Perangkingan Dokumen Berbahasa Arab Menggunakan *Latent Semantic Indexing*

Aminul Wahib[1], Pasnur[2], Putu Praba Santika[3], Agus Zainal Arifin[4]
Jurusan Teknik Informatika, Institut Teknologi Sepuluh Nopember
Kampus ITS Keputih, Sukolilo, Surabaya 60111, Jawa Timur
E-mail: [1]wahib13@mhs.if.its.ac.id, [2]pasnur13@mhs.if.its.ac.id, [3]praba13@mhs.if.its.ac.id, [4]agusza@cs.its.ac.id

Masuk: 11 Juni 2014; Direvisi: 26 Juni 2014; Diterima: 8 Juli 2014

Abstract. *Various ranking methods for documents in Information Retrieval applications have been developed and implemented. One of the most popular method is document ranking with the use of vector space model based on TF.IDF term weighting. The method only performs term weighting based on frequency of its occurence in documents regardless semantic relations between terms. In fact, semantic relations between terms have an important role in increasing the relevance of thedocument-searching results. This research develops the TF.IDF.ICF.IBF method by adding the Latent Semantic Indexing to find out semantic relations between terms in the case of ranking documents in Arabic language. Datasets used are taken from document collections in Maktabah Syamilah software. The result shows that the proposed method outperforms the TF.IDF.IBF.ICF method. Respectively, f-measure values for TF.IDF.ICF.IBF.LSI method at cosine similarity thresholds of 0.3, 0.4, and 0.5 are 45%, 51%, and 60. However, the proposed method has higher average computation time than TF.IDF.ICF.IBF method by 2 minutes and 8 seconds.*
Keywords: *Book Index, Class Index, Latent Semantic Indexing, Ranking Documents*

Abstrak. *Berbagai metode perangkingan dokumen dalam aplikasi Information Retrieval telah dikembangkan dan diimplementasikan. Salah satu metode yang sangat populer adalah perangkingan dokumen menggunakan vector space model berbasis pada nilai term weighting TF.IDF. Metode tersebut hanya melakukan pembobotan term berdasarkan frekuensi kemunculannya pada dokumen tanpa memperhatikan hubungan semantik antar term. Dalam kenyataannya hubungan semantik antar term memiliki peranan penting untuk meningkatkan relevansi hasil pencarian dokumen. Penelitian ini mengembangkan metode TF.IDF.ICF.IBF dengan menambahkan Latent Semantic Indexing untuk menemukan hubungan semantik antar term pada kasus perangkingan dokumen berbahasa Arab. Dataset yang digunakan diambil dari kumpulan dokumen pada perangkat lunak Maktabah Syamilah. Hasil pengujian menunjukkan bahwa metode yang diusulkan memberikan nilai evaluasi yang lebih baik dibandingkan dengan metode TF.IDF.ICF.IBF. Secara berurut nilai f-measure metode TF.IDF.ICF.IBF.LSI pada ambang cosine similarity 0,3, 0,4, dan 0,5 adalah 45%, 51%, dan 60%. Namun metode yang disulkan memiliki waktu komputasi rata-rata lebih tinggi dibandingkan dengan metode TF.IDF.ICF.IBF sebesar 2 menit 8 detik.*
Kata Kunci: *Indeks Buku, Indeks Kelas, Latent Semantic Indexing, Perangkingan Dokumen*

1. Pendahuluan

Jumlah dokumen digital mengalami peningkatan yang sangat pesat. Hal ini menyebabkan permasalahan untuk menemukan dokumen yang paling relevan dalam proses pencarian. Para peneliti mencoba menyelesaikan permasalahan tersebut dengan melakukan pengembangan metode dalam aplikasi *Information Retrieval.*

Information Retrieval adalah tindakan untuk menemukan materi (biasanya dokumen) dari bentuk yang tidak terstruktur (biasanya teks) yang memenuhi kebutuhan informasi dalam koleksi besar (biasanya disimpan dalam komputer) (Manning dkk., 2009). Pada definisi

tersebut, yang dimaksud data tidak terstruktur adalah data yang tidak memiliki bentuk jelas yang mudah dipahami oleh komputer.

Dokumen yang sesuai dengan kata kunci pencarian mungkin saja lebih dari satu, sehingga perlu dilakukan perangkingan dokumen berdasarkan kecocokannya dengan kata kunci yang dicari. Metode perangkingan dokumen telah banyak diimplementasikan untuk dokumen berbahasa Inggris, sedangkan implementasi pada bahasa lain relatif lebih sedikit.

Penelitian (Fauzi, 2013) melakukan perangkingan hasil pencarian dokumen menggunakan pembobotan berbasis indeks buku dan kelas untuk dokumen berbahasa Arab. Pencarian dengan kata kunci tertentu akan menghasilkan buku dan halaman yang sesuai dengan kata kunci tersebut. Masing-masing halaman pada buku dipandang sebagai sebuah dokumen. Penelitian tersebut melakukan perbaikan metode pembobotan *term* TF.IDF dengan menggabungkannya dengan metode pembobotan *Inverse Class Frequency* (ICF) dan *Inverse Book Frequency* (IBF). Penggabungan metode menghasilkan sebuah metode baru yang disebut TF.IDF.ICF.IBF. Pembobotan ICF memberikan nilai yang tinggi terhadap *term* yang jarang muncul pada kumpulan kelas/kategori. Pembobotan *IBF* memberikan nilai yang tinggi terhadap *term* yang jarang muncul pada kumpulan buku/kitab. Penelitian tersebut menunjukkan hasil pengujian bahwa metode TF.IDF.ICF.IBF mampu memperbaiki relevansi hasil pencarian dokumen berbahasa Arab bila dibandingkan dengan metode TF.IDF. Namun demikian penelitian tersebut mengabaikan hubungan semantik antar *term* yang merupakan salah satu faktor untuk meningkatkan relevansi hasil pencarian.

