Sistem Deteksi Kemiripan Antar Dokumen Teks Menggunakan Model Bayesian Pada Term *Latent Semantic Analysis* (LSA)

Danang Wahyu Wicaksono, Mohammad Isa Irawan, dan Alvida Mustika Rukmi
Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
e-mail: alvidamustika@ymail.com

***Abstrak*—Metode *Latent Semantic Analysis*(LSA) adalah suatu metode yang mampu merepresentasikan hubungan antar dokumen teks melalui *term* serta dapat menilai kemiripan antar dokumen teks tersebut. Namun, metode LSA hanya menilai kemiripan antar dokumen teks melalui frekuensi *term* yang ada pada masing-masing dokumen teks sehingga mempunyai kelemahan yaitu tidak memperhatikan urutan atau tata letak *term* tersebut yang secara tidak langsung berpengaruh pada makna yang terkandung pada masing-masing dokumen. Oleh karena itu, digunakan model Bayesian pada *term* yang dihasilkan oleh LSA tersebut untuk menjaga dan memperhatikan urutan *term*dalam mendeteksi kemiripan antar dokumen teks sehingga struktur kalimat tetap terjaga dan mendapat hasil penilaian kemiripan antar dokumen teks yang lebih baik. Jika terdapat dua dokumen yang saling salin (*copy*) namun struktur kalimatnya diubah dan dibandingkan pada LSA dengan menggunakan *cosine similarity* maka akan didapat hasil yang sama seperti kedua dokumen ini dibandingkan tanpa perubahan struktur kalimat, sedangkan jika dibandingkan dengan menggunakan model Bayesian pada *term*, dokumen-dokumen yang mempunyai perbedaan struktur kalimat akan diperlakukan berbeda.**

Kata Kunci*—model Bayesian, LSA, *document similarity

I. PENDAHULUAN

DIGITALISASI pengolahan informasi dengan menggunakan komputer menghasilkan fasilitas yang *copy-paste* (salin-tempel) sehingga memudahkan pengolahan data sesuai dengan kebutuhan misalnya memenuhi tugas kuliah, membuat *paper*, dan sebagainya. Hal ini tentu berpotensi terjadinya tindakan penjiplakan suatu karya tulis tanpa ijin seperti plagiat.

Plagiarisme atau sering disebut plagiat adalah penjiplakan atau pengambilan karangan, pendapat, dan sebagainya dari orang lain dan menjadikannya seolah karangan dan pendapat sendiri [1]. Namun jika pengambilan karangan tersebut disertai dengan mencantumkan asalnya yaitu nama pengarang serta judul karangan yang diambil, maka tindakan pengambilan karangan tersebut bukan merupakan plagiat. Oleh karena itu, perlu dilakukan pemeriksaan kemiripan antar dokumen, dalam hal ini adalah dokumen teks, sebagai langkah validasi (*validation*) keterkaitan dan hubungan antar dokumen tersebut.

Salah satu metode untuk mendeteksi kemiripan antar dokumen teks yaitu LSA (*Latent Semantic Analysis*) [2] namun metode ini menilai kemiripan antar dokumen teks dengan memanfaatkan frekuensi kemunculan dari *term* [3] yang dihasilkannya sehingga keakuratan penilaian masih belum tentu ketika dokumen teks yang sedang dibandingkan memiliki tata letak *term* yang berbeda yang secara tidak langsung berpengaruh pada makna kalimat pada dokumen yang memuat *term* tersebut.

Penelitian yang dilakukan oleh Georgina Cosma menyebutkan bahwa pada deteksi plagiat (*plagiarism detection*), LSA membutuhkan algoritma atau metode lain sebagai pelengkap dan penyempurna untuk hasil deteksi yang lebih baik [4]. Oleh karena itu, dibutuhkan lebih banyak penelitian untuk metode deteksi plagiat dalam mencapai tujuan yaitu mendapatkan kesempurnaan metode deteksi plagiat pada dokumen teks.

Pada tugas akhir ini, akan digunakan kombinasi antara metode LSA dengan konsep model Bayesian. Metode LSA digunakan untuk mencari hubungan, keterkaitan, atau kesamaan antar dokumen teks dengan menghasilkan *term* kemudian pada langkah selanjutnya akan digunakan model Bayesian untuk menentukan pola (urutan) *term*pada dokumen yang diuji.

