Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)
Vol. 2, No. 2, Oktober 2015, hlm. 132-137

PEMBOBOTAN KATA BERBASIS PREFERENSI DAN HUBUNGAN SEMANTIK PADA DOKUMEN FIQIH BERBAHASA ARAB

Septiyawan R. Wardhana[1], Dika R. Yunianto[2], Agus Zainal Arifin[3], Diana Purwitasari[4]

[1,2,3,4]Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya
Email: [1]rosetya.wardhana15@mhs.if.its.ac.id, [2]dika15@mhs.if.its.ac.id, [3]agusza@cs.its.ac.id, [4]diana@if.its.ac.id

(Naskah masuk: 11 Juni 2015, diterima untuk diterbitkan: 22 Juli 2015)

Abstrak

Dalam proses pencarian dokumen, pengguna sering menginginkan hasil pencarian yang sesuai dengan preferensi yang diinginkannya. Maka, untuk memperoleh hasil pencarian yang sesuai dengan preferensi tersebut dibutuhkan suatu metode pembobotan kata yang didasarkan pada preferensi tersebut. Metode pembobotan tersebut perlu mempertimbangkan hubungan semantik antar kata untuk meningkatkan relevansi hasil pencarian. Dalam penelitian ini diusulkan metode pembobotan kata berbasis preferensi berdasarkan hubungan semantik antar kata pada dokumen fiqih berbahasa Arab. Latent Semantic Indexing merupakan salah satu metode indexing dalam sistem temu kembali informasi yang mempertimbangkan hubungan semantik antar kata. Hasil pembobotan kata berdasarkan preferensi dijadikan sebuah matriks untuk perhitungan Latent Semantic Indexing yang menghasilkan sebuah vektor. Vektor tersebut dihitung similaritasnya antara vektor query dengan vektor-vektor dokumen yang ada. Metode pembobotan kata berbasis preferensi yang mempertimbangkan hubungan semantik antar kata dapat digunakan dalam perankingan dokumen fiqih bahasa Arab berbasis preferensi. Hal tersebut dapat dilihat dari nilai maksimal precision, recall dan f-measure yang meningkat menjadi 88.75 %, 89.72% dan 87.91%.

Kata kunci: *Bahasa Arab, Latent Semantic Indexing, Pembobotan Kata, Preferensi*

Abstract

In the document search process is not uncommon users want search results that correspond to the desired preferences. Thus, to obtain the search results according to user preferences needed a word weighting method based on user preference. The term weighting method needs to consider the semantic relationships between words to improve the relevance of search results. This paper propose a new method of term weighting based preference by considering the semantic relationships between term in documents fiqh Arabic. Latent Semantic Indexing is a method of indexing in information retrieval system that takes the semantic relationships between words. Term weighting results based on preferences made a matrix for calculation of Latent Semantic Indexing which generate a vector for the calculated similarity between the query vector of vectors documents. Term weighting based preference by considering the semantic relationships between term method can be used on the rank documents fiqh Arabic. It can be seen from the value of the precision, recall, and F-measure which increase to 88.75 %, 89.72 % and 87.91 %.

Keywords: **:** *Arabic, Latent Semantic Indexing, Term Weighting, Preference*

1. PENDAHULUAN

Pencarian dokumen pada *search engine* menghasilkan sebuah dokumen-dokumen yang diranking berdasarkan relevansi terhadap *query* yang dimasukkan oleh pengguna [1]. Terkadang pengguna menginginkan hasil perankingan dokumen berdasarkan preferensi tertentu. Preferensi tersebut akan merubah hasil dari perankingan dokumen [2].