Pada penelitian ini diusulkan penggabungan metode perangkingan hasil pencarian menggunakan pembobotan TF.IDF.ICF.IBF dengan metode *Latent Semantic Indexing* (LSI). Metode LSI mampu mengidentifikasi hubungan semantik antar *term* berdasarkan pola dan hubungan antara istilah dan konsep-konsep yang terkandung dalam koleksi teks. Hubungan semantik tersebut merupakan hubungan yang tersembunyi tetapi dapat dimunculkan melalui pendekatan statistik. Penggunaan LSI pada metode TF.IDF.ICF.IBF diharapkan mampu meningkatkan relevansi hasil pencarian dokumen berbahasa Arab dengan melakukan perangkingan dokumen berdasarkan nilai similaritas tertinggi.

2. Penelitian Terkait

Penelitian ini berdasarkan penelitian terdahulu yang dilakukan (Fauzi, 2013) pada kasus perangkingan dokumen berbahasa Arab. Penelitian tersebut mengusulkan metode TF.IDF.ICF.IBF untuk memperbaiki metode lama yang hanya menggunakan pembobotan TF.IDF. Metode TF.IDF.ICF.IBF merupakan metode yang menggabungkan empat jenis pembobotan, yaitu *Term Frequency* (TF)*, Inverse Document Frequency* (IDF), *Inverse Class Frequency* (ICF), dan *Inverse Book Frequency* (IBF). Penelitian tersebut menggunakan pembobotan TF untuk menghitung frekuensi kemunculan *term* pada sebuah dokumen. Pembobotan IDF digunakan untuk memberikan nilai yang tinggi terhadap *term* yang jarang muncul pada kumpulan dokumen. Pembobotan ICF digunakan untuk memberikan nilai yang tinggi terhadap *term* yang jarang muncul pada kumpulan kelas/kategori dokumen. Pembobotan IBF digunakan untuk memberikan nilai yang tinggi terhadap *term* yang jarang muncul pada kumpulan buku/kitab. Kombinasi keempat pembobotan tersebut memberikan dampak peningkatan relevansi hasil pencarian dokumen yang dibuktikan dengan pengukuran evaluasi metode. Metode TF.IDF.ICF.IBF yang diusulkan menunjukkan hasil yang lebih baik dari metode perangkingan TF.IDF pada kasus perangkingan dokumen berbahasa Arab.

Pada penelitian (Zhang, dkk, 2011) dilakukan sebuah studi komparasi antara metode pembobotan dengan TF.IDF, LSI, dan *Multi-Word* untuk melakukan *text categorization* pada dokumen berbahasa Cina dan Inggris. Hasil penelitian tersebut menunjukkan bahwa LSI memiliki hasil evaluasi yang lebih baik di antara ketiga metode yang diuji untuk melakukan *text categorization*, baik pada dokumen berbahasa Cina maupun dokumen berbahasa Inggris.

3. Kajian Pustaka

3.1. Pembobotan *Term*

Perangkingan dokumen menggunakan representasi *vector space model* dari kumpulan dataset. Dokumen dalam *vector space model* direpresentasikan dalam bentuk matriks yang berisi bobot kata pada dokumen. Bobot tersebut menyatakan kepentingan/kontribusi kata terhadap suatu dokumen dan kumpulan dokumen. Kepentingan suatu kata dalam dokumen dapat dilihat dari frekuensi kemunculannya terhadap dokumen. Biasanya kata yang berbeda memiliki frekuensi yang berbeda. Beberapa jenis pembobotan *term* yang digunakan antara lain: TF, IDF, ICF, dan IBF.

TF merupakan metode yang paling sederhana dalam membobotkan setiap *term*. Setiap *term* diasumsikan memiliki kepentingan yang proporsional terhadap jumlah kemunculan term pada dokumen. Bobot dari *term* t pada dokumen d dapat dilihat pada persamaan (1).

$$TF(d,t) = f(d,t) \tag{1}$$

di mana f(d,t) adalah frekuensi kemunculan term t pada dokumen d.

Bila TF memperhatikan kemunculan *term* di dalam dokumen, maka *Inverse Document Frequency* (IDF) memperhatikan kemunculan *term* pada kumpulan dokumen. Latar belakang pembobotan ini adalah *term* yang jarang muncul pada kumpulan dokumen sangat bernilai. Kepentingan tiap *term* diasumsikan memilki proporsi yang berkebalikan dengan jumlah dokumen yang mengandung *term*. Faktor IDF dari *term* t dapat dilihat pada persamaan (2).

$$IDF(t) = 1 + \log\left(\frac{Nd}{df(t)}\right) \tag{2}$$

di mana Nd adalah jumlah seluruh dokumen, dan df(t) jumlah dokumen yang mengandung *term* t.

