**BAB I**

**PENDAHULUAN**

**1.1 Latar Belakang**

Dewasa ini, bidang pendidikan mengalami perkembangan yang cukup pesat. Sekolah tinggi seperti STMIK Denpasar dengan kuantitas jumlah mahasiswa yang semakin bertambah telah menghasilkan lulusan di setiap tahunnya. Tentunya buku tugas akhir sebagai syarat kelulusan mahasiswa disimpan untuk digunakan sebagai referensi bagi pihak-pihak yang membutuhkan, misalnya mahasiswa-mahasiswa yang sedang mengerjakan tugas akhir.

Dalam hal ini, perpustakaaan STMIK Denpasar adalah salah satu wadah yang menyediakan fasilitas di mana mahasiswa dapat meminjam tugas akhir. Kumpulan tugas akhir tersebut semakin bertambah kuantitas dan jenisnya setiap tahunnya.

Mahasiswa tentunya akan mengalami kesulitan mencari sekelompok tugas akhir yang sesuai dengan kata penting tertentu. Misalnya kata “basis data” yang terdapat pada tugas akhir kelompok sistem informasi, tetapi tidak menutup kemungkinan kata “basis data” juga terdapat pada tugas akhir di kelompok lainnya.

Untuk membantu memecahkan masalah di atas tentunya dibutuhkan suatu aplikasi. Aplikasi yang dimaksud adalah Text Mining dengan metode *Vector Space Model*. Aplikasi ini akan melakukan *pem-*filter*-an* pada sekumpulan tugas akhir yang disimpan dalam folder situs e-library STMIK Denpasar*,* Dengan aplikasi ini nantinya diperoleh *list* judul tugas akhir dari yang paling berhubungan dengan keyword yang di-*input*-kan user sampai dengan yang paling sedikit berhubungan dengan *keyword* yang di-*input*-kan.

* 1. **RUMUSAN MASALAH**

Berdasarkan latar belakang diatas maka permasalahan yang didapat adalah :

* + 1. Bagaimana membangun sistem pencarian topik tugas akhir dari kumpulan tugas akhir dalam *e-library* STMIK Denpasar dengan memanfaatkan *text mining*?
    2. Seperti apa data mentahnya dan bagaimana proses awal yang dilakukan terhadap data tersebut?
    3. Bagaimana cara menampilkan hasil tersebut sebagai informasi kepada *user*?

Digunakannya text mining disebabkan karena data mentah yang diteliti dan diolah adalah abstrak dari kumpulan tugas akhir STMIK Denpasar. Metode *Vector space model* digunakan untuk menghindari kemungkinan adanya dokumen yang memiliki bobot yang sama saat proses pencarian, sehingga *ambigu* untuk diurutkan.

**1.3 BATASAN MASALAH**

Dari pokok masalah diatas maka dapat ditentukan batasan masalah sebagai berikut :

* + 1. Studi kasus untuk penelitian ini adalah STMIK Denpasar.
    2. Data presprocessing dilakukan pada abstrak tugas akhir STMIK Denpasar.
    3. Abstrak yang dijadikan *target data* merupakan Abtrak berbahasa Indonesia dengan asumsi kata asing sebagai kata penting.
    4. Metode *vector space model* digunakan hanya untuk melakukan perangkingan hasil pencarian.
    5. Penggunaan database MySQL hanya untuk menyimpan informasi dokumen, stoplist sedangkan untuk menyimpan data hasil *preprocessing* mengunakan XML.
    6. Keyword yang di-*input*-kan oleh *user* pada *search engine* hanya diproses jika merupakan kata penting.
    7. Tugas akhir harus berformat text, misalnya PDF, Docx, Doc, odt.

**1.4 TUJUAN**

Tujuan yang ingin dicapai adalah terciptanya sebuah sistem pencarian topik tugas akhir dari alumni mahasiswa di e-library STMIK Denpasar dalam waktu yang cepat dengan memanfaatkan text mining, crontab dan metode *vector space model* berdasarkan asosiasinya dengan *keyword* yang di-*input*-kan oleh *user*.

* 1. **HIPOTESA**

Dalam penelitian ini dikembangkan sebuah e-library atau perpustakaan *online* yang nantinya akan melakukan pengurutan data (tugas akhir) berdasarkan dari hasil bobot masing-masing dokumen dari *keyword* dengan proses text mining dan algoritma *vector space model*, dimana proses pembobotan dokumen dilakukan setelah dilakukannya proses text mining dengan bantuan crontab sehingga proses pencarian menjadi lebih cepat.

Text mining digunakan untuk mendapatkan informasi dari sekumpulan dokumen yang tidak terstruktur dan crontab digunakan untuk melakukan penjadwalan proses text mining (*Tokenizing, Stoplist, Stemming dan pembobotan*) pada dokumen melalui server (tidak melalui *browser*), hal ini diharapkan dapat memperingan kinerja server.

.

