*Jurnal Sains dan Matematika* Vol. 20 (3): 64-70 (2012)

# Sistem Temu Kembali Informasi pada Dokumen Teks Menggunakan Metode Term Frequency Inverse Document Frequency (TF-IDF)

**[1]Dhony Syafe'i Harjanto, [2]Sukmawati Nur Endah, dan [2]Nurdin Bahtiar**
[1]Jurusan Matematika, [2]Jurusan Ilmu Komputer / Informatika
Fakultas Sains dan Matematika Universitas Diponegoro

## ABSTRAK

Banyaknya informasi yang disimpan dalam dokumen teks mengakibatkan pengguna sistem informasi mengalami kesulitan untuk mendapatkan informasi yang diinginkan, maka diperlukan sebuah mesin pencarian yang dapat menentukan dan menemukan dokumen yang relavan sesuai dengan query pengguna. Penelitian ini menggunakan metode Term Frequency Inverse Document Frequency (TF-IDF) yang didasarkan pada kemunculan term pada tiap dokumen dan pengurangan dominasi term yang sering muncul di berbagai dokumen. Hasil Penelitian ini adalah program simulasi Sistem Temu Kembali Informasi pada dokumen teks menggunakan Metode Term Frequency Inverse Document Frequency (TF-IDF) yang menghasilkan perhitungan pembobotan Term Frequency Inverse Document Frequency (TF-IDF) dan mendapatkan dokumen relevan yang teranking sesuai tingkat pembobotannya berdasarkan query masukan oleh pengguna.

*Keywords: Mesin Pencarian, Query, Term Frequency, TF-IDF*

## PENDAHULUAN

Belakangan ini terdapat beragam informasi berbasis teks, yaitu informasi yang disimpan dalam dokumen-dokumen berupa file text. Sejalan dengan banyaknya dokumen yang dapat disimpan, maka seseorang mengalami kesulitan untuk mendapatkan informasi yang diinginkan karena harus melihat dokumen satu demi satu untuk mendapatkan informasi yang tepat, dan tentunya akan membutuhkan waktu yang lama. Sehingga diperlukan suatu cara agar pengguna dapat mengakses informasi secara cepat dan tepat.

Sebuah dokumen khususnya dokumen ilmiah dapat ditemukan dengan mudah oleh manusia, tetapi jika dilakukan secara terkomputerisasi akan membawa permasalahan tersendiri. Begitu pula dengan tingkat relevansi atau kecocokan suatu dokumen dengan dokumen lainnya. Manusia dapat dengan mudah menentukan tingkat relevansi suatu dokumen dengan dokumen lainnya. Salah satu cara untuk menemukan dokumen secara terkompuetrisasi adalah dengan menggunakan *query* (kata kunci) [1]. Dengan mengetikkan kata kunci, sistem atau aplikasi tertentu akan menampilkan dokumen-dokumen tersebut, biasanya terurut menurut tingkat relevansinya. Hal ini disebut Sistem Temu Kembali Informasi (*information retrieval*).

Sistem Temu Kembali Informasi merupakan sistem yang berfungsi untuk menemukan informasi yang relevan dengan kebutuhan pemakai. Salah satu hal yang perlu diingat adalah bahwa informasi yang diproses terkandung dalam sebuah dokumen yang bersifat tekstual. Dalam konteks ini, temu kembali informasi berkaitan dengan representasi, penyimpanan, dan akses terhadap dokumen representasi dokumen. Dokumen yang ditemukan tidak dapat dipastikan apakah relevan dengan kebutuhan informasi pengguna yang dinyatakan dalam *query* [2,3].