II. DASAR TEORI

A. Latent Semantic Analysis (LSA)

LSA adalah suatu metode untuk menemukan hubungan, keterkaitan, dan kemiripan antar dokumen-dokumen, penggalan dari dokumen-dokumen, dan kata-kata yang muncul pada dokumen-dokumen dengan memanfaatkan komputasi statistik untuk menggali dan merepresentasikan konteks yang digunakan sebagai sebuah arti kata untuk sejumlah *corpus* yang besar. *Corpus* adalah kumpulan teks yang memiliki kesamaan subjek atau tema.

Metode LSA menerima masukan (*input*) berupa dokumen teks yang selanjutnya akan dibandingkan kata-kata unik yang digunakan atau yang ada pada dokumen kemudian direpresentasikan sebagai matriks, dimana indeks dokumen-dokumen yang dibandingkan merupakan kolom matriks, kata unik (*term*) merupakan baris matriks, dan nilai dari matriks tersebut adalah banyaknya atau frekuensi kemunculan sebuah kata (*term*) di setiap dokumen [5].

Contoh *corpus* pada LSA dapat dilihat pada Gambar 1 dan bentuk matriks yangdihasilkan LSA dapat dilihat padaTabel 1.

Example of text data: Titles of Some Technical Memos

c1: *Human* machine *interface* for ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management *system*
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user* perceived *response time* to error measurement

m1: The generation of random, binary, ordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

Gambar 1. *Corpus* pada LSA

Tabel 1.
Matriks Hasil LSA

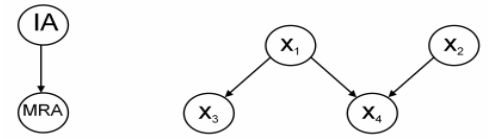
Term\Doc	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	0	1	1	0	1	0	0	0	0
user	0	1	1	2	0	0	0	0	0
system	0	1	0	0	1	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minor	0	0	0	0	0	0	0	1	1

B. Graphical Model

Sebuah model grafikal (*graphical model*) mempresentasikan bentuk distribusi peluang melalui sebuah grafik. Titik-titik (*nodes*) pada grafik menunjukkan variabel yang didefinisikan, penghubung (*edge*) antar *node* menunjukkan ketergantungan antar *node*.

Terdapat dua jenis model grafikal, dilihat dari jenis penghubung antar *node*. Jika penghubung antar *node* mengindikasikan hubungan antar variabel tanpa menunjukkan arah, maka disebut model grafikal tidak langsung (*undirected graphical model*). Jika penghubung antar *node* menunjukan arah ketergantungan antar variabel, maka disebut model grafikal langsung (*directed graphical model*). Contoh model grafikal ini dapat dilihat pada Gambar 2 [6].

Konsep model grafis inilah yang akan digunakan penulis sebagai implementasi model Bayesian dalam penentuan pola *term* dimana kemunculan term ke-n didahului oleh kemunculan *term* ke-(n-1), *term*ke-(n-2), dan seterusnya untuk n>1.



Gambar 2. *Directed Graphical Model*

C. Model Bayesian

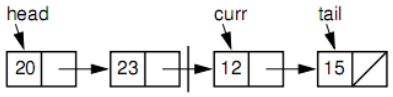
Secara umum, model Bayesian adalah DAG (*Directed Acyclic Graph*) yang dimana *node* merepresentasikan variabel acak, dapat berupa jumlah nilai tertentu, variabel laten, parameter tak tentu, atau hipotesis. *Edge* merepresentasikan ketergantungan kondisional atau kebersyaratan, jika terdapat *node* yang tidak terhubung melalui sebuah *edge* maka *node* tersebut dikatakan independen kondisional dengan *node* yang lain.

Dilihat dari namanya, model Bayesian tidak perlu atau tidak harus mengimplikasikan komitmen statistika Bayesian. Ini merupakan hal umum untuk menggunakan metode frekuentis untuk estimasi parameter-parameter dari CPD (*Conditional Probability Distribution*). Sebaliknya, disebut model Bayesian karena menggunakan aturan bayes yaitu kejadian bersyarat untuk dijadikan inferensi probabilitas, disebut *directed graphical model* yang lebih tepatnya. Namun demikian, model Bayesian adalah representasi yang berguna untuk hierarki Bayesian, dimana membentuk fondasi dasar dari aplikasi statistikaBayesian [7].