**BAB II**

**TINJAUAN PUSTAKA**

* 1. **Text mining**

Text mining adalah salah satu bidang khusus dari *data mining*. Sesuai dengan buku *The Text Mining Handbook* yang ditulis oleh *Ronen Feldman* dari *Hebrew University of Jerusalem*, text mining dapat didefinisikan sebagai suatu proses menggali informasi dimana seorang user berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis yang merupakan komponen-komponen dalam *data mining* yang salah satunya adalah kategorisasi. Tujuan dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Jadi, sumber data yang digunakan pada *text mining* adalah kumpulan teks yang memiliki format yang tidak terstruktur atau minimal semi terstruktur. Adapun tugas khusus dari *text mining* antara lain yaitu pengkategorisasian teks *(text categorization)* dan pengelompokan teks *(text clustering)*.

*Text mining* bisa dianggap subjek riset yang tergolong baru. Text mining dapat memberikan solusi dari permasalahan seperti pemrosesan, pengorganisasian atau pengelompokkan dan menganalisa unstructured text dalam jumlah besar.

Dalam memberikan solusi, text mining mengadopsi dan mengembangkan banyak teknik dari bidang lain, seperti *Data mining, Information Retrieval, Statistik dan Matematik, Machine Learning, Linguistic, Natural Languange Processing, dan Visualization*. Kegiatan riset untuk text mining antara lain ekstraksi dan penyimpanan *text*, *preprocessing* akan konten *text*, pengumpulan data statistik dan *indexing* dan analisa konten.

Permasalahan yang dihadapi pada text mining sama dengan permasalahan yang terdapat pada data mining, yaitu jumlah data yang besar, dimensi yang tinggi, data dan struktur yang terus berubah, dan data *noise.* Perbedaan di antara keduanya adalah pada data yang digunakan. Pada data mining, data yang digunakan adalah *structured data*, sedangkan pada text mining, data yang digunakan text mining pada umumnya adalah *unstructured data*, atau minimal *semistructured*. Hal ini menyebabkan adanya tantangan tambahan pada text mining yaitu struktur text yang complex dan tidak lengkap, arti yang tidak jelas dan tidak standard, dan bahasa yang berbeda ditambah translasi yang tidak akurat.

* 1. **Metode Vector Space Model**

Vector space model adalah suatu model yang digunakan untuk mengukur kemiripan antara suatu dokumen dengan suatu query. Pada model ini, query dan dokumen dianggap sebagai vektor-vektor pada ruang n-dimensi, dimana n adalah jumlah dari seluruh term yang ada dalam leksikon. Leksikon adalah daftar semua term yang ada dalam indeks. Salah satu cara untuk mengatasi hal tersebut dalam *Model Vector Space* adalah dengan cara melakukan perluasan vektor. Proses perluasan dapat dilakukan pada vektor query, vektor dokumen, atau pada kedua vektor tersebut.

Dalam vektor space model, dokumen dan query direpresentasikan sebagai vektor dalam ruang vektor yang disusun dalam indeks *term*, kemudian dimodelkan dengan persamaan geometri. Ada beberapa yang perlu diperhatikan pada model ruang vektor yaitu:

1. Menggunakan bobot index *term*
2. Adanya vektor dokumen dari kueri
3. Perhitungan *Cosine* menentukan kesamaan dokumen *query*

Untuk mendapatkan nilai W dari tiap dokumen dan kata kunci digunakan algoritma *TF/IDF* (*Term Frequency – Inversed Document Frequency*) . *TF (Term Frequency)* merupakan frekuensi untuk menghitung bobot dari kata ke- *I* pada dokumen (*Steinbach, Micheal, Karypis, George dan Kumar, Vipin. 2000 : 4*) dan *IDF* (*Inversed Document Frequency*) untuk menghitung bobot kata pada dokumen sehingga bentuk representasi dokumen dengan vektor seperti berikut ini:

*dt dt t W* *tf* \* *IDF*

Keretangan:

d = dokumen ke-d

t = kata ke-t dari kata kunci

W = bobot dokumen ke-d terhadap kata ke-t

tf = banyaknya kata yang dicari pada sebuah dokumen

IDF = *Inversed Document Frequency*

D = total dokumen

df = banyak dokumen yang mengandung kata yang dicari

Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses *sorting*/pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya.

Apabila hasil pengurutan bobot dokumen tidak dapat mengurutkan secara tepat, karena nilai W keduanya sama, maka diperlukan proses perhitungan dengan algoritma *vector-space model*. Ide dari metode ini adalah dengan menghitung nilai cosinus sudut dari dua vektor, yaitu W dari tiap dokumen dan W dari kata kunci.

Rumus yang digunakan adalah sebagai berikut :

Co sin *q) = =*

Keterangan :

q = kata kunci

d = kata di dokumen

t = kata di database

Vector space model solusi atas permasalah yang dihadapi jika menggunakan algoritma *TF/IDF*. Karena pada algoritma *TF/IDF* terdapat kemungkinan antar dokumen memiliki bobot yang sama, sehingga *ambigu* untuk diurutkan. Dengan alasan ini penulis menggunakan metode *Vector space model* dengan pendekatan *Cosine Similarity* (ukuran kemiripan antara 2 vektor) untuk menghindari kesamaan bobot pada saat pengurutan hasil pencarian.