Metode pembobotan yang menggabungkan antara TF dan IDF dikenal dengan nama TF.IDF. Metode ini memperbaiki hasil pembobotan dokumen karena mempertimbangkan frekuensi kemunculan *term* pada sebuah dokumen serta kelangkaan kemunculan *term* pada kumpulan dokumen. Persamaan klasik untuk menghitung nilai pembobotan TF.IDF ditunjukkan pada persamaan 3.

$$w_{i,j} = tf_{i,j} \times log\left(\frac{N}{df_i}\right) \tag{3}$$

di mana $w_{i,j}$ merupakan bobot *term* i pada dokumen j, N merupakan jumlah dokumen pada seluruh koleksi, $tf_{i,j}$ merupakan frekuensi kemunculan *term* i pada dokumen j dan df_i merupakan frekuensi dokumen dari *term* i pada koleksi (Zhang dkk., 2011). Nilai pembobotan TF.IDF dapat juga diketahui dengan melakukan perkalian antara nilai TF dengan nilai IDF (Padhiyar & Rekh, 2013).

Jika IDF memperhatikan kemunculan *term* pada kumpulan dokumen, maka *Inverse Class Frequency* (ICF) memperhatikan kemunculan *term* pada kumpulan kategori/kelas. *Term* yang jarang muncul pada banyak kelas adalah *term* yang bernilai untuk klasifikasi. Kepentingan tiap *term* diasumsikan memilki proporsi yang berkebalikan dengan jumlah kelas yang mengandung *term*. Faktor *ICF* dari *term* t dapat dilihat pada persamaan (4).

$$ICF(t) = 1 + \log\left(\frac{Nc}{cf(t)}\right) \tag{4}$$

di mana Nc adalah jumlah seluruh kelas, cf(t) jumlah kelas yang mengandung *term* t.

Jika ICF memperhatikan kemunculan *term* pada kumpulan kelas, maka IBF memperhatikan kemunculan *term* pada kumpulan kitab/buku. *Term* yang jarang muncul pada banyak buku adalah *term* yang sangat bernilai. Kepentingan tiap *term* diasumsikan memiliki

proporsi yang berkebalikan dengan jumlah buku yang mengandung *term*. Faktor IBF dari *term* t ditunjukkan pada persamaan 5.

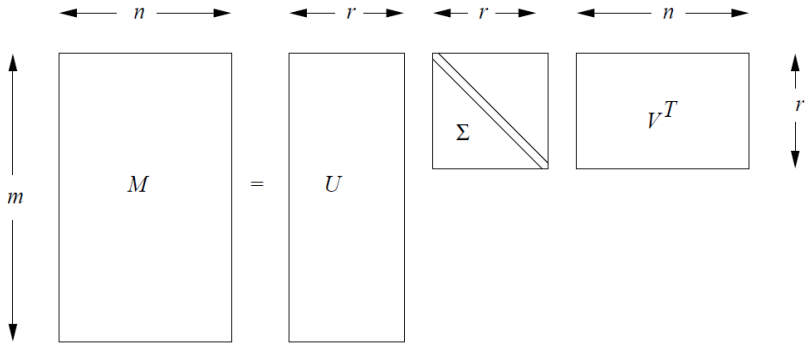
$$IBF(t) = 1 + \log\left(\frac{Nb}{bf(t)}\right) \quad (5)$$

di mana Nb adalah jumlah seluruh buku, bf(t) jumlah buku yang mengandung *term* t.

3.2. *Latent Semantic Indexing* (LSI)

Latent Semantic Indexing (LSI) adalah metode *indexing* pada *information retrival* yang menggunakan teknik *singular value decomposition* (SVD) untuk mengidentifikasi makna semantik kata-kata berdasarkan pola dan hubungan antara istilah dan konsep-konsep yang terkandung dalam koleksi teks (Froud dkk., 2013) (Papadimitriou dkk, 1998). LSI mengikuti logika bahwa kata-kata yang digunakan dalam konteks yang sama cenderung memiliki makna semantik yang sama. Salah satu sifat utama LSI adalah kemampuannya untuk membangun hubungan antara istilah yang muncul dalam konteks yang serupa.

Sebuah matrik M yang merupakan frekuensi *term* pada dokumen berukuran m x n dengan *rank* r dapat didekomposisi dengan SVD menjadi U, ∑, V seperti pada Gambar 1. Hasil SVD adalah matriks U berukuran m x k dan matriks V berukuran n x k. Kedua matriks tersebut mempunyai kolom-kolom orthogonal. Sedangkan ∑ adalah matriks diagonal, di mana semua elemen selain diagonalnya bernilai 0. Elemen diagonal dari matriks ∑ disebut sebagai *singular values* dari matrik M (Rajaraman & Ullman, 2011).

