KLASIFIKASI DOKUMEN TEKS BERBAHASA INDONESIA MENGGUNAKAN METODE *ROCCHIO*

SKRIPSI

Oleh :
WILDAN SUHARSO
0210960056-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2008

UNIVERSITAS BRAWIJAYA

KLASIFIKASI DOKUMEN TEKS BERBAHASA INDONESIA MENGGUNAKAN METODE *ROCCHIO*

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar
Sarjana pada Program S1 Ilmu Komputer

Oleh :
WILDAN SUHARSO
0210960056-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2008

LEMBAR PENGESAHAN SKRIPSI

KLASIFIKASI DOKUMEN TEKS BERBAHASA INDONESIA MENGGUNAKAN METODE *ROCCHIO*

Oleh:
WILDAN SUHARSO
0210960056-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 22 Februari 2008
dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana
dalam bidang Ilmu Komputer

Pembimbing I	Pembimbing II
Drs. Achmad Ridok, M.Kom	**Bayu Rahayudi, ST., MT**
NIP. 132 090 392	**NIP. 132 318 424**

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas
Brawijaya

Dr. Agus Suryanto, M.Sc
NIP. 132 126 049

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Wildan Suharso
NIM : 0210960056
Program Studi : Ilmu Komputer
Jurusan : Matematika
Skripsi berjudul : Klasifikasi Dokumen Teks Berbahasa Indonesia Menggunakan Metode *Rocchio*.

Dengan ini menyatakan bahwa :

2. **Isi dari Skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.**
3. **Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 22 Februari 2008
Yang menyatakan,

(Wildan Suharso)
NIM. 0210960056-96

UNIVERSITAS BRAWIJAYA

KLASIFIKASI DOKUMEN TEKS BERBAHASA INDONESIA MENGGUNAKAN METODE *ROCCHIO*

ABSTRAK

Klasifikasi dokumen merupakan proses untuk mengklasifikasi dokumen ke dalam kategori kategori atau kelas tertentu. Sistem pengklasifikasian berita merupakan salah satu penerapan klasifikasi dokumen karena bertujuan mengklasifikasi berita ke dalam kategori tertentu. Beberapa tahapan yang dipakai klasifikasi dokumen adalah *preprocessing, features selection*, dan metode pembelajaran. *Text preprocessing* mengolah data awal agar menjadi data yang siap diproses pada tahapan selanjutnya, misalnya dengan melakukan penghilangan tanda baca. *Features selection* merupakan tahapan untuk memisah *good features* dari *all features*. *Good features* merupakan isi yang dianggap penting pada proses klasifikasi, sedangkan *all features* merupakan isi secara keseluruhan setelah melewati tahapan *text preprocessing*. Salah satu metode *features selection* adalah penghilangan kata yang sering muncul tapi tidak memiliki makna *(stopword)*. Tahapan metode pembelajaran merupakan tahapan terpenting dalam klasifikasi dokumen, yang berusaha menemukan pola dari keseluruhan teks. Metode yang digunakan pada tahapan ini adalah metode *rocchio*, yang merepresentasikan seluruh data ke dalam ruang vektor dengan *features* atau kata sebagai dimensi vektor, dengan pemakaian prototipe vektor untuk setiap kelas atau kategori. Hasil pengujian efektifitas menghasilkan rata-rata sebesar 0,8703 (87%) dan rata-rata efisiensi sebesar 10,231 detik, dengan menggunakan 679 data *training* dan 315 data *test*. Pengujian efisiensi dilakukan untuk mengetahui sejauh mana pengaruh jumlah data dengan waktu komputasi. Peningkatan jumlah data *training* meningkatkan efektifitas sistem tetapi menurunkan efisiensi sistem.

UNIVERSITAS BRAWIJAYA

TEXT DOCUMENT CLASSIFICATION IN INDONESIAN USING ROCCHIO CLASSIFIERS

ABSTRACT

Document classification is the process of grouping documents into different categories or classes. Document classification processes are text preprocessing, features selection, and classifiers method. Text preprocessing prepares text to data that will have been proceed in a next process, one of examples in that process is case folding. Features selection reduces the number of features, one of examples in features selection process is eliminating stopword. This research use rocchio classifiers to learn pattern from all of textual database. Rocchio classifiers represent all documents into vector space, features are dimensions of vector. Rocchio classifiers represent vector prototype for each class in all training documents. To evaluate a system classification probability effectiveness. The result shows that system have average effectiveness 0.8703 (87%) and average efficiency 10,231 second. Evaluation is also used to know the impact of the training set on effectiveness and efficiency of the system. Increasing training set will increase effectiveness of the system but will reduce efficiency of the system.

UNIVERSITAS BRAWIJAYA

KATA PENGANTAR

Segala puji dan syukur penulis panjatkan kepada Allah yang Maha Menentukan segala hal. Karena berkat ketentuanNyalah skripsi yang berjudul ”Klasifikasi Dokumen Teks Berbahasa Indonesia Menggunakan Metode Rocchio” dapat terselesaikan.