Pada penelitian yang terkait, perankingan dokumen berbahasa Arab telah dilakukan dengan melakukan pembobotan kata atau *term* terhadap indeks buku dan kelas. Peneliti menyarankan sebuah metode pembobotan baru yang memperhatikan frekuensi kemunculan *term* terhadap indeks buku dan kelas yang disebut sebagai *inverse book frequency* (IBF) dan *inverse class frequency* (ICF). Metode yang digabungkan tersebut menjadi metode pembobotan *term* TF.IDF.ICF.IBF. Dari hasil pengujian terbukti bahwa metode TF.IDF.ICF.IBF memiliki nilai *precision*, *recall*, dan *f-measure* lebih besar dibandingkan dengan metode TF.IDF saja [3].

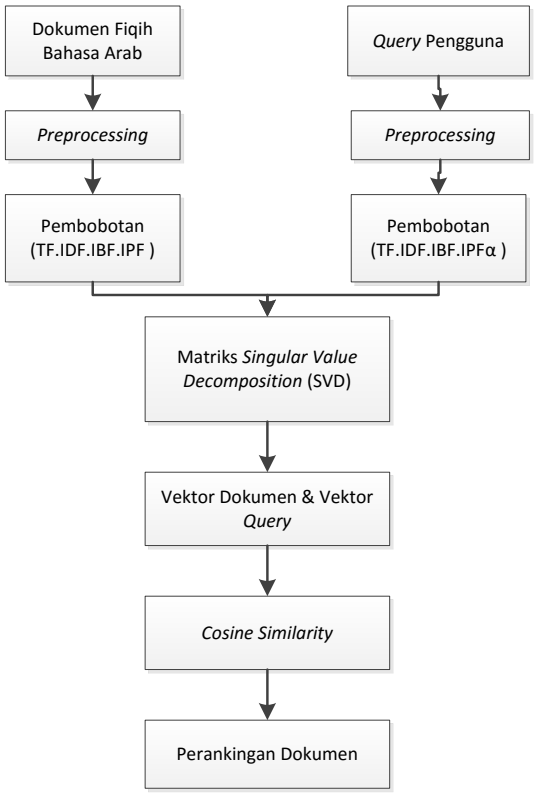
Dalam perankingan dokumen hubungan semantik antar *term* juga mempengaruhi relevensi dari hasil perankingan dokumen tersebut. *Latent semantic indexing* (LSI) merupakan sebuah metode yang mampu menemukan hubungan semantik antar *term*. Metode pembobotan *term* TF.IDF.ICF.IBF ditambahkan dengan LSI mampu meningkatkan relevansi hasil perankingan dokumen dilihat dari meningkatnya nilai *f-measure* TF.IDF.ICF.IBF.LSI

dibandingkan dengan metode TF.IDF.ICF.IBF [4].

Pada penelitian lainnya, terdapat metode pembobotan *term* berdasarkan preferensi pengguna atau *inverse preference frequency* (IPF) yang ditambahkan nilai α sebagai nilai penguat preferensi yang dipilih oleh pengguna. Metode IPFα digabungkan dengan metode pembobotan *term* berdasarkan dokumen dan kitab atau buku menjadi TF.IDF.IBF.IPFα. Metode tersebut dianggap mampu mengatasi permasalahan pencarian dokumen fiqih berbahasa Arab yang memiliki preferensi tersendiri dalam pencariannya [5]. Untuk meningkatkan relevansi *query* terhadap hasil perankingan dokumen dengan preferensi dari pengguna maka, dibutuhkan pembobotan yang juga mempertimbangkan hubungan semantik antar *term*.

Pada penelitian ini diusulkan metode pembobotan *term* berdasarkan preferensi untuk perangkingan dokumen fiqih berbahasa Arab dengan memperhatikan hubungan semantik antar *term*. Penambahan metode LSI pada metode pembobotan TF.IDF.IBF.IPFα diharapkan mampu meningkatkan relevansi hasil perankingan dokumen fiqih berbahasa Arab dengan preferensi dari pengguna.