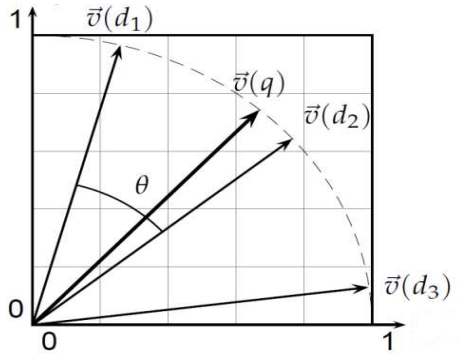


Gambar 1. Ilustrasi Dekomposisi Matriks dengan SVD

SVD mampu menangkap hubungan antar *term* sehingga *term* dan kalimat dapat dikelompokkan secara semantik. Jika pola kombinasi kata yang menonjol dan berulang dalam dokumen, pola ini akan ditangkap dan diwakili oleh salah satu vektor *singular*. (Toure & Gangopadhyay, 2013). Besarnya nilai *singular* yang sesuai menunjukkan tingkat pentingnya pola ini dalam dokumen. Setiap kalimat yang mengandung pola kombinasi kata akan diproyeksikan sepanjang vektor tunggal, dan kalimat yang paling mewakili pola ini akan memiliki nilai indeks terbesar. Setiap pola kombinasi kata tertentu menjelaskan topik/konsep tertentu dalam dokumen, sehingga setiap vektor tunggal merupakan topik/konsep penting dari dokumen, dan besarnya nilai *singular* yang sesuai merupakan tingkat dari pentingnya topik/konsep tersebut.

3.3. *Cosine Similarity*

Hasil pembobotan *term* pada dokumen digunakan sebagai representasi vektor. Dari representasi bobot tersebut dapat dihitung nilai kemiripan suatu dokumen dengan *query*. Nilai kemiripan ini biasa dihitung dengan rumusan *cosine similarity*, yaitu perhitungan tingkat kemiripan berdasar pada besar sudut kosinus antara dua vektor, dalam hal ini adalah vektor dokumen. Representasi perumusan ini dalam bidang kartesian seperti diperlihatkan pada Gambar 2.



Gambar 2. Representasi Perumusan *Cosine Similarity*

Pada Gambar 2 terdapat tiga vektor dokumen d_1, d_2 dan d_3 dan satu vektor *query* q. *Cosine similarity* menghitung nilai kosinus θ dari *query* dan tiga dokumen lain. Nilai ini menunjukkan derajat kemiripan dokumen dengan *query*.

Berdasarkan kosinus sudut antara dua vektor, maka nilainya berkisar pada 0 sampai dengan 1, di mana 0 menandakan bahwa kedua dokumen tidak mirip sama sekali, dan 1 menandakan bahwa antara *query* dan dokumen benar-benar identik. *Cosine similarity* dinyatakan pada persamaan (6).

$$cos(q, d_j) = \frac{\sum_{t_k}[TF.IDF(t_k,q)].[TF.IDF(t_k,d_j)]}{\sqrt{\sum TF.IDF_q|^2}.\sqrt{\sum |TF.IDFd_j|^2}} \quad (6)$$

di mana $cos(q,d_j)$ merupakan nilai *cosinus* antara *query* dan dokumen j, sedangkan $TF.IDF(t_k,q)$ dan $TF.IDF(t_k,d_j)$ adalah pembobotan TF.IDF kata t_k pada *query* dan dokumen j. $|TFIDFq|$ dan $|TFIDFd_j|$ adalah panjang dari vektor *query* q dan dokumen. Sebagai contoh $\|d_i\|^2 = (TFIDFt_1^2 + TFIDFt_2^2 + TFIDFt_3^2 + ... + TFIDFt_k^2)^{1/2}$, di mana $TFIDFt_k$ adalah bobot kata ke-t_k pada vektor dokumen d_i.

4. Metode Penelitian

Penelitian ini menggunakan *corpus* atau sekumpulan dokumen teks berbahasa Arab yang diambil dari kumpulan kitab dalam perangkat lunak Maktabah Syamilah. Dalam penelitian ini setiap halaman pada kitab tersebut dianggap sebagai satu dokumen. Jumlah dokumen yang digunakan adalah sebanyak 230 dokumen. Kemudian dari 230 dokumen tersebut setelah dilakukan *preprocessing* menghasilkan 9194 kata berbeda *(distinct term).* Alur metode penelitian yang digunakan digambarkan secara detail pada Gambar 3.