Salah satu model sistem temu kembali informasi yang paling sederhana namun produkif adalah model ruang vektor. Vektor model ini mempresentasikan *term* yang terdapat pada dokumen dan *query*. Elemen vektor tersebut adalah bobot term yang menjadi penilaian dan perankingan dokumen. Dalam model ruang vektor ini hal yang perlu diperhatikan adalah pembobotan *term* (*term weighting*) [4]. Metode pembobotan yang umumnya diunggulkan dalam beberapa penelitian menggunakan model ruang vektor yaitu *Term Frequency Inverse*

*Document Frequency* TF-IDF [4]. Dalam perhitungan bobot *term*, sekalipun *term frequency* banyak digunakan, namun hal itu hanya mendukung proporsi jumlah dokumen yang dapat ditemukan kembali oleh proses pencarian pada sistem temu kembali informasi. Sedangkan proporsi jumlah dokumen yang ditemukan dan dianggap relevan untuk kebutuhan pengguna akan lebih meningkat bila vektor bobot tersebut menggunakan *term* yang jarang muncul pada koleksi dokumen. *Term* demikian diharapkan mampu mengelompokkan sejumlah dokumen yang memuatnya, sehingga berbeda dengan seluruh anggota koleksi dokumen lain yang tidak memilikinya. Kriteria ini dapat diakomodasi dengan menghitung invers frekuensi dokumen. Dengan digabungkannya kedua metode ini yaitu frekuensi kemunculan *term* dan invers frekuensi yang mengandung kata tersebut, diharapkan mampu meningkatkan proporsi jumlah dokumen yang dapat ditemukan kembali dan yang dianggap relevan secara sekaligus. Sehingga kriteria *term* yang paling tepat adalah *term* yang sering muncul dalam dokumen secara individu, namun jarang dijumpai pada dokumen lainnya [5].

Dari penjelasan di atas maka dalam penelitian ini membuat suatu perangkat lunak yang dapat mencari dokumen-dokumen penulisan ilmiah dari mahasiswa Jurusan Matematika Universitas Diponegoro yang relevan sesuai tingkat pembobotannya. Aplikasi ini mencari dokumen pendek berupa abstrak tugas akhir berbahasa Indonesia dari mahasiswa Jurusan Matematika Universitas Diponegoro yang berformat teks (.txt). Query inputan yang digunakan adalah berupa kata, bukan simbol atau notasi tertentu. Aplikasi ini dibangun menggunakan perangkat lunak Borland Delphi 6.0.

## GAMBARAN UMUM SISTEM

Perangkat lunak yang dibangun, selanjutnya disingkat STKI merupakan sistem pencarian dokumen dalam bahasa Indonesia berupa Abstrak Tugas Akhir mahasiswa Jurusan Matematika Universitas Diponegoro. Proses yang berjalan dalam STKI adalah proses *indexing subsystem* dan *searching subsystem* (*macthing system*) [6].

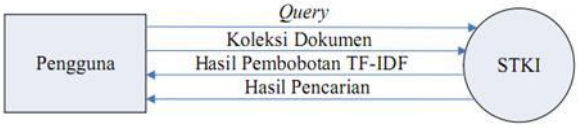
*Indexing* merupakan proses persiapan ulang dilakukan terhadap dokumen sehingga dokumen siap diproses. Proses *indexing* sendiri meliputi 2 proses, yaitu *document indexing* dan *term indexing*. Dalam proses *document indexing*, koleksi dokumen yang digunakan adalah kumpulan Abstrak Tugas Akhir dan *term indexing* merupakan *query* yang diinputkan oleh pengguna. Dari *term indexing* akan dihasilkan koleksi kata yang akan digunakan untuk meningkatkan performasi pencarian pada tahap selanjutnya.

*Search subsystem* (*matching*) merupakan proses menemukan kembali informasi (dokumen) yang relevan terhadap query yang diberikan. Tidak semua dokumen yang diambil (*retrieved*) oleh sistem merupakan dokumen yang sesuai dengan keinginan user (*relevant*).

## PEMODELAN FUNGSIONAL

### 1. Data Context Diagram

Data Context Diagram (DCD) aplikasi STKI dapat dilihat pada Gambar 1.