* 1. **PHP**

PHP adalah bahasa pemrograman script yang paling banyak dipakai saat ini. PHP banyak dipakai untuk memprogram situs web dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain.

PHP dapat digunakan di berbagai mesin (*Linux, Unix, Macintosh, Windows*) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

Contoh terkenal dari aplikasi PHP adalah forum (phpBB) dan MediaWiki (software di belakang Wikipedia). PHP juga dapat dilihat sebagai pilihan lain dari ASP.NET/C#/VB.NET Microsoft, ColdFusion Macromedia, JSP/Java Sun Microsystems, dan CGI/Perl. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan PHP adalah Mambo, Joomla!, Postnuke, Xaraya, dan lain-lain.

Kelebihan PHP dari bahasa pemrograman lain :

1. Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. Web Server yang mendukung PHP dapat ditemukan dimana – mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis – milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa scripting yang paling mudah karena memiliki referensi yang banyak.
   1. **MYSQL**

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia (http://id.wikipedia.org/wiki/MySQL). MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL) (http://id.wikipedia.org/wiki/MySQL). MySQL dapat dijalankan pada berbagai *platform*, diantaranya Linux, Windows, dan lain-lain. Keunggulan lain yang dimiliki MySQL adalah mampu mendukung *Relational Database Manajement System* (RDBMS), sehingga dengan kemampuan itu MySQL akan mampu menangani datadata sebuah perusahaan yang berukuran sangat besar hingga ukuran *Giga Byte*. Untuk melakukan administrasi dalam basis data MySQL, dapat menggunakan modul yang sudah termasuk yaitu *command-line* (perintah: mysql dan mysqladmin).

MySQL memiliki beberapa keistimewaan, antara lain :

1. **Portabilitas**. MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
2. **Perangkat lunak sumber terbuka**. MySQL didistribusikan sebagai [perangkat lunak sumber terbuka](http://id.wikipedia.org/wiki/Perangkat_lunak_sumber_terbuka), dibawah lisensi [GPL](http://id.wikipedia.org/wiki/GPL) sehingga dapat digunakan secara gratis.
3. **Multi-user**. MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. *'****Performance tuning'***, MySQL memiliki kecepatan yang menakjubkan dalam menangani query sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. **Ragam tipe data**. MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double, char, text, date, timestamp*, dan lain-lain.
6. **Perintah dan Fungsi**. MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).
7. **Keamanan**. MySQL memiliki beberapa lapisan keamanan seperti level [subnetmask](http://id.wikipedia.org/w/index.php?title=Subnetmask&action=edit&redlink=1), nama [host](http://id.wikipedia.org/w/index.php?title=Host&action=edit&redlink=1), dan izin akses *user* dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8. **Skalabilitas dan Pembatasan**. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman ([*records*](http://id.wikipedia.org/w/index.php?title=Records&action=edit&redlink=1)) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya.
9. **Konektivitas**. MySQL dapat melakukan koneksi dengan klien menggunakan p*rotokol* [*TCP/IP*](http://id.wikipedia.org/wiki/TCP/IP)*,* [*Unix soket*](http://id.wikipedia.org/w/index.php?title=Unix_soket&action=edit&redlink=1) *(*[*UNIX*](http://id.wikipedia.org/wiki/UNIX)*), atau* [*Named Pipes*](http://id.wikipedia.org/w/index.php?title=Named_Pipes&action=edit&redlink=1) *(*[*NT*](http://id.wikipedia.org/wiki/NT)*)*.
10. **Lokalisasi**. MySQL dapat mendeteksi pesan kesalahan pada klien dengan menggunakan lebih dari dua puluh bahasa. Meski pun demikian, bahasa Indonesia belum termasuk di dalamnya.
11. **Antar Muka**. MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi [API](http://id.wikipedia.org/wiki/API) (*Application Programming Interface*).
12. **Klien dan Peralatan**. MySQL dilengkapi dengan berbagai peralatan ([tool](http://id.wikipedia.org/wiki/Tool))yang dapat digunakan untuk administrasi basis data, dan pada setiap peralatan yang ada disertakan petunjuk online.
13. **Struktur tabel**. MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basis data lainnya semacam [PostgreSQL](http://id.wikipedia.org/wiki/PostgreSQL) ataupun [Oracle](http://id.wikipedia.org/wiki/Oracle).
    1. **XML**

XML (*Extensible Markup Language*) adalah bahasa markup yang digunakan untuk menyimpan data (tidak ada program) dan tidak tergantung dengan tools tertentu (seperti editor, dbms, compiler, dan sebagainya).

