APLIKASI PENGGALI POLA STATISTIK NON-LINEAR PADA DOKUMEN TEKS UNTUK JARINGAN SYARAF TIRUAN SEMINAR NASIONAL TEKNOIN 2008

Purnomo Husnul Khotimah[1)]
Pusat Penelitian Informatika LIPI [1)]
Jl Sangkuriang No 21/154D Cisitu Bandung 40135
Telepon (022) 2504711
E-mail : hkhotimah@informatika.lipi.go.id[1)]

Abstrak

Perkembangan teknologi digital yang cukup pesat ini, salah satunya ditandai dengan usaha-usaha untuk membawa intelegensia ke dalam dunia digital. Jaringan syaraf tiruan (JST) adalah salah satu usaha tersebut yang berusaha meniru cara kerja syaraf dalam meneruskan data yang kemudian diproses untuk menghasilkan sebuah informasi dengan memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data. Dalam mengembangkan model JST, terkadang pola-pola statistik yang diperlukan cukup bervariasi untuk memecahkan antara permasalahan satu dengan yang lainnya dan masukan yang dibutuhkan model JST biasanya berupa data numerik dalam bentuk matriks. Akan tetapi aplikasi word count biasanya hanya terbatas pada menghitung jumlah kata, kalimat, jumlah baris, dan jumlah karakter.

Pada penulisan ini akan dijelaskan mengenai pengembangan sebuah aplikasi penggali pola-pola statistik non-linear pada dokumen teks yang diperlukan untuk penelitian ataupun pengembangan model JST. Aplikasi dikembangkan menggunakan metode pemrograman berorientasi objek dan diimplementasikan menggunakan Java (J2SE). Oleh karena itu, mesin yang menjadi host aplikasi ini memiliki dependensi, yaitu pada mesin harus terdapat java virtual machine (java run time environment). Saat ini aplikasi yang telah dikembangkan dapat digunakan pada dokumen teks yang terdiri dari karakter alphabet pada 6 bahasa, yaitu Indonesia, Malaysia, Inggris, Jerman, Italia dan Portugis. Pola-pola statistik non-linear yang bisa diperoleh dari aplikasi ini adalah rasio vokal dan konsonan dalam satu kalimat, distribusi vokal dan konsonan dalam satu kalimat, persentasi rasio vokal dan konsonan dalam satu kalimat, rasio vokal dan konsonan spesial, rasio vokal, rasio konsonan, persentase rasio vokal, persentase rasio konsonan, distribusi vokal dan konsonan spesial dalam satu kalimat.

Kata Kunci : aplikasi penggali, pola statistik non-linear, dokumen teks, J2SE, jaringan syaraf tiruan

PENDAHULUAN

Perkembangan teknologi digital yang cukup pesat ini, salah satunya ditandai dengan usaha-usaha untuk membawa intelegensia ke dalam dunia digital. Dalam usaha-usaha tersebut, muncul bidang keilmuan kognitif, yaitu suatu bidang ilmu yang mempelajari bagaimana terjadinya intelegensia. Misalnya manusia dengan kesadarannya membuat suatu keputusan. Dari sini diketahui bahwa untuk membawa intelegensia ke dalam dunia digital tidak bisa dengan mudah diselesaikan dengan algoritma biasa. Salah satu metode yang banyak digunakan untuk menyelesaikan permasalah non algoritmik ini adalah penggunaan model jaringan syaraf tiruan (JST).

JST merupakan sebuah usaha untuk meniru proses kerja sel syaraf (neuron) manusia dalam meneruskan data yang kemudian diproses untuk menghasilkan sebuah informasi. Secara sederhana JST dapat dijelaskan sebagai alat pemodelan data statistik non-linear yang dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data. Metode statistik yang sering dipakai untuk keperluan JST adalah statistik deskriptif, yaitu sebuah metode statistik yang menggunakan metode numerik dan grafis untuk mencari pola dalam sebuah data set, untuk meringkas informasi yang ditemukan dalam sebuah data set, dan untuk menampilkan informasi tersebut sedemikian rupa

sehingga dapat digunakan oleh pengguna (Bekker,2008).