D. Struktur Data *Linked-List*

Berhubungan dengan model Bayesian yang akan digunakan, dimana kemunculan suatu *term* adalah bersyarat yaitu didahului oleh suatu *term* atau *term-term* lainnya sehingga dibutuhkan suatu struktur data yang mampu mengadaptasi model tersebut. Struktur data *linked-list*digunakan karena karakter dari *linked-list* yang berbentuk sekuen (urutan) dengan alokasi memori dinamis yang dinilai lebih efisien dibandingkan dengan *list* berbasis *array* yang kebutuhan memorinya bersifat statis.

Linked-list merupakan pengembangan atau modifikasi dari struktur data *list* dan terbentuk dari serangkaian objek-objek, yang disebut dengan *node*. Kebutuhan memori pada *linked-list* bersifat dinamis dimana kebutuhan memori akan bertambah ketika terdapat elemen *list* yang baru [8]. Contoh struktur data pada *linked-list* dapat dilihat pada Gambar 3.



Gambar 3. Contoh Strukur Data *Linked-List*

III. METODE PENELITIAN

Metode penelitian yang dilakukan pada tugas akhir ini adalah sebagai berikut:

1. Studi Literatur

   Melakukan identifikasi permasalahan dengan mencari referensi guna menunjang penelitian serta mempelajari konsep kerja LSA, model Bayesian, dan struktur data *linked-list.*

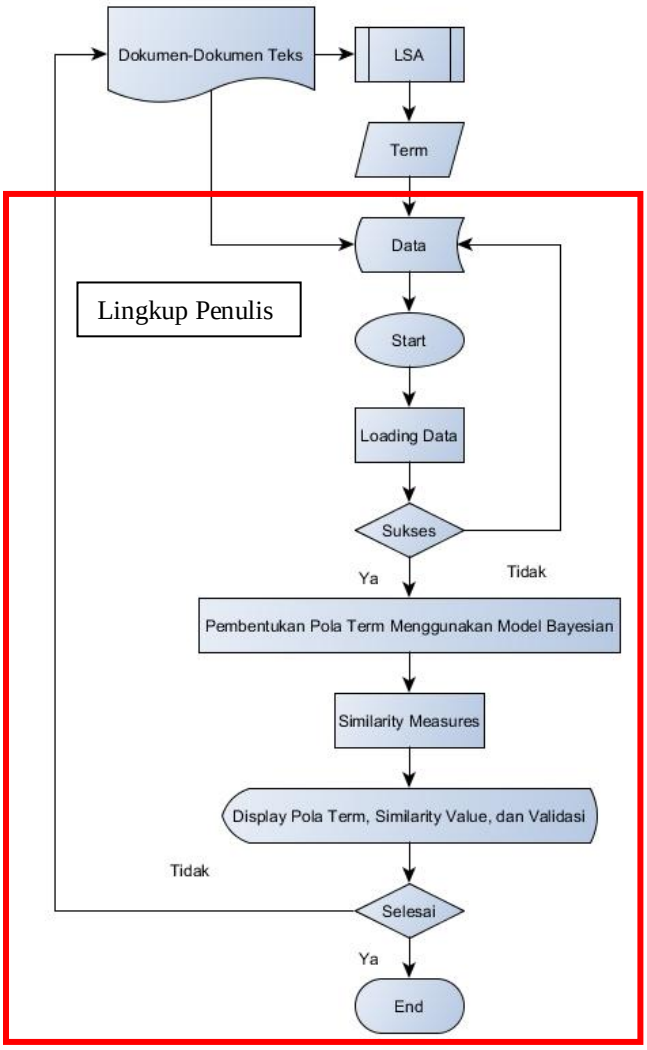
2. Pembuatan Data Uji

   Data yang dibutuhkan dan akan digunakan ada 2 jenis, yaitu dokumen-dokumen teks yang diolah LSA serta *term* hasil pengolahan LSA terhadap dokumen-dokumen tersebut. Data-data yang digunakan pada Tugas Akhir ini

dibuat dan disebut sebagai data uji. Data uji ini digunakan sebagai *input* atau bahan pengujian pada sistem yang dibuat untuk penentuan pola *term* serta penilaian kemiripan antar dokumen.