XML merupakan kelanjutan dari [HTML](http://id.wikipedia.org/wiki/HTML) ([HyperText Markup Language](http://id.wikipedia.org/wiki/HyperText_Markup_Language)) yang merupakan bahasa standar untuk melacak [Internet](http://id.wikipedia.org/wiki/Internet). XML didesain untuk mempu menyimpan data secara ringkas dan mudah diatur. [Kata kunci](http://id.wikipedia.org/wiki/Kata_kunci) utama XML adalah [data](http://id.wikipedia.org/wiki/Data) ([jamak](http://id.wikipedia.org/wiki/Jamak) dari datum) yang jika diolah bisa memberikan informasi.

XML menyediakan suatu cara terstandarisasi namun bisa dimodifikasi untuk menggambarkan isi dari dokumen. Dengan sendirinya, XML dapat digunakan untuk menggambarkan sembarang *view* [database](http://id.wikipedia.org/wiki/Database), tetapi dengan suatu cara yang standar.

Keuntungan XML adalah :

* + 1. Ekstensibilitas : dapat ditukar/digabung dengan dokumen XML lain
    2. Memungkinkan pemrograman yang lebih baik maka dibuat suatu software pengolah XML
    3. Memisahkan data dan presentasi. Yang akan direpresentasikan dalam XML dan XSLT
    4. Pencarian data cepat karena XML merupakan data dalam format yang terstruktur
    5. Plain Text dan platform independent untuk pertukaran data.

## Contoh dokumen XML sederhana :

*<?xml version="1.0" encoding="UTF-8" ?>*

*<module>*

*<name>blockadvertising</name>*

*<displayName><![CDATA[Block advertising]]></displayName>*

*<version><![CDATA[0.3]]></version>*

*<description><![CDATA[Description]]></description>*

*<author><![CDATA[PrestaShop]]></author>*

*</module>*

* 1. **Crontab**

Crontab adalah sebuah perintah yang sangat berguna untuk menjalankan tugas-tugas yang terjadwal, sehingga akan mengurangi waktu administrasi.

Dalam melakukan administrasi sistem, pengaturan cron dilakukan melalui file crontab, yang berisi jadwal waktu dan script yang harus dieksekusi. System Linux memiliki file crontab default, yaitu /etc/crontab, yang akan menjalankan beberapa script pada waktu yang telah ditentukan, misalnya setiap jam, harian, mingguan, dan bulanan.

Terdapat dua buah file yang menentukan user mana yang bisa menggunakan crontab: /etc/cron.allow dan /etc/cron.deny. Biasanya, hanya ada file cron.deny pada sistem, dan jika file ini ada, dan terdapat nama user di dalamnya (satu user per baris), maka user tersebut tidak diperbolehkan menggunakan perintah crontab. Jika terdapat file cron.allow, maka hanya user yang namanya terdapat pada file ini yang diperbolehkan menggunakan perintah crontab.

Contoh perintah crontab:

# min(0-59) hours(0-23) day(1-31) month(1-12) dow(0-6) command

34 2 \* \* \* sh /root/backup.sh

**BAB III**

**ANALISA DAN PERANCANGAN SISTEM**

* 1. Gambaran umum sistem

Sistem yang dibangun ini adalah perpustakaan digital yang diakses melalui piranti bergerak yang didalamnya diimplementasikan metode text mining dalam catalog abstrak skripsi untuk optimasi pencarian. Tujuannya adalah untuk membuat peringkatan dokumen terhadap kata kunci yang dimasukkan. Tahapan dalam pembuatan peringkatan dokumen ini ada 2, yaitu preprocessing dan analisa.

3.1.1 Tahap Preprocessing

Text ming

Text ming

Database

Crontab

Dokumen

**Gambar 3.1 Tahap *preprocessing* terhadap dokumen**

Proses preprocessing dilakukan beberapa kali, pertama pada saat data diupload ke server dan lainnya dilakukan sesuai dengan jadwal crontab pada server. Adapun tahap preprocessing yang dilakukan meliputi yaitu *case folding, tokenizing, filtering, stemming.*

Secara garis besar, preprocessing ini terdiri dari 4 proses yaitu :

1. Proses pengenalan *file* berbasis *hiperteks* yang sudah ada.
2. Proses *Tokenizer*, yaitu unit pemrosesan dokumen menghasilkan token dan proses parsing dokumen untuk pengenalan token, yang terdapat di dalam file hyperteks yang sudah diinputkan.
3. Proses *Filtering* berdasarkan *Stoplist,* yaitu proses menghilangkan kata-kata buang yang didapat dari file hyperteks.
4. Proses *Stemming*, yaitu proses untuk menghilangkan imbuhan, awalan dan akhiran dari hasil *stoplist*.