JST terdiri dari elemen-elemen pemrosesan yang keluarannya dikalkulasi dengan mengkalikan inputnya dengan sebuah vektor weight, menjumlahkan hasilnya, dan menerapkan sebuah fungsi aktivasi pada jumlah. Kemudian untuk mengembangkan sebuah model jaringan syaraf tiruan biasanya parameter input yang digunakan berupa data-data numerik dalam bentuk matriks. Aplikasi word count biasanya hanya terbatas pada menghitung jumlah kata, kalimat, jumlah baris, dan jumlah karakter. Pada penelitian pengembangan model JST terkadang pola-pola statistik yang diperlukan cukup bervariasi untuk memecahkan antara permasalahan satu dengan yang lainnya. Selain itu bentuk input yang dibutuhkan model JST biasanya berupa data numerik dalam bentuk matriks.

Pada penulisan ini akan dijelaskan mengenai pengembangan sebuah aplikasi penggali pola-pola statistik non-linear pada dokumen teks yang diperlukan untuk penelitian ataupun pengembangan model JST.

METODOLOGI PENELITIAN

Upaya pengembangan aplikasi dilakukan dengan mengacu pada model Waterfall. Tahapan pengembangan dimulai dengan inisialisasi, analisa, desain, implementasi, pengujian, dan dokumentasi (Bennett, McRobb, dan Farmer, R. 2003).. Tahapan-tahapan tersebut ada yang bersifat iteratif, yaitu desain, implementasi dan pengujian.

Pada tahap inisialisasi diperoleh hasil berupa ide pembuatan aplikasi penggali pola-pola statistik non-linear. Kemudian pada tahap analisa dihasilkan beberapa poin. Poin pertama yaitu bahwa pemrograman akan dilakukan dengan menggunakan menggunakan metode pemrograman berorientasi objek. Dengan demikian diharapkan aplikasi dapat mudah dikembangkan kedepannya, misalnya jika diperlukan kustomisasi lebih lanjut agar sesuai dengan keperluan penelitian lainnya

Poin ke dua hasil tahap analisa adalah perancangan akan dikembangkan dengan menggunakan pemodelan *Unified Modelling Language* (UML). UML menetapkan sembilan diagram untuk memodelkan sebuah sistem, yaitu *usecase diagram, class diagram, object* diagram, *statechart diagram, sequence diagram, collaboration diagram, activity diagram, component diagram, dan deployment diagram* (Saleh, 2001). Dari kesembilan diagram tersebut, yang akan digunakan dalam pengembangan adalah *use case, statechart*, dan *class diagram*. *Usecase diagram* dan *statechart diagram* menunjukkan perilaku sistem, sedangkan class diagram akan menunjukkan truktur sistem. Pada tahap analisa juga dihasilkan poin yang menyebutkan bahwa pengembangan akan dilakukan dengan bahasa pemrograman Java (J2SE). Hal ini menyebabkan implementasinya memiliki dependensi pada mesin yang menjadi *host* aplikasi ini, yaitu pada mesin harus terdapat *java virtual machine*/*java run time environment* (Deitel 2003).

Tahapan desain, implementasi dan pengujian dilakukan secara berulang, yaitu dari desain awal yang sederhana yang kemudian diimplementasikan dan diuji begitu seterusnya sehingga desain dan aplikasi menjadi lebih berkembang dan lengkap. Dengan demikian pengerjaan program lebih mudah dan cepat untuk dilakukan tanpa harus menunggu desain lengkap aplikasi selesai. Hal ini mempertimbangkan juga aplikasi yang akan dikembangkan berukuran kecil. Selain itu untuk mempercepat pemrograman, dilakukan daur ulang terhadap *class-class* yang bersifat *open source* dengan tetap memperhatikan syarat-syarat penggunaannya.

Pengujian dilakukan dengan melakukan *functional specification*, yaitu pengujian dengan memberikan kondisi-kondisi dan dilihat apakah keluaran yang diperoleh sesuai dengan spesifikasi fungsinya (Chillarege, R. 1999). Selain pengujian yang dilakukan secara iteratif seperti yang telah disebutkan sebelumnya, juga dilakukan pengujian akhir pada aplikasi yang telah selesai diprogram. Tahapan terakhir, yaitu dokumentasi dilakukan dengan mendokumentasikan rancangan dan source code aplikasi.