2. METODOLOGI



Gambar 1. Bagan Metodologi

Secara umum penelitian ini dilakukan sesuai dengan Gambar 1. Pada metode pembobotan yang diusulkan, dilakukan beberapa proses perhitungan pembobotan term antara lain adalah *TF (Term Frequency)*, *IDF (Inverse Document Frequency)*, *IBF (Inverse Book Frequency)*, dan *IPF (Inverse Preference Frequency)*. Pembobotan *IPF* akan diimplemetasikan dengan menambahkan nilai α yang merupakan nilai khusus keberpihakan terhadap suatu preferensi. Hasil pembobotan pada setiap proses dikalikan menjadi *TF.IDF.IBF.IPFα* yang digunakan sebagai nilai pembobotan *term* terhadap preferensi yang dipilih. Nilai pembobotan *term* tersebut dibentuk menjadi sebuah matriks yang digunakan untuk menemukan hubungan semantik antar *term* dengan perhitungan *Latent Semantic Indexing (LSI)*.

2.1. Data


Metodologi Fiqih Imam Hanafi

Kitab al mabsuth (المبسوط)

Dokumen 1

[كتاب الصلاة]

ثم إنه بدأ بكتاب الصلاة؛ لأن الصلاة من أقوى الأركان بعد الإيمان بالله - تعالى - قال الله تعالى: {فإن تابوا وأقاموا الصلاة} [التوبة: 5] وقال - عليه الصلاة والسلام -: «الصلاة عماد الدين» فمن أراد نصب خيمة بدأ بنصب العماد، والصلاة من أعلى معالم الدين ما خلت عنها شريعة المرسلين - صلوات الله وسلامه عليهم أجمعين -.

وقد سمعت شيخنا الإمام الأستاذ شمس الأئمة الحلواني - رحمه الله تعالى - يقول في تأويل قوله تعالى: {وأقم الصلاة لذكري} [طه: 14] أي لأتي ذكرتها في كل كتاب منزل على لسان كل نبي مرسل وفي قوله - عز وجل -: {ما سلككم في سقر} [المدثر: 42] {قالوا لم نك من المصلين} [المدثر: 43] ما يدل على وكادتها، فحين وقعت بها البداية، دل على أنها في القوة بأعلى النهاية، وفي اسم الصلاة ما يدل على أنها ثانية الإيمان فالمصلي في اللغة هو التالي للسابق في الخيل قال القائل

ولا بد لي من أن أكون مصليا ... إذا كنت أرضى أن يكون لك السبق

وفي رواية

أما كنت ترضى أن أكون مصليا

والصلاة في اللغة عبارة عن الدعاء والثناء قال الله - تعالى -: {وصل عليهم إن صلاتك سكن لهم} [التوبة: 103] أي دعاءك، وقال القائل:

وقابلها الريح في دنها ... وصلى على دنها وارتسم

أي دعا وأثنى على دنها وفي الشريعة عبارة عن أركان مخصوصة كان فيها الدعاء أو لم يكن فالاسم شرعي ليس فيه معنى اللغة، فالدلائل من الكتاب والسنة على فرضيتها مشهورة يكثر تعدادها.

Gambar 2. Contoh dokumen fiqih

Preferensi pada dokumen merupakan sebuah kelas yang karakteristik dari kelas tersebut sulit dilihat pada term-term yang terdapat pada dokumen tersebut. Pada dokumen fiqih, preferensi dilihat dari keberpihakan dokumen terhadap suatu mahzab. Terdapat 4 mahzab yaitu yaitu mahzab Imam Hambali, mahzab Imam Hanafi, mahzab Imam Maliki, dan mahzab Imam Syafi'i.

Pada penilitian data yang digunakan diambil dari 3 kitab pada setiap mahzabnya dan 10 dokumen pada setiap kitabnya. Jadi total dokumen yang digunakan pada penelitian ini sebanyak 120 dokumen. Dokumen-dokumen tersebut diambil dari Maktabah Syamilah. Contoh dari dokumen fiqih dapat dilihat pada Gambar 2.

2.2. Preprocessing

Dokumen fiqih berbahasa Arab terlebih dahulu melalui tahapan *preprocessing*. Pada tahap ini yang pertama dilakukan adalah tokenisasi. Tokenisasi merupakan metode yang digunakan untuk menghilangkan spasi, simbol, dan tanda baca yang ada pada dokumen.