3. Desain dan Analisis Algoritma

Pembentukan pola *term* dengan menggunakan model Bayesianpada dokumen-dokumen uji membutuhkan struktur data yang mampu mengadaptasi konsep model Bayesian dimana kemunculan *term* ke-n harus didahului *term* ke-(n-1), *term* ke-(n-2), dan seterusnya untuk n>1. Sehingga desain sistem dan analisis algoritma pada aplikasi harus memenuhi sifat model Bayesian tersebut yaitu salah satunya dengan menerapkan struktur data *linked-list* sebagai penyimpanan dan pembentukan pola *term*.Bentuk *interface* dari sistem ini adalah sebuah aplikasi. Algoritma aplikasi dapat dilihat Gambar 4.



Gambar 4. Diagram Alur Aplikasi (dalam region garis merah)

4. Implementasi Desain dan Algoritma

Implementasi ini menggunakan bahasa pemrograman Java dengan bantuan *tool* NetBeans 8.0 serta beberapa *library* yang dibutuhkan.

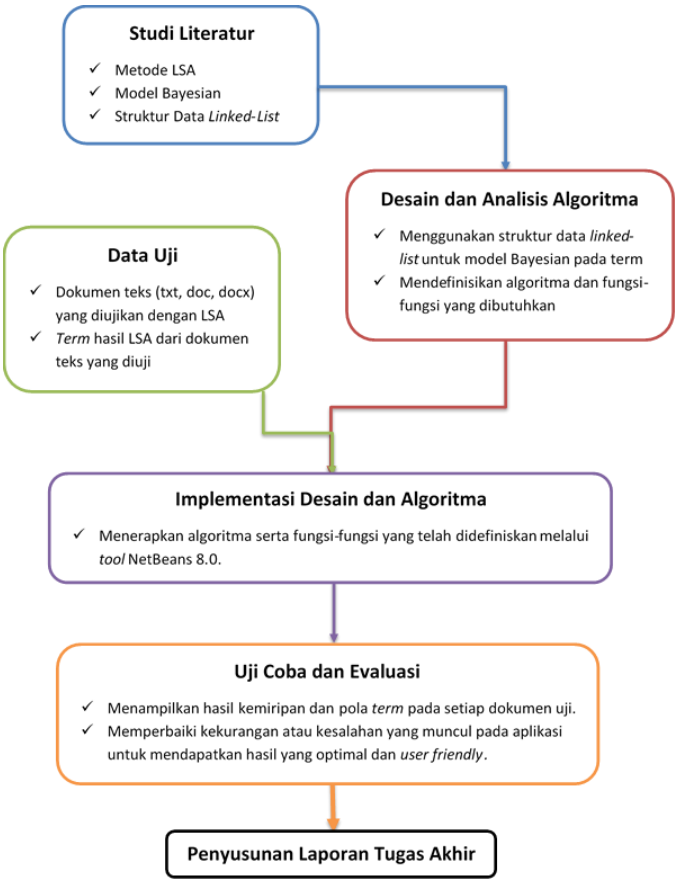
5. Uji Coba dan Evaluasi

Uji coba dilakukan pada aplikasi yang telah dibuat serta melihat hasil yang didapatkan yaitu didapatkan pola *term* serta hasil penilaian kemiripan antar dokumen uji.

Jika terdapat kekurangan atau kesalahan, maka dilakukan *maintenance* pada aplikasi untuk mendapatkan hasil yang sesuai target dan menghasilkan aplikasi yang *user friendly*.

6. Penarikan Kesimpulan dan Penyusunan Laporan

Alur metode penelitian dapat dilihat pada Gambar 5.



