.00
rata-rata	6.5	5.43	8.97	6.5	5.11	11.47

Berdasarkan tabel 4.9, kinerja model VSM dan SVD tidak terpaut terlalu jauh. Hanya saja jumlah dokumen teraktivasi model SVD lebih banyak yaitu sebesar 11.47% dibandingkan dengan model VSM yang hanya 8.97%. Hal ini sesuai dengan hasil percobaan ujicoba pada semua koleksi dokumen di mana model SVD selalu menghasilkan jumlah dokumen teraktivasi lebih banyak. Mengingat jumlah dokumen yang diujicoba masih sedikit yaitu 300 dokumen maka hasil ujicoba tidak terlampau beda.

Ujicoba selanjutnya adalah menambahkan koleksi A1 pada koleksi AA dengan pengindeksan ulang VSM dan SVD, serta metode penambahan *folding-in* dan *SVD-Update*.

Tabel 4.10. Hasil ujicoba koleksi AAA1

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	310											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	2.78	11.61	5	2.44	13.23	5	2.63	12.26	5	2.63	12.26
s2	0	0.00	1.29	5	6.25	5.16	5	5.88	5.48	5	5.88	5.48
s3	5	8.33	3.87	5	5.88	5.48	5	3.70	6.71	5	3.85	6.39
s4	0	0.00	2.58	0	0.00	3.55	0	0.00	3.87	0	0.00	3.87
s5	0	0.00	11.29	0	0.00	14.52	0	0.00	14.84	0	0.00	14.84
s6	10	11.76	5.48	10	11.11	5.81	10	11.76	5.48	10	11.76	5.48
s7	20	16.30	10.00	15	15.15	10.65	15	13.51	11.94	15	13.51	11.94
s8	10	7.69	12.58	10	5.88	16.45	10	5.88	16.45	10	5.88	16.45
s9	15	6.98	13.87	15	5.46	17.74	15	5.66	17.10	15	5.66	17.10
s10	0	0.00	15.16	0	0.00	19.03	0	0.00	18.39	0	0.00	18.39
rata-rata	6.5	5.38	8.77	6.5	5.22	11.16	6.5	4.90	11.45	6.5	4.82	11.42

Tabel 4.10 adalah hasil ujicoba pada koleksi AAA1 baik yang dilakukan pengindeksan ulang dengan VSM dan SVD maupun yang dilakukan secara dinamis yaitu dengan *folding-in* dan *SVD-Update*. Data yang ditampilkan pada tabel 4.10 adalah persentase penduga presisi peringkat 20 besar, persentase penduga presisi dokumen terambil, dan persentase dokumen teraktivasi.

Pada pengindeksan ulang model VSM dan SVD kinerja yang dihasilkan tidak terpaut jauh mengingat jumlah dokumen yang diujicobakan masih sedikit. Namun

sekali lagi model SVD memiliki persentase penduga presisi yang lebih kecil dibandingkan model VSM diakibatkan adanya perluasan kueri sehinga dokumen yang teraktivasi lebih banyak.

Kemudian ketika dilakukan pengindeksan dinamis secara konseptual, kinerja yang dihasilkan menurun namun tidak terlalu signifikan. Hal ini terlihat baik pengindeksan secara *Folding-in* maupun *SVD-Update*. Namun berdasarkan tabel 4.10 hasil ujicoba, model *SVD-Update* lebih baik dibanding model *Folding-in*, mengingat model *SVD-Update* berupaya mempertahankan ortogonalitas indeks dokumen yang sudah ada. Sehingga hasilnya baik dari persentase penduga presisi maupun jumlah dokumen teraktivasi *SVD-Update* lebih baik.

Selanjutnya 310 dokumen yang sudah diujicoba tersebut dilakukan pengindeksan kembali baik dengan pengindeksan ulang maupun pengindeksan dinamis. Hasilnya dapat dilihat pada tabel 4.11.

Tabel 4.11. Hasil ujicoba koleksi AAA2

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	330											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	0	2.50	12.12	5	2.04	14.85	5	2.33	13.03	5	2.33	13.03
s2	5	20.00	1.52	5	7.69	3.84	5	5.88	5.15	5	5.88	5.15
s3	5	7.69	3.84	5	6.25	4.85	5	3.57	8.48	5	3.70	8.18
s4	0	0.00	2.73	0	0.00	4.55	0	0.00	3.64	0	0.00	3.64
s5	0	0.00	12.12	0	0.00	15.15	0	0.00	14.85	0	0.00	14.85
s6	10	10.53	5.76	10	10.00	6.06	10	10.53	5.76	10	10.53	5.76
s7	20	15.15	10.00	15	14.29	10.61	15	12.82	11.82	15	12.82	11.82
s8	10	6.82	13.33	10	5.66	18.06	10	5.26	17.27	10	5.45	16.67
s9	25	10.20	14.85	25	7.94	19.09	25	8.62	17.58	25	8.62	17.58
s10	0	0.00	14.85	0	0.00	17.88	0	0.00	17.88	0	0.00	17.88
rata-rata	7.5	7.29	9.12	7.5	5.39	11.30	7.5	4.90	11.55	7.5	4.93	11.45

Berdasarkan tabel 4.11. rata-rata persentase penduga presisi peringkat 20 besar untuk semua metode tidak ada yang berbeda, semuanya bernilai 7,5%. Namun secara penduga presisi terjadi penurunan pada pengindeksan dinamis baik *Folding-in* maupu *SVD-Update* dikarenakan jumlah dokumen teraktivasinya

lebih banyak jika dibandingkan dilakukan pengindeksan ulang. Secara keseluruhan penurunan kinerja tidak terlalu berpengaruh dikarenakan seorang user biasanya lebih tertarik pada dokumen-dokumen terambil yang awal-awal dan cenderung mengabaikan dokumen terambil di bagian akhir. Sedangkan persentase penduga presisi peringkat 20 besar nilai sama semua untuk semua metode yaitu 7,5%.

Tabel 4.12. Hasil ujicoba AAA3

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	370											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	0	3.85	14.05	5	1.72	15.68	5	1.82	14.86	5	1.82	14.86
s2	5	20.00	1.35	5	7.69	3.51	5	5.56	4.86	5	5.56	4.86
s3	10	13.33	4.05	5	5.88	9.19	10	6.25	8.65	5	6.67	8.11
s4	0	0.00	2.43	0	0.00	2.16	0	0.00	3.24	0	0.00	3.24
s5	5	2.13	12.70	5	1.79	15.14	5	1.79	15.14	5	1.79	15.14
s6	10	9.09	5.95	10	8.70	6.22	10	9.52	5.68	10	9.52	5.68
s7	30	18.42	10.27	15	16.28	11.62	15	15.56	12.16	15	15.56	12.16
s8	10	6.25	12.97	10	4.84	16.76	10	4.62	17.57	10	4.76	17.03
s9	25	9.09	14.66	25	6.76	20.00	25	7.46	18.11	25	7.46	18.11
s10	0	0.00	15.14	0	0.00	16.49	0	0.00	17.84	0	0.00	17.84
rata-rata	9.5	8.22	9.38	8	5.37	11.68	8.5	5.26	11.81	8	5.31	11.70

Selanjutnya tabel 4.12 adalah hasil ujicoba ketika koleksi ditambahkan kembali sebanyak 40 dokumen. Hasil yang didapat menunjukkan terjadi penurunan kinerja pada pengindeksan dinamis sebesar 1% hingga 1,5%. Penurunan terbesar terjadi pada persentase rata-rata penduga presisi peringkat 20 besar model *SVD-Update* yaitu sebesar 1,5% menjadi hanya 8% dari 9,5% pada pengindeksan biasa dengan model VSM. Seperti pada hasil ujicoba pada tabel 4.11. penurunan penduga presisi diakibatkan meningkatnya jumlah dokumen yang teraktivasi khususnya pada model pengindesan konseptual baik yang dilakukan secara ulang dari awal maupun secara dinamis.

