PERANCANGAN SISTEM TEMU BALIK INFORMASI DENGAN METODE PEMBOBOTAN KOMBINASI TF-IDF UNTUK PENCARIAN DOKUMEN BERBAHASA INDONESIA

Meisya Fitri

Program Studi Teknik Informatika
Jurusan Teknik Elektro Fakultas Teknik Universitas Tanjungpura
meisya.fitri@yahoo.co.id

Abstrak - **Information will be easily to access especially from internet that has increased the number of various of digital information. It has increased need of user about information retrieval tools, which it can describe the most relevant document based on user question. Information retrieval can support in managing and finding information effectively and efficiently. It designed to handle searching based on keyword from user. This implemented query expansion technique and term weighting method with combination of TF-IDF (*term frequency-inverse document frequency*). Query expansion is a method that increases the term in query for improving search performance. Term weighting based on combination of TF-IDF gives more weight for more impotant term. In this system, searching query add based on synonim of searching keywords that the result is not only document which have keywords, but also document associated with searching keyword based on thesaurus and root word. This system can find all of relevant document from available collection. Searching result sort based on document weight toward to keywords that depict the similarity of document with user question that it can be facilitating user to find information properly.**

Keywords - information retrieval system, natural language processing, term weighting, TF-IDF, query expansion

1. Pendahuluan

Kemudahan dalam mendapatkan informasi khususnya melalui internet, mendorong pertambahan jumlah informasi digital menjadi semakin banyak dan beragam. Hal ini mengakibatkan meningkatnya kebutuhan pengguna akan perangkat pencarian informasi yang efektif dan efisien sehingga menghasilkan dokumen yang paling relevan berdasarkan pertanyaan pengguna.

Pencarian informasi atau yang dikenal dengan istilah sistem temu balik informasi (*Information Retrieval*) bertujuan menghasilkan dokumen yang paling relevan berdasarkan *keyword* pada *query* yang diberikan pengguna. Dokumen dianggap relevan jika suatu dokumen cocok dengan pertanyaan pengguna. Akan tetapi, istilah-istilah yang terdapat dalam dokumen dan *query* sering memiliki banyak varian morfologik, sehingga pasangan istilah seperti "bergeser" dan "menggeser" tidak akan dianggap ekuivalen oleh sistem tanpa suatu bentuk *Natural Language Processing* (NLP).

Pembobotan dasar dilakukan dengan menghitung frekuensi kemunculan istilah dalam dokumen karena dipercaya bahwa frekuensi kemunculan istilah merupakan petunjuk sejauh mana istilah tersebut mewakili isi dokumen [1]. Pembobotan kata diharapkan dapat menemukan kembali informasi yang paling relevan dengan indeks istilah terbaik. Metode pembobotan kata berdasarkan kombinasi TF-IDF memberikan bobot lebih kepada istilah yang lebih penting. Dengan menggunakan metode pembobotan kata dapat diperoleh informasi penting dari suatu dokumen berdasarkan kata-kata yang ada dalam dokumen tersebut.

Penambahan istilah pada *query* juga diperlukan untuk meningkatkan performa dalam *Information Retrieval* (IR) atau dikenal dengan istilah *Query Expansion* atau perluasan *query* [2]. Dalam konteks *web search engine*, hal ini termasuk evaluasi input user dan memperluas *query* pencarian.

Informasi berupa dokumen teks yang tersedia di internet memiliki banyak variasi. Dalam penelitian ini, pencarian dilakukan terhadap dokumen hasil *crawling* dari beberapa situs yang sudah ditentukan. Fokus utama penelitian ini, berupa dokumen hasil *crawling* dari situs berita, tanpa menutup kemungkinan untuk melakukan pencarian terhadap dokumen hasil *crawling* dari situs lain.

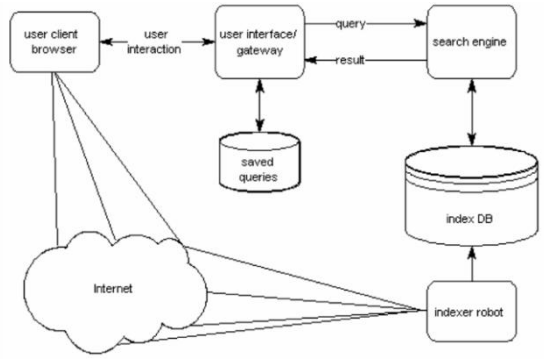
2. Teori Dasar

Sistem temu balik informasi bertujuan menghasilkan dokumen yang paling relevan berdasarkan *keyword* pada *query* yang diberikan pengguna. Sistem temu balik informasi ini digunakan untuk mengurangi informasi yang terlalu banyak sehingga sulit untuk dikelola.

