ISSN: 2302-3805

SISTEM REKOMENDASI TAG PADA DOKUMEN BLOG MENGGUNAKAN *LATENT SEMANTIC INDEXING*

Lailil Muflikhah[1)] , Nurul Fadilah[2)] , Achmad Ridok[3)]

[1,3)] Program Teknologi Informasi dan Ilmu Komputer
[2)] Program Studi Ilmu Komputer
[1,2,3)] Universitas Brawijaya, Malang

email : lailil@ub.ac.id[1)], aishteruu@gmail.com[2)], acridok@gmail.com[3)]

Abstrak

Sebagian besar dokumen web memiliki bentuk berupa blog. Pelabelan dokumen web dengan tag membutuhkan waktu lebih lama karena banyaknya tag di dalamnya. Oleh karena itu, pada penelitian ini dimaksudkan untuk membuat sistem rekomendasi pemberian tag yang berfungsi sebagai navigasi dan titik temu hasil pencarian dokumen. Adapun metode yang digunakan adalah Latent Semantic Indexing (LSI) dengan menggunakan pendekatan matematis sehingga mendapatkan hubungan yang tidak nampak antara kata kunci yang terdapat pada teks dokumen, yaitu Tag-In (TI) dan kata kunci yang tidak terdapat pada teks dokumen, yaitu Tag-Out (TO). Ekstraksi dari TI pada dokumen didapatkan dengan menghitung bobot kata. Dari sejumlah TI yang memiliki bobot tertinggi, didapatkan TO yang muncul bersama sejumlah TI tersebut dari graf data latih. Kemudian dibentuk sebuah matriks yang berisi nilai co-occurrence antara TI dan TO yang telah diperoleh. Matriks tersebut didekomposisi menggunakan Singular Value Decomposition (SVD). Hasil dekomposisi yang merupakan vektor masing-masing TO dihitung kedekatannya dengan vektor query dan diambil jarak yang terdekat. Hasil pengujian menunjukkan nilai recall terbaik dicapai pada saat TI=3 dan TO=6, yaitu 0.69, dimana jumlah rata-rata tag tiap dokumen uji adalah 5 tag. Nilai k yang tepat dan menghasilkan akurasi terbaik bergantung pada rank matriks dengan F-Measure terbaik untuk rank = 2.

Kata kunci : ***dokumen, blog, LSI, tag, SVD***

1. Pendahuluan

Saat ini semakin tinggi kebutuhan informasi terkini dalam berbagai aspek, dimana setiap pengguna *internet* dapat ikut berperan serta mempublikasikan data informasi melalui media populer yang dikenal sebagai *Weblog* atau *blog*. *Weblog* atau *blog* adalah dokumen *web* yang terdiri dari isi (*content*), yang diberi label tanggal dan disajikan secara kronologis [1]. Seiring dengan semakin banyaknya *blog* yang terpublikasikan, maka dibutuhkan sistem untuk mengorganisasikan arsip *blog* berdasarkan topik sehingga memudahkan dalam pencariannya. Salah satu metode yang populer adalah *social tagging*, yaitu pemberian *tag* pada setiap artikel *blog*.

Tag adalah kata kunci (*keyword*) deskriptif atau frase yang mewakili isi dari sebuah dokumen atau objek [1]. Namun, pengertian *tag* berbeda dengan *keyword*. Jika *keyword* mengacu pada intisari dokumen, maka untuk memberikan *tag* pada sebuah *content* tidak diperlukan adanya ekstraksi dokumen terlebih dahulu. Pengguna bebas memberikan *string* apa saja sebagai *tag*, yang dapat mendeskripsikan dan menyimpulkan isi dokumen. Selain memudahkan dalam pencarian, *tag* juga berfungsi sebagai alat navigasi *web* sehingga pengguna dapat memperoleh informasi yang relevan dan berhubungan satu sama lain. Pemberian *tag* secara otomatis disebut juga *autotag* atau rekomendasi *tag*.

2. TINJAUAN PUSTAKA

Penelitian terdahulu mengenai sistem rekomendasi *tag* content based atau ekstraksi *keyword* telah dilakukan dengan berbagai metode. Diantaranya adalah menggunakan pendekatan statistik dan keterhubungan antar kata. Dalam pendekatan statistik, salah satu penelitian yang telah dilakukan adalah ekstraksi *keyword* menggunakan Naive Bayes dimana perhitungan pembobotan kata salah satunya menggunakan TF/IDF [2]. Sedangkan dalam pendekatan keterhubungan antar kata, penelitian yang telah dilakukan adalah ekstraksi *keyword* menggunakan TextRank [3], dimana pembobotan kata menggunakan perhitungan kemunculan bersama (*co-occurrence*) antar kata dalam dokumen menggunakan graf.