Tabel 4.13. Hasil ujicoba AAA4

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	430											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	0	3.03	15.35	5	1.32	17.67	5	1.45	16.05	5	1.47	15.81
s2	5	12.50	1.86	0	0.00	5.12	5	5.26	4.42	5	5.26	4.42
s3	10	13.33	3.49	10	5.88	7.91	10	5.71	6.14	10	6.06	7.67
s4	0	0.00	2.33	0	0.00	2.33	0	0.00	3.26	0	0.00	3.26
s5	5	3.92	11.86	5	3.08	15.12	5	1.67	13.95	5	1.64	14.19
s6	10	9.09	5.12	10	8.70	5.35	10	9.09	5.12	10	9.52	4.88
s7	30	17.50	9.30	20	16.67	9.77	15	14.58	11.16	15	14.58	11.16
s8	10	5.56	12.66	10	4.05	17.21	10	3.95	17.67	10	4.05	17.21
s9	25	8.20	14.19	15	5.95	19.53	25	6.49	17.91	25	6.49	17.91
s10	0	0.00	15.35	0	0.00	17.67	0	0.00	18.84	0	0.00	18.84
rata-rata	9.5	7.31	9.14	7.5	4.56	11.77	8.5	4.82	11.65	8.5	4.91	11.53

Selanjutnya tabel 4.13 adalah hasil ujicoba ketika koleksi ditambahkan kembali sebanyak 50 dokumen. Hasil yang didapat menunjukkan terjadi penurunan kinerja pada pengindeksan dinamis sebesar 1%. Penurunan terbesar terjadi pada persentase rata-rata penduga presisi peringkat 20 besar model *SVD-Update* yaitu sebesar 1,5% menjadi 8,5% dari pada pengindeksan biasa dengan model VSM. Seperti pada hasil ujicoba pada tabel 4.11. dan tabel 4.12. penurunan persentase penduga presisi diakibatkan meningkatnya jumlah dokumen yang teraktivasi khususnya pada model pengindesan konseptual baik yang dilakukan secara ulang dari awal maupun secara dinamis. Namun penurunan yang terjadi tidak terlalu signifikan mengingat rata-rata 20 dokumen terambil peringkat atas dengan semua model masih terdapat dokumen yang relevan. Berdasarkan tabel di atas, model *SVD-Update* memiliki kinerja lebih baik dibandingkan model *Folding-in* hal ini ditunjukkan baik dari rata-rata persentase penduga presisi peringkat 20 besar dokumen terambil dan rata-rata persentase penduga presisinya.

Tabel 4.14. Hasil ujicoba AAAS

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	510											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	2.47	15.68	10	2.04	19.22	10	2.22	17.65	10	2.27	17.25
s2	5	12.50	1.57	0	0.00	3.33	5	4.17	4.71	5	4.55	4.31
s3	10	12.50	3.14	10	5.71	6.86	10	4.44	8.82	10	5.00	7.84
s4	0	0.00	2.16	0	0.00	3.53	0	0.00	2.75	0	0.00	2.75
s5	5	3.33	11.76	5	2.70	14.51	5	1.45	13.53	5	1.47	13.33
s6	10	7.69	5.10	10	7.14	5.49	10	7.41	5.29	10	7.69	5.10
s7	25	17.78	8.82	15	15.69	10.00	15	14.29	10.98	15	14.55	10.78
s8	10	4.64	12.16	10	3.61	16.27	10	3.33	17.65	10	3.45	17.06
s9	25	7.78	17.65	20	6.14	22.35	25	6.54	20.98	30	6.73	20.39
s10	0	0.00	14.51	0	0.00	17.45	0	0.00	17.45	0	0.00	17.45
rata-rata	9.5	6.89	9.28	8	4.30	11.90	9	4.39	11.98	9.5	4.57	11.63

Selanjutnya tabel 4.14 adalah kembali hasil ujicoba ketika koleksi ditambahkan kembali sebanyak 80 dokumen. Hasil yang didapat menunjukkan terjadi penurunan kinerja pada pengindeksan dinamis sebesar 0,5% untuk pengindeksan dinamis. Hal sebaliknya menunjukkan bahwa terjadi penurunan 2% untuk rata-rata persentase penduga presisi peringkat 20 besar untuk pengindeksan ulang model SVD dibandingkan model VSM. Ketika dilakukan pengindeksan dinamis baik *Folding-in* maupun *SVD-Update* kinerjanya dihasilkan tidak menurun terlalu jauh. Namun sebagaimana hasil percobaan sebelumnya jumlah dokumen teraktivasi memang lebih banyak dibandingkan dengan pengindeksan ulang dengan VSM. Hal ini sesuai dengan konsep perluasan kueri (*query expansion*). Kembali terlihat bahwa model pengindeksan dinamis menggunakan *SVD-Update* lebih baik dibandingkan *Folding-in* hal ini terlihat dari rata-rata persentase penduga presisi peringkat 20 besar *SVD-Update* tidak mengalami penurunan atau sama dengan model VSM sebesar 9,5%.

Tabel 4.15. Hasil ujicoba AAA6

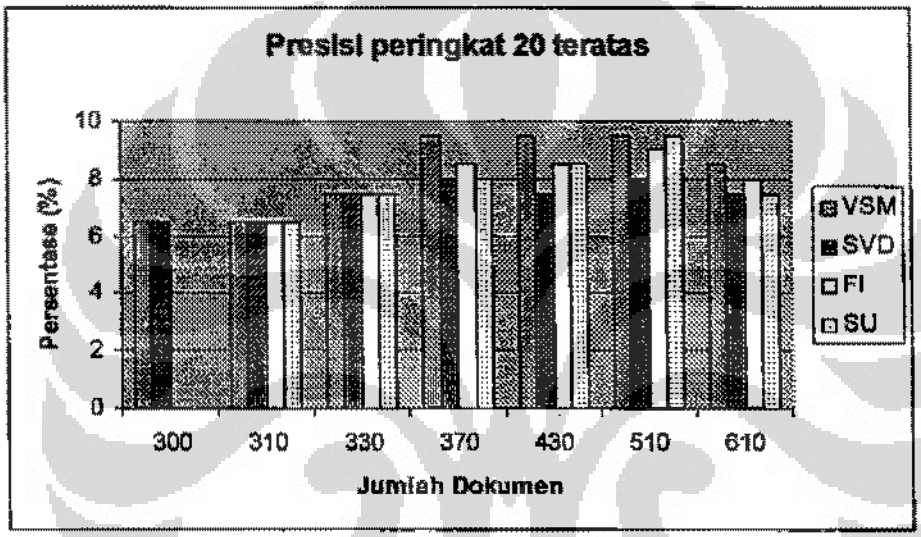
metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	610											
kuori	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	3.30	14.92	10	2.56	19.18	10	1.89	17.38	5	1.96	16.72
s2	5	14.29	1.15	0	0.00	1.64	5	3.85	4.26	5	4.17	3.93
s3	10	11.11	2.95	10	5.88	5.57	10	3.77	8.69	10	4.55	7.21
s4	0	0.00	1.80	0	0.00	2.30	0	0.00	2.95	0	0.00	2.48
s5	5	2.60	12.62	5	2.13	15.41	5	1.14	14.43	5	1.12	14.59
s6	10	6.06	5.41	10	5.88	5.57	10	5.88	5.57	10	6.25	5.25
s7	20	15.69	8.36	10	14.29	10.33	15	12.70	10.33	15	12.90	10.16
s8	10	4.17	11.80	10	3.30	14.92	10	2.94	16.72	10	3.06	16.07
s9	20	6.25	16.36	20	4.96	23.11	15	5.15	22.30	15	5.30	21.64
s10	0	1.10	14.92	0	0.96	17.05	0	0.83	17.70	0	0.93	17.54
rata-rata	8.5	6.46	9.23	7.5	4.00	11.51	8	3.83	12.03	7.5	4.02	11.56