Langkah selanjutnya dalam tahap ini adalah melakukan stopword removal dengan menghilangkan kata yang tidak memiliki nilai informasi. Setelah didapatkan kata dari hasil *stopword removal*, kemudian setiap kata dilakukan

normalisasi *stopword*.

Normalisasi dalam bahasa Arab penting untuk dilakukan, mengingat terdapat banyak variasi penulisan untuk sebuah kata yang sama. Normalisasi dilakukan dengan merubah أ, إ, آ, kedalam alif (ا), merubah ta marbutoh (ة) menjadi ha (ه), merubah ya (ى) menjadi ya (ي) [6].

Setelah normalisasi selesai kemudian dilakukan *stemming* untuk mendapatkan kata dasar dari setiap kata. *Stemmer* yang digunakan untuk melakukan *stemming* pada penelitian ini merupakan *stemming* dari Khoja [7]. Tahapan *preprocessing* ini akan diterapkan baik untuk dokumen fiqih maupun *query* masukan dari pengguna.

2.3. TF.IDF.IBF.IPFα.LSI

TF merupakan perhitungan yang paling sederhana dalam proses pembobotan *term*. Proses ini dilakukan dengan menghitung jumlah kemunculan (*f*) *term* t_i dalam setiap dokumen d_j . Untuk proses perhitungan TF dapat dilihat pada persamaan (1).

$$W_{TF}(t_i, d_j) = f(t_i, d_j) \quad (1)$$

Sedangkan IDF merupakan pembobotan untuk mengetahui seberapa signifikan pengaruh *term* dari suatu dokumen terhadap dokumen lain. *Term* yang jarang sekali muncul dalam suatu dokumen merupakan *term* yang sangat bernilai. Pembobotan ini juga bisa diasumsikan sebagai nilai *opposite proportion* dari jumlah dokumen yang mengandung *term* tersebut. Log dari jumlah dokumen secara keseluruhan *D* dibagi dengan jumlah dokumen *d* yang mengandung *term* t_i menghasilkan sebuah bobot *term* terhadap dokumen-dokemen yang ada.

$$W_{IDF}(t_i, d_j) = 1 + \log\left(\frac{D}{d(t_i)}\right) \quad (2)$$

IBF dihitung dengan memperhatikan kemunculan *term* pada buku atau kitab. Persamaan (3) menunjukkan proses perhitungan IBF. Term yang jarang muncul dalam buku atau kitab justru merupakan *term* yang paling bernilai. Kepentingan tiap *term* diasumsikan memiliki proporsi yang berkebalikan dengan jumlah buku yang mengandung *term*. IBF dapat diperoleh dari hasil log pada jumlah keseluruhan kitab atau buku *B* dibagi dengan jumlah kitab atau buku *b* yang mengandung *term* t_i menghasilkan suatu pembobotan *term* terhadap kitab atau buku yang ada.

$$W_{IBF}(t_i, b_k) = 1 + \log\left(\frac{B}{b(t_i)}\right) \quad (3)$$

IPF dihitung dengan memperhatikan kemunculan *term* pada kelompok yang merupakan kandidat *user preference* seperti pada persamaan (4). Dalam penelitian ini, yang dimaksud dengan kelompok *user preference* adalah empat metodologi fiqih yang telah dijelaskan pada sub bab data. *Term* yang jarang muncul pada keempat metodologi fiqih adalah *term* yang sangat bernilai. Kepentingan tiap *term* diasumsikan memiliki proporsi yang berkebalikan dengan jumlah metodologi fiqih yang mengandung *term*. Nilai IPF dapat diperoleh dari hasil log pada proses pembagian antara jumlah keseluruhan preferensi *P* dengan jumlah preferensi *p* yang mengandung *term* t_i.