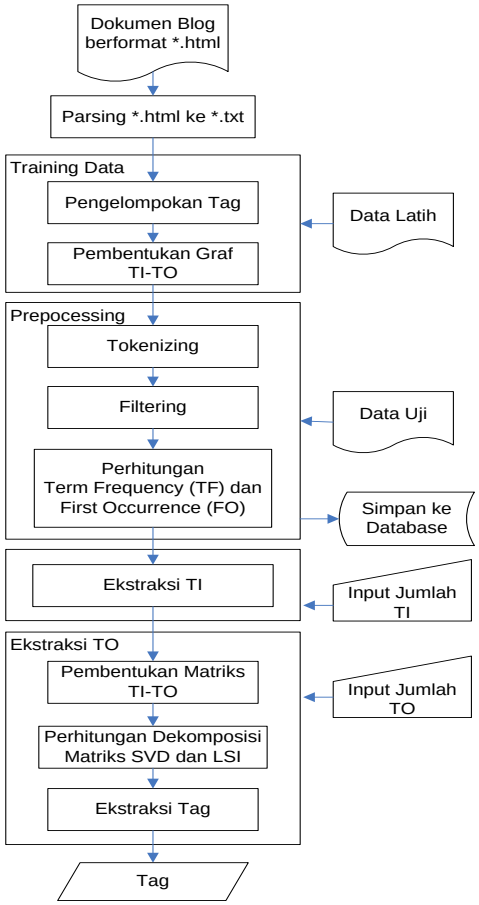
Dalam penelitian ini, *tag* yang akan direkomendasikan tidak hanya berasal dari ekstraksi *keyword* dari isi dokumen saja, melainkan juga berupa *string* atau kata kunci yang tidak terdapat dalam teks. Sekitar 43% *tag* muncul dalam teks dokumen *blog* dan yang lainnya sebanyak 56% tidak terdapat dalam isi teks [5]. *Tag* yang berada didalam tubuh teks dokumen *blog* disebut dengan *Tag-In* dan yang tidak berada dalam teks

ISSN: 2302-3805

disebut dengan *Tag-Out*. *Tag-In* diperoleh melalui pendekatan statistik, yaitu menggunakan perhitungan pembobotan kata *TermScore* dengan atribut frekuensi dan jarak kemunculan kata pertama dalam dokumen. Sedangkan *Tag*-Out diperoleh melalui pendekatan hubungan antar kata, yaitu hubungan antara *Tag-In* dan *Tag-Out* dengan menggunakan metode LSI.

3. Metode Penelitian

Sistem yang dibuat merupakan sistem yang dapat mengekstraksi sekumpulan *tag* secara otomatis ada *content blog* berdasarkan hasil ekstraksi *keyword* dan hubungan antar *tag*. Ekstraksi keyword (*Tag-in*) didapatkan dari pengurutan hasil nilai bobot kata dalam sebuah dokumen uji. Sedangkan hubungan antar *tag* didapatkan dari graf antara *tag* yang ada dalam tubuh *content* (*Tag-in*) dan *tag* yang tidak terdapat dalam tubuh *content* (*Tag-out*) menggunakan metode *Latent Semantic Indexing*. Hasil *tag* dari *Tag-in* dan *Tag-out* digabungkan menjadi sekumpulan *tag* yang direkomendasikan. Secara keseluruhan terlihat pada Gambar 1.



Gambar 1 Rancangan Alur Sistem

3.1. *Prepocessing*

Preprocessing dalam system ini meliputi *tokenizing*, *filtering*, perhitungan *term frequency* (TF) dan perhitungan *first occurrence*(FO). *Stemming* tidak diperlukan karena sifat dari tag yang bebas dari kaidah tata bahasa. Perhitungan TF dan FO digunakan untuk pembobotan kata. FO adalah menyatakan jarak posisi kemunculan kata pertama pada dokumen. Hal ini didasarkan pada asumsi bahwa kata-kata penting yang menjadi topik isi dokumen berada pada beberapa paragraf pertama.