Gambar 5. Diagram Alur Metode Penelitian

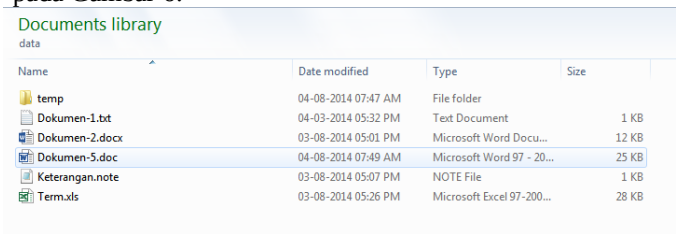
IV. UJI COBA DAN PEMBAHASAN

Prosedur kerja aplikasi adalah sebagai berikut:

1. *Loading* Data

Data yang dibutuhkan ada 2 jenis, yaitu data yang meliputi dokumen-dokumen teks yang diujikan di LSA (berekstensi .txt, .doc, .docx) dan data *term* yang dihasilkan oleh LSA (berekstensi .xls) dimana data-data ini ditempatkan di *folder* "data" yang akan diakses oleh aplikasi saat *running*.

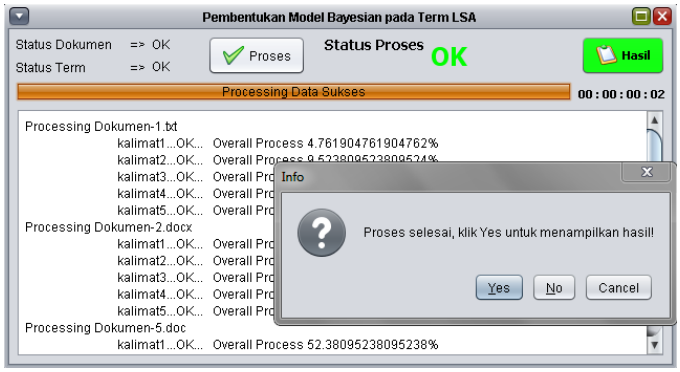
Gambaran data-data yang ada di *folder* "data" dapat dilihat pada Gambar 6.



Gambar 6. *File* Data pada *Folder* "Data"

2. Pembentukan Pola *Term*

Scanning term dilakukan pada masing-masing dokumen teks yang diuji untuk pembentukan pola (urutan) *term* yang ada pada setiap kalimat pada dokumen-dokumen tersebut. Proses pembentukan pola *term* dapat dilihat pada Gambar 7.



Gambar 7. Proses Pembentukan Pola Term Menggunakan Model Bayesian

Proses pembentukan pola *term* menggunakan model Bayesian tersebut menghasilkan pola *term* berbentuk urutan (sekuensial) kemunculan*term* yang terjadi di setiap kalimat pada setiap dokumen teks yang diuji. Pola*term* yang terbentuk adalah sebagai berikut:

Dokumen-1.txt kalimat-1 = T19 T2 T1
Dokumen-1.txt kalimat-2 = T9 T7 T1 T2 T1 T3
Dokumen-1.txt kalimat-3 = T1 T4 T18 T9 T8 T8 T2
Dokumen-1.txt kalimat-4 = -
Dokumen-1.txt kalimat-5 = T1 T5 T7 T2 T18 T2 T5 T4 T3

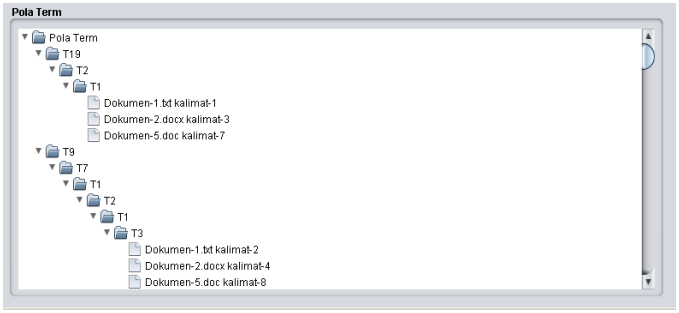
Dokumen-2.docx kalimat-1 = T1 T4 T18 T9 T8 T8 T2
Dokumen-2.docx kalimat-2 = T1 T5 T7 T2 T18 T2 T5 T4 T3
Dokumen-2.docx kalimat-3 = T19 T2 T1
Dokumen-2.docx kalimat-4 = T9 T7 T1 T2 T1 T3
Dokumen-2.docx kalimat-5 = -