Sistem temu balik informasi terdiri dari tiga komponen utama, yaitu masukan (*input*), pemroses (*processor*), dan keluaran (*output*). Pengguna adalah pemilik kebutuhan akan informasi, yang kemudian menerjemahkan kebutuhannya menjadi sebuah *query*.

Input harus berupa representasi yang tepat dari setiap dokumen dan *query* agar dapat diolah oleh pemroses. Pemroses (*Processor*) bertugas menstrukturkan informasi dalam bentuk yang tepat, misalnya dengan pengindeksan dan klasifikasi serta melakukan proses temu balik, yaitu dengan menjalankan suatu strategi pencarian sebagai respon dari *query*. *Output* adalah keluaran yang diberikan oleh pemroses, biasanya berbentuk informasi tentang suatu dokumen, dokumen itu sendiri, dan acuan ke dokumen lain (*citation*).

Penggabungan Hiperteks dan temu-kembali informasi dapat memecahkan masalah-masalah dalam bidang temu-kembali informasi [3]. Misalnya, sistem temu-kembali informasi yang didasarkan pada penggunaan operator *boolean*, mengandalkan kemampuan pemakai dalam memformulasikan *query*. Dengan adanya sistem hiperteks, hal ini dapat dipermudah dengan penyediaan antarmuka yang memakai pencarian dengan metode *browsing*. Menurut Hasibuan, penelitian yang dilakukan Budi Yuwono (1995) menggunakan rancangan arsitektur seperti pada gambar 1.



Gambar 1. Arsitektur Sistem Temu Balik Informasi [3]

Arsitektur yang dirancang ini terdiri dari dua komponen utama yaitu: *Index Builder* dan *Search Engine*. *Index builder* merupakan sebuah sistem pengindeksan yang memanfaatkan "*robot*" atau "*crawler*" yang berkomunikasi dengan menggunakan HTTP (*Hypertext Transfer Protocol*) untuk mencari informasi yang akan diindeks.

Sedangkan *search engine* atau mesin pencari merupakan teknik dari temu-kembali dalam menemukan dokumen dan sekaligus mengeksekusi algoritma peringkat dalam menampilkan dokumen. Pengguna dapat mencari halaman *web* yang dibutuhkan melalui *search engine*. *Search engine* tidak lain sebuah mesin pencari yang ulet dan teliti, yang melakukun eksplorasi atas informasi-informasi yang di-*request* tanpa memandang kapan, di mana dan oleh siapa itu dilakukan [4]. Sedangkan komunikasi antara pemakai dan *search engine* dalam memformulasikan *query* dilakukan melalui *user interface*. Setelah pemakai menemukan dokumen yang relevan dengan *query*, dapat langsung melakukan *browsing* ke sumber informasi dalam hal ini adalah alamat tempat www.

Mesin pencari menggunakan indeks (yang sudah dibuat dan disusun secara teratur) untuk mencari file setelah pengguna memasukkan kriteria pencarian. Informasi yang ditampilkan mengandung atau berhubungan dengan suatu istilah spesifik.

Crawler merupakan program yang berjalan secara otomatis, berisi *script* program yang melakukan *crawling* melalui halaman *website* untuk mengumpulkan data berdasarkan indeks dari halaman *web* yang ditemukan [5]. Tujuan dari *crawler* adalah dengan cepat dan efisien mengumpulkan banyak informasi dari halaman *web* yang berguna, berikut dengan struktur *link* yang terkoneksi dengan halaman *web* tersebut. Mesin pencari menggunakan *crawler* untuk mengumpulkan informasi yang terdapat pada halaman *website* sehingga ketika pengguna internet memasukkan kata kunci pencarian dapat dengan cepat memberikan informasi yang relevan kepada user.