3.2 Ekstraksi *Keyword* (*Tag-In*)

Tahap ekstraksi TI merupakan proses untuk mendapatkan sekumpulan kata kunci TI sebanyak jumlah input TI dari *user*. TI yang diambil adalah sebanyak *m* kata dengan bobot *score* terbesar. Semakin tinggi nilai TF dan semakin rendah nilai FO maka semakin besar kemungkinan kata tersebut merupakan kata kunci. Pembobotan *term score* dirumuskan sebagai berikut[6]:

$$TermScore = \left[\frac{1}{2} + \frac{1}{2} . \frac{nrWords - FO}{nrWords}\right] . TF \quad (1)$$

dimana :
nrWords = jumlah total kata dalam dokumen
FO = jarak posisi pertama kemunculan kata
TF = frekuensi kata

3.3 *Singular Value Decomposition* (SVD)

Setiap proses dekomposisi akan memfaktorkan sebuah matriks menjadi lebih dari satu matriks. Salah satu teknik dekomposisi matriks adalah *Singular Value Decomposition* (SVD). SVD berkaitan erat dengan singular value atau nilai singular dari sebuah matriks yang merupakan salah satu karakteristik matriks. Bentuk dekomposisi SVD sebagaimana dalam persamaan 2 [8] :

$$A = USV^T \quad (2)$$

Dimana :
A = matriks mxn
U = matriks yang dibentuk oleh eigen vektor normal matriks AA^T
S = matriks singular : matriks diagonal mxn yang entri-entrinya adalah nilai singular dari matriks A yang elemen diagonalnya terurut turun dan non-negatif.
V = matriks yang dibentuk oleh eigen vektor normal matriks A^TA
V^T = transpose dari matriks V

ISSN: 2302-3805

3.4 *Latent Semantic Indexing* (LSI) Untuk Ekstraksi Tag (*Tag-Out*)

Matriks berukuran mxs dibentuk pada saat proses dokumen uji, setelah tahap ekstraksi TI. Sebanyak m jumlah TI dicari pasangan *co-occurrence*-nya di dalam graf. Maka didapatkan sebanyak s jumlah TO yang paling sedikit memiliki nilai *co-occurrence* 1 dengan paling sedikit 1 buah TI. Selanjutnya dibentuk matriks A_{mxs} yang berisikan bobot untuk m TI dan s TO yang memiliki *co-occurrence* pada *bipartite* graph (bigraf). Bigraf dibentuk saat training data.

Setelah matriks terbentuk, matriks tersebut didekomposisi menggunakan persamaan (2) sehingga menghasilkan matriks U, S, dan V. Kemudian dilakukan reduksi matriks menjadi U_k, V_k, dan S_k. Selanjutnya rumus LSI diterapkan pada matriks-matriks tersebut sebagaimana dalam persamaan 3: [5][9]:

$$q = q^T U_k S_k^{-1} \qquad (3)$$

Dimana :
q = vector query baru yang dihasilkan
q^T = transpose dari vector query TI-TO berukuran m, dengan masing-masing elemennya adalah 1.
k = besaran untuk mereduksi matriks, k ≤ r (dimana r adalah *rank* matriks)
U_k = matriks U_{mxk}
S_k^{-1} = inverse dari matriks S_{kxk}
Baris-baris pada matriks V_k adalah kumpulan dari vektor eigen, maka tiap baris matriks V_k merupakan koordinat vektor masing-masing dokumen. Tahap terakhir adalah mengukur tingkat kesamaan (*similiarity*) antara vektor queri dan masing-masing vektor dokumen menggunakan rumus perhitungan *cosine similiarity*.

$$sim(q,d) = \frac{q \bullet d}{|q||d|} \qquad (4)$$

Setelah didapatkan hasil kedekatan masing-masing vector kemudian dilakukan pengurutan secara *descending* hasil *similiarity* antara *query* dan semua dokumen. Semakin besar nilai *similiarity* artinya semakin dekat hubungan antara TO dengan m TI yang telah dipilih pada proses ekstraksi keyword.

3.5 Metode Evaluasi

Ukuran evaluasi terhadap sistem rekomendasi *tag* adalah sebagai berikut [7]:

1. *Top*-k *accuracy*. Prosentase dari dokumen yang rekomendasi *tag*nya benar paling sedikit 1 dari sejumlah k *tag* teratas yang direkomedasikan (*top-*k^{th} *tag*).
2. *Exact-k Accuracy*. Prosentase banyaknya dokumen yang hasil rekomendasi *tag*nya benar sebanyak k dari sejumlah k *tag* teratas yang direkomendasikan (*exact-*k^{th} *tag*)
3. *Tag-recall*. Prosentase hasil *tag* rekomendasi yang benar dari seluruh *tag* yang telah diberikan oleh *user*. *Tag* rekomendasi yang benar adalah *tag* yang sama antara *tag* yang dihasilkan oleh program dan *tag* yang dibuat oleh *user*.
4. *Tag-precision*. Prosentase hasil *tag* rekomendasi yang benar dari seluruh *tag* yang dihasilkan oleh algoritma program.
5. *F-Measure* merupakan gabungan antara *precision* dan *recall*.