Dalam menyelesaikan skripsi ini, berbagai pihak telah memberikan bantuan baik moral maupun materiil. Atas bantuan yang telah diberikan, penulis mengucapkan terima kasih kepada :

1. Drs. Achmad Ridok, M.Kom. Terima kasih atas saran dan kritik, waktu, dorongan, semangat dan bantuan yang diberikan.
2. Bayu Rahayudi, ST., MT. Terima kasih atas saran dan kritik, waktu, dorongan, semangat dan bantuan yang diberikan.
3. Wayan F Mahmudy, S.Si., MT. Selaku Penasihat Akademik dan Ketua Program Studi Ilmu Komputer Universitas Brawijaya Malang.
4. Mama, Papa, mbah Uti, mbak Wida, mas Irfan dan Ninda yang senantiasa memberikan dorongan.
5. Semua Ilkomers dan Brawijaya. Terima kasih atas waktu dan bantuannya.
6. Semua karyawan dan staf jurusan Matematika.
7. Pihak lain yang tidak bisa penulis sebutkan satu persatu.

Semoga penulisan laporan skripsi ini bermanfaat bagi pembaca. Penulis menyadari bahwa dalam penulisan skripsi ini jauh dari kesempurnaan, atas segala kerendahan hati penulis mengharap saran dan kritik dari pembaca.

Malang, 22 Februari 2008

Penulis

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

Halaman

HALAMAN JUDUL i
LEMBAR PENGESAHAN SKRIPSI iii
LEMBAR PERNYATAAN v
ABSTRAK vii
KATA PENGANTAR xi
DAFTAR ISI xiii
DAFTAR GAMBAR xvii
DAFTAR TABEL xix
DAFTAR LAMPIRAN xxi
BAB I PENDAHULUAN
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan Penelitian 2
1.5 Manfaat Penelitian 2
1.6 Metodologi Penelitian 3
1.7 Sistematika Penulisan 3

BAB II TINJAUAN PUSTAKA
2.1 Konsep Klasifikasi Dokumen 5
2.1.1 Tipe Klasifikasi 5
2.1.2 Jenis Data 6
2.1.3 Tahapan Klasifikasi Dokumen 6
2.1.4 Pengukuran Efektifitas 7
2.2 Metode Pembelajaran *Rocchio* 8
2.2.1 Pembobotan *TFIDF* 9
2.2.2 Representasi Dokumen 9
2.2.3 Prototipe Vektor 10
2.2.4 Klasifikasi *TFIDF* 10
2.3 Berita 11
2.4 *Stemming* Pada Bahasa Indonesia 12
2.4.1 Morfologi 12
2.4.2 Proses Stemming Bahasa Indonesia 14

BAB III METODOLOGI DAN PERANCANGAN
3.1 Perancangan Sistem 17

3.1.1 Deskripsi Umum Sistem 17
3.1.2 Batasan Sistem 18
3.2 Proses Klasifikasi 18
3.2.1 Proses *Text Preprocessing* 18
3.2.2 Proses *Features Selection* 19
3.2.3 Proses Metode Pembelajaran 22
3.3 Perancangan Database 22
3.3.1 Perancangan Tabel 22
3.3.2 ERD 25
3.4 Implementasi Metode *Rocchio* 25
3.4.1 Representasi Dokumen 26
3.4.2 Membangun Prototipe Vektor 27
3.4.2.1 Prototipe Vektor Musik 27
3.4.2.2 Prototipe Vektor Sport 28
3.4.3 Data *Test* 28
3.4.4 Menghitung Kedekatan Sudut 28
3.4.4.1 Kedekatan Data *Test* Dan Kategori Musik .. 29
3.4.4.2 Kedekatan Data *Test* Dan Kategori Sport ... 29
3.4.5 Rangking Kedekatan 29

BAB IV IMPLEMENTASI DAN PEMBAHASAN
4.1 Implementasi 31
4.1.1 Spesifikasi Komputer 31
4.1.2 Persiapan Data 31
4.1.3 Implementasi *Database* 32
4.1.4 Deskripsi Program 34
4.1.4.1 *Preprocessing* 34
4.1.4.2 *Filterstring* 35
4.1.4.3 *Stemming* 36
4.1.4.4 Metode Pembelajaran 38
4.2 Penerapan Aplikasi 40
4.2.1 Halaman Dictionary 40
4.2.2 Halaman Stopword 40
4.2.3 Halaman Preprocessing 41
4.2.4 Halaman News Parser 42
4.2.5 Halaman News Classification 43
4.3 Analisa Hasil 44
4.3.1 Efektifitas 44
4.3.2 Efisiensi 46