Tabel 4.15 adalah penambahan secara progresif terakhir untuk jumlah koleksi dokumen yang kecil. Terlihat bahwa terjadi penurunan kinerja baik dari semua parameter yang disediakan khususnya pada model VSM dibandingkan dengan model pengindeksan konseptual lainnya baik yang dilakukan secara ulang maupun dinamis. Dari pengindeksan ulang, model SVD terjadi p20-nya lebih kecil dibandingkan dengan p20-nya VSM dikarenakan memang jumlah dokumen teraktivasinya lebih banyak. Sedangkan pada pengindeksan dinamis, model *Folding-in* maupun *SVD-Update* keduanya terjadi peningkatan jumlah dokumen teraktivasi dan otomatis menurunkan rata-rata persentase penduga presisi peringkat 20 besar dibandingkan dengan pengindeksan ulang. Hal itu terlihat dari p20 *Folding-in* sebesar 8% dan dokumen teraktivasinya sebesar 12,03%. Sedangkan pada model *SVD-Update* p20 sebesar 7,5% dan dokumen teraktivasinya sebesar 11,56%.

Berdasarkan rangkuman dari tabel 4.10 hingga tabel 4.15 didapatkan tabel 4.16 dan gambar 4.4. berikut ini yang berdasarkan persentase penduga presisi peringkat 20 besar dari koleksi AA, AAA1, hingga AAA6.

Tabel 4.16. Penduga presisi peringkat 20 teratas

#dok	model			
	VSM	SVD	FI	SU
300	6.5	6.5	-	-
310	6.5	6.5	6.5	6.5
330	7.5	7.5	7.5	7.5
370	9.5	8	8.5	8
430	9.5	7.5	8.5	8.5
510	9.5	8	9	9.5
610	8.5	7.5	8	7.5



Gambar 4.4. Penduga presisi peringkat 20 teratas kelompok pertama

Berdasarkan tabel 4.16 dan gambar 4.4 terlihat bahwa rata-rata persentase penduga presisi peringkat 20 besar baik pengindeksan ulang maupun pengindeksan dinamis tidak terdapat perbedaan yang terlalu signifikan. Untuk pengindeksan ulang, model SVD lebih kecil dikarenakan jumlah dokumen yang teraktivasi lebih banyak sedangkan jumlah dokumennya berdasarkan dataset BATAN jumlahnya tetap sehingga menurunkan rata-rata persentase penduga presisi peringkat 20 besar. Penurunan yang terjadi juga tidak terlalu besar yaitu berkisar 1% hingga 2%.

Sedangkan pada pengindeksan dinamis, jika dibandingkan dengan pengindeksan ulang berdasarkan tabel 4.16 terlihat justru rata-rata persentase presisi peringkat 20 besar lebih baik jika dibandingkan SVD dengan *Folding-in* dan *SVD-Update*.

Untuk sesama pengindeksan konseptual dinamis, model *SVD-Update* masih lebih baik dibandingkan *Folding-in* jika dilihat dari jumlah dokumen yang teraktivasi dengan rata-rata persentase presisi peringkat 20 besar dokumen terambil.

4.5.2 Ujicoba Kelompok Kedua

Pada ujicoba kelompok kedua ini, 500 dokumen koleksi BATAN, koleksi BB, dengan ID 0001 hingga ID 0500 diindeks menggunakan VSM dan SVD masing-masing dengan rank k = 50, 75, dan 100 (lihat tabel 4.4). Kemudian dilakukan penambahan dokumen-dokumen baru terhadap koleksi BB tersebut. Koleksi yang akan ditambahkan adalah koleksi B1, B2, dan B3. Sama dengan ujicoba pada kelompok pertama, langkah pertama adalah menambah koleksi B1 ke koleksi BB, hasilnya adalah koleksi BBB1. Selanjutnya penambahan koleksi B2 ke BBB1 menjadi BBB2, dan terakhir menambah koleksi B3 ke BBB2 menjadi BBB3. Pada ujicoba ini juga akan dilihat efektivitas STKI namun pada jumlah dokumen yang lebih besar dibandingkan pada ujicoba kelompok kedua.

Hasil ujicoba yang dicatat adalah persentase presisi 20 peringkat teratas, persentase presisi dokumen teraktivasi, dan persentase dokumen teraktivasi. Sama dengan kelompok ujicoba pertama, ujicoba SVD hanya dilakukan pada rank k = 50. Karena berdasarkan ujicoba antara VSM dan SVD pada subbab 4.4, kinerja SVD terbaik diraih saat rank $k = 50$.

Tabel 4.17 adalah presisi 20 peringkat teratas hasil ujicoba pada koleksi BB dengan menggunakan VSM dan SVD. Koleksi BB inilah yang dijadikan indeks acuan sebelum indeks ditambah dengan koleksi dokumen lainnya. Selanjutnya akan dilakukan pengindeksan dinamis pada koleksi BB.

Tabel 4.17. Penduga presisi peringkat 20 teratas koleksi BB

metode	VSM			SVD		
#dokumen	500			500		
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	3.75	16.00	10	3.03	19.80
s2	5	12.50	1.60	5	5.00	4.00
s3	10	13.33	3.00	10	6.45	6.20
s4	0	0.00	2.00	0	0.00	2.40
s5	5	3.39	11.80	5	2.78	14.40
s6	10	7.69	5.20	10	7.14	5.60
s7	25	17.39	9.20	10	15.38	10.40
s8	10	4.92	12.20	10	3.57	16.80
s9	25	6.98	17.20	20	5.36	22.40
s10	0	0.00	14.40	0	0.00	17.60
rata-rata	9.5	7.00	9.26	8	4.87	11.96

Tabel 4.17 adalah hasil ujicoba pada 500 dokumen BATAN menggunakan model VSM dan SVD. Terlihat bahwa penduga presisi peringkat 20 teratas model VSM pada koleksi BB sebesar 9.5% atau 1.5% lebih besar dari SVD yang sebesar 8%. Untuk presisi dokumen teraktivasi, hasil VSM didapatkan sebesar 7%, sedangkan SVD didapat 4.87%. Sedangkan untuk dokumen yang teraktivasi, model VSM mencapai 9.26% lebih kecil sekitar 2% dibandingkan dengan model SVD yang mencapai 11.96%. Dari tabel tersebut terlihat bahwa model persentase jumlah dokumen teraktivasi lebih banyak dibandingkan dengan model VSM hal ini mengakibatkan jumlah rata-rata persentase penduga presisi peringkat 20 besar maupun rata-rata penduga presisinya lebih kecil dibandingkan dengan model VSM. Hal ini kembali memperlihatkan bahwa dengan pengindeksan konseptual dokumen yang tadinya tidak memiliki bobot menjadi memiliki bobot dan menjadi terambil ketika kueri diberikan.

Tabel 4.18. Hasil ujicoba BBB1

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	700											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	5	3.00	14.29	10	1.55	18.43	10	2.27	18.86	10	2.36	18.14
s2	5	10.00	1.43	0	0.00	2.43	0	4.35	3.29	5	4.35	3.29
s3	20	19.05	3.00	10	5.41	5.29	10	5.71	5.00	10	5.56	5.14
s4	5	7.14	2.00	0	0.00	1.86	0	0.00	1.66	0	0.00	1.71
s5	5	2.35	12.14	5	1.87	15.29	5	1.87	15.29	5	1.80	15.00
s6	15	6.25	9.14	15	6.25	6.86	15	6.12	7.00	15	6.36	6.71
s7	15	13.64	9.43	5	12.50	11.43	10	12.50	10.29	10	12.50	10.29
s8	10	4.55	12.57	10	3.70	15.43	10	3.51	16.29	10	3.51	16.29
s9	20	6.29	20.43	10	4.84	26.57	15	4.92	26.14	15	5.08	25.29
s10	0	1.04	13.71	0	1.02	14.00	0	0.92	15.57	0	0.92	15.57
rata-rata	10	7.33	9.81	6.5	3.71	11.76	7.5	4.22	11.88	8	4.26	11.74

Tabel 4.18 adalah hasil ujicoba terhadap koleksi BBB1 yaitu koleksi BB ditambah 200 dokumen baru.