$$W_{IPF}(t_i, p_l) = 1 + \log\left(\frac{P}{p(t_i)}\right) \quad (4)$$

Tingkat preferensi atau keberpihakan juga dapat mempengaruhi hasil IPF. Oleh karena itu, pada perhitungan IPF ditambahkan nilai α yang merupakan nilai keberpihakan terhadap suatu preferensi yang bernilai antara 0 sampai 1 sehingga menjadi IPFα. Pada proses perhitungan IPFα, nilai W_{IPF_α} dapat dihitung dengan beberapa rumus yang berbeda, tergantung dari *user preference* (*UP*) dan query (*Q*) seperti pada persamaan (5).

$$W_{IPF_\alpha}(t_i, d_j, p_l) = \begin{cases} \left(1 + log\left(\frac{P}{p_{(t_i)}}\right)\right) \times \left(\frac{\alpha}{2} + 0.5\right), & d_j \in UP, t_i \in Q \\ \left(1 + log\left(\frac{P}{p_{(t_i)}}\right)\right) \times \left(1 - \left(\frac{\alpha}{2} + 0.5\right)\right), & d_j \notin UP, t_i \in Q \\ \left(1 + log\left(\frac{P}{p_{(t_i)}}\right)\right), & t_i \notin Q \end{cases} \quad (5)$$

P merupakan jumlah preferensi yang ada pada sistem. Sedangkan $p(t_i)$ merupakan jumlah preferensi atau metodologi *fiqih* yang mengandung *term* yang bersangkutan. α merupakan nilai antara 0 sampai 1, yang ditentukan oleh pengguna. Apabila nilai $\alpha = 0$ maka tidak ada preferensi yang dijadikan prioritas oleh pengguna. Jika nilai $\alpha = 1$ maka akan dianggap sebagai prioritas penuh terhadap preferensi dari pengguna.

Untuk mendapatkan pembobotan secara keseluruhan [5]. Maka, pembobotan-pembobotan yang telah dilakukan sebelumnya digabungkan menjadi satu dengan operasi perkalian seperti pada persamaan (6).

$$W_{TF.IDF.IBF.IPF_\alpha}(t_i, d_j, b_k, p_l) = W_{TF.IDF}(t_i, d_j) \times W_{IBF}(t_i, b_k) \times W_{IPF_\alpha}(t_i, d_j, p_l) \quad (6)$$

LSI merupakan metode *indexing* pada *information retrieval* 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 [6][8].

Hasil dari pembobotan TF.IDF.IBF.IPFα dipetakan menjadi sebuah matriks. Nilai TF.IDF.IBF.IPFα dari setiap *term* terhadap dokumen akan dipetakan menjadi matriks *A*. Dimana T_i merupakan *term* ke-i pada D_j dokumen ke-j. Sedangkan nilai TF.IDF.IBF.IPFα dari setiap *term* terhadap *query* akan dipetakan menjadi matriks Q. Dimana T_i merupakan *term* ke-i pada *query* q. Proses tersebut dapat dilihat pada persamaan (7). Matriks *A* akan melalui proses *decomposition* menggunakan operasi SVD pada persamaan (8) sehingga menghasilkan matriks *U, S* dan *V* yang dapat dilihat pada persamaan (9)(10)(11). Sedangkan formula $V_{transpose}$ dapat dilihat pada persamaan (12).

$$A = \begin{bmatrix} T_1D_1 & T_1D_2 & \cdots & T_1D_j \\ T_2D_1 & T_2D_2 & \cdots & T_2D_j \\ \vdots & \vdots & \ddots & \vdots \\ T_iD_1 & T_iD_2 & \cdots & T_iD_j \end{bmatrix} = Q \begin{bmatrix} T_1Q \\ T_2Q \\ \vdots \\ T_iQ \end{bmatrix} \quad (7)$$

$$A = U.S.V^T \quad (8)$$

Setelah matriks *U, S* dan *V* terbentuk, proses selanjutnya yang akan dilakukan adalah proses *rank 2 approximation*. Proses *rank 2 approximation* merupakan tahapan proses dalam LSI yang dilakukan untuk mereduksi matriks *U, S* dan *V* dengan tujuan untuk mengambil nilai koordinat vektor dari setiap dokumen maupun *query*. Proses ini dilakukan dengan mengambil 2 kolom pertama dari matriks *U* dan matriks *V* dan mengambil 2 baris dan kolom pertama pada matriks S. Proses *Rank* 2 *Approximation* dapat dilihat pada persamaan (13) (14) (15).