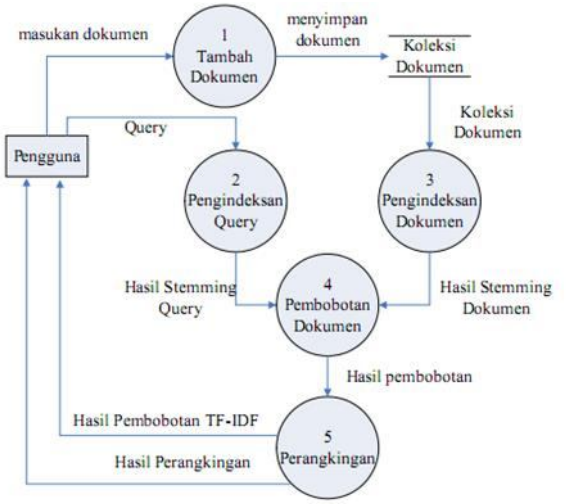



**Gambar 1.** DCD Aplikasi STKI

Pada Gambar 1 di atas menjelaskan bahwa pengguna dapat memberikan inputan kepada sistem berupa *query* dan koleksi dokumen yang akan diproses pembobotannya.

### 2. DFD Level 1

DFD Level 1 di Gambar 2 merupakan *breakdown* dari Gambar 1.



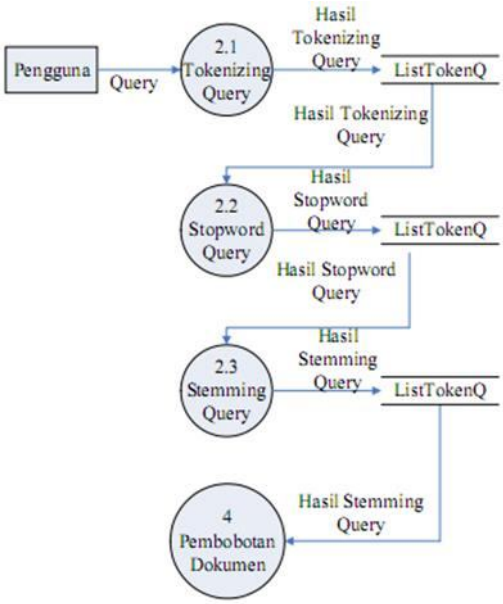

**Gambar 2.** DFD Level 1

Pada DFD level 1, dipecah menjadi proses-proses kecil yang berguna untuk menjelaskan fungsi-fungsi dan arus data yang mengalir pada sistem. Berikut ini adalah fungsi-fungsi yang terdapat pada aplikasi perangkat lunak ini:

1. Tambah Dokumen, yaitu proses menambahkan dokumen kemudian sitem akan menyimpan dalam database koleksi dokumen.
2. Pengindeksan *Query*, yaitu proses pengolahan *query* masukan dengan menggunakan tiga tahapan *tokenizing, stopword removal*, stemming untuk mendapatkan *term query* setelah mendapat masukkan dari pengguna yang berupa query masukan.
3. Pengindeksan Dokumen, yaitu proses pengolahan dokumen masukan dengan menggunakan tiga tahapan *tokenizing, stopword removal*, stemming untuk mendapatkan *term* koleksi dokumen setelah mendapat masukkan dari pengguna yang berupa koleksi dokumen. Untuk proses stemming mengacu pada stemming dari hasil penelitian Fadillah Z Tala [7].
4. Pembobotan Dokumen, yaitu proses penghitungan bobot masing-masing dokumen dengan menggunakan metode pembobotan TF-IDF untuk mendapatkan bobot nilai dari masing-masing dokumen sesuai dengan *query* masukan dari pengguna
5. Perangkingan, yaitu proses penentuan urutan dari dokumen-dokumen relevan yang akan diberikan kepada pengguna, perangkingan ini berdasarkan pada besarnya nilai bobot setiap dokumen sebagai ukuran tingkat relevansi dokumen tersebut terhadap *query*.