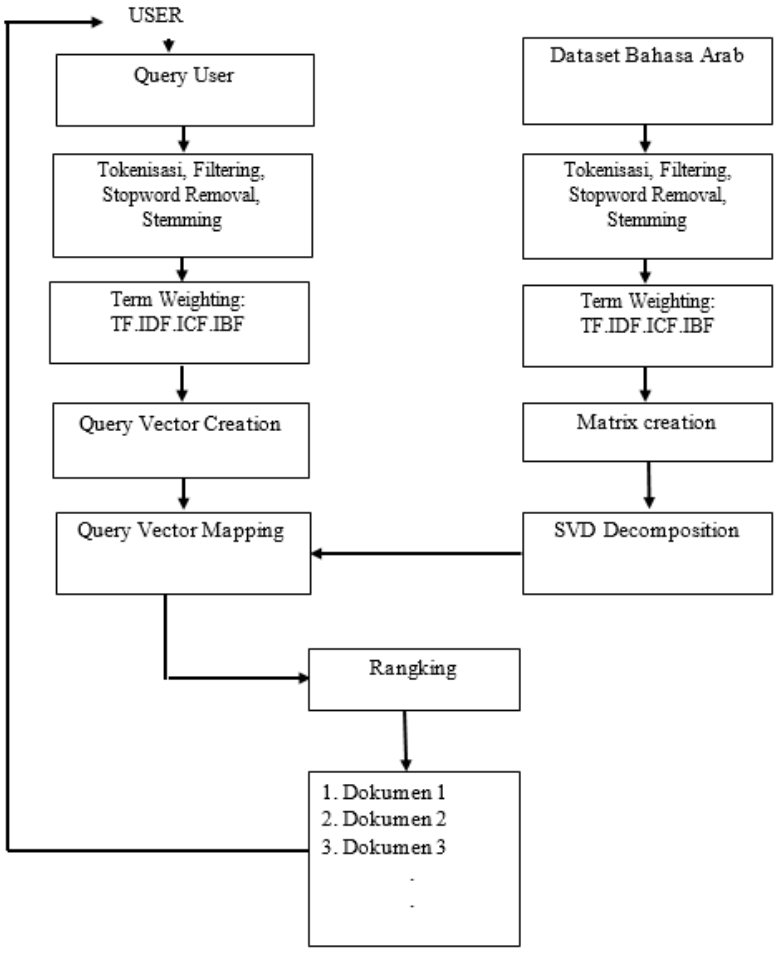
Pada Gambar 3 dapat dilihat bahwa sekumpulan dokumen harus melalui tahap *preprocesing* yaitu *tokenization*, *filtering*, *stopword removal* dan *stemming*. Hal ini dilakukan agar diperoleh bentuk kata dasar dari setiap kata maupun kalimat. Jumlah kata yang terbentuk merupakan representasi dari baris matriks. Sedangkan banyaknya dokumen merupakan representasi dari kolom matriks, dan frekuensi kemunculan kata pada dokumen tertentu merupakan nilai dari sebuah matrik yang berada pada baris dan kolom sesuai dengan letak *term* dan dokumen yang terkait. Setelah diperoleh matriks A kemudian dilakukan proses dekomposisi menggunakan *single value decompotition* sehingga diperoleh persamaan (7)

$$A = USV^t \quad (7)$$

Pada *query user* juga dilakukan *preprocesing* untuk memperoleh bentuk dasar dari setiap kata, kemudian dibangun sebuah vektor *query* dengan ukuran panjang kolom sesuai dengan matriks A dan banyak kolom adalah 1. Jika *term* pada matriks A tidak terdapat dalam *query user* maka nilainya adalah 0, sedangkan jika *term* pada matriks A terdapat pada *query user* maka nilai *term* tersebut sesuai dengan frekuensi kemunculan kata di dalam *query* tersebut.

Setelah itu vektor *query* dipetakan ke dalam ruang berdimensi 2 *(low dimensional space)* dengan transformasi seperti pada persamaan 8.

$$Q^t U S^{-1} \quad (8)$$



Gambar 3. Alur Metode Penelitian

Hasil dari perkalian pada persamaan 8 akan menghasilkan vektor *query* yang selanjutnya akan di cari similaritasnya dengan vektor dokumen (V pada persamaan 7). Melalui perhitungan menggunakan rumus *cosinus* maka akan diperoleh rangking dokumen yang paling relevan berdasarkan nilai kosinusnya.

Selanjutnya dilakukan evaluasi untuk menganalisa performa dari metode perangkingan yang diusulkan. Analisa performa ini menggambarkan efektifitas metode yang direlasikan dengan kemampuan metode untuk mengembalikan dokumen yang relevan dengan *query*. Pengukuran evaluasi pengujian metode dinyatakan dengan nilai akurasi, *precision, recall,* dan *f-measure.*

Nilai akurasi menunjukkan kemampuan sistem untuk memisahkan secara benar semua dokumen yang relevan dan semua dokumen yang tidak relevan. Akurasi akan bernilai maksimal (100%) jika seluruh dokumen yang relevan berhasil dikembalikan oleh sistem dan tak satu pun dokumen yang dikembalikan tersebut yang tidak relevan. Sebaliknya nilai akurasi akan bernilai minimal (0%) jika semua dokumen yang dikembalikan oleh sistem merupakan seluruh dokumen yang tidak relevan.

Nilai *precision* menunjukkan rasio antara dokumen relevan yang dikembalikan oleh sistem terhadap total dokumen yang dikembalikan oleh sistem. *Precision* akan bernilai

maksimal (100%) jika seluruh hasil pencarian dokumen merupakan dokumen relevan. Sebaliknya *precision* akan bernilai minimal (0%) jika seluruh hasil pencarian merupakan dokumen yang tidak relevan.

Nilai *recall* menunjukkan kemampuan sistem untuk menemukan semua dokumen relevan yang terdapat di dalam sistem. *Recall* akan bernilai maksimal (100%) jika semua dokumen relevan berhasil ditemukan. Sebaliknya *recall* akan bernilai minimal (0%) jika tak satu pun dokumen relevan yang ditemukan.

Jika dimisalkan TP merupakan dokumen relevan yang dikembalikan oleh sistem, FN merupakan dokumen relevan yang tidak dikembalikan oleh sistem, FP merupakan dokumen tidak relevan yang dikembalikan oleh sistem, dan TN merupakan dokumen tidak relevan yang tidak dikembalikan oleh sistem, maka nilai akurasi dihitung dengan menggunakan persamaan 9, nilai *precision* dihitung dengan menggunakan persamaan 10, dan nilai *recall* dihitung dengan menggunakan persamaan 11.