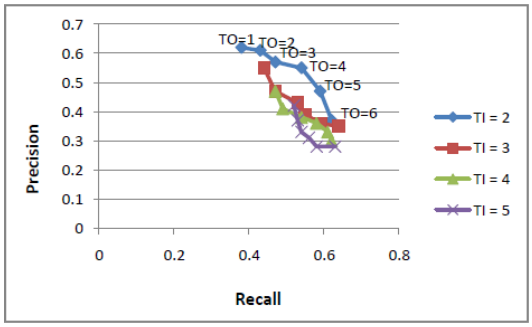
$$F = \frac{2\ x\ recall\ x\ precision}{recall + precision} \qquad (5)$$

4. Hasil dan Pembahasan

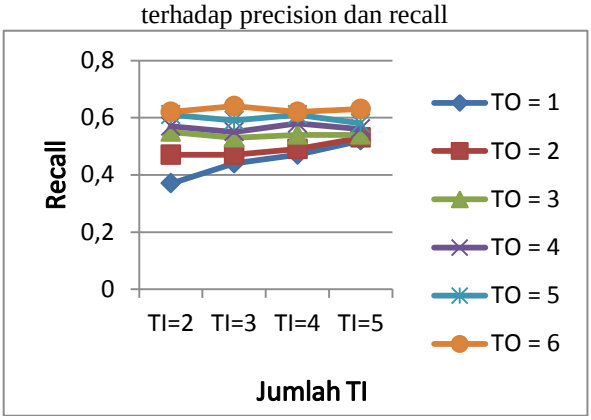
Uji coba terhadap sistem rekomendasi *tag* dengan metode LSI dilakukan dengan 2 parameter, yaitu banyaknya TI dan TO serta jumlah nilai k terhadap *rank* matriks. Kedua parameter tersebut digunakan untuk mengetahui hasil akurasi terbaik dari sistem. Nilai *k* merupakan suatu variabel yang digunakan untuk mereduksi matriks. Semakin kecil nilai *k* maka semakin kecil ukuran matriks. Syarat dari nilai k adalah lebih kecil atau sama dengan nilai *rank* (r) matriks. Maka pada percobaan ini digunakan nilai *k = r, k = r-1,* dan *k = r-2*.
Percobaan dilakukan sebanyak 4 kali untuk masing-masing k = r dan k = r-1, dengan *input* TI sebanyak 2 sampai dengan 5. Untuk k= r-2 percobaan dilakukan sebanyak 3 kali, yaitu dengan *input* TI sebanyak 3 sampai dengan 5. Masing-masing percobaan dengan *input* TI, diinputkan TO sebanyak 1 sampai dengan 6.

Pada Gambar 2 nilai *precision* dan *recall* selalu berbanding terbalik dalam setiap percobaan. Semakin banyak TO yang dihasilkan maka semakin tinggi nilai *recall* yang didapatkan, akan tetapi semakin kecil nilai *precision* yang dihasilkan. Semakin banyak nilai TI yang diinputkan semakin tinggi pula *recall* pada setiap hasil TO dan jumlah *tag* yang benar pun semakin banyak. Semakin banyak TO yang direkomendasikan semakin banyak pula *tag* yang sama dengan *tag* yang diberikan oleh *user*. Meskipun nilai *precision* semakin kecil, belum tentu semakin banyak juga *tag* yang tidak sama dengan *tag* dari *user* karena jumlah total *tag* pada data uji, yaitu *tag* dari *user*, lebih kecil daripada jumlah *tag* yang dihasilkan sistem jika TI + TO > 5, dengan jumlah rata-rata *tag* per dokumen adalah 5 buah *tag*. Karena nilai *precision* semakin kecil dengan bertambahnya jumlah *tag* yang direkomendasikan maka digunakan analisa nilai *recall* untuk mengukur akurasi yang dipengaruhi oleh jumlah TI dan TO.