## 3. DFD Level 2

DFD Level 2 terjadi hanya pada proses pengindeksan *query* dan proses pengindeksan dokumen. Gambar 3 menunjukkan DFD Level 2 proses pengindeksan query, sedangkan Gambar 4 menunjukkan DFD Level 2 proses pengindeksan dokumen.

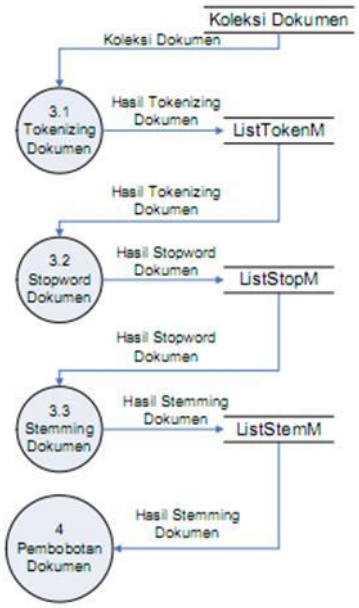



**Gambar 3.** DFD Level 2 Proses Pengindeksan *Query*

Pada proses pengindeksaan *query* terdapat tiga tahapan proses, yaitu:

1. *Tokenizing Query*, yaitu tahap pemotongan string inputan dari query berdasarkan kata yang menyusunnya. Eleman teks (string input) dipisahkan dengan teknik token menggunakan fungsi split dimana pemisahan string dilakukan berdasarkan *white space* (space dan tab) untuk kemudian diletakkan pada array.
2. *Stopword Removal Query*, yaitu tahap penghilangan kata-kata yang tidak digunakan dalam proses pencarian dengan mencocokan hasil dari tahap tokenizing dengan daftar stoplist, sehingga kata yang dibuang tidak akan dimasukan dalam proses selanjutnya.

3. *Stemming Query*, yaitu tahap penghilangan imbuhan sehingga didapatkan kata dasar dari *query* inputan.



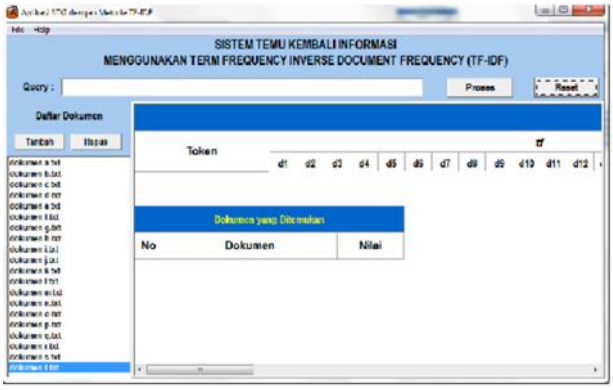

**Gambar 4.** DFD Level 2 Proses Pengindeksan Dokumen

Pada proses pengindeksaan dokumen terdapat tiga tahapan proses, yaitu:

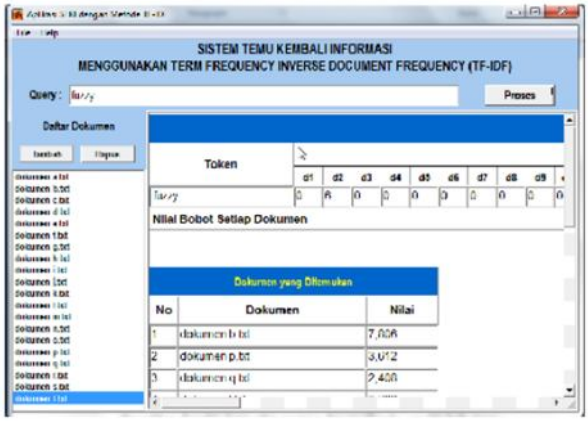
1. *Tokenizing* Dokumen, yaitu tahap pemotongan string inputan dari koleksi dokumen berdasarkan kata yang menyusunnya. Eleman teks (string input) dipisahkan dengan teknik token menggunakan fungsi split dimana pemisahan string dilakukan berdasarkan *white space* (*space* dan tab) untuk kemudian diletakkan pada array.
2. *Stopword Removal* Dokumen, yaitu tahap penghilangan kata-kata yang tidak digunakan dalam proses pencarian dengan mencocokan hasil dari tahap *tokenizing* dengan daftar *stoplist*, sehingga kata yang dibuang tidak akan dimasukan dalam proses selanjutnya.
3. Stemming Dokumen, yaitu tahap penghilangan imbuhan sehingga didapatkan kata dasar dari *term-term* dokumen inputan.

## IMPLEMENTASI SISTEM

Gambar 5 sampai Gambar 12 merupakan hasil implementasi dari perancangan aplikasi yang dibuat dengan menggunakan perangkat lunak Borland Delphi 6.0.