Dokumen-5.doc kalimat-1 = T10 T17 T12 T17 T16 T15 T11 T10
Dokumen-5.doc kalimat-2 = T12 T10 T15
Dokumen-5.doc kalimat-3 = T10 T19 T10
Dokumen-5.doc kalimat-4 = T10 T11 T11
Dokumen-5.doc kalimat-5 = T11 T10 T16 T14 T13
Dokumen-5.doc kalimat-6 = T12 T14 T13 T12 T13
Dokumen-5.doc kalimat-7 = T19 T2 T1
Dokumen-5.doc kalimat-8 = T9 T7 T1 T2 T1 T3
Dokumen-5.doc kalimat-9 = T1 T4 T18 T9 T8 T8 T2
Dokumen-5.doc kalimat-10 = -
Dokumen-5.doc kalimat-11 = T1 T5 T7 T2 T18 T2 T5 T4 T3

Dimana T1, T2,…, TX adalah kode *term* dan angka yang ada di belakang huruf "T" adalah indeks *term* pada data "Term.xls".

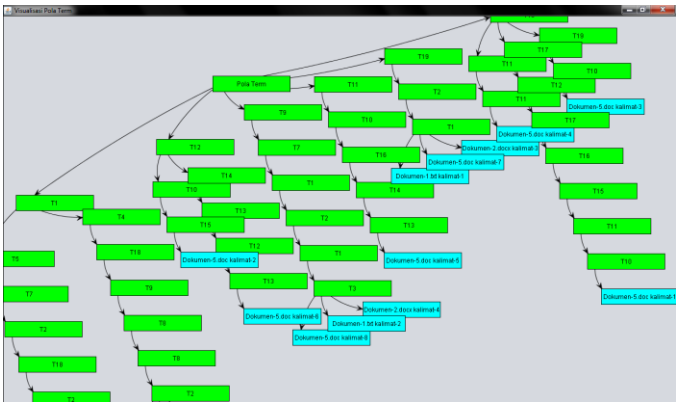
3. *Display* Pola *Term*

Pola *term* yang telah terbentuk dapat ditampilkan secara sekuensial seperti pada Gambar 8.



Gambar 8. *Display* Pola *Term* secara Sekuensial

Pola *term* juga dapat ditampilkan secara visual dengan *layout* hierarki. Pada *endpoint-endpoint*, dapat dilakukan validasi untuk memeriksa pola *term* yang terbentuk sekaligus melihatterjadinya "saling ambil" kalimat jika terdapat pola *term* yang sama namun terdapat pada tempat (dokumen) yang berbeda. Visualisasi pola *term* dapat dilihat pada Gambar 9.



Gambar 9. *Display* Visualisasi Pola *Term*

4. Penilaian Persentase Kemiripan antar Dokumen

Data pola *term* yang terbentuk pada masing-masing dokumen disimpan yang kemudian dijadikan acuan untuk menghitung presentase kemiripan antar dokumen uji.

Untuk menghitung kemiripan antar dokumen (dalam presentase) digunakan algoritma berikut yang diadopsi dari rumus *cosine similarity* [9].

$$similarity(doc_i, doc_j)_{\ dimana\ i \neq j} = \frac{jml\_tb(doc_i, doc_j)}{tot\_term(doc_i, doc_j)} x100\%$$

Keterangan:

$similarity(doc_i, doc_j)$ = nilai kemiripan antara dokumen-i dan dokumen-j dimana i≠j.

$jml\_tb(doc_i, doc_j)$ = jumlah *term* yang sama secara berurutan pada kedua dokumen-i dan dokumen-j.

$tot\_term(doc_i, doc_j)$ = total *term* pada dokumen-i dan dokumen-j.

Contoh penghitungan persentase kemiripan:

Total *term* yang terkandung pada dokumen-dokumen uji adalah sebagai berikut:

Nama Dokumen	Total Term
Dokumen-1.txt	25
Dokumen-2.docx	25
Dokumen-5.doc	52

Dengan pola *term* yang terbentuk pada Dokumen-1.txt dan Dokumen-5.doc sebagai berikut:

Dokumen-1.txt kalimat-1 = T19 T2 T1
Dokumen-1.txt kalimat-2 = T9 T7 T1 T2 T1 T3
Dokumen-1.txt kalimat-3 = T1 T4 T18 T9 T8 T8 T2
Dokumen-1.txt kalimat-4 = -
Dokumen-1.txt kalimat-5 = T1 T5 T7 T2 T18 T2 T5 T4 T3