Web crawler dimulai dengan sekumpulan URL, kemudian mendownload setiap halamannya, mendapatkan link dari setiap halaman yang dikunjungi kemudian mengulangi kembali proses *crawling* pada setiap *link* halaman tersebut. Proses *crawling* berlangsung terus sampai antrian URL kosong atau kalau kondisi berhenti sudah terpenuhi. Penelusuran dapat difokuskan pada kedalaman (*depth-first spidering*) atau pada keluasan (*breadth first spidering*) situs.

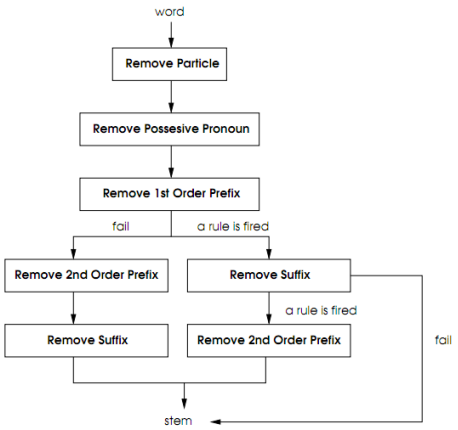
Indexing atau pengindeksan merupakan proses membangun basis data indeks dari koleksi dokumen. *Indexing* dilakukan terhadap dokumen sebelum pencarian dilakukan. Dewasa ini, sistem pengindeksan secara manual mulai digantikan oleh sistem pengindeksan otomatis. Sistem pengindeksan otomatis dapat dilakukan dengan implementasi *Natural Language Processing* (NLP). NLP bertujuan untuk memahami arti yang diberikan dalam bahasa alami dan memberikan respon yang sesuai, misalnya dengan melakukan suatu aksi tertentu atau menampilkan data tertentu [6].

Adapun tahapan dari pengindeksan adalah sebagai berikut:

1. *Parsing* dokumen yaitu proses pengambilan kata-kata dari kumpulan dokumen.
2. *Stoplist* yaitu proses pembuangan kata buang seperti: tetapi, yaitu, sedangkan, dan sebagainya.
3. *Stemming* yaitu proses penghilangan/ pemotongan dari suatu kata menjadi bentuk dasar.
4. *Term Weighting* dan *Inverted File* yaitu proses pemberian bobot pada istilah.

Parsing merupakan proses pengambilan kata-kata dari kumpulan dokumen. *Stemming* adalah salah satu cara yang digunakan untuk meningkatkan performa IR. *Stemming* digunakan untuk mencari kata dasar dari bentuk berimbuhan. Algoritma *stemming* untuk bahasa yang satu berbeda dengan algoritma *stemming* untuk bahasa lainnya. Bahasa Inggris memiliki morfologi yang berbeda dengan Bahasa Indonesia sehingga memiliki algoritma *stemming* juga berbeda. Proses *stemming* pada teks berbahasa Indonesia lebih rumit/kompleks karena terdapat variasi imbuhan yang harus dibuang untuk mendapatkan *root word* dari sebuah kata. Algoritma *stemming* untuk Bahasa

Indonesia telah dikembangkan, seperti Algoritma Porter untuk teks berbahasa Indonesia yang dikembangkan oleh Tala, Algoritma Nazief & Adriani untuk teks berbahasa Indonesia. Adapun langkah-langkah algoritma ini adalah sebagai berikut [7]:



Gambar 2. Porter *Stemming* untuk Bahasa Indonesia

Dalam pencarian yang dilakukan oleh sistem temu balik informasi, semua istilah yang dicari tidak memiliki bobot yang sama. Untuk itulah dibutuhkan metode pembobotan kata agar pencarian lebih mudah difokuskan. *Term weighting* atau pembobotan kata dilakukan dengan menghitung frekuensi kemunculan istilah dalam dokumen karena dipercaya bahwa frekuensi kemunculan istilah merupakan petunjuk sejauh mana istilah tersebut mewakili isi dokumen [1]. Metode pembobotan kata berdasarkan kombinasi TF-IDF memberikan bobot lebih kepada istilah yang lebih penting.