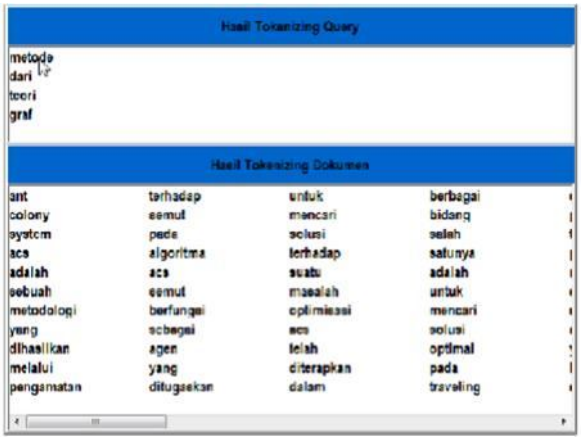



**Gambar 5.** Form Utama

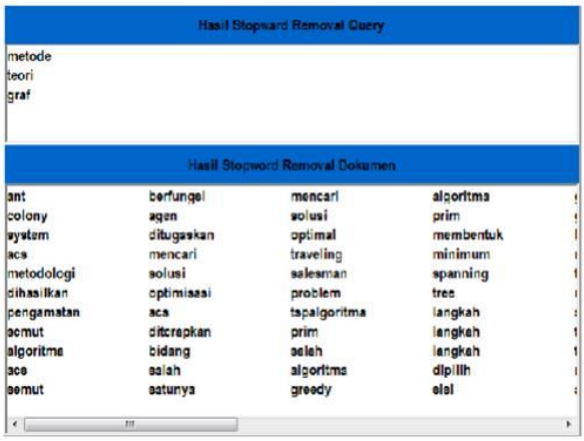



**Gambar 6.** Form Hasil

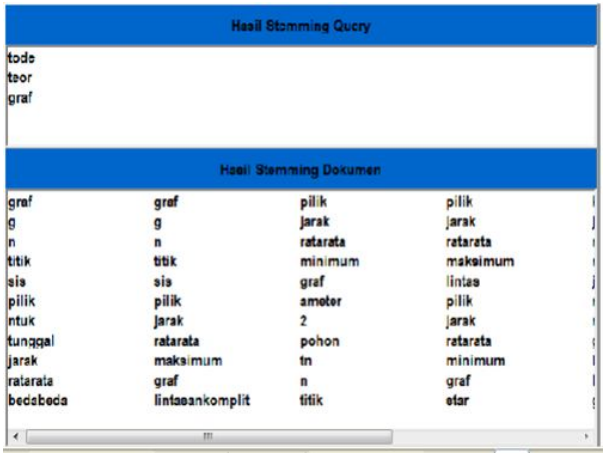

**Gambar 7.** Form Pengindeksan



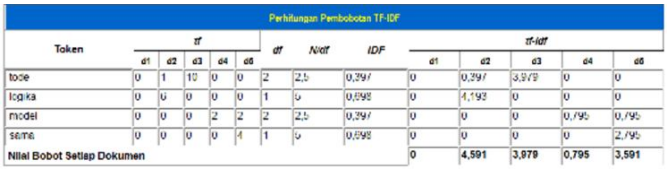

**Gambar 8.** Antarmuka Hasil *Tokenizing Query* dan Dokumen




**Gambar 9.** Antar muka Hasil Stopword Removal *Query* dan Dokumen




**Gambar 10.** Antar Muka Hasil Stemming *Query* dan Dokumen




**Gambar 11.** Antar Muka Perhitungan Pembobotan TF-IDF


**Gambar 12.** Antar Muka Hasil Perangkingan

## PENGUJIAN

Pada pelaksanaan pengujian ini akan digunakan koleksi data uji berupaAbstrak Tugas Akhir Mahasiswa S1 Matematika Universitas Diponegoro tahun 2004-2006 sebanyak 20 dokumen. Selama proses pelaksanaan pengujian perangkat lunak ini, beberapa langkah yang harus ditempuh, yaitu:

1. Memasukkan/menambah dokumen.
   Memasukkan atau menambah dokumen dengan menekan tombol "Tambah". Batasan-batasan dokumen yang digunakan adalah dokumen dalam bahasa Indonesia berupa Abstrak Tugas Akhir mahasiswa Jurusan Matematika Universitas Diponegoro,yang berformat file teks (*.txt).
2. Masukkan *query*
   *Query* yang digunakan sebagai pengujian sebanyak 5 *query*, yaitu Algoritma genetika, Graf G, Persamaan diferensial, Persamaan diferensial non linier dan Sistem.
3. Selanjutnya jalankan aplikasi dengan menekan tombol "Proses".

Dalam aplikasi Temu Kembali Informasi untuk mengetahui efektifitas sebuah sistem perlu dilakukan pengukuran performansi. Pengukuran dilakukan dengan menghitung nilai *precision* dan *recall*. Dalam perhitungan *precision* perlu memperhatikan probabilitas jumlah dokumen yang relevan dari semua dokumen yang ter-*retrieve* dibandingkan dengan jumlah keseluruhan dokumen yang ter-*retrieve*. Sedangkan pada perhitungan *recall*

perlu memperhatikan probabilitas dokumen relevan yang ter-*retrieve* dibandingkan dengan jumlah dokumen-dokumen yang relevan [8].

### 1. Hasil Pengujian

Pada pengujian terhadap koleksi data uji yang terdiri dari 20 dokumen Abstrak Tugas Akhir Mahasiswa S1 Matematika Universitas Diponegoro tahun 2004-2006 dan 5 *query*, diperoleh perhitungan nilai *precision* dan *recall* seperti dalam Tabel 1.

**Tabel 1.** Perhitungan Precision dan Recall

| Query | Algoritma genetika | Graf G | Persamaan Diferensial | Persamaan Diferensial Non Linier | Sistem |
|---|---|---|---|---|---|
| dokumen relevan yang ter-retrive | 1 | 4 | 3 | 3 | 6 |
| dokumen relevan yang tidak ter-retrive | 0 | 0 | 0 | 0 | 1 |
| dokumen tidak relevan yang ter-retrive | 5 | 2 | 3 | 7 | 0 |
| Precision | 1/6=0,17 | 4/6=0,67 | 3/6=0,50 | 3/10 = 0,3 | 6/6 =1 |
| Recall | 1/1 = 1 | 4/4=1 | 3/3=1 | 3/3=1 | 6/7=0,86 |