Pada pengindeksan ulang, persentase dokumen teraktivasi model SVD lebih banyak dibandingkan dengan model VSM yaitu sebesar 11,76% dibandingkan dengan 9,81%. Dikarenakan jumlah dokumen relevan tidak bertambah pada dataset maka otomatis rata-rata persentase presisi peringkat 20 besar dan rata-rata persentase presisi keseluruhan model SVD lebih kecil dibandingkan dengan model VSM.

Begitu juga ketika dilakkan pengindeksan konseptual dinamis, persentase dokumen teraktivasi SVD lebih banyak dibandingkan model VSM. Namun ternyata walau tidak dilakukan pengindeksan ulang, rata-rata persentase presisi peringkat 20 besar *Folding-in* dan *SVD-Update* lebih baik baik dibandingkan dengan model SVD pengindeksan ulang, yaitu sebesar 7,5% dan 8% dibandingkan dengan 6,5%. Hal ini terjadi akibat berubahnya peringkat atau posisi dokumen yang terambil.

Berdasarkan tabel 4.18 ujicoba yang dilakukan terhadap pengindeksan dinamis untuk koleksi dokumen yang lebih besar masih dapat ditolerir karena tidak terjadi penurunan rata-rata penduga presisi peringkat 20 besar secara signifikan.

Tabel 4.19. Hasil ujicoba BBB2

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	1000											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	10	2.80	14.30	15	2.17	18.40	15	2.19	18.30	15	2.27	17.60
s2	5	9.09	1.10	0	0.00	1.60	0	0.00	1.10	0	0.00	1.00
s3	15	9.52	4.20	10	11.76	1.70	10	3.77	5.30	10	4.00	5.00
s4	20	10.81	3.70	20	13.79	2.90	0	0.00	1.30	0	0.00	1.20
s5	5	2.42	12.40	5	2.38	16.80	5	2.42	16.50	5	2.45	16.30
s6	25	6.10	8.20	25	5.75	8.70	15	6.58	7.60	20	6.76	7.40
s7	15	8.57	10.50	0	8.70	13.80	5	9.48	11.80	5	8.57	11.50
s8	10	3.25	12.30	10	2.36	16.60	5	2.34	17.10	5	2.37	16.90
s9	30	6.61	22.70	15	5.47	27.40	15	5.12	29.30	20	5.30	28.30
s10	5	1.48	13.50	5	1.23	16.20	5	1.31	15.30	5	1.32	15.10
rata-rata	14	6.07	10.29	10.5	5.36	12.43	7.5	3.32	12.34	8.5	3.40	12.03

Tabel 4.19 adalah hasil ujicoba penambahan dokumen sebanyak 300 dokumen dari 700 dokumen menjadi 1000 dokumen. Ujicoba dilakukan terhadap pengindeksan ulang maupun pengindeksan dinamis.

Pada pengindeksan ulang, model SVD kembali menghasilkan jumlah dokumen yang lebih banyak dibandingkan model VSM. Yaitu sebesar 12,43% dibandingkan dengan 10,29%. Sedangkan jumlah dokumen relevan pada dataset tidak bertambah akibatnya rata-rata penduga presisi peringkat 20 besar VSM lebih baik yaitu sebesar 14% dibandingkan 10,5% pada SVD. Begitu pula pada rata-rata persentase penduga presisinya yaitu 6,07% dibandingkan 5,36% pada SVD.

Sedangkan pada pengindeksan dinamis, selain jumlah dokumen teraktivasi baik *Folding-in* maupun *SVD-Update* dibandingkan model SVD pengindeksan ulang, jumlahnya lebih sedikit. Namun rata-rata persentase presisi peringkat 20 besar maupun rata-rata persentase secara keseluruhan tidak juga lebih baik. Artinya

terjadi penurunan kinerja ketika jumlah dokumen makin besar dan penambahan terus dilakukan juga dalam jumlah besar tanpa dilakukan pengindeksan ulang.

Tabel 4.20. Hasil ujicoba BBB3

metode	Pengindeksan Ulang						Indeks Dinamis					
	VSM			SVD			*Folding-in*			*SVD-Update*		
#dokumen	1162											
kueri	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)	p20 (%)	p (%)	ak (%)
s1	15	4.00	15.06	15	3.57	19.26	15	3.21	18.76	15	3.33	18.07
s2	5	7.69	1.12	0	0.00	1.46	0	0.00	0.95	0	0.00	0.66
s3	15	7.55	4.56	5	7.41	2.32	10	3.08	5.59	10	3.33	5.16
s4	20	15.79	3.27	20	17.86	2.41	0	0.00	0.43	0	0.00	0.69
s5	5	3.50	12.31	10	2.58	16.70	10	3.08	16.78	10	3.11	16.61
s6	25	5.56	7.75	25	5.15	8.35	15	5.68	7.57	20	5.58	7.31
s7	15	9.60	10.76	0	9.94	13.86	5	10.49	12.31	5	10.79	11.96
s8	5	2.68	12.82	10	2.03	16.95	5	1.88	17.38	5	2.02	17.04
s9	40	8.08	22.38	20	6.33	28.57	20	6.23	29.00	15	5.46	27.97
s10	5	1.69	13.66	5	1.55	16.70	5	1.62	15.92	5	1.67	15.49
rata-rata	15	6.63	10.37	11	5.64	12.66	8.5	3.54	12.47	8.5	3.66	12.11

Selanjutnya tabel 4.20 adalah ujicoba terakhir untuk penambahan dokumen sebesar 162 dokumen terhadap 1000 dokumen sehingga didapatkan keseluruhan koleksi dokumen sebesar 1162 dokumen.

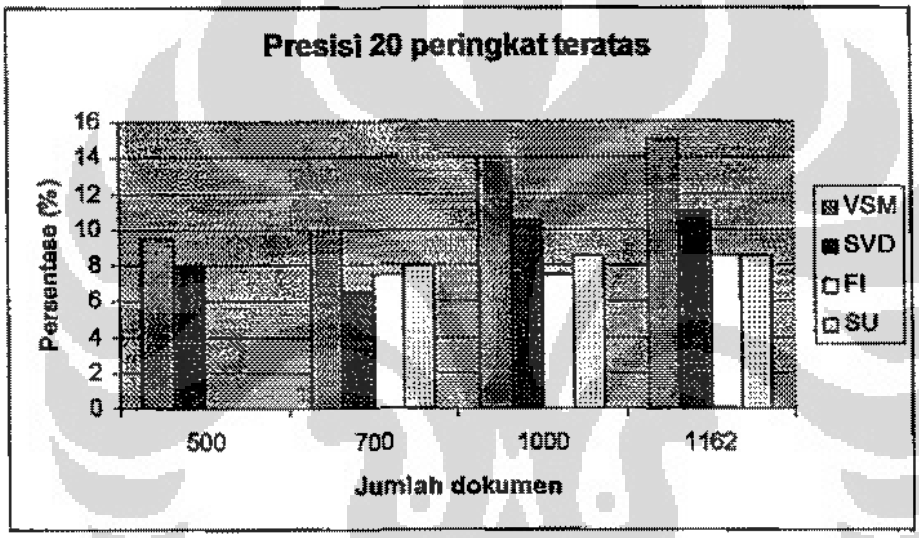
Berdasarkan tabel tersebut, untuk pengindeksan ulang, jumlah dokumen teraktivasi pada model SVD lebih baik dibandingkan dengan model VSM yaitu 12,66% dibandingkan dengan 10,37%. Namun rata-rata persentase penduga presisi peringkat 20 besar VSM jauh mengungguli model SVD yaitu 15% dibanding 11%. Hal ini memang sesuai dengan jumlah dokumen relevan pada dataset yang tidak bertambah.