Dalam penelitian ini, akan digunakan metode pembobotan TF-IDF (*term frequency-inverse document frequency*). Mengenai efektivitas kinerja dari sebuah *search engine* selalu dikaitkan dengan tingkat relevansi hasil pencarian. Untuk menemukan dokumen yang relevan, metode pembobotan TF-IDF memberikan bobot lebih kepada istilah yang lebih penting. Istilah yang lebih penting yang dimaksud adalah istilah yang jika muncul pada sebuah dokumen maka dokumen tersebut dapat dianggap relevan dengan *query*.

TF adalah algoritma pembobotan heuristik yang menentukan bobot dokumen berdasarkan kemunculan *term* (istilah). Semakin sering sebuah istilah muncul, semakin tinggi bobot dokumen untuk istilah tersebut, dan sebaliknya. Terdapat empat buah algoritma TF yaitu *Raw TF*, *Logarithmic TF*, *Binary TF*, *Augmented TF* [8]. Dalam penelitian ini digunakan algoritma *Raw TF*. *Raw TF* diperoleh dari perhitungan frekuensi kemunculan suatu istilah pada dokumen.

IDF merupakan banyaknya istilah tertentu dalam keseluruhan dokumen, dapat dihitung dengan persamaan:

$$idfj = \log \frac{n}{nj} \qquad (1)$$

n = jumlah seluruh dokumen
nj = jumlah dokumen yang mempunyai istilah j

Dokumen-dokumen yang ditampilkan oleh sistem temu balik informasi harus memenuhi persyaratan *recall*, *precision* dan NIAP (*Non Interpolated Average Precision*) [8]. *Recall* didefinisikan dengan menemukan seluruh dokumen yang relevan dalam koleksi dokumen.

$$\text{Recall} = \frac{\text{jumlah dokumen relevan ditemukan}}{\text{jumlah dokumen relevan dalam koleksi}} \qquad (2)$$

Nilai *recall* tertinggi adalah 1, yang berarti seluruh dokumen dalam koleksi berhasil ditemukan. *Precision* didefinisikan dengan menemukan hanya dokumen yang relevan saja dalam koleksi.

$$\text{Precision} = \frac{\text{jumlah dokumen relevan ditemukan}}{\text{jumlah dokumen ditemukan}} \qquad (3)$$

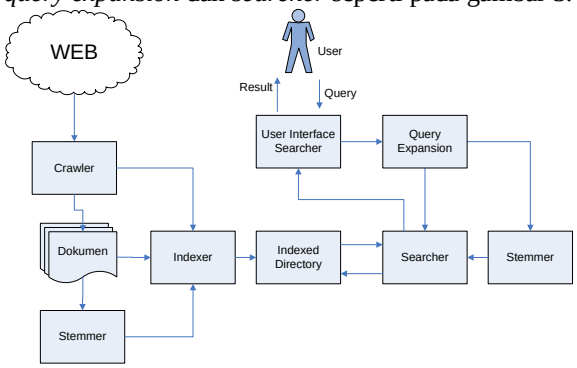
Nilai *precision* tertinggi adalah 1, yang berarti seluruh dokumen yang ditemukan adalah relevan. NIAP adalah penggabungan dari *recall* dan *precision*.

$$\text{NIAP} = \sum_{i=1}^{n} \frac{Precision \text{ pada dokumen ke}-\text{i}}{\text{Jumlah dokumen relevan dalam koleksi}} \qquad (4)$$

Di mana n menunjukkan jumlah dokumen yang dicari hingga seluruh dokumen relevan ditemukan. Nilai NIAP tertinggi adalah 1, yang berarti seluruh dokumen relevan berhasil ditemukan dengan seluruh dokumen relevan tersebut ditempatkan pada urutan teratas dalam hasil pencarian. Nilai NIAP akan digunakan untuk mengecek kebenaran hasil pencarian dari perangkat lunak yang dibangun.

3. Hasil Eksperimen

Secara garis besar, sistem terdiri dari lima komponen utama yaitu *crawler*, *indexer*, *stemmer*, *query expansion* dan *searcher* seperti pada gambar 3.



Gambar 3 Desain Arsitektur Sistem

Proses yang dilakukan *crawler* akan menghasilkan dokumen berupa metadata yang dikumpulkan dari beberapa situs. Dokumen yang berupa metadata akan melalui proses *parsing* sehingga berita akan terekstraksi menjadi dokumen. *Document parsing* berupa pengambilan kata-kata dalam dokumen hasil *crawling* yang digunakan untuk memudahkan dalam ekstraksi kata. *Crawler* juga mengumpulkan link dari metadata dan disimpan dalam dokumen untuk dijelajahi kemudian.