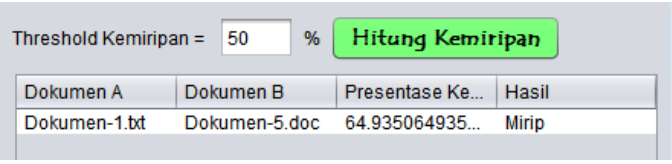
Dokumen-5.doc kalimat-1 = T10 T17 T12 T17 T16 T15 T11 T10
Dokumen-5.doc kalimat-2 = T12 T10 T15
Dokumen-5.doc kalimat-3 = T10 T19 T10
Dokumen-5.doc kalimat-4 = T10 T11 T11
Dokumen-5.doc kalimat-5 = T11 T10 T16 T14 T13
Dokumen-5.doc kalimat-6 = T12 T14 T13 T12 T13
Dokumen-5.doc kalimat-7 = T19 T2 T1
Dokumen-5.doc kalimat-8 = T9 T7 T1 T2 T1 T3
Dokumen-5.doc kalimat-9 = T1 T4 T18 T9 T8 T8 T2
Dokumen-5.doc kalimat-10 = -
Dokumen-5.doc kalimat-11 = T1 T5 T7 T2 T18T2 T5 T4 T3

---

didapat,

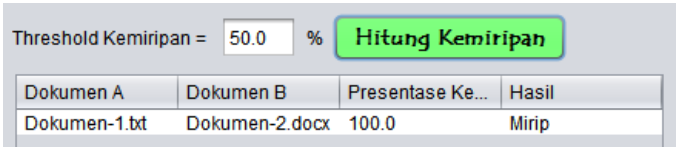
$$similarity(doc_1, doc_5) = \frac{(3+6+7+9)+(3+6+7+9)}{25+52} x100\ \% = \frac{50}{77} x100\ \% = \mathbf{64.935}\ \%$$

Nilai *threshold* dibutuhkan dan digunakan dalam penentuan hasil kemiripan yaitu dalam bentuk predikat "mirip" atau "tidak mirip", jika persentase kemiripan kurang dari nilai *threshold*, maka predikat kemiripan adalah "tidak mirip" dan jika nilai kemiripan lebih dari sama dengan nilai *threshold* maka predikat kemiripan adalah "mirip". Pada Gambar 10, diperlihatkan hasil kemiripan antara Dokumen-1.txt dan Dokumen-5.doc dengan nilai *threshold* 50%.



Gambar 10. Display Hasil Kemiripan (dalam persentase dan *threshold* 50%)

Dengan cara yang sama, maka dapat dihitung bahwa nilai kemiripan Dokumen-1.txt dengan Dokumen-2.docx adalah 100% karena dapat dilihat bahwa pola *term* yang ada pada Dokumen-1.txt sama dengan Dokumen-2.docx, hanya saja tempat pola *term*-nya ada di kalimat yang berbeda (dapat dilihat pada Gambar 11).



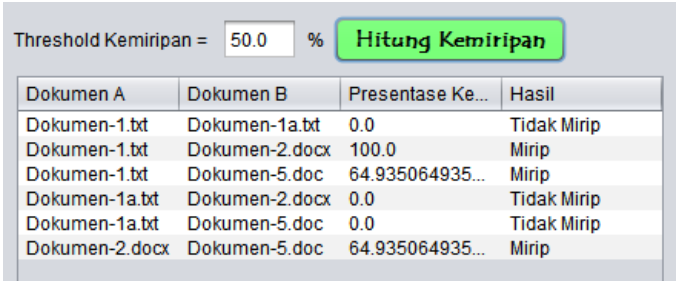
Gambar 11. Hasil Kemiripan antaraDokumen-1.txt dan Dokumen-2.docx (*threshold* 50%)

Selanjutnya dapat dilakukan manipulasi data untuk mendapatkan hasil deteksi kemiripan yang berbeda dari aplikasi ini, misal isi dari berkas Dokumen-1.txt diacak sehingga dokumen tersebut tidak mempunyai arti, namun komposisi *term* yang ada pada dokumen tersebut tidak berubah yang selanjutnya dinamakan Dokumen-1a.txt. Pola*term* pada dokumen-1a.txt setelah diacak adalah sebagai berikut