Gambaran Umum proses preprocessing dapat digambarkan dalam flowchart pencarian dan pengurutan seperti dibawah ini :

|  |  |
| --- | --- |
| Masukkan  dokumen  Proses pre processing  (case folding, tokenizing, filtering,  stemming)  Menyimpan data asli dan data hasil pre processing ke dalam database | Mengambil data dalam DB  Ada perubanhan?  Proses pre processing  (case folding, tokenizing, filtering,  stemming)  Menyimpan data asli dan data hasil pre processing ke dalam database |
| **Gambar 3. 2 Proses *preprocessing*** | **Gambar 3.3 Proses *crontab*** |

**Penjelasan :**

Seperti digambar pada Gambar 3.1, dalam tahap preprocessing, proses yang dilakukan adalah:

1. Administrator memasukkan dokumen ke dalam perangkat lunak
2. Sistem akan melakukan pemrosesan terhadap dokumen tersebut, yaitu *case folding, tokenizing, filtering, stemming*.
3. Hasil proses tersebut, kemudian disimpan kedalam database termasuk juga data aslinya.

Untuk optimalisasi text mining dimanfaatkan sistem crontab pada server, tujuannya adalah melakukan proses preprocessing secara berskala dengan menyeting jadwal crontab, sehingga apabila terdapat penambahan stoplist pada database maka dokumen hasil preprocessing pada saat upload file bisa dilakukan perubahan lagi.

3.1.2 Tahap analisa

Preprocessing keyword

Dokumen hasil preprocessing

Pembobotan

*TF/IDF & VSM*

Keyword

Tampilkan hasil

**Gambar 3.4 Proses pencarian**

Proses analisa merupakan tahap penentuan seberapa jauh keterhubungan kata-kata dalam dokumen dengan kata kunci yang dimasukkan oleh *user*, sehingga mengahasilkan bobot dari masing-masing dokumen terhadap kata kunci. Dengan tahap analisa tersebut diharapkan dapat user mendapatkan data yang terurut sesuai dengan tingkat kedekatan kata kunci dengan isi dari dokumen (abstrak skripsi), dimana data yang ditampilkan berupa judul skripsi yang disertai dengan link untuk mendownload file skripsi.

Gambaran Umum proses pencarian dapat digambarkan dalam flowchart pencarian dan pengurutan seperti dibawah ini :

Ambil dokumen asli dari database

Dan diperingkatkan sesuai bobot

Hitung Pembobotan dokumen terhadap kata kunci

**(*TF/IDF & VSM )***

Masukkan kata kunci

Tampilkan peringkat dokumen beserta link download

Y

Kata kunci dimasukkan ke dalam proses *preprocessing*

For =1 to nKata

Hitung banyak kata nKata

K > nKata

Array [] bobot nKata

Ambil data hasil pre processing

sesuai kata kunci

**Gambar 3.5 Gambar diagram alir analisa pencarian konten**

**PENJELASAN :**

Dari flowchart diatas dapat dijelaskan proses / gambaran umum sistem pencarian dalam konten tugas akhir. Alur dari pencarian tugas akhir dimulai dengan *user* meng-*input*-kan kata kunci . Kemudian proses penghapusan kata-kata yang dianggap tidak penting dari kata kunci dengan membandingkan dengan daftar *stoplist* yang telah tersimpan dalam database. Jika proses pembuangan *stoplist* selesai dan terdapat kata penting maka proses pencarian dilanjutkan dengan proses *stemming* pada kata kunci, dilanjutkan dengan pencarian kata dalam kumpulan data tugas akhir dan proses perangkingan dokumen dengan metode TF/IDF dan*Vector Space Model* diakhiri dengan menampilkan judul tugas akhir secara berurutan sesuai dengan bobot atau *intensitas* kemunculan dari kata kunci beserta *link* untuk men-*dowload* dokumen tersebut.

3.1.3 Contoh perhitungan manual *text mining*

Dalam contoh perhitungan manual ini akan dijabarkan proses dalam text mining, yaitu *preprocessing(case folding, tokenizing, filtering, stemming)* dan analisa algoritma(*algoritma tf/idf dan vector space model*).

1. *Preprocessing*

Dokumen 1: *AJAX adalah suatu teknik pemrograman berbasis web untuk menciptakan aplikasi web lebih interaktif dan dinamis.*

Dokumen 2: *Fungsi AJAX adalah memindahkan sebagian besar interaksi web server, serta melakukan pertukaran data dengan server dibelakang layar sehingga halaman web tidak harus dibaca ulang secara keseluruhan setiap kali seorang pengguna melakukan perubahan.*

Dokumen 3: *Ada beberapa keuntungan menggunakan “AJAX” antara lain lebih interaktif , merubah data hanya mengambil halaman yang relevan dan lebih cepat.*

1. *Case folding*

Dalam tahap ini, dilakukan proses penerimaan karakter a-z dan selain karakter tersebut akan dibuang. Hasil proses *case folding* adalah sebagai berikut :

Dokumen 1 : *ajax adalah suatu teknik pemrograman berbasis web untuk menciptakan aplikasi web lebih interaktif dan dinamis*

Dokumen 2 : *fungsi ajax adalah memindahkan sebagian besar interaksi web server serta melakukan pertukaran data dengan server dibelakang layar sehingga halaman web tidak harus dibaca ulang secara keseluruhan setiap kali seorang pengguna melakukan perubahan*

Dokumen 3 : *ada beberapa keuntungan menggunakan ajax antara lain lebih interaktif merubah data hanya mengambil halaman yang relevan dan lebih cepat*