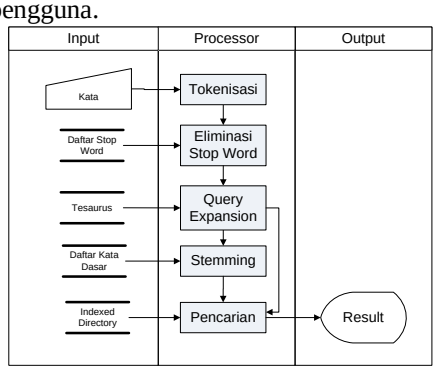
Dalam *indexing* yang dilakukan *indexer*, terdiri atas dua (2) bagian yaitu *document pre-processing* dan pembobotan kata pada masing-masing dokumen. *Document pre-processing* melakukan ekstraksi kata

dari dokumen (tokensisasi dokumen), eliminasi *stop words* dan *stemming*. Proses eliminasi *stop words* digunakan untuk menghilangkan kata-kata buang yang memiliki frekuensi kemunculan yang tinggi seperti: tetapi, yaitu, sedangkan, dan sebagainya. Proses *stemming* digunakan untuk mencari bentuk kata dasar dari kata dalam dokumen untuk selanjutnya dihitung bobotnya. Dalam proses *stemming*, digunakan algoritma porter untuk teks berbahasa Indonesia yang dikembangkan oleh Tala.

Dalam sistem yang dirancang, digunakan kata dasar yang berasal dari kamus bahasa Indonesia [9] dan tesaurus yang berasal dari tesaurus bahasa Indonesia pusat bahasa [10] dari pusat bahasa departemen pendidikan nasional yang diterbitkan tahun 2008 versi elektronik. Kata dasar dan tesaurus dari pusat bahasa berupa file .pdf yang kemudian telah dikonversi menjadi file .babylon oleh Steven Haryanto (http://steven.haryanto.org). File .babylon kemudian dapat dibaca sebagai file .txt untuk diinputkan ke dalam database.

Dalam pengindeksan dokumen digunakan metode pembobotan kata berdasarkan TF, IDF dan kombinasi keduanya.

Proses pencarian merupakan proses menemukan kembali informasi (dokumen) yang relevan terhadap *query* kata kunci yang diberikan oleh pengguna dan menyediakan daftar dokumen terurut yang relevan dengan *query* tersebut. Dalam proses pencarian yang dilakukan *searcher*, *user* akan memasukkan kata yang akan dicari. Kata tersebut akan melewati beberapa proses sehingga menjadi kata kunci pencarian. Proses tersebut yaitu proses ekstraksi kata (tokenisasi), eliminasi kata-kata yang tidak layak dijadikan kata kunci pencarian (*stop words*), *query expansion* dan *stemming* untuk memperkuat pencarian. Daftar dokumen yang memiliki keterkaitan akan diurutkan berdasarkan frekuensi kemunculan kata kunci. Daftar terurut inilah yang akan dikembalikan kepada pengguna.

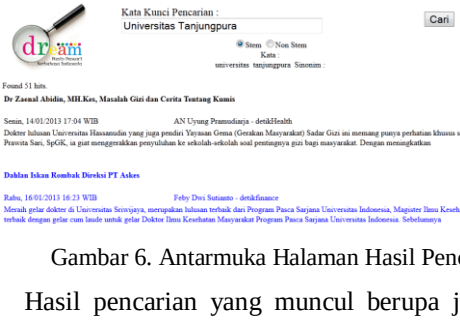


Gambar 4 Metode *Searching* Dokumen

Sistem yang telah dirancang menggunakan JSP sebagai bahasa pemrograman, Apache Tomcat sebagai *web server*, MySql sebagai pengelola database serta JDK dan JRE sebagai *tools* pendukung. *User* dan Admin dapat mengakses sistem dengan *web browser* seperti Mozila Firefox dan Internet Explorer. Antarmuka halaman utama sistem ini dapat dilihat pada gambar 5.