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \quad (9)$$

$$Precision = \frac{TP}{TP+FP} \quad (10)$$

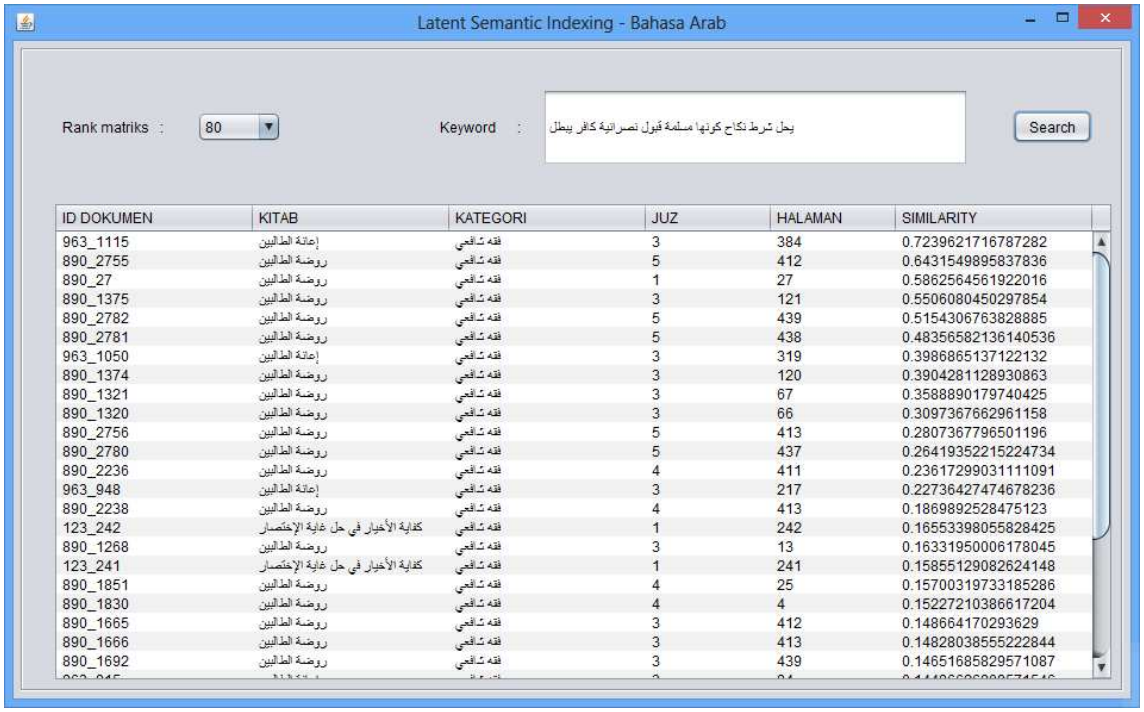
$$Recall = \frac{TP}{TP+FN} \quad (11)$$

Evaluasi dengan menggunakan nilai *precision* dan *recall* dianggap kurang cukup untuk menunjukkan kualitas sebuah sistem. Oleh karena itu, penelitian ini juga menggunakan pengukuran *f-measure* yang merupakan *harmonic mean* dari *precision* dan *recall.* Persamaan 12 menunjukkan cara menghitung nilai *f-measure* dari sebuah sistem *information retrieval.*

$$f - measure = \frac{2\ x\ Precision\ x\ Recall}{Precision+Recall} \quad (12)$$

Penelitian ini menggunakan aplikasi yang dikembangkan dengan menggunakan bahasa pemrograman Java. Contoh hasil pencarian dokumen dengan menggunakan metode TF.IDF.ICF.IBF diperlihatkan pada Gambar 4, sedangkan contoh hasil pencarian dengan menggunakan metode yang diusulkan (TF.IDF.ICF.IBF.LSI) diperlihatkan pada Gambar 5. Pada Gambar 4 dan Gambar 5, setiap hasil pencarian berdasarkan kata kunci tertentu akan mengembalikan identitas dokumen yang diurutkan berdasarkan nilai similaritas tertinggi. Kata kunci yang diuji berdasarkan *ground truth* yang telah ditetapkan oleh pakar pada dataset kumpulan kitab pada perangkat lunak Maktabah Syamilah.

Gambar 4. *Output* pencarian dokumen dengan metode TF.IDF.ICF.IBF



Gambar 5. *Output* pencarian dokumen dengan metode TF.IDF.ICF.IBF.LSI

5. Hasil Uji Coba

Pada penelitian ini dilakukan dua tahap pengujian. Tahap pertama dilakukan untuk menganalisa akurasi dari metode yang diusulkan menggunakan beberapa variasi nilai *rank*. Tahap kedua dilakukan pengukuran nilai *precision, recall,* dan *f-measure* pada kondisi nilai *rank* terbaik yang dihasilkan pada tahap pertama.