Dari proses sebelumnya maka diperoleh matriks U_k, S_k dan V_k yang merupakan hasil reduksi dari proses *Rank* 2 *Approximation*. Matriks U_k dan S_k akan digunakan kembali untuk menghitung koordinat vektor *query*, sedangkan pada matriks V_k akan dilakukan proses *transpose* menjadi matriks V^T_k untuk mendapatkan nilai koordinat vektor dokumen. Nilai koordinat vektor dokumen direpresentasikan pada setiap baris dari matriks V^T_k. Berdasarkan proses perhitungan pada persamaan (16), maka nilai koordinat vektor dokumen diperoleh dari matriks *Vk(transpose)* Setelah nilai koordinat vektor dokumen ditemukan, proses selanjutnya adalah menghitung nilai koordinat vektor *query* dengan menggunakan persamaan (17)(18).

$$U = \begin{bmatrix} \text{TF. IDF. IBF. IPF t1, D1} & \cdots & \text{TF. IDF. IBF. IPF t1, Dm} \\ \vdots & \ddots & \vdots \\ \text{TF. IDF. IBF. IPF tn, D1} & \cdots & \text{TF. IDF. IBF. IPF tn, Dm} \end{bmatrix} \quad (9)$$

$$S = \begin{bmatrix} \text{TF. IDF. IBF. IPF s(1,1)} & \cdots & \text{TF. IDF. IBF. IPF s(1, m)} \\ \vdots & \ddots & \vdots \\ \text{TF. IDF. IBF. IPF s(n, 1)} & \cdots & \text{TF. IDF. IBF. IPF s(n, m)} \end{bmatrix} \quad (10)$$

$$V = \begin{bmatrix} \text{TF. IDF. IBF. IPF v(1,1)} & \cdots & \text{TF. IDF. IBF. IPF v(1, m)} \\ \vdots & \ddots & \vdots \\ \text{TF. IDF. IBF. IPF v(n, 1)} & \cdots & \text{TF. IDF. IBF. IPF v(n, m)} \end{bmatrix} \quad (11)$$

$$V_{transpose} = \begin{bmatrix} V_{1,1} & \cdots & V_{n,1} \\ \vdots & \ddots & \vdots \\ V_{1,m} & \cdots & V_{n,m} \end{bmatrix} \quad (12)$$

$$U_k = \begin{bmatrix} U_{1,1} & \cdots & U_{1,2} \\ \vdots & \ddots & \vdots \\ U_{n,1} & \cdots & U_{n,2} \end{bmatrix} \quad (13)$$

$$S_k = \begin{bmatrix} S_{1,1} & S_{1,2} \\ S_{2,1} & S_{2,2} \end{bmatrix} \quad (14)$$

$$V_k = \begin{bmatrix} V_{1,1} & \cdots & V_{1,2} \\ \vdots & \ddots & \vdots \\ V_{n,1} & \cdots & V_{n,2} \end{bmatrix} \quad (15)$$

$$V_{k_{transpose}} = \begin{bmatrix} V_{1,1} & \cdots & V_{n,1} \\ \vdots & \ddots & \vdots \\ V_{2,1} & \cdots & V_{n,2} \end{bmatrix} \quad (16)$$

$$q = q^T.Uk.Sk \quad (17)$$

Hasil dari koordinat vektor dokumen dan koordinat vektor *query* pada persamaan (19) digunakan sebagai acuan untuk mengukur kemiripan terhadap vektor-vektor dokumen dengan menggunakan metode *cosine similarity*.