Gambar 5. Antarmuka Halaman Utama untuk *User*

Antarmuka halaman utama untuk user berisi fitur pencarian yang dapat digunakan *user* untuk mencari berita. Pada bagian paling bawah terdapat 2 pilihan fitur pencarian antara lain stem dan non stem. Hasil eksekusi pencarian akan muncul halaman hasil pencarian. Antarmuka halaman hasil pencarian berisi *header* dan hasil pencarian. *Header* berisi kata kunci pencarian, 2 fitur pencarian, kata kunci yang dicari dan sinonim dari kata kunci yang dicari sebagai fitur *query expansion*.



Gambar 6. Antarmuka Halaman Hasil Pencarian

Hasil pencarian yang muncul berupa judul berita dan potongan isi berita. Hasil pencarian yang ditampilkan terurut berdasarkan pembobotan tertinggi. Hasil pencarian disajikan sebagai link halaman sehingga ketika ingin membuka halaman sumber berita dapat mengklik langsung dibagian hasil pencarian.

Form crawling berisi perintah untuk memulai dan menghentikan *crawling*. *Crawling* akan dilakukan terhadap situs yang sudah ditentukan sebelumnya. Data hasil *crawling* akan disimpan dalam *database* bernama dokumen. Antarmuka halaman manajemen *crawler* dapat dilihat pada gambar 7.

Gambar 7. Antarmuka Halaman Manajemen *Crawler*

Halaman admin merupakan halaman yang digunakan untuk memanajemen pencarian baik memanajemen tabel pendukung dalam pencarian dan tabel admin maupun memanajemen proses *crawling* dan *indexing*. Halaman manajemen pencarian berisi informasi mengenai *indexing* dan *form indexing*. *Form indexing* berisi pilihan tipe *indexing* yang akan digunakan yaitu stem atau non stem serta tombol untuk memulai *indexing*. *Indexing* akan dilakukan ke dalam

folder yang dinamai dengan tanggal dan waktu *indexing* serta tipe yang digunakan. Antarmuka halaman manajemen pencarian dapat dilihat pada gambar 8.

Gambar 8. Antarmuka Halaman Manajemen Pencarian

Pada pengujian digunakan data hasil *crawling* sebanyak masing-masing 20 berita dari 3 *website*. Sehingga jumlah dokumen yang digunakan adalah 30 berita. Pengujian dilakukan terhadap 5 macam kata kunci. Dokumen yang digunakan memiliki variasi jumlah kata dan halaman. Hal ini akan mempengaruhi bobot dari setiap dokumen pada saat pencarian.

Tabel 1. Pengujian Pembobotan Kata dengan *Indexing* Mode Stem

Kata Kunci	*Indexing* Mode Stem					
	Recall		Precision		NIAP	
	S	NS	S	NS	S	NS
Penculikan anak	1	1	0.1316	0.1429	0.2282	0.2184
Pendidikan SMA	1	1	0.2778	0.3125	0.835	0.9667
Korupsi	1	1	1	1	1	1
Dahlan Iskan	1	1	1	1	1	1
Banjir Jakarta	1	1	0.3488	0.3488	0.7131	0.7323

Tabel 2. Pengujian Pembobotan Kata dengan *Indexing* Mode Non Stem

Kata Kunci	*Indexing* Mode Stem					
	Recall		Precision		NIAP	
	S	NS	S	NS	S	NS
Penculikan anak	1	1	0.1429	0.1471	0.2108	0.2308
Pendidikan SMA	1	1	0.3333	0.3333	1	0.9429
Korupsi	1	1	0.5000	1	1	1
Dahlan Iskan	1	1	1	1	1	1
Banjir Jakarta	1	1	0.3659	0.3659	0.8956	0.7763

Dari perhitungan yang dilakukan pada tabel 1 dan tabel 2 didapatkan nilai *Recall*, *Precision* dan NIAP dari setiap kata kunci. Setiap kata kunci baik dalam pencariannya terhadap data *indexing* mode stem maupun data *indexing* mode non stem menghasilkan nilai *recall* 1 yang berarti semua dokumen relevan ditemukan dalam pencarian. Sebaliknya, Nilai perhitungan *precision* pada setiap kata kunci menghasilkan nilai yang bervariasi. Semakin tinggi nilai *precision* yang dihasilkan berarti semakin banyak dokumen relevan dalam dokumen yang ditemukan. Nilai *precision* memiliki nilai tertinggi 1 yang berarti seluruh dokumen yang ditemukan adalah relevan. Dari pengujian terhadap 5 kata kunci, didapatkan bahwa nilai *precision* akan meningkat ketika digunakan pada kata kunci non stem.