Sedangkan pada pengindeksan dinamis, baik *Folding-in* maupun *SVD-Update* terjadi penurunan rata-rata persentase presisi peringkat 20 besar maupun rata-rata persentase presisi secara keseluruhan. Hal ini diakibatkan indeks dokumen baru hasil pengindeksan dinamis semakin kehilangan ortogonalitasnya ketika dokumen terus ditambah secara berulang-ulang.

Berdasarkan tabel 4.18 hingga tabel 4.20 dianalisa didapatkan tabel 4.21 dan gambar 4.5. berikut ini berdasarkan persentase penduga presisi peringkat 20 besar dari koleksi BB, BBB1, hingga BBB3.

Tabel 4.21. Penduga presisi peringkat 20 teratas kelompok kedua

		model			
		VSM	SVD	FI	SU
#dok	500	9.5	8	-	-
	700	10	6.5	7.5	8
	1000	14	10.5	7.5	8.5
	1162	15	11	8.5	8.5



Gambar 4.5. Grafik penduga presisi peringkat 20 teratas kelompok kedua

Berdasarkan tabel 4.21 dan gambar 4.5 terlihat bahwa rata-rata persentase penduga presisi peringkat 20 besar model VSM lebih baik dibandingkan dengan model pengindeksan konseptual baik setelah dilakukan pengingdeksan ulang maupun pengindeksan dinamis secara progresif. Berdasarkan hasil ujicoba juga menunjukkan penambahan dokumen secara terus menerus tanpa dilakukannya pengindeksan ulang mengakibatkan penurunan kinerja yang cukup penting terutama jika jumlah dokumen banyak. Hal ini sebaliknya tidak terlalu signifikan pada percobaan kelompok pertama dimana jumlah dokumen yang diujicoba dan ditambahkan tidak sebanyak pada ujicoba kelompok kedua ini.

4.6 Rangkuman Hasil Ujicoba

Berdasarkan ujicoba antara model VSM dan SVD serta pengujian pada koleksi dokumen dinamis antara pengindeksan ulang VSM dan SVD dengan pengindeksan *folding-in* dan *SVD-Update* maka didapatkan analisa berikut:

- Pada percobaan seluruh dokumen BATAN sebanyak 1162 dokumen, dengan penilaian relevansi yang terbatas, model VSM lebih baik dibandingkan dengan model SVD. Hal ini diakibatkan jumlah dokumen yang teraktivasi pada model SVD lebih banyak dibandingkan dengan model VSM. Terlihat bahwa dengan pengindeksan konseptual membuat jumlah dokumen yang teraktivasi lebih banyak. Artinya terjadi perluasan kueri (*query expansion*).
- Pada percobaan seluruh dokumen BATAN, khususnya model SVD, jumlah nilai singular terbaik yang digunakan adalah sebanyak 50. Hal ini terlihat dari penduga presisi, penduga presisi peringkat 20 besar, hingga jumlah dokumen teraktivasi yang lebih baik jika digunakan rank $k = 75$ dan $k = 100$.
- Pada penambahan dokumen untuk kelompok pertama yaitu jumlah dokumen yang sedikit dan penambahan bertahap yang jumlahnya juga sedikit, terlihat bahwa pengindeksan ulang lebih baik dibandingkan dengan pengindeksan dinamis. Namun perbedaan yang terdapat tidak terlampau signifikan.
- Khusus, masih dalam kelompok pertama, untuk pengindeksan dinamis model *SVD-Update* lebih baik dibandingkan dengan model *Folding-in*.
- Berdasarkan ujicoba kelompok pertama, setelah dilakukan penambahan dokumen secara progresif dalam jumlah kecil, penurunan kinerja tidak terlalu signifikan yaitu berkisar antara 0,5% hingga 2%.
- Jumlah dokumen teraktivasi pengindeksan dinamis baik yang dilakukan ulang maupun dinamis lebih banyak dibandingkan dengan pengindeksan non konseptual. Hal ini tentu saja merupakan akibat dari terbobotnya dokumen terhadap suatu istilah sehingga ketika kueri diberikan dokumen tersebut menjadi terambil.

- Pada ujicoba kelompok kedua yaitu jumlah dokumen awal dalam jumlah banyak dan penambahan dokumen juga dalam jumlah banyak terlihat bahwa kinerja yang dihasilkan pada pengindeksan konseptual menurun baik yang dilakukan secara ulang maupun dinamis.
- Namun pada pengindeksan ulang penurunan kinerja tidak sejauh pada pengindeksan dinamis.
- Walau belum dilakukan penilaian relevansi secara keseluruhan pada dataset BATAN, jumlah dokumen yang teraktivasi pada kelompok kedua baik pengindeksan ulang maupun dinamis lebih banyak dibandingkan dengan pengindeksan biasa model VSM.
- Hal ini mengakibatkan rata-rata persentase presisi peringkat 20 besar VSM pada semua dokumen tidak dapat ditandingi oleh pengindeksan konseptual yaitu sebesar 15% dibandingkan dengan 11% dan 8,5% (tabel 4.21).
- Untuk penambahan dalam jumlah sedikit pengindeksan dinamis secara progresif tidak terjadi penurunan kinerja terlalu signifikan
- Sebaliknya pada penambahan dalam jumlah dokumen yang lebih besar, pengindeksan secara progresif menjadi tidak efektif lagi dan hasil dilakukan pengindeksan ulang.
- Hal ini diakibatkan menurunnya ortogonalitas indeks konseptual yang sudah ditambahkan dokumen-dokumen baru secara progresif.
- Pengindeksan konseptual dinamis dengan *SVD-Update* lebih baik dibandingkan dengan *Folding-in* hal ini terlihat pada hasil ujicoba baik pada kelompok pertama maupun kelompok kedua.
- Berdasarkan semua ujicoba dapat disimpulkan bahwa pengindeksan konseptual merupakan perluasan kueri dikarenakan jumlah dokumen yang teraktivasi lebih banyak.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan ujicoba yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

- Pada pengindeksan model VSM, dokumen yang istilahnya tidak ada pada kueri tidak akan memiliki bobot sebaliknya pada pengindeksan konseptual, setelah diadakannya dekomposisi matriks menggunakan SVD dengan rank k tertentu (pada penelitian ini digunakan k = 50, 75, dan 100) dokumen tersebut dapat memiliki bobot sehingga ketika kueri diberikan dokumen tersebut dapat ikut terambil.
- Akibatnya jumlah dokumen teraktivasi atau terambil pada model pengindeksan konseptual lebih banyak dibandingkan dengan model VSM.
- Hal ini menunjukkan bahwa masalah sinonim dan polisemi pada pengindeksan konseptual dapat dianggap sebagai perluasan kueri (*query expansion*) tanpa menggunakan thesaurus.
- Model pengindeksan konseptual memungkin ditambahnya dokumen dan istilah baru tanpa perlu dilakukan pengindeksan ulang sebagaimana layaknya sistem temu kembali informasi lainnya.
- Pada ujicoba pengindeksan dinamis menunjukkan bahwa penambahan koleksi baik pada koleksi dokumen besar maupun kecil dapat dilakukan untuk pertama kali dan tidak terjadi penurunan kinerja terlalu signifikan. Ini diakibatkan penurunan ortogonalitas tidak terlalu tinggi.
- Pada ujicoba pengindeksan dinamis secara progresif menunjukkan bahwa setelah dilakukan penambahan dokumen berkali-kali terjadi penurunan efektivitas. Hal ini disebabkan ortogonalitas representasi koleksi dokumen yang tidak dapat dipertahankan ketika dilakukan penambahan dokumen secara terus menerus baik dengan metode *Folding-in* maupun *SVD-Update*.