1. *Tokenizing*

Setelah dokumen melewati proses case folding maka dokumen tersebut akan di-*parsing* berdasarkan spasi agar terbentuk kumpulan kata seperti tabel berikut :

**Tabel 3.1 Tabel Tokenizer Dokumen 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Ajax | 9 | menciptakan |
| 2 | Adalah | 10 | aplikasi |
| 3 | Suatu | 11 | Web |
| 4 | Teknik | 12 | menjadi |
| 5 | pemprograman | 13 | lebih |
| 6 | Berbasis | 14 | dinamis |
| 7 | Web | 15 | Dan |
| 8 | Untuk | 16 | interaktif |

**Tabel 3.2 Tabel Tokenizer Dokumen** **2** :

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Ajax | 7 | server |
| 2 | Berfungsi | 8 | melalui |
| 3 | Memindahkan | 9 | Layar |
| 4 | Interaksi | 10 | belakang |
| 5 | Data | 11 | Web |
| 6 | Dengan | 12 |  |

**Tabel 3.3 Tabel Tokenizer Dokumen 3 :**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Ajax | 17 | layar |
| 2 | Adalah | 18 | sehingga |
| 3 | memindahkan | 19 | halaman |
| 4 | sebagian | 20 | web |
| 5 | Besar | 21 | tidak |
| 6 | interaksi | 22 | harus |
| 7 | Web | 23 | dibaca |
| 8 | Server | 24 | ulang |
| 9 | Serta | 25 | secara |
| 10 | melakukan | 26 | keseluruhan |
| 11 | pertukaran | 27 | setiap |
| 12 | Data | 28 | kali |
| 13 | dengan | 29 | seorang |
| 14 | Server | 30 | pengguna |
| 15 | dibelakang | 31 | melakukan |
| 16 | perubahan |  |  |

1. *Filtering*

Pada proses *filtering* ini kata-kata yang tidak penting akan dibuang. Kumpulan kata-kata tidak penting itu disebut *stoplist.* Adapun kata-kata yang termasuk *stoplist* seperti : tidak, yang, setiap dan lain sebagainya, maka akan mendapat hasil seperti berikut.

**Tabel 3.4 Tabel Filtering Dokumen** **1**:

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Ajax | 6 | menciptakan |
| 2 | Teknik | 7 | aplikasi |
| 3 | Peprograman | 8 | web |
| 4 | Berbasis | 9 | dinamis |
| 5 | Web | 10 | interaktif |

**Tabel 3.5 Table Filtering Dokumen** **2** :

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Fungsi | 10 | server |
| 2 | Ajax | 11 | dibelakang |
| 3 | Memindahkan | 12 | layar |
| 4 | Interaksi | 13 | halaman |
| 5 | Web | 14 | web |
| 6 | Server | 15 | dibaca |
| 7 | Melakukan | 16 | ulang |
| 8 | Pertukaran | 17 | keseluruhan |
| 9 | Data | 18 | perubahan |

**Tabel 3.6 Tabel Filtering Dokumen 3 :**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Keuntungan | 5 | data |
| 2 | Ajax | 6 | halaman |
| 3 | Interaktif | 7 | relevan |
| 4 | Merubah | 8 | cepat |

1. *Stemming*

Pada proses stemming ini tiap kata akan dihilangkan awalan dan akhiran sehingga membentuk kata dasar. Hasil dari proses stemming ini dapat dilihat pada Tabel berikut:

**Tabel 3.7 Tabel Stemming Dokumen 1**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Ajax | 7 | web |
| 2 | Teknik | 8 | aplikasi |
| 3 | Program | 9 | web |
| 4 | Basis | 10 | interaktif |
| 5 | Web | 11 | dinamis |
| 6 | Cipta |  |  |

**Tabel 3.8 Tabel Stemming Dokumen 2**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Fungsi | 8 | tukar |
| 2 | Ajax | 9 | data |
| 3 | Pindah | 10 | server |
| 4 | Interaksi | 11 | belakang |
| 5 | Web | 12 | layar |
| 6 | Server | 13 | halaman |
| 7 | Web |  |  |

**Tabel 3.9 Tabel Stemming Dokumen 3**

|  |  |  |  |
| --- | --- | --- | --- |
| NO | KATA | NO | KATA |
| 1 | Baca | 8 | interaktif |
| 2 | Ulang | 9 | ubah |
| 3 | Seluruh | 10 | data |
| 4 | Guna | 11 | halaman |
| 5 | Ubah | 12 | Relevan |
| 6 | Untung | 13 | cepat |
| 7 | Ajax |  |  |

1. Analisa

Pada tahap text mining di atas telah dihasilkan 3 buah data yang telah melewati preprocessing.

Kata kunci :

“perubahan web”

Kata kunci kemudian dimasukkan dalam proses *preprocessing,* sehingga menghasilkan kata kunci “ubah web”.