## ANALISIS HASIL PENGUJIAN

Metode pembobotan dokumen TF-IDF tidak selalu memberikan hasil performansi yang baik pada koleksi pengujian, sebab tidak semua dokumen ter-*retrive* merupakan dokumen yang relevan. Hal ini dapat terjadi karena sebuah *term* yang muncul dibanyak dokumen tidak dianggap lebih penting daripada *term* lain yang hanya muncul pada sedikit dokumen, namun jika sebuah *term* memiliki frekuensi kemunculan lebih tinggi dalam sebuah dokumen daripada dalam dokumen lain, maka dokumen tersebut dianggap lebih penting. Sifat ini merupakan kelemahan dari Metode TF-IDF, ditambah dengan karakteritik model ruang vektor yang simetris hingga pada akhirnya akan memberikan nilai output yang sama.

Dari Tabel 1 dapat dilihat, dengan memperhatikan hasil *recall* yang berrata-rata 97,2 % mengindikasikan bahwa dokumen yang relevan memungkinkan untuk selalu ter-*retrieve*. Di sisi lain, banyak juga dokumen yang tidak relevan ikut ter-*retrive*. Hal ini kemungkinan disebabkan karena term yang menjadi *query* bukanlah kata kunci yang unik untuk dokumen-dokumen ilmiah pada abstrak Tugas Akhir mahasiswa Jurusan Matematika. Namun demikian, nilai presisi ini tidak dapat dijadikan patokan karena jumlah informasi yang dapat bertambah dan di-update setiap waktunya. Peng-update-an ini menambah jumlah dokumen yang di-*retrieve* dan mungkin menambah jumlah dokumen yang relevan. Walaupun terkadang, penambahan ini cenderung menurunkan tingkat presisi suatu *search engine* karena jumlah dokumen yang harus di-*retrieve* pasti jauh lebih banyak dibandingkan dengan dokumen yang relevan. Nilai *recall* merupakan ukuran sensitivitas suatu STKI dalam menemukan informasi-informasi yang sebenarnya relevan dengan *query* yang dimasukkan. Semakin besar nilai *precision* dan *recall* maka semakin baik sistem.

## KESIMPULAN

Kesimpulan yang dapat ditarik dari penelitian ini adalah metode pembobotan dokumen TF-IDF dapat me-*retrieve* dokumen sesuai dengan *query* pengguna. Metode pembobotan dokumen TF-IDF tidak selalu memberikan hasil performansi yang baik pada koleksi pengujian, dikarenakan tidak semua dokumen ter-retrive merupakan dokumen yang relevan.

## DAFTAR PUSTAKA

[1] Herwansyah, Adhit.____. Aplikasi Pengkategorian Dokumen dan Pengukuran Tingkat Similaritas Dokumen Menggunakan Kata Kunci Pada Dokumen Penulisan Ilmiah Universitas Gunadarma. Jakarta: Universitas Gunadarma.

[2] Baeza-Yates dan Ribeiro-Neto, 1999. Introduction of informasi retrieval. Addison Wesley.

[3] Tarto. 2008. Konsep Dasar Sistem Temu Kembali Informasi. [Online] Tersedia : http://tartojogja.wordpress.com/2008/09/23/konsep-dasar-sistem-temu-kembali-informasi/. Tanggal Akses: 3 November 2010.

[4] Arifin, A. 2002. Penggunaan Digital Tree Hibrida pada Aplikasi Information Retrieval untuk Dokumen Berita. Surabaya, Indonesia: Institut Teknologi Sepuluh Nopember.

[5] Husni.2010. Sistem Temu Kembali Informasi. Universitas Trunojoyo.

[6] William B. Frakes dan Richardo B. Yates.1992. Information Retrieval : Data Structures and Algorithms. New Jersey: Prentice-Hall.
[7] Tala, Fadillah Z., 2003. A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia. Institute for Logic, Language and Computation Universeit Van Amsterdam.
[8] Desrianti, Gita. 2010. Akurasi dalam Pencarian pada Searchs Engines. Bandung: Institut Teknologi Bandung.