5.2 Saran

Berdasarkan hasil penelitian, ada beberapa saran untuk pengembangan dan penyempurnaan lebih lanjut, yaitu:

- Penelitian dilakukan pada koleksi dokumen selain dari BATAN yang telah memiliki penilaian relevansi yang lengkap atau dilakukan penilaian relevansi yang lengkap pada koleksi dokumen BATAN.
- Dilakukannya pengujian nilai relevansi lebih lanjut setiap dokumen pada koleksi terhadap kueri yang diberikan terutama oleh real user sehingga dapat ditentukan nilai *recall* dan *precission*.
- Jumlah koleksi dokumen BATAN hanya 1162 buah dokumen dan sudah lama tidak diperbaharui. Sehingga sistem belum diuji pada koleksi dokumen yang jumlahnya sangat besar.
- Perlu dilakukan perbaikan metode SVD-Update guna mengatasi hilangnya ortogonalitas pada saat penambahan koleksi dokumen baru. Seperti yang ditawarkan oleh Obrien (1994) dengan adanya teknik perbaikan bobot koleksi dokumen yang sudah diperbaharui.
- Dapat dilakukan metode penambahan dokumen lainnya yaitu metode baru SVD-Update bernama *low-rank-plus-shift* (Hongyuan Zha, 1999).

DAFTAR PUSTAKA

Allan, James. Materi kuliah *Information Retrieval* University of Massachusetts, (Fall Course 2004). Diakses pada tanggal 11 November 2005 dari http://ciir.umass.edu/cmpsci646/

Ariwibowo, Anung B. (2001). *Pendekatan Multi-dimensi Dokumen dalam Sistem Temu-kembali Informasi Menggunakan Model Spreading Activation*. Tesis S2. Depok: Fasilkom UI.

Baeza-Yates, Ricardo and Ribeiro-Neto, Berthier. (1999). *Modern Information Retrieval*. New York: Addison-Wesley Publishing Company.

Berry, Michael W., et al. (1999). *Matrices, Vector Spaces, and Information Retrieval*. SIAM Review. Vol 41, No. 2 pp. 335-362.

Budi, Indra. (2002). *Pengindeksan dan Kemiripan Dokumen dalam Sistem Temu Kembali Informasi*. Tesis S2. Depok: Fasilkom UI.

Bondan, Alit. (2001). *Aljabar Linier*. Penerbit Universitas Trisakti, Jakarta.

Deerwester, S. C., et al. (1990). *Indexing by Latent Semantic Analysis*. Journal of American Society of Information Science. Vol 41 no. 6. pp. 391-407.

Dumais, Susan T. (1991). *Improving the Retrieval of Information from External Sources*. Behavior Research Methods, Instruments, & Computers, 23(2) pp. 229-236.

Fauzan, Mohamad. (2002). *Penerapan Dekomposisi SDD dan Nilai Singular untuk Peningkatan Kinerja Sistem Temu Kembali Informasi*. Skripsi S1. Depok: Fasilkom UI.

Fitriyanti, Masayu. (1997). *Sistem Temu Kembali Informasi dengan Mengimplementasikan Operasi Boolean, Sistem Peringkat, Perbaikan Query, dan Pemanfaatan Tesaurus*. Skripsi S1. Depok: Fasilkom UI.

Frakes, W.B., and Baeza Yates, R. (1992). *Information Retrieval: Data Structures & Algorithms*. USA: Prentice Hall.

Furnas, George W., et al. (1988). *Information Retrieval using a Singular Value Decomposition Model of Latent Semantic Structure*. Proceedings of the 11th annual international ACM SIGIR, Grenoble, France.

Herdiyeni, Yeni and Hasibuan, Zainal A. (2003). *Information Retrieval System in Bahasa Indonesia Using Latent Semantic Indexing and Semi-discrete Decomposition*. Proceedings of IiWAS 2003. Jakarta. Indonesia.

Kolda, Tamara G. And O'Leary, Dianne P. (1998). *A Semidiscrete Matrix Decomposition for Latent Semantic Indexing in Information Retrieval*. Proceedings of the 21th annual international ACM SIGIR, Melbourne, Australia.

Korfhage, Robert R. (1997). *Information Storage and Retrieval*. Wiley Computer Publishing.

Lancaster, F. W. (1998). *Indexing and Abstracting in Theory and Practice*. Library Assosiation Publishing. London.

Leach, Sonia. (1995). *Singular Value Decomposition – A Primer*. Tutorial Draft Version. Departement of Computer Science. Brown University. Diakses pada tanggal 6 Juli 2005 dari http://www.cs.brown.edu/research/ai/dynamics/tutorial/Postscript/SingularvalueDecomposition.ps

Leon, Steven J. (2001). *Aljabar Linier dan Aplikasinya edisi ke-5*. Penerbit Erlangga. Jakarta.

Liao, Yu. (2000). *TSVD in LSI: project paper for Numerical Computation*.

Mustangimah. (1998). *Efektifitas Sistem Temu Kembali dan Analisis Bibliometrik: Aplikasi pada Dokumen Bidang Nuklir Berbahasa Indonesia*. Tesis S2. Depok: Pasca Sarjana Universitas Indonesia.

O'Brien, Gavin W. (1994). *Information Management Tools for Updating an SVD-Encoded Indexing Scheme*. Diakses dari http://www.cs.utk.edu/library/TechReports/1994/ut-cs-94-258.ps.Z tanggal 5 Desember 2004.

Rijsbergen, C.J. Van. (1979). *Information Retrieval 2ed*. Butterworth & Co Ltd. London.

Rosario, Barbara. (2000). *Latent Semantic Indexing: An Overview*. Final Paper. Infosys 240.

Salton, Gerard. (1989). *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. New York: Addison-Wesley Publishing Company.

Savoy, Jacques. (1994). *A Learning Scheme for Information Retrieval in Hypertext*. Information Processing and Management 30.

Skillicorn, D. B., et al. (2003). *Handbook of Data Mining using Matrix Decompositions*. Report paper. School of Computing Queen's University Kingston Canada.

Supranto, J. (1992). *Pengantar Matrix Edisi Lima*. Lembaga Penerbit Fakultas Ekonomi Universitas Indonesia.

Witter, Dian I. and Berry, Michael W. (1998). *Downdating the Latent Semantic Indexing Model for Conceptual Information Retrieval*. The Computer Journal, Vol 14, No. 8.

Xu, Jinxi and Croft, W. Bruce. (1996). *Query Expansion Using Local and Global Document Analysis*. Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval. Zurich, Switzerland.
http://umass.edu/~xu/sigir96final.ps

Zha, Hongyuan. (1999). On Matrices with Low-Rank-Plus-Shift Structure Partial SVD and Latent Semantic Indexing. SIAM Journal Volume 21, Issue 2.