Tabel 1. Pengujian akurasi dengan variasi nilai *rank*

Query	*Rank* = 40	*Rank* =80	*Rank* =120	*Rank* =180	*Rank* =220
Q1	7: 0,7	1: 0,8	1:0,79	1:0,7	1:0,6
Q2	1: 0,9	1 : 0,9	1: 0,8	1:0,78	1:0,7
Q3	1:0,6	1: 089	1: 0,8	3:0,49	2:0,4
Q4	3:0,9	1: 0,9	1:79	1:0,68	1:0,6
Q5	1: 0,9	1:0,88	1: 85	1:0,7	1:0,6
Q6	1: 0,8	1:0,7	1: 0,6	1:0,49	1:0,4
Q7	1:0,7	1:0,48	1: 0,3	6:0,2	9:0,1
Akurasi	**71%**	**100%**	**100%**	**71%**	**71%**

Hasil dari pengujian tahap pertama diperlihatkan pada Tabel 1. Format nilai pada Tabel 1 adalah x:y, di mana x adalah rangking dokumen dan y adalah nilai *cosine similarity* dokumen. Hasil pengujian pada Tabel 1 memperlihatkan bahwa pengaturan nilai *rank* yang menghasilkan akurasi tertinggi adalah 80 dan 120. Nilai *rank* menunjukkan jumlah *singular value* yang dipertahankan untuk digunakan pada proses pengukuran similaritas. Nilai-nilai yang dibuang dianggap tidak memberikan pengaruh yang kuat terhadap nilai similaritas. Nilai-nilai yang dibuang tersebut dapat juga dianggap sebagai *noise* yang bilamana dipertahankan akan memberikan penurunan nilai akurasi. Seperti yang terlihat pada Tabel 1, jika nilai *rank* dinaikkan menjadi 180, maka akurasi mengalami penurunan daripada ketika menggunakan *rank* 120. Demikian juga nilai *rank* tidak boleh terlalu rendah, karena hal tersebut justru akan menghilangkan informasi penting yang juga akan memberikan dampak pada penurunan nilai akurasi. Pada Tabel 1 terlihat bahwa jika nilai *rank* adalah 40, maka akurasi yang dihasilkan hanya 71%. Dengan demikian, maka harus ditentukan nilai *rank* yang optimal, yaitu pada angka 80 dan 120.

Selanjutnya dilakukan pengujian tahap 2, di mana dilakukan pengukuran *precision, recall,* dan *f-measure* dengan menggunakan nilai *rank* terbaik dari pengujian 1. Dari pengujian 1 diperoleh akurasi terbaik dengan nilai *rank* adalah 80 dan 120. Namun demikian, nilai *cosinus* tertinggi diperoleh pada posisi nilai *rank* 80, sehingga nilai tersebut yang akan dipergunakan pada pengujian tahap 2.

Tabel 2. *Precision, Recall* dan *F-Measure* dengan nilai ambang *cosine similarity* 0,3

Query	TF.IDF.ICF.IBF			TF.IDF.ICF.IBF.LSI		
	Precission	***Recall***	***F-Measure***	***Precission***	***Recall***	***F-Measure***
Q1	1	0,5	0,66667	0,20	1	0,33333
Q2	0	0	0	0,66667	0,66667	0,66667
Q3	0,5	0,25	0,33333	0,4	1	0,57143
Q4	0	0	0	0,13333	1	0,23529
Q5	0	0	0	0,22222	1	0,36364
Q6	0,66667	1	0,8	0,33333	1	0,5
Rata-Rata	**36%**	**29%**	**30%**	**33%**	**94%**	**45%**

Tabel 3. *Precision, Recall* dan *F-Measure* dengan nilai ambang *cosine similarity* 0,4

Query	TF.IDF.ICF.IBF			TF.IDF.ICF.IBF.LSI		
	Precission	***Recall***	***F-Measure***	***Precission***	***Recall***	***F-Measure***
Q1	1	0,5	0,66667	0,29	1	0,44444
Q2	0	0	0	1	0,66667	0,8
Q3	0	0	0	0,4	1	0,57143
Q4	0	0	0	0,25	1	0,4
Q5	0	0	0	0,25	0,5	0,33333
Q6	1	0,5	0, 66667	0,33333	1	0,5
Rata-Rata	**33%**	**17%**	**22%**	**42%**	**86%**	**51%**

Tabel 4. *Precision, Recall* dan *F-Measure* dengan nilai ambang *cosine similarity* 0,5

Query	TF.IDF.ICF.IBF			TF.IDF.ICF.IBF.LSI		
	Precission	***Recall***	***F-Measure***	***Precission***	***Recall***	***F-Measure***
Q1	0	0	0	0,40	1	0,57143
Q2	0	0	0	1	0,66667	0,8
Q3	0	0	0	0,4	1	0,57143
Q4	0	0	0	0,4	1	0,57143
Q5	0	0	0	0,5	0,5	0,5
Q6	0	0	0	0,4	1	0,57143
Rata-Rata	**0%**	**0%**	**0%**	**52%**	**86%**	**60%**

Pada pengujian tahap 2 ini, metode yang diusulkan akan dibandingkan dengan penelitian (Fauzi, 2013) dan data yang digunakan akan disesuaikan dengan data yang digunakan pada penelitian ini dengan tujuan dapat memperoleh hasil yang objektif. Nilai ambang *cosine similarity* yang digunakan adalah 0,3, 0,4 dan 0,5. Nilai-nilai tersebut diperoleh dari beberapa hasil percobaan baik menggunakan metode TF.IDF.ICF.IBF.LSI maupun menggunakan metode TF.IDF.ICF.IBF yang digunakan pada penelitian (Fauzi, 2013).