Nilai NIAP tertinggi adalah 1, yang berarti seluruh dokumen relevan berhasil ditemukan dengan seluruh dokumen relevan tersebut ditempatkan pada urutan teratas dalam hasil pencarian. Nilai NIAP yang didapatkan dalam pengujian memiliki nilai bervariasi. Nilai NIAP yang dihasilkan terhadap 5 kata kunci mencapai angka 1 sehingga untuk kata kunci tertentu, sistem dapat mengurutkan semua dokumen relevan ke urutan teratas.

Pengujian kecepatan pencarian dilakukan pada aplikasi Dream Search Engine dan sebagai pembanding digunakan aplikasi Giggle Search Engine yang juga dibangun dengan java namun menggunakan teknik dasar pembobotan yang dibangun Lucene yaitu TF, IDF dan *normalization*. Hal ini akan mempengaruhi kecepatan eksekusi pada saat pencarian.

Tabel 3 Pengujian Kecepatan Pencarian dalam detik

Kata Kunci	Kecepatan Pencarian Dream Search Engine				Kecepatan Pencarian Giggle Search Engine			
	Indexing Mode Stem		Indexing Mode Non Stem		Indexing Mode Stem		Indexing Mode Non Stem	
	S	NS	S	NS	S	NS	S	NS
Pendidikan SMA	3.7	3.9	2.9	3.6	1.7	0.01	1.7	0.05
Penculikan Anak	4.2	4.4	4.0	4.4	1.7	1.7	1.7	0.04
Dahlan Iskan	0.4	0.2	0.2	0.3	1.5	0.02	1.5	0.04
Banjir Jakarta	3.8	4.0	3.6	3.9	1.3	0.02	1.3	0.05
Korupsi	0.6	0.3	0.4	0.3	1.7	0.01	2	0.04

Dari perhitungan yang dilakukan pada tabel 3 didapatkan bahwa kecepatan pencarian Dream Search Engine jauh dibawah kecepatan Giggle Search Engine terkecuali pada kata kunci "Dahlan Iskan" dan "Korupsi". Perbedaan yang signifikan ini terjadi dikarenakan teknik *query expansion* yang diimplementasikan dalam Dream Search Engine sehingga *query* yang dieksekusi untuk pencarian berita menjadi panjang dan lebih banyak bobot yang diperhitungkan, sedangkan pada kata kunci "Dahlan Iskan" tidak ditemukan sinonim kata dalam *query expansion* sehingga kecepatan pencarian mengungguli kecepatan Giggle Search Engine. Kecepatan pencarian juga unggul ketika digunakan kata kunci "Korupsi" yang memiliki sinonim antara lain curang, manipulasi, gelap dan seleweng. Hal ini dikarenakan sinonim yang berjumlah 4 kata tidak secara signifikan mempengaruhi kecepatan pencarian Dream Search Engine sehingga pencarian dapat dilakukan kurang dari 1 detik.

Kecepatan pencarian bertambah secara signifikan hingga lebih dari 3 detik pada kata kunci "Penculikan Anak", "Pendidikan SMA" dan "Banjir Jakarta". Ini dikarenakan sinonim yang dihasilkan oleh proses *query expansion* untuk kata kunci tersebut berjumlah banyak sehingga *query* menjadi panjang dan bobot yang harus diperhitungkan menjadi banyak.