LAMPIRAN

LAMPIRAN 1

CONTOH ABSTRAK DOKUMEN

DOC: 0001

TIT: Pengaruh Pemupukan P Dan Pengapuran Pada Tumpangsari Jagung Kedelai Terhadap Hasil dan Fiksasi N Simbiotik

AUT: Widjang H. Sisworo, N. Abdullah, Havid Rasyid, B. Soeminto

AFF: Pusat Aplikasi Isotop dan Radiasi, BATAN

JOU: Majalah BATAN Vol. XIX No. 1/2 Apzil/Juli 1986

ABS: Percobaan lapang yang memiliki tujuh perlakuan dan enam ulangan
diatur dalam rancangan acak kelompok, bertujuan untuk mempelajari pengaruh penempatan pupuk P dan pemberian kapur pada tumpangsari jagung dan kedelai terhadap hasil dan banyaknya N yang difiksasi kedelai secara sembiotik. Hasil dari penelitian ini menunjukkan cara penempatan pupuk P mempengaruhi hasil biji kedelai, tetapi tidak terhadap hasil biji jagung. Pupuk P yang ditempatkan dalam alur di bawah biji menyebabkan hasil biji kedelai lebih tinggi daripada yang ditebar dan diaduk dengan tanah atau yang dialurkan di dekat baris tanaman. Pengapuran meningkatkan hasil biji kedelai atau jagung sebesar 31 persen lebih tinggi dari yang tidak dikapur. Pengapuran juga meningkatkan banyaknya N yang difiksasi kedelai secara simbiotik. Kebutuhan N tanaman kedelai diperoleh dari tanah, fiksasi dan pupuk berturut-turut sebesar 50, 46 dan 6 persen.

BIB:

1. ANONIM#(1982)#Fert#. Plant Nutr Bull No 5 210., FOOD AND AGRICULTURE ORGANIZTION, FAO. (1982) 210.

2. W.H. SISWORO, N. ABDULLAH, S. GANDANEGARA, B.SOEMINTO#(1982)#Coordinated Research Programme on Nuclear Techniques in the Development of Fertilizer and Water Management Practices for Multiple Cropping System#. A Report to the Second Coordination Meeting. Viena 8-12 Nov. 1982.

3. W.H. SISWORO, N. ABDULLAH, M. MARDJO, B. SOEMINTO, S. GANDANEGARA, E. SURYATNA, G. SOEPARDI#(1983)#Risalah Pertemuan Ilmiah Aplikasi Teknik Nuklir di Bidang Pertanian dan Biologi#. Pusat Aplikasi Isotop dan Radiasi (1983) 274.

4. H. SISWORO, N. ABDULLAH, H. RASYID, B. SOEMINTO#(0000)#Penempatan pupuk fosfat dan pengapuran dalam pertanaman tumpangsari jagung-kedelai#. belum diterbitkan.

5. D. ROBSON#(1978)#Mineral Nutrition of Legumes in Tropical and Subtropical Soils#. (C. S. ANDREW, and E. J. KAMPRATH, eds).CSIRO (1978) 277.

6. K.R. HELYAR#(1978)#Mineral Nutrition of Legumes in Tropical and subtropical Soils#. (C.S. ANDREW, and E.J. KAMPRATH, eds). CSIRO (1978) 207.

7. D.N. MUNNS#(1978)#Mineral Nutrition of Legumes in Tropical and Subtropical Soils#. (C.S. ANDREW, and E.J. KAMPRATH, eds). CSIRO (1978) 247.

8. H.S. RUSSEL#(1978)#Mineral Nutrition of Legumes in Tropical and Subtropical Soils#. (C.S. ANDREW, and E.J. KAMPRATH, eds). CSIRO (1978) 361.

9. FRIED, V. MIDDEL BOE#(1977)#Mesurement of amount of nitrogen fixed by a Legumes Crop#. Plant Soil (1977) 47:713-715.

10. Anonim#(1978)#IAEA Tracer Manual On Crop and Soils#. Tech. Rep. Ser. No. 171 (1978) 277.

11. R.W.E. HARDY, U.D. HAVELKA#(1976)#Simbiotic Nitrogen Fixation in Plant#. (P.S. NUTMAN, ed) I.B.P.J. Cambridge Univ. Press. (Cambridge (1976) 421.

12. M.B. PARKER, H.B. HARRIS#(1977)# J Agron 69 551#., (1977) 551.

13. D.F. BEZDICEK, D.W. EVANS, B. ABEDE, R.A. WITTERS#(1978)#J Agron 70 865#., (1978) 865.

14. A.N. NELSON, R.W. WEAVER#(1980)#J agron 72 613#., (1980) 613.

DOC: 0002

TIT: Sistem Deteksi Spektroskopi Korelasi Foton I. Penguat Dan Diskriminator Tabung Foto Pengganda Pencacah Foton.(2)

AUT: Soewono Prawiroatmodjo

AFF: Pusat Penelitian Bahan Murni dan Instrumentasi, BATAN

JOU: Majalah BATAN Vol. XIX No. 1/2 April /Juli 1986 (11-20)

ABS: Diuraikan rangkaian elektronik yang dapat memperkuat, mendiskriminasi dan membentuk
warta keluaran tabung foto pengganda foton serta keluarannya dapat ditanggapi secara digital. Alat tersebut terdiri dari penguat operasional MCI733C sebagai penguat, komparator UA760 sebagai komponen utama diskriminator dan rangkaian pembentuk pulsa.
Sistem tersebut dapat dipasang di antara tabung foto pengganda pencacah foton dan peralatan elektronika digital.

BIB:

1. B. CHU#(1974)#Laser Light Scattering#. Academic Press, New York, 1974.

2. Anonim#(1983)#Characteristic of Photomultiplier#. AMAMATZU, Japan, 1983.

3. KOWALSKI#(1970)#Nuclear Electonics#. Springer Verlag, New York, 1970.

4. Anonim#(1978)#Photomultiplier Tubes#. RCA, Lancaster, USA, 1978.

5. SOEWONO PRAWIROATMODJO#(0000)#Sistem Deteksi Spektroskopi Korelasi Foton II Korelator Digital (akan terbit)#.

6. Anonim#(1978)#Linear Data Book#. National Semiconductor, 1978.

LAMPIRAN 2

SOURCE CODE

Membaca matriks sparse

```
function [A, m, n] = mtxread(infile);

%MTXREAD Read matrix A from a file in matrix market format.
%
%    A = mtxread(FILENAME) reads a MatrixMarket formatted matrix
from the
%    file FILENAME.
%
%SDDPACK: Software for the Semidiscrete Decomposition.
%Copyright (c) 1999 Tamara G. Kolda and Dianne P. O'Leary.

% This program is free software; you can redistribute it and/or
modify it
% under the terms of the GNU General Public License as published
by the Free
% Software Foundation; either version 2 of the License, or (at
your option)
% any later version.
%
% This program is distributed in the hope that it will be useful,
but
% WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY
% or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
License
% for more details.
%
% You should have received a copy of the GNU General Public
License along
% with this program; if not, write to the Free Software
Foundation, Inc., 59
```

```
% Temple Place - Suite 330, Boston, MA 02111-1307, USA.

fid = fopen(infile, 'rt');

if (fid == -1)
  error('Error opening file.');
end

line = fgets(fid);
while line(1) == '%'
  line = fgets(fid);
end

[data, cnt] = sscanf(line,'%d');
m = data(1);
n = data(2);
nnzs = data(3);

[data, cnt] = fscanf(fid, '%d %d %e', [3, inf]);

I = data(1,:);
J = data(2,:);
S = data(3,:);
A = sparse(I, J, S, m, n);

fclose(fid);

return;
```

Proyeksi Kueri

```
%-----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% query projection
% file name: qprj.m
%-----------------------------------------

function newq = qprj(q, Gi, Uk, Sk)
```

```
for i = 1 : length(q)
    wq(i, 1) = log2(1 + q(i)) * Gi(i);
end

[M, N] = size(Sk);
T = Sk;
if M < N
    for i = 1 : M
        T(i, i) = 1 / T(i, i);
    end
else
    for i = 1 : N
        T(i, i) = 1 / T(i, i);
    end
end

newq = wq' * Uk * T;
```