Hasil pada Tabel 2, Tabel 3 dan Tabel 4 menunjukkan bahwa metode TF.IDF.ICF.IBF.LSI memberikan hasil evaluasi yang lebih baik jika dibandingkan dengan metode TF.IDF.ICF.IBF. Pada ambang *cosine similarity* 0,3, nilai *f-measure* metode TF.IDF.ICF.IBF.LSI adalah 45% dan lebih tinggi 15% dibandingkan nilai *f-measure* metode TF.IDF.ICF.IBF yang hanya mencapai angka 30%. Pada ambang *cosine similarity* 0,4, nilai *f-measure* metode TF.IDF.ICF.IBF.LSI adalah 51% dan lebih tinggi 29% dibandingkan nilai *f-measure* metode TF.IDF.ICF.IBF yang hanya mencapai angka 22%. Pada ambang *cosine similarity* 0,5, nilai *f-measure* metode TF.IDF.ICF.IBF.LSI adalah 60% sedangkan metode TF.IDF.ICF.IBF bernilai 0%.

Hasil pengujian juga memperlihatkan bahwa pada metode TF.IDF.ICF.IBF.LSI terdapat kecenderungan di mana semakin tinggi nilai ambang batas *cosine similarity* yang diberikan (0,3, 0,4 dan 0,5), maka nilai *f-measure* juga semakin meningkat. Hal ini berbeda dengan metode TF.IDF.ICF.IBF yang memiliki nilai *f-measure* semakin menurun pada kenaikan nilai ambang *cosine similarity* tersebut. Hal ini membuktikan bahwa metode TF.IDF.ICF.IBF.LSI

menghasilkan nilai similaritas yang lebih tinggi jika dibandingakan dengan metode TF.IDF.ICF.IBF.

Tabel 5 merupakan hasil pengukuran waktu terhadap proses pencarian data. Dari hasil tersebut terlihat bahwa waktu komputasi rata-rata metode TF.IDF.ICF.IBF lebih baik dibandingkan dengan metode TF.IDF.ICF.IBF.LSI. Selisih rata-rata waktu kedua metode adalah sebesar 2 menit 8 detik. Waktu komputasi yang lebih tinggi pada metode TF.IDF.ICF.IBF.LSI disebabkan oleh proses dekomposisi matriks terhadap vektor dokumen dan *query.*

Tabel 5. Rata-rata waktu komputasi

Metode	Waktu Komputasi
TF.IDF.ICF.IBF	15 detik
TF.IDF.ICF.IBF.LSI	2 menit, 23 detik

6. Kesimpulan

Pada penelitian ini telah dilakukan perangkingan dokumen berbahasa Arab dengan menggunakan *Latent Semantic Indexing* pada metode TF.IDF.ICF.IBF yang telah dilakukan sebelumnya. Hasil pengujian menunjukkan bahwa metode yang diusulkan (TF.IDF.ICF.IBF.LSI) memberikan nilai evaluasi (*precision, recall,* dan *f-measure*) yang lebih baik dibandingkan dengan metode TF.IDF.ICF.IBF. Secara berurut nilai *f-measure* metode TF.IDF.ICF.IBF.LSI pada ambang *cosine similarrity* 0,3, 0,4, dan 0,5 adalah 45%, 51%, dan 60%.

Hasil pengujian juga menunjukkan bahwa metode yang disulkan (TF.IDF.ICF.IBF.LSI) memiliki waktu komputasi rata-rata lebih tinggi 2 menit 8 detik dibandingkan dengan metode TF.IDF.ICF.IBF. Hal tersebut disebabkan oleh adanya proses dekomposisi matriks pada metode yang diusulkan. Kelemahan tersebut menjadi peluang pengembangan pada penelitian selanjutnya agar diperoleh sebuah metode yang memiliki waktu komputasi yang lebih rendah.

Referensi

Fauzi, A. 2013. *Term Weighting Berbasis Indeks Buku Dan Kelas Untuk Perangkingan Dokumen Berbahasa Arab*. Thesis tidak diterbitkan. ITS, Surabaya.

Froud, H., Lachkar, A., & Ouatik, SA. 2013. *Arabic Text Summarization Based On Latent Semantic Analysis to Enhance Arabic Documents Clustering*. International Journal of Data Mining & Knowledge Management Process, vol. 3, no. 1, pp. 79-95, 2013.

Manning, C. D., Raghavan, P. & Schütze, H. 2009. *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press

Padhiyar, H. & Rekh, P., 2013. *Improving Accuracy of Text Classification for SMS Data*. International Journal for Scientific Research & Development, .vol. 1, no. 10, 2321-0613.

Papadimitriou, C.H., Tamaki, H., Raghavan, P., & Vempala, S. 1998. *Latent Semantic Indexing: a Probabilistic Analysis*. PODS '98 Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, New York.

Rajaraman, A. & Ullman, J. 2011. *Mining of Massive Datasets*. Cambridge: Cambridge University Press.

Toure, I. & Gangopadhyay, A., 2013. *Analyzing Terror Attacks Using Latent Semantic Indexing*. Technologies for Homeland Security (HST), Waltham.

Zhang, W., Yoshida, T., & Tang, X., 2011. *A comparative study of TFIDF, LSI and multi-words for text classification*. Expert Systems with Applications, 38: 2758–2765.