* + 1. Algoritma TF/IDF

Formula yang digunakan untuk menghitung bobot(w) masing-masing dokumen terhadap kata kunci adalah :

*­­­­­Wd.f = tfd.t \* IDF*

*IDF= D/df*

Untuk memudahkan perhitungan maka akan dibuat tabel perhitungan frekuensi kata seperti tertera pada tabel 3.5.

**Tabel 4.0 Tabel perhitungan TF/IDF**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Token | Tf | | | | df | D/df | IDF | W | | | |
| Kk | D1 | D2 | D3 | Kk | D1 | D2 | D3 |
| Ajax | 0 | 1 | 1 | 1 | 3 | 1 | 0 | 0 | 0 | 0 | 0 |
| Teknik | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Program | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Basis | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Web | 1 | 2 | 2 | 0 | 2 | 1,5 | 0,176 | 0,176 | 0,352 | 0,352 | 0 |
| Cipta | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Aplikasi | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Interaktif | 0 | 1 | 0 | 1 | 2 | 1,5 | 0,176 | 0 | 0,176 | 0 | 0,176 |
| Dinami,s | 0 | 1 | 0 | 0 | 1 | 3 | 0,477 | 0 | 0,477 | 0 | 0 |
| Fungsi | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Pindah | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Interaksi | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Server | 0 | 0 | 2 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,954 | 0 |
| Tukar | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Data | 0 | 0 | 1 | 1 | 2 | 1,5 | 0,176 | 0 | 0 | 0,176 | 0,176 |
| Belakang | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Layar | 0 | 0 | 1 | 0 | 1 | 3 | 0,477 | 0 | 0 | 0,477 | 0 |
| Halaman | 0 | 0 | 1 | 1 | 2 | 1,5 | 0,176 | 0 | 0 | 0,176 | 0,176 |
| Baca | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Ulang | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Seluruh | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Guna | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Ubah | 1 | 0 | 0 | 2 | 1 | 3 | 0,477 | 0,477 | 0 | 0 | 0,954 |
| Untung | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Relevan | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
| Cepat | 0 | 0 | 0 | 1 | 1 | 3 | 0,477 | 0 | 0 | 0 | 0,477 |
|  |  |  |  |  |  |  |  |  |  |  |  |

Bobot (W) untuk Dokumen 1 : 0,176+0,477= 0,352

Bobot (W) untuk Dokumen 2 : 0,352+0 = 0,352

Bobot (W) untuk Dokumen 3 : 0+0,954 = 0,954

Setelah dilakukan perhitungan, dapat diketahui bahwa nilai bobot dari D1 dan D2 adalah sama. Apabila diurutkan maka proses sorting tidak dapat mengurutkan secara tepat, kara nilai keduanya sama. Maka untuk mengatasinya, digunakan algoritma dari vector space model.

2. Vector Space Model (*Cosine Similarity*)

Untuk mempermudah perhitungan, maka akan digambarkan dalam tabel perhitungan vector space model seperti pada Tabel 4.1. Pada tahap ini akan dihitung cosines similiraty antar dokumen dimana bobot dokumen dokumen digambarkan sebagai matrik. Untuk menghitung kosinus sudu antar dokumen terhadap kata kunci maka dicari dulu token-token dari dokumen. Token-token diperoleh dari pencarian kata dalam dokumen setelah proses preprocessing. Token diambil satu kata jika mempunya dua atau lebih kata yang sama pada kumpulan dokumen.

Kemudian setiap token-token akan dihitung bobot nya terhadap dokumen-dokumen yang ada dan terhadap kata kunci yang dimasukkan. Setelah mendapatkan bobot tiap token, dicari bobot kuadrat dari bobot dokumen dan perkalian bobot kata kunci terhadap bobot dokumen.

**Tabel 3.6 Tabel perhitungan Vector Space Model**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Token | W | | | | W2 | | | | Kk\*D1 | Kk\*D2 | Kk\*D3 |
| Kk | D1 | D2 | D3 | Kk2 | D12 | D2 | D32 |
| Ajax | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Teknik | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 | 0 |
| Program | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 | 0 |
| Basis | 0 | 0,477 | 0 | 0 | 0 | 0,288 | 0 | 0 | 0 | 0 | 0 |
| Web | 0,176 | 0,352 | 0,352 | 0 | 0,031 | 0,124 | 0,124 | 0 | 0,062 | 0,062 | 0 |
| Cipta | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 | 0 |
| Aplikasi | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 | 0 |
| Interaktif | 0 | 0,176 | 0 | 0,176 | 0 | 0,031 | 0 | 0,031 | 0 | 0 | 0 |
| Dinamis | 0 | 0,477 | 0 | 0 | 0 | 0,228 |  | 0 | 0 | 0 | 0 |
| Fungsi | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Pindah | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Interaktif | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Server | 0 | 0 | 0,954 | 0 | 0 | 0 | 0,910 | 0 | 0 | 0 | 0 |
| Tukar | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Data | 0 | 0 | 0,176 | 0,176 | 0 | 0 | 0,031 | 0,031 | 0 | 0 | 0 |
| Belakang | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Layar | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 | 0 |
| Halaman | 0 | 0 | 0,176 | 0,176 | 0 | 0 | 0,031 | 0,031 | 0 | 0 | 0 |
| Baca | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Ulang | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Seluruh | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Guna | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Ubah | 0,477 | 0 | 0 | 0,954 | 0,228 | 0 | 0 | 0,910 | 0 | 0 | 0,455 |
| Untung | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Relevan | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,288 | 0 | 0 | 0 |
| Cepat | 0 | 0 | 0 | 0,477 | 0 | 0 | 0 | 0,228 | 0 | 0 | 0 |
| Perhitungan | | | | | Sum(kk2) | Sum(w2Di) | | | Sum (kk dot Di) | | |
| 0,259 | 1,523 | 2,464 | 2,599 | 0,062 | 0,062 | 0,455 |
| Sqrt (sumkk2) | Sqrt (sum w2Di) | | |
| 0,509 | 1,234 | 1,570 | 1,612 |