ISSN: 2302-3805



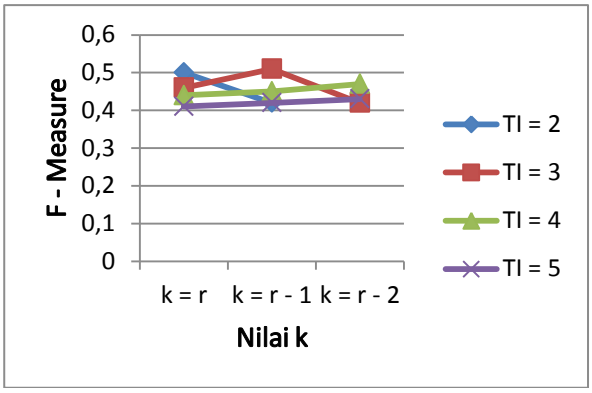
Gambar 2 Hubungan antara jumlah TI dan TO terhadap precision dan recall



Gambar 3 Grafik Hubungan Nilai Recall dari Jumlah TO dengan Jumlah TI

Nilai *recall* yang dihasilkan dalam setiap eksekusi jumlah TO bergantung pada jumlah TI yang diinputkan. Pada gambar 3 dapat dilihat bahwa semakin banyak jumlah TI, nilai *recall* pada jumlah TO yang sama semakin tinggi. Hal ini disebabkan oleh jumlah TI yang berperan dalam pembentukan kompleksitas matriks co-occurrence dan vector query. Semakin banyak jumlah TI maka semakin tinggi nilai *rank* matriks. Tingginya nilai *rank* matriks menyebabkan vektor TO dan vektor *query* baru yang dihasilkan semakin kompleks dan melibatkan banyak *node* dalam proses perhitungan nilai *similiarity*-nya sehingga nilai *recall* pun meningkat. Pada gambar 4.10 hasil *recall* terbaik dicapai pada saat TI=5. Pada posisi tersebut nilai *recall* dari masing-masing TO memiliki jarak yang semakin kecil karena nilai *recall* tersebut adalah nilai maksimum yang dapat dicapai oleh sistem.

Untuk mengukur akurasi berdasarkan nilai k digunakan *F-Measure* rata-rata dari masing jumlah TI. Pada gambar 4 dapat disimpulkan bahwa meskipun *input* jumlah TI sama namun didapatkan hasil yang berbeda pada setiap nilai *input* k yang berbeda. Nilai rata-rata *F-Measure* pada TI=2 dan k=r-2 tidak ada karena pada *input* TI=2 menghasilkan matriks co-occurrence dengan maksimal nilai rank adalah 2 sehingga k=0 dan menghasilkan matriks *null* yang tidak mungkin dapat diolah.



Gambar 4 Grafik Hubungan Nilai F-Measure dari Jumlah TI dengan Nilai k

Pada gambar 4 nilai *F-Measure* tertinggi pada TI=2 dihasilkan pada saat nilai k=r. Kemudian nilai *F-Measure* tertinggi pada TI=3 dihasilkan pada saat k=r-1, dan nilai *F-Measure* tertinggi pada TI=4 dan TI=5 dihasilkan pada saat nilai k=r-2. Hal ini menunjukkan bahwa uji coba dengan nilai k yang berbeda menghasilkan nilai *recall* dan *precision* yang berbeda pula meskipun jumlah TI dan TO yang diinputkan sama. Jadi tidak ada satu nilai *F-Measure* yang paling maksimum untuk k=r sampai dengan k=r-2 karena adanya ketergantungan antara nilai k dan jumlah TI yang diinputkan.

Nilai *F-Measure* maksimum untuk *k=r* ada pada TI=2 dan *rank* matariks yang dihasilkan dari *input* tersebut adalah 2. Jadi *k=r-2*. Begitupula dengan nilai *F-Measure* maksimum untuk *k=r-1* ada pada TI=3 dimana *rank* matriksnya adalah 3, maka *k=r-1=2*. Hasil yang sama juga didapatkan pada *k=r-2* dengan TI=4 yang menghasilkan *rank* matriks 4, maka nilai *k=r-2=2*. Dari pola tersebut dapat disimpulkan bahwa untuk matriks yang memiliki *rank* sebanyak 2 hingga 4 nilai k terbaik adalah 2. Jadi belum tentu semakin banyak *rank* matriks semakin banyak pula nila k yang dibutuhkan. Nilai k untuk hasil yang maksimum bekerja pada *range* interval *rank* matriks tertentu.