$$q^{transpose} = [\text{TF. IDF. IBF. IPF t1 ... TF. IDF. IBF. IPF tn}] \begin{bmatrix} U_{1,1} & \cdots & U_{1,2} \\ \vdots & \ddots & \vdots \\ U_{n,1} & \cdots & U_{n,2} \end{bmatrix} S_k^{-1} \quad (18)$$

$$q = (x, y) \quad (19)$$

2.4. *Cosine Similarity*

Cosine similarity digunakan untuk mengukur kemiripan antara dokumen dengan *query* yang dimasukkan oleh pengguna. Nilai kosinus sudut antara vektor *query* dan setiap vektor dokumen diurutkan dari yang terbesar hingga yang terkecil. Hasil nilai kosinus berkisar antara 0 hingga 1, dimana nilai 0 menandakan bahwa *query* dan dokumen tidak mirip sama sekali, dan 1 menandakan bahwa antara *query* dan dokumen benar-benar identik.

Nilai kosinus didapat dari perkalian *dot product* dari vektor *query* $\vec{q}$ dengan vektor dokumen $\vec{d}_j$ dibagi dengan hasil kali panjang vektor $\vec{q}$ dengan vektor $\vec{d}_j$ persamaan (20).

$$\cos(q, d_j) = \frac{\vec{q} \bullet \vec{d}_j}{|\vec{q}||\vec{d}_j|}, \quad (20)$$

3. HASIL DAN PEMBAHASAN

TABEL 1.
CONFUSION MATRIKS

	Relevan	Tidak relevan
Ditampilkan	TP	FP
Tidak ditampilkan	TN	FN

Pengujian pada penelitian ini menggunakan pendekatan *precision*, *recall*, dan *f-measure*. Ketiga jenis pengujian tersebut merupakan cara untuk melakukan evaluasi yang umum digunakan dalam *Information Retrieval* (IR). Berdasarkan Tabel 1 penghitungan *precission*, *recall*, *dan f-measure* dapat dilakukan dengan merujuk pada persamaan (21), (22), dan (23).

$$precision\ (P) = \frac{tp}{tp+fp} \quad (21)$$

$$recall\ (R) = \frac{tp}{tp+tn} \quad (22)$$

$$Fmeasure\ (F) = \frac{2 \times precision \times recall}{precision+recall} \quad (23)$$

Pengujian dilakukan dengan memasukkan *query* kedalam sistem untuk menemukan dokumen-dokumen yang relevan terhadap *query* tersebut. Daftar *query* yang diujicobakan dapat dilihat pada Tabel 2.

TABEL 2.
QUERY PENGUJIAN

ID	*Query*
Q1	استعمال آنية من لا تحل ذبيحته
Q2	الماء الذي ينجس والذي لا ينجس
Q3	الماء المسخن
Q4	الماء المشمس
Q5	الوضوء من النوم
Q6	غسل الجنابة

Dari *query-query* yang diujicobakan menghasilkan sebuah perankingan dokumen berdasarkan relevansi serta preferensi dari pengguna. Berdasarkan pendekatan persamaan (9),(10), dan (11) dihasilkan sebuah nilai pada setiap *query* yang diujicobakan. Hasil tersebut dapat dilihat pada Tabel 3.

TABEL 3.
HASIL PENGUJIAN

ID	Precision (%)	Recall (%)	F-Measure (%)
Q1	86,25	88,33	87,09
Q2	78,75	83,87	81,16
Q3	86,25	89,72	87,91
Q4	83,75	87,07	85,31
Q5	83,75	85,11	84,34
Q6	88,75	86,58	87,51

Nilai precision tertinggi sebesar 88,75% pada *query* ke 6. Hal ini dikarenakan relevansi *term* pada *query* terhadap dokumen yang ada memiliki similaritas yang cukup tinggi. Sedangkan nilai *precision* terendah terdapat pada ujicoba dengan *query* ke 2. Hal tersebut dikarenakan hanya sedikit dokumen dalam sistem yang memiliki kesamaan dengan *query* tersebut. Nilai *f-measure* tertinggi sebesar 87,91% hal ini dikarenakan nilai *recall* yang besar atau jumlah dokumen yang relevan banyak yang di-*retrieve*. LSI yang digabungkan dengan metode pembobotan TF.IDF.IBF.IPFα meningkatkan nilai *cosine similarity*.