Perhitungan:

Formula :

Cosine (Di) = sum(kk dot Di)/([sqrt(kk2)]\*[sqrt(Di2)])

Perhitungan bobot tiap dokumen.

Cosine (D1) = 0,062/(0,509\*1,234)

= 0,062/0,628

= 0.098

Cosine (D2) = 0,062/(0,509\*1,570)

= 0,078

Cosine (D3) = 0,455/(0,509\*1,612)

= 0,455/0,821

= 0,554

Setelah mendapatkan bobot tiap dokumen, maka dokumen-dokumen akan diurutkan dari bobot terbesar ke bobot terkecil, sehingga hasil dari perangkingan dokumen dapat dilihat pada Tabel 3.7.

**Tabel 4.2 Tabel Rangking Dokumen**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rangking Dokumen** | **1** | **2** | **3** |
| **Dokumen ke** | 3 | 1 | 2 |
| **Bobot** | 0,554 | 0,098 | 0,078 |

**3.2 Perancangan antarmuka Sistem**

Pada perancangan antarmuka akan dibagi menjadi beberapa bagian antara lain:

* + 1. Perancangan halaman admin

Footer admin

Navigator admin

Header

Form Penambahan data Tugas akhir tahun lalu.

Tautan input data

Tautan tampil data

**Gambar 3.6 Tampilan penambahan dokumen**

Keterangan :

Input : tambah data abstak skripsi berformat text

tambah data file skripsi full.

Proses : jika data behasil disimpan maka akan dilanjutkan dengan

preprocessing secara otomatis.

* + 1. Perancangan form pengunjung

*Masukkan keyword*

GO

Form pencarian topik TA STMIK Denpasar

Footer

Navigasi web pengunjung

Header

**Gambar 3.6 Tampilan form pencarian**

**Gambar 3.7 Tampilan penambahan dokumen**

Keterangan :

Input : Kata kunci

Proses : Jika kata kunci yang dimasukkan terdapat dalam dokumen

yang tersimpan di dalam database maka akan diarahkan ke

halaman hasil pencarian.

3.1.3 Perancangan tampilamn hasil pencarian

Header

.

Footer

Navigasi web pengunjung

GO

*Masukkan keyword*

Grid hasil pencarian

**Gambar 3.8 Tampilan Hasil pencarian**

Keterangan :

Input : Kata kunci

Proses : Jika kata kunci yang dimasukkan terdapat dalam dokumen

yang tersimpan di dalam database maka halaman loading otomatis dengan menggunakan *ajax* data menampilkan data hasil pencarian.

.

**JADWAL PELAKSANAAN KEGIATAN**

Jadwal/kalender pelaksanaan kegiatan dirancang seperti contoh berikut :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Keterangan** | **Juni** | | | | **Juli** | | | | **Agustus** | | | | **September** | | |
| **II** | **III** | **IV** | **I** | **II** | **III** | **IV** | **I** | **II** | **III** | **IV** | **I** | **II** | **III** | **IV** |
| 1 | Usulan Proyek Penelitian |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 | Desain Sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 | Pembuatan Program |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 | Implementasi Sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 | Dokumentasi Sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Keterangan :**

**= titik pelaksanaan kegiatan**

**DAFTAR PUSTAKA**

XML, 2012. *http://id.wikipedia.org/wiki/EML.* Diakses Pada hari Rabu, 30 -05-

2012, 16.36.

Penambangan text, 2012. *http://id.wikipedia.org/wiki/Penambangan\_teks.* Diakses hari Rabu, 30-05-2012, 08:15.

Pencarian denga Vector Space Model, 2012. [*http://liyantanto.wordpress.com/2011/06/28/pencarian-dengan-metode-space-model-vsm*](http://liyantanto.wordpress.com/2011/06/28/pencarian-dengan-metode-space-model-vsm). Diakses hari Kamis, 31-05-2012, 15:20.

PHP, 2012. *http://id.wikipedia.org/wiki/PHP.* Diakses Pada hari Rabu, 30 -05-

2012, 16.36.