HASIL DAN PERANCANGAN

Aplikasi dirancang untuk menawarkan kemudahan bagi pengguna untuk menentukan pola statistik apa saja yang akan digunakan, ukuran keluaran matriks, dan menyimpan hasil dalam sebuah dokumen bertipe teks sehingga memudahkan untuk digunakan oleh aplikasi lain yang akan memodelkan jaringan syaraf tiruan. Ketiga kasus penggunaan aplikasi tersebut dapat dilihat pada gambar 1, yaitu *usecase diagram* aplikasi.

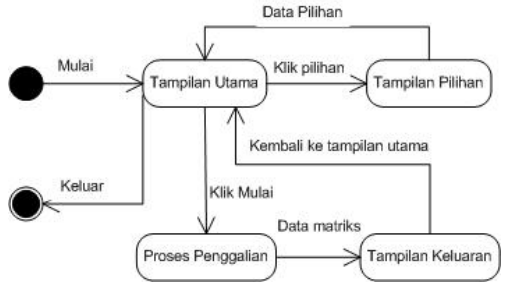
Gambar 1. *Usecase diagram* sistem

Skenario penggunaan aplikasi lebih lanjut dapat dijelaskan seperti di bawah ini :

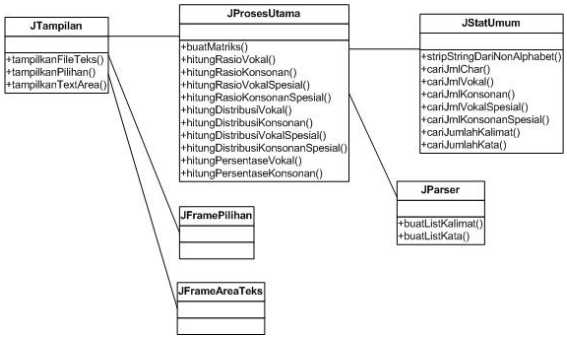
1. pengguna membuka aplikasi
2. pengguna mengklik tombol Buka File
3. aplikasi akan menampilkan File Dialog sehingga pengguna dapat memilih dokumen teks yang akan digunakan
4. pengguna mengklik tombol Pilihan
5. aplikasi akan menampilkan *window frame* Pilihan yang terdapat daftar pilihan pola statistik

6. pengguna memilih pola-pola statistik yang akan digali pada dokumen teks
7. pengguna mengklik tombol Mulai
8. aplikasi akan memulai proses penggalian data
9. aplikasi akan menampilkan keluaran dalam *window frame* area teks
10. pengguna mengklik tombol save untuk menyimpa keluaran pada *window frame* area teks

Selain dalam bentuk *usecase diagram* dan skenario penggunaan , perancangan juga dilakukan dengan membuat *statechart diagram* dan *class diagram*, seperti yang dapat dilihat pada Gambar 3 dan Gambar 4.



Gambar 3. *Statechart diagram* sistem



Gambar 3. *Class diagram* sistem

Agak berbeda dengan *usecase diagram* yang memperlihatkan perilaku pengguna aplikasi, *statechart diagram* pada Gambar 3 memperlihatkan perilaku aplikasi. Disini dapat dilihat bahwa aplikasi memiliki 4 kondisi utama, yaitu :

1. Tampilan utama : tampilan utama akan memperlihatkan menu-menu yang bisa dipilih oleh pengguna dan dokumen teks yang telah dipilih pengguna.
2. Tampilan Pilihan : tampilan pilihan akan memperlihatkan daftar pilihan pola-pola statistik yang dapat digali oleh aplikasi.
3. Proses Penggalian : proses penggalian merupakan proses utama aplikasi, yaitu proses pencarian dan penghitungan pola-pola statistik yang telah dipilih pengguna.
4. Tampilan Keluaran : tampilan keluaran akan memperlihatkan hasil dari proses utama berupa matriks yang ditampilkan dalam sebuah *pop up* window.