---

Dokumen-1a.txt kalimat-1 = T1 T2 T5 T19
Dokumen-1a.txt kalimat-2 = T7 T1 T3 T2 T1
Dokumen-1a.txt kalimat-3 = T1 T9 T4 T18 T8 T9 T8 T2
Dokumen-1a.txt kalimat-4 = T2
Dokumen-1a.txt kalimat-5 = T1 T7 T3 T18 T2 T5 T4

Dengan data pola *term* tersebut, jika penilaian kemiripan dijalankan kembali pada seluruh data uji, maka akan didapatkan hasil pada Gambar 12 berikut.



Gambar 12. Hasil Kemiripan setelah Data dokumen-1.txt Diubah (*threshold* 50%)

Akan tetapi, jika data Dokumen-1.txt setelah diacak yaitu Dokumen-1a.txt dibandingkan kembali dengan Dokumen-2.docx di LSA dan dilakukan penilaian kemiripan berdasarkan frekuensi *term*, maka dapat dipastikan akan didapat nilai kemiripan 100% karena komposisi *term* yang ada pada dokumen-dokumen tersebut sama jumlahnya, disamping kandungan arti pada dokumen-dokumen tersebut yang jelas berbeda.Hal inilah yang membedakan hasil dari penelitian Tugas Akhir ini terhadap deteksi kemiripan antar dokumen yang hanya dilakukan pada LSA saja.

V. KESIMPULAN

Deteksi kemiripan antar dokumen teks pada LSA (*Latent Semantic Analysis*) hanya mengacu pada frekuensi kata (*term*) yang ada di dokumen dan tidak memperhatikan urutan tata letak kata sehingga struktur kalimat pada dokumen diabaikan, dan hal ini berpengaruh pada makna pada setiap dokumen yang diujikan. Oleh karena itu, dilakukan kombinasi metode LSA dengan model Bayesian yang mana model Bayesian

berperan dalam menjaga urutan *term* yang secara tidak langsung berarti menjaga struktur kalimat yang ada pada dokumen tersebut. Sehingga hasil deteksi kemiripan yang dihasilkan bisa lebih baik karena deteksi kemiripan yang dilakukan tidak hanya mengacu pada frekuensi *term* tetapi juga menjaga makna yang terkandung pada dokumen yang dibandingkan.

DAFTAR PUSTAKA

[1] Kamus Besar Bahasa Indonesia Daring (Dalam Jaringan). 2008. http://bahasa.kemdiknas.go.id/kbbi/index.php. Diakses tanggal 17 Juli 2014.

[2] Cosma, Georgina & Mike Joy. 2012. *Evaluating the Performance of LSA for Source-code Plagiarism Detection*. Journal of Informatica, Vol. 36, Hal. 409-424.

[3] Mozgovoy, Maxim, Tuomo Kakkonen & Georgina Cosma. 2010. *Automatic Student Plagiarism Detection: Future Perspectives*. Journal of Educational Computing Research, Vol. 43, Hal. 511-531.

[4] Cosma, Georgina. 2008. *An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis*. Thesis for Doctor of Philosophy in Computer Science, University of Warwick.

[5] Landauer, Thomas K., Peter W. Foltz & Darrell Laham. 1998. *"An Introduction to Latent Semantic Analysis"*. Department of Psychology, University of Colorado.

[6] Griffiths, Thomas L., Charles Kemp & Joshua B. Tenenbaum. 2004. *"Bayesian Models of Cognition"*. Journal of Annual Meeting of Cognitive Science Society.

[7] Murphy, Kevin. 1998. *A Brief Introduction to Graphical Models and Bayesian Networks*. http://www.cs.ubc.ca/~murphyk/Bayes/bnintro.html. Diakses tanggal 18 Juli 2014.

[8] Shaffer, Clifford A. 2012. "*Data Structures and Algorithm Analysis*". Blackburg: Virginia Tech.

[9] Huang, Anna. 2009. *Similarity Measures for Text Document Clustering*. Department of Computer Science, The University of Waikato.