4. Kesimpulan

Berdasarkan hasil analisis dan pengujian terhadap Sistem Temu Balik Informasi dengan Metode Pembobotan Kombinasi Tf-Idf yang digunakan untuk pencarian dokumen Berbahasa Indonesia, maka dapat ditarik kesimpulan sebagai berikut:

1. Sistem mampu mengumpulkan dokumen berita melalui proses *crawling website* dan memberikan bobot dengan mengimplementasikan metode pembobotan kata dengan metode kombinasi Tf-Idf secara lengkap sehingga lebih banyak data relevan yang dapat diperhitungkan dalam pencarian.
2. Sistem dapat melakukan proses pencarian dan menemukan informasi yang relevan berdasarkan hasil pengujian yang dilakukan pada 5 kata kunci. Data hasil pengujian menghasilkan nilai recall 1 yang menunjukkan bahwa semua dokumen yang relevan dapat ditemukan sistem dan nilai precision antara 0.1316 dan 1 yang menunjukkan terdapat dokumen lain selain dokumen relevan yang ikut ditemukan oleh sistem. Nilai NIAP yang dihasilkan mencapai nilai 1 yang menunjukkan sistem dapat mengurutkan dokumen relevan ke dalam urutan hasil pencarian teratas.
3. Sistem Dream Search Engine tidak lebih efisien dari Giggle Search Engine walaupun keduanya sama-sama menggunakan proses *indexing* terkecuali ketika memiliki kata kunci yang tidak banyak memiliki sinonim.

Referensi

[1] Hamzah, Amir. *Temu Kembali Informasi Berbasis Kluster untuk Sistem Temu Kembali Informasi Teks Bahasa Indonesia*. Jurnal Teknologi, 2009. [Online] Unduh: http://jurtek.akprind.ac.id/sites/default/files/1-7_Amir.pdf [3 Januari 2012]

[2] Nugroho, Susetyo Adi. *Query Expansion Dengan Menggabungkan Metode Ruang Vektor Dan Wordnet Pada Sistem Information Retrieval*. Jurnal Informatika, 2009. [Online] Unduh: http://ti.ukdw.ac.id/ojs/index.php/informatika/article/download/67/30 [13 Januari 2012]

[3] Hasibuan, Zainal A., dan Andri, Yofi. *Penerapan Berbagai Teknik Sistem Temu-Kembali Informasi Berbasis Hiperteks*. Jurnal Komunikasi Ilmu Komputer Dan Teknologi Informasi, Jakarta, 2001. [Online] Unduh: http://repository.ui.ac.id/dokumen/lihat/6396.pdf [1 Januari 2012]

[4] Rafiudin, Rahmat. *Praktis Membangun Search Engine untuk Website Anda*. Andi, Edisi I. Yogyakarta, 2003.

[5] Sasongko, Jati. *Aplikasi untuk Membangun Corpus dari Data Hasil Crawling dengan Berbagai Format Data Secara Otomatis*. Jurnal Teknologi Informasi DINAMIK, 2010. [Online] Unduh: http://www.unisbank.ac.id/ojs/index.php/fti1/article/download/107/102 [3 Januari 2012]

[6] Suciadi, James. *Studi Analisis Metode-Metode Parsing dan Interpretasi Semantik pada Natural Language Processing. Jurnal Informatika*. 2001. [Online] Unduh: http://puslit2.petra.ac.id/ejournal/index.php/inf/article/viewFile/15799/15791 [18 Februari 2013]

[7] Agusta, Ledy. *Perbandingan Algoritma Stemming Porter dengan Algoritma Nazief & Adriani Untuk Stemming Dokumen Teks Bahasa Indonesia*. Konferensi Nasional Sistem dan Informatika, Bali, 2009. [Online] Unduh: http://yudiagusta.files.wordpress.com/2009/11/196-201-knsi09-036-perbandingan-algoritma-stemming-porter-dengan-algoritma-nazief-adriani-untuk-stemming-dokumen-teks-bahasa-indonesia.pdf [3 Januari 2012]

[8] Safriadi, Novi dan Wibowo, Ari. *Uji Relevansi dan Performansi Sistem Temu Balik Informasi pada Giggle Search Engine*. ELKHA, Pontianak, 2011.

[9] Sugono, Dendy. dkk. *Kamus Bahasa Indonesia.* Pusat Bahasa Departemen Pendidikan Nasional, Jakarta, 2008

[10] Sugono, Dendy. dkk. *Tesaurus Bahasa Indonesia Pusat Bahasa*. Pusat Bahasa Departemen Pendidikan Nasional, Jakarta, 2008

Biografi

Meisya Fitri, lahir di Pontianak, Indonesia, 9 Mei 1989. Memperoleh gelar Sarjana dari Program Studi Teknik Informatika Universitas Tanjungpura, Pontianak, Indonesia, 2013.