Class diagram pada Gambar 3, memperlihatkan struktur sistem. Pada dasarnya class sistem terdiri dari 2 *class* utama, yaitu *class* komputasi dan *class* tampilan. Hal ini dilakukan untuk mempermudah pemrograman aplikasi dan agar pengembangan lebih lanjut dapat dengan mudah dilakukan. Dengan adanya pembagian seperti ini, modifikasi pada penampilan ataupun penambahan fungsi komputasi dapat dilakukan dengan mudah dibandingkan jika metode untuk penampilan data dan komputasinya dicampur dalam satu *class*. *Class* komputasi, yaitu *class* yang mempunyai fungsi utama untuk melakukan perhitungan, terdiri dari 3 *class*, yaitu :

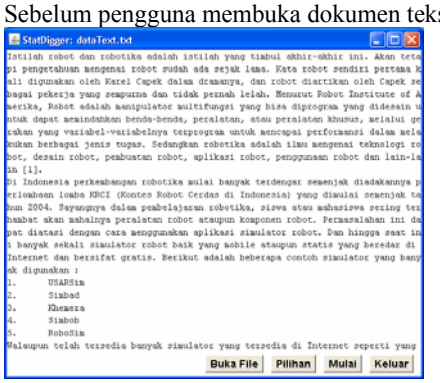
1. JProsesUtama
   JProsesUtama merupakan class komputasi utama penghitungan elemen matriks. Pada *class* JProsesUtama terdapat metode-metode penghitungan pola-pola statistik yang terdapat pada menu pilihan.
2. JStatUmum
   JstatUmum merupakan *class* komputasi pola-pola statistik umum, yaitu jumlah karakter, jumlah vokal, jumlah konsonan, jumlah vokal spesial, jumlah konsonan spesial, jumlah kalimat dan jumlah kata.
3. JParser
   JParser merupakan *class* yang berfungsi untuk melakukan proses pemilahan kalimat dan kata.

Class tampilan yang mempunyai fungsi utama untuk melakukan interaksi dengan pengguna, juga terdiri dari 3 *class*, yaitu :

1. JTampilan
   JTampilan merupakan *class* tampilan utama yang berfungsi selain untuk menampilkan tampilan utama juga menjadi antar muka bagi class tampilan lainnya untuk mengakses hasil perhitungan dan bagi *class* komputasi untuk mengakses hasil interaksi pengguna dengan aplikasi.
2. JFramePilihan
   JFramePilihan merupakan *class* untuk mengambil daftar pola statistik pilihan pengguna. Metode-metode pada *class* JFramePilihan memungkinkan pengguna untuk memilih dan menghapus pilihan.
3. JFrameAreaTeks
   JFrameAreaTeks merupakan *class* yang berfungsi untuk menampilkan keluaran dan memungkinkan pengguna untuk menyimpan keluaran dalam bentuk dokumen teks.

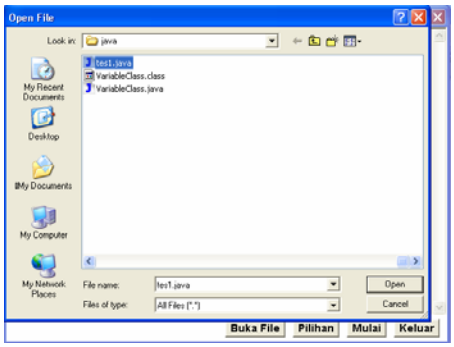
Hasil tampilan aplikasi penggali pola statistik non-linear dapat dilihat pada Gambar 4, Gambar 5, Gambar 6, Gambar 7, Gambar 8. Gambar 4 menunjukkan tampilan utama aplikasi. Tampilan pada Gambar 4a dan Gambar 4b memperlihatkan tampilan utama sebelum dan sesudah pengguna membuka dokumen teks. Dokumen teks dapat dibuka dengan pengguna memilih tombol Buka Dokumen yang akan mengaktifkan *file dialog* Open Dokumen seperti yang terlihat pada Gambar 5.