Peringkat Dokumen

```
%----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Rank-order the documents
% file name: rankdoc.m
%----------------------------------------

function ord = rankdoc(D, q)

[M, N] = size(D);
for j = 1 : N
    DV = D(1:M, j);
    r(j) = (DV'*q) / (norm(DV) * norm(q));
end

[V, ord] = sort(r);
```

Peringkat Istilah

```
%-----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Rank-order the terms
% file name: rankterm.m
%-----------------------------------------

function ord = rankterm(T, q)

[M, N] = size(T);
for i = 1 : M
    TV = T(i, 1:N);
    r(i) = abs(TV * q) / (norm(TV) * norm(q));
end

[Y, ord] = sort(r);
```

Pembobotan Koleksi Dokumen

```
%-----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% weighting
% file name: weight.m
%-----------------------------------------

function [W, Gi, df] = weight(A)

[M, N] = size(A);

for i = 1 : M

    temp = 0;
    for j = 1 : N
        if ( A(i,j) ~= 0 )
            temp = temp + 1;
        end
    end
```

```
    df(i) = temp;

    sum = 0;
    denom = log10(N);
    for j = 1 : N
        p(i,j) = A(i, j) / df(i);
        if (p(i,j) ~= 0)
            sum = sum + p(i,j) * log10(p(i,j)) / denom;
        end
    end
    Gi(i) = 1 - sum;

    for j = 1 : N
        W(i, j) = log10(A(i, j) + 1) * Gi(i);
    end
end
```

Update Istilah (*Folding-in*)

```
%----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Updating-SVD updating term
% file name: upterm.m
%----------------------------------------

function [newU, newS, newV] = upterm(T, Uk, Sk, Vk)

Wt = weight(T);

H1 = Wt * Vk;
H = [Sk; H1]
[Uh, Sh, Vh] = svd(H);

% ambil ukuran matrik term baru
[mt, nt] = size(T);
% ambil ukuran term
[mu, nu] = size(Uk);
```

```
temp = [Uk zeros(mu, mt); zeros(mt, nu) eye(mt)];

newU = temp * Uh;
newV = Vk * Vh;
newS = Sh;
```

Update Istilah (*SVD-Update*)

```
%-----------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Updating terms using svd new method
% file name: upterm2.m
%-----------------------------------

function [newU, newS, newV] = upterm2(T, Uk, Sk, Vk)

Wt = weight(T);
[Mv, Nv] = size(Vk);
[Q, R] = qr((eye(Mv) - Vk * Vk') * Wt');

[M, N] = size(Sk);
H1 = Wt * Vk;
H = Sk;

[M1, N1] = size(H1);
H(M+1:M+M1, 1:N) = H1;
H(M+1:M+M1, N+1:N+Mv) = R';

[Uh, Sh, Vh] = svd(R);

[Mu, Nu] = size(Uk);
[Mt, Nt] = size(Wt);
Lmat = Uk;
Lmat(Mu+1:Mu+Mt, Nu+1:Nu+Mt) = eye(Mt);
[Muh, Nuh] = size(Uh);
newU = Lmat * Uh(1:Muh, 1:M)
newS = Sh(1:M, 1:N)
V = Vk;
```

```
V(1:Mv, Nv+1:Nv+Mv) = Q;
[Mvh, Nvh] = size(Vh);
newV = V * Vh(1:Mvh, 1:N)
```

Update Dokumen (*Folding-in*)

```
%-----------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Updating-SVD-updating documents
% file name: updoc.m
%-----------------------------------

function [newU, newS, newV] = updoc(D, Gi, Uk, Sk, Vk)

[Md, Nd] = size(D);
for i = 1 : Md
    for j = 1 : Nd
        Wd(i, j) = log10(1 + D(i, j)) * Gi(i);
    end
end

F1 =Uk' * Wd;
F = [Sk F1];
[Uf, Sf, Vf] = svd(F);

newU = Uk * Uf;
newS = Sf;

[m, n] = size(Vk);
kanan = zeros(m, Nd);
kiri = zeros(Nd, n);
V1 = [Vk kanan; kiri eye(Nd)];
newV = V1 * Vf;
```

Update Dokumen (*SVD-Update*)

```
%-----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Updating docs using svd new method
% file name: updoc2.m
%-----------------------------------------

function [newU, newS, newV] = updoc2(D, Gi, Uk, Sk, Vk)

[Md, Nd] = size(D);
for i = 1 : Md
    for j = 1 : Nd
        Wd(i, j) = log10(1 + D(i, j)) * Gi(i);
    end
end

[Mu, Nu] = size(Uk);
[Q, R] = qr((eye(Mu) - Uk * Uk') * Wd);

[M, N] = size(Sk);
F1 = Uk' * Wd;

[M1, N1] = size(F1);
F = Sk;
F(1:M , M+1 : N+N1) = F1

[Mr, Nr] = size(R);
F(M+1 : M+Mr, N+1:N+Nr) = R;

[Uf, Sf, Vf] = svd(F);
[Muf, Nuf] = size(Uf);
Uf1 = Uf(1:Muf, 1:M);
U = Uk;
U(1:Mu, Nu+1:Nu+Mu) = Q;
newU = U * Uf1;

newS = Sf(1:M, 1:N)
```

```
LMat = Vk;
[Mv, Nv] = size(Vk);
LMat(Mv+1:Mv+Nd, Nv+1:Nv+Nd) = eye(Nd);
newV = LMat *Vf(1:(Nv+Nd), 1:N)
```

Pemberian *Score* pada Dokumen

```
%-----------------------------------------
% Latent Semantic Indexing (LSI) / TSVD
% Score the documents using Cosine
% file name: rankdoc.m
%-----------------------------------------

function SR = scoredoc(D, q)

[M, N] = size(D);
for j = 1 : N
    DV = D(:, j);
    DV;
    SR(j) = (DV'*q) / (norm(DV) * norm(q));
end
```

LAMPIRAN

OUTPUT SEBAGIAN PROGRAM

Peringkat 20 besar kueri s1 terhadap 300 dokumen BATAN dengan VSM beserta bobotnya.

219	0.26136
47	0.23799
126	0.23468
183	0.21593
239	0.19535
159	0.15752
221	0.15293
146	0.15214
4	0.15209
227	0.14989
52	0.1465
230	0.14578
24	0.14467
271	0.14186
163	0.14134
276	0.14117
82	0.14004
206	0.13995
139	0.13816
284	0.12745

Peringkat 20 besar kueri s2 terhadap 300 dokumen BATAN dengan VSM beserta bobotnya

37	0.12875
137	0.12269
273	0.11876
109	0.1074
270	0.097408
133	0.093346
288	0.092818

271	0.092022
36	0.091094
13	0.090368
220	0.088254
213	0.079872
67	0.074505
124	0.069816
269	0.066421
175	0.066333
179	0.061327
57	0.054146
300	0
299	0

Peringkat 20 besar kueri s3 terhadap 300 dokumen BATAN dengan SVD $k = 50$ beserta bobotnya

96	0.081333
279	0.059587
138	0.051102
72	0.049924
105	0.047396
231	0.046464
50	0.046257
56	0.04491
256	0.044594
6	0.043319
161	0.043046
126	0.042735
264	0.042226
151	0.041825
140	0.039964
94	0.03996
4	0.039852
19	0.039825
89	0.037609
157	0.037196

Peringkat 20 besar kueri s5 terhadap 500 dokumen BATAN dengan SVD $k = 50$

327	0.35641
241	0.35363
62	0.34538
367	0.33881
473	0.29977
426	0.28434
474	0.27973
322	0.27636
127	0.2699
43	0.25225
476	0.25211
232	0.25072
361	0.24789
482	0.24065
203	0.23758
404	0.22723
47	0.22657
421	0.22087
52	0.21997
424	0.2197