5. Kesimpulan

Dari hasil uji dan analisis yang telah dilakukan, dapat diambil kesimpulan sebagai berikut :

1. Metode *Latent Semantic Indexing* (LSI) dapat diimplementasikan untuk rekomendasi *tag* dokumen *blog* dengan tingkat akurasi yang dipengaruhi oleh jumlah TI, TO dan nilai *k* yang diinputkan.
2. Pengaruh dari parameter tersebut meliputi:
   a. *Input* jumlah TI dan TO. Semakin banyak jumlah TO yang diinputkan semakin besar nilai *recall* tetapi semakin kecil nilai *precision*nya. Juga semakin banyak jumlah TI yang diinputkan semakin besar nilai *recall* untuk setiap jumlah TO tetapi semakin kecil *precision*nya. Tetapi semakin

ISSN: 2302-3805

Sedikit jumlah TI dan TO semakin besar nilai *exact-k accuracy*.

b. Nilai k, dimana nilai k adalah besaran untuk mereduksi matriks. Namun hasil maksimum untuk setiap *input* k juga dipengaruhi oleh jumlah TI. Juga dapat disimpulkan bahwa untuk mencapai hasil *F-Measure* terbaik digunakan nilai k=2 untuk *range rank* matriks 2 hingga 4.

Daftar Pustaka

[1] LEE, S. O., & CHUN, A. H. 2008. *Automatic Tag Recommendation for Web 2.0 Blogosphere by Extracting Keywords from Similar Blogs*. http://www.wseas.us/e-library/conferences/2008/hangzhou/acacos/51-586-352.pdf tanggal akses : 11 Februari 2011

[2] Uzun, Y. 2008. *Keyword Extraction Using Naïve Bayes*. www.cs.bilkent.edu.tr/~guvenir/courses/cs550/Workshop/Yasin_Uzun.pdf tanggal akses : 22 Februari 2011

[3] Mihalcea, R., & Tarau, P. 2004. *TextRank: Bringing Order into Texts*. http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf tanggal akses : 16 Februari 2011

[4] Moldovan, A., Bot, R. I., & Wanka, G. 2008. *Latent Semantic Indexing for Patent Documents*. http://www-*user*.tu-chemnitz.de/~rabot/pdf/jour05-04.pdf tanggal akses : 18 Februari 2011

[5] Liu, Y., Liu, M., Xiang, L., Yang, Q., & Chen, X. 2008. *Automatic Tag Recommendation for Weblogs*.www.nlpr-*eb*.ia.ac.cn/2009papers/gjhy/gh64.pdf tanggal akses : 05 Januari 2011

[6] Chirita, P. A., Costache, S., Handschuh, S., & Nejdl, W. 2007. *PTAG: Large Scale Automatic Generation of Personalized Annotation TAGs for the Web*. http://www2007.org/papers/paper481.pdf tanggal akses : 11 Februari 2011

[7] Song, Y., Zhang, L., & Giles, C. L. 2008. *Automatic Tag Recommendation Algorithms for Social Recommender Systems*. ACM Transactions on Computational Logic

[8] Garcia, E. 2005. *SVD and LSI Tutorial 3 Computing the Full SVD of a Matrix*. http://www.miislita.com/information-retrieval-tutorial/svd-lsi-tutorial-3-full-svd.html tanggal akses :7 Januari 2011

[9] Garcia, E. 2005. *SVD and LSI Tutorial 4: Latent Semantic Indexing (LSI) How-to Calculations*. http://www.miislita.com/information-retrieval-tutorial/svd-lsi-tutorial-4-lsi-how-tocalculations.html tanggal akses :7 Januari 2011

Biodata Penulis

Lailil Muflikhah, memperoleh gelar Sarjana Komputer (S.Kom.), Program Studi Teknik Informatika Institut Teknologi Sepuluh Nopember, lulus tahun 1998. Tahun 2010 memperoleh gelar Master of Science (M.Sc.) pada Information Technology UTP Malaysia. Saat ini sebagai staf pengajar program Sarjana Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang. **Nurul Fadilah**, memperoleh gelar Sarjana Komputer (S.Kom.) pada program studi Ilmu Komputer Universitas Brwijaya Malang bulan Desember tahun 2011. **Achmad Ridok**, memperoleh gelar Sarjana (Drs.), pada jurusan Matematika Universitas Brawijaya, lulus tahun 1992. Kemudian tahun 1999 memperoleh gelar Master pada bidang ilmu komputer (M.Kom.) Universitas Indonesia. Saat ini, beliau sebagai staf pengajar Sarjana Program Teknologi Informasi dan Ilmu Komputer Universitas Indonesia.

ISSN: 1979 - .....