a. Sebelum pengguna membuka dokumen teks



b. Sesudah pengguna membuka dokumen teks

Gambar 4. Tampilan utama aplikasi

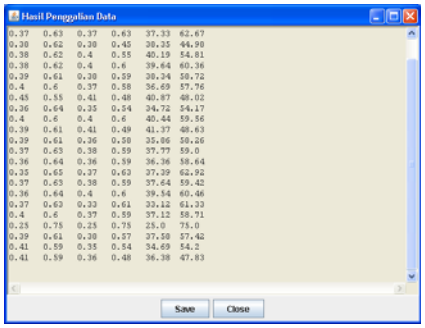


Gambar 5. Tampilan *file dialog* Open File

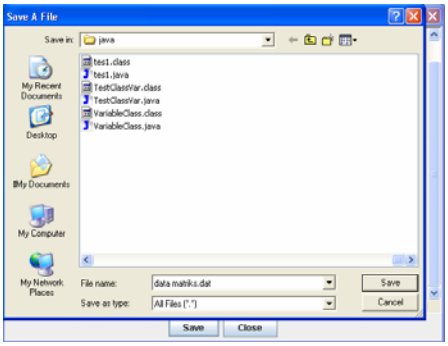
Di dalam pemrogramannya, untuk memudahkan pengaturan komponen-komponen GUI, window frame tampilan utama menggunakan Layout Manager tipe BorderLayout, yaitu pengaturan tata letak komponen dengan membagi frame menjadi 5 area, yaitu : NORTH, SOUTH, EAST, WEST dan CENTER. Area CENTER window frame utama berisi JtextArea, area SOUTH berisi panel tombol dan area yang lainnya tidak digunakan. Pengaturan seperti ini memungkinkan tampilan seperti terbagi menjadi dua bagian yaitu bagian tampilan dokumen teks dan bagian tombol menu. Panel tombol menggunakan tipe Layout Manager yang berbeda, yaitu FlowLayout. Flow layout memungkinkan komponen untuk disusun seperti aliran komponen dari kiri ke kanan. Panel tombol terdiri dari tombol Buka Dokumen, Pilihan, Mulai dan Keluar.
Gambar 6 berikut ini menunjukkan tampilan daftar pilihan pola-pola statistik yang bisa dipilih oleh pengguna. Seperti pada tampilan utama, layout yang digunakan pada bagian dasar adalah BoderLayout. Area CENTER digunakan untuk panel JList, sedangkan area SOUTH diisi dengan panel tombol. Panel Jlist menggunakan layout bertipe FlowLayout dan GridLayout untuk pengaturan tombol “>>” dan tombol “<<”. GridLayout adalah pengaturan tata letak komponen menjadi kolom dan baris. Panel JList terdiri dari dua komponen JList, dan dua tombol untuk memilih dan menghapus pilihan. Komponen JList yang memungkinkan pengguna untuk memilih satu atau lebih obyek dari sebuah daftar obyek. Untuk mempermudah manipulasi JList, di dalam pemrograman tampilan daftar pilihan digunakan DefaultListModel untuk merepresentasikan isi dari daftar pilihan pada saat menyatakan . Kemudian tombol “>>” dan tombol “<<” secara default akan tampil tidak aktif. Hanya ketika pengguna memilih pada daftar, tombol akan aktif seperti terlihat pada Gambar .

Gambar 6. Tampilan daftar pilihan pola-pola statistik

Gambar 7 di bawah ini menunjukkan tampilan matriks keluaran. Frame tampilan matriks keluaran menggunakan tipe layout manager yang sama dengan window frame tampilan utama, yaitu BorderLayout. Ketika tombol Save diklik oleh pengguna, sebuah *file dialog* Save A File akan terlihat seperti pada Gambar 8.



Gambar 7. Tampilan matriks keluaran



Gambar 8. Tampilan *file dialog* Save A File

Proses pengujian akhir dilakukan dengan terlebih dahulu membuat daftar membuat daftar spesifikasi fungsi yang terbagi menjadi 3 bagian, yaitu pengujian fitur, kondisi, dan keluaran. Pengujian fitur dilakukan untuk memastikan bahwa aplikasi memenuhi diagram use case. Pengujian fitur dilakukan dengan memberikan data-data normal (*Non-unique case*). Pengujian selanjutnya yaitu pengujian kondisi merupakan pengujian yang dilakukan untuk mendekteksi kesalahan yang mungkin terjadi pada interaksi aplikasi dengan pengguna (*exception case*). Sedangkan pengujian keluaran adalah pengujian yang dilakukan untuk memastikan kebenaran matriks keluaran dan kevalidan dokumen yang berisi matriks keluaran. Secara lebih rinci, berikut adalah daftar spesifikasi fungsi untuk pengujian akhir aplikasi:

1. Pengujian fitur :
   a. Pengguna dapat membuka dokumen teks.
   b. Pengguna dapat memilih pola-pola statistik non linear.
   c. Pengguna dapat menjalankan proses penggalian pola-pola statistik non linear.
   d. Pengguna dapat menyimpan matriks keluaran dalam bentuk dokumen teks.
2. Pengujian kondisi :
   a. Kondisi dokumen teks yang dipilih tidak ada, yaitu kondisi ketika pengguna langsung menuliskan nama dokumen secara random dalam *file dialog* Open File. Hasil yang diharapkan adalah aplikasi menampilkan pemberitahuan bahwa dokumen yang diminta tidak ada.
   b. Kondisi pengguna belum memilih pola-pola statistik non-linear, yaitu kondisi setelah membuka dokumen teks, pengguna langsung mengklik tombol Mulai. Hasil yang diharapkan adalah aplikasi menampilkan pemberitahuan bahwa pengguna belum melakukan pemilihan pola-pola statistik non-linear.
   c. Kondisi dokumen teks untuk matriks keluaran sudah ada, yaitu kondisi ketika nama dokumen yang dimasukkan oleh pengguna dalam *file dialog* Save A File sama dengan nama dokumen dalam direktori yang sedang dibuka. Hasil yang diharapkan adalah aplikasi menampilkan pemberitahuan bahwa terdapat dokumen dengan nama yang sama.
3. Pengujian keluaran :
   a. Hasil matriks keluaran sama dengan perhitungan manual
   b. Ukuran matriks sesuai dengan jumlah kalimat x jumlah pola statistik non-linear yang dipilih.
   c. Dokumen yang menyimpan matriks keluaran dapat dibuka.

Hasil pengujian akhir yang dilakukan pada aplikasi penggali pola-pola statistik non-linear menunjukkan bahwa aplikasi dapat memenuhi semua daftar pengujian spesifikasi fungsi-fungsi yang telah disebutkan sebelumnya.

KESIMPULAN

Saat ini aplikasi yang telah dikembangkan dapat menggali pola-pola statistik pada dokumen teks yang terdiri dari karakter alphabet pada 6 bahasa, yaitu Indonesia, Malaysia, Inggris, Jerman, Italia dan Portugis. Pola-pola statistik non linear yang dapat diperoleh aplikasi ini adalah rasio vokal dan konsonan dalam satu kalimat, distribusi vokal dan konsonan dalam satu kalimat, persentasi rasio vokal dan konsonan dalam satu kalimat, rasio vokal dan konsonan spesial, rasio vokal, rasio konsonan, persentase rasio vokal, persentase rasio konsonan, distribusi vokal dan konsonan spesial dalam satu kalimat.

Untuk ke depannya aplikasi ini akan dikembangkan supaya pengguna dapat menambahkan karakter-karakter spesial dari bahasa selain 6 bahasa yang telah disebutkan sebelumnya. Selain itu juga akan ditambahkan fitur untuk mendefinisikan pola statistik sendiri berdasarkan pola-pola statistik umum yang disediakan oleh aplikasi. Dengan demikian diharapkan aplikasi penggali pola-pola statistik non-linear pada dokumen teks dapat semakin memenuhi kebutuhan pengembangan model JST.

DAFTAR PUSTAKA

[1] Chillarege, R. (1999). *Software Testing Best Practices*. IBM Research. Di akses 28 Oktober 2008, dari http://www.chillarege.com/authwork/TestingBestPratice.pdf

[2] Saleh, K. (2001). Documenting electronic commerce systems and software using the unified modelling language. Diakses 28 Oktober 2008, dari http://www.sciencedirect.com

[3] Bennett, S., McRobb, S. dan Farmer, R. (2003). *Object Oriented Systems Analysis And Design Using UML* (Edisi 2). McGraw-Hill, Inc.

[4] Deitel (2003). *Java How To Program* (Edisi 5). Prentice Hall

[5] Bekker, L (2008). *Statistics, Data & Statistical Thinking*. Diakses 31 Agustus 2008, dari http://www.fiu.edu/~bekkerl/chap1(ism10).pdf