Dilihat dari hasil pengujian, metode yang diusulkan dapat meningkatkan nilai *f-measure* dibandingkan dengan metode yang belum mempertimbangkan hubungan semantik antar *term* [5]. Perbandingan tersebut dapat dilihat pada Tabel 4.

TABEL 4.
PERBANDINGAN METODE

Metode	*Precisionl (%)*	*Recall (%)*	*F-Measure (%)*
TF.IDF.IBF.IPFα	100	75	85,7
TF.IDF.IBF.IPFα.LSI	88,75	89,72	87,91

4. KESIMPULAN

Dalam pencarian dokumen berbasis preferensi, hasil perankingan perlu memepertimbangkan hubungan semantik antar *term*. *TF.IDF.IBF.IPFα.LSI* merupakan metode pembobotan *term* berbasis preferensi pengguna yang dibentuk kedalam matriks dan didekomposisi dengan *Latent Semantic Indexing*. Metode usulan tersebut dapat mempertimbangkan preferensi dari pengguna dan juga hubungan semantik antar *term*. Hal tersebut dibuktikan dengan nilai *precision* yaitu 88,75 %, *recall* 89,72% dan *f-measure* 87,91% yang meningkat dibandingkan dengan metode berbasis preferensi yang tidak mempertimbangkan hubungan semantik antar term.

Untuk kedepannya metode ini dapat dikembangkan dengan adanya tambahan preferensi dari pengguna atau multi preferensi dan juga diuji cobakan dengan studi kasus dokumen berbahasa lain, seperti bahasa Indonesia, Inggris atau bahasa lainnya.

DAFTAR PUSTAKA

[1] ELRAOUF, A., ESRAA, BADR, L., NAGWA, TOLBA, FAHMY, M. 2010. An Efficient Ranking Module for an Arabic Search Engine. *International Journal of Computer Science and Network Security*, 10(2), 218-225.

[2] HERBRICH, R., GRAPAEL, T., BOLLMAN-SDORRA, P., OBERMAYER, K.1998. Learning preference relations for information retrieval. *ICML-98 Workshop: text categorization and machine learning*, July, Berlin. 80-84.

[3] FAUZI, M. A., ARIFIN, A. Z., YUNIARTI, A. 2013. Term Weighting Berbasis Indeks Buku dan Kelas untuk Perankingan Dokumen Berbahasa Arab. *Jurnal Lontar Komputer*,5(2).

[4] WAHIB, A., PASNUR, SANTIKA P.P., ARIFIN, A. Z. 2015. Perankingan Dokumen Berbahasa Arab Menggunakan Latent Semantic Indexing. *Jurnal Buana Informatika*, 6(2), 83-92.

[5] HOLLE, K. F. H., ARIFIN, A. Z., PURWITASARI, D. 2015. Preference Based Term Weighting for Arabic Fiqh Document Ranking. *Jurnal Ilmu Komputer dan Informasi*, 8(1).

[6] LARKEY, L. S., CONNELL, M. E. 2006. Arabic Information Retrieval at UMass in TREC-10. *Massachusetts Univ Amherst Center for Intelligent Information Retrieval.*

[7] KHOJA, S., GARSIDE, R. 1999. Stemming Arabic Text. Computing Department, Lancaster University, Lancaster.

[8] PAPADIMITRIOU, C. H., TAMAKI, H., RAGHAVAN, P., VEMPALA, S. 1998. Latent semantic indexing: A probabilistic analysis. *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, May. 159-168.