BAB V PENUTUP
5.1 Kesimpulan 49
5.2 Saran 49

DAFTAR PUSTAKA 51
LAMPIRAN 53

UNIVERSITAS BRAWIJAYA

DAFTAR GAMBAR

		Halaman
Gambar 3.1	Blok diagram proses sistem	18
Gambar 3.2	*Flowchart* proses *features selection*	19
Gambar 3.3	*Flowchart* proses *stemming*	21
Gambar 3.4	ERD	25
Gambar 3.5	Representasi dokumen	27
Gambar 4.1	Halaman dictionary	40
Gambar 4.2	Halaman stopword	41
Gambar 4.3	Halaman preprocessing	41
Gambar 4.4	Halaman news parser	42
Gambar 4.5	Hasil parsing	42
Gambar 4.6	Halaman news classification	43
Gambar 4.7	Hasil klasifikasi	43
Gambar 4.8	Rata-rata Efektifitas	45
Gambar 4.9	Rata-rata Efisiensi	47

UNIVERSITAS BRAWIJAYA

DAFTAR TABEL

		Halaman
Tabel 2.1	Aturan partikel infleksional	14
Tabel 2.2	Aturan kata ganti infleksional	14
Tabel 2.3	Aturan prefiks derivasional pertama	14
Tabel 2.4	Aturan prefiks derivasional kedua	15
Tabel 2.5	Aturan sufiks derivasional	15
Tabel 3.1	Aturan peleburan	20
Tabel 3.2	Atribut tabel news	22
Tabel 3.3	Atribut tabel news_category	23
Tabel 3.4	Atribut tabel news_part	23
Tabel 3.5	Atribut tabel stopword	24
Tabel 3.6	Atribut tabel dictionary	24
Tabel 3.7	Atribut tabel rank	24
Tabel 3.8	Daftar kata	25
Tabel 3.9	Daftar bobot kata	26
Tabel 4.1	Hasil uji efektifitas 1	44
Tabel 4.2	Hasil uji efektifitas 2	45
Tabel 4.3	Hasil uji efisiensi 1	46
Tabel 4.4	Hasil uji efisiensi 2	47

UNIVERSITAS BRAWIJAYA

DAFTAR LAMPIRAN

Halaman

Lampiran 1 Daftar *Stopword* 53

UNIVERSITAS BRAWIJAYA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kemajuan dalam bidang informasi sangatlah pesat, dari media cetak sampai media elektronik. Seiring dengan kemajuan tersebut, mendorong penyedia informasi untuk meningkatkan fasilitas dalam penyampaian informasi. Salah satu fasilitas yang mendukung dalam penyampaian informasi adalah kemampuan media dalam melakukan klasifikasi, karena dengan adanya klasifikasi dapat memudahkan pencarian informasi sesuai dengan kategori yang diinginkan.

Salah satu contoh aplikasinya adalah klasifikasi pada web berita berbahasa Indonesia, yang memudahkan pembaca web mencari berita sesuai dengan kategori yang diinginkan. Secara sederhana klasifikasi dokumen atau *document classification* dapat diartikan sebagai metode pengelompokan dokumen ke dalam kelompok-kelompok menggunakan aturan-aturan tertentu. Aturan-aturan tersebut yang menjadikan klasifikasi dokumen memiliki metode pembelajaran yang beragam.

Secara garis besar klasifikasi dokumen dibagi menjadi 2 metode dasar, yaitu *unsupervised document classification* dan *supervised document classification*. *Unsupervised document classification* merupakan teknik klasifikasi dokumen yang memisahkan data-data ke dalam kelompok dengan menilai kemiripan antar data dari segi isi. Sedangkan s*upervised document classification* mengasosiasikan satu bagian data dengan yang lainnya dengan aturan aturan tertentu. Pada media teks, metode *supervised document classification* disebut juga dengan *text mining*.

Text mining merupakan proses penemuan informasi baru yang sebelumnya tidak terungkap dengan memproses dan menganalisa data dalam jumlah besar. Dalam menganalisa sebagian atau keseluruhan *unstructured text, text mining* mengasosiasikan satu bagian teks dengan yang lainnya berdasarkan aturan-aturan tertentu. Hasil yang diharapkan adalah informasi baru yang sebelumnya tidak terungkap dengan jelas **(Adiwijaya, 2006)**.

Banyak metode pembelajaran yang dikembangkan untuk *text mining*, dan setiap metode mempunyai kelebihan dan kekurangan yang berbeda. Salah satu contoh metode *text mining* adalah *rocchio classifiers*. Klasifikasi *rocchio* mengadaptasi teknik *information retrieval*, yaitu dengan merepresentasikan data-data ke dalam ruang vektor dengan seluruh kata sebagai dimensi vektor **(Joachims, 1997)**. Klasifikasi

rocchio membandingkan kedekatan sudut antara data yang digunakan untuk pembelajaran atau data *training* dan data yang akan diklasifikasi atau data *test*.

Text mining yang dilakukan pada media teks berbahasa Indonesia pun masih terus dalam pengembangan. Oleh karena itu perlu adanya analisa mengenai *text mining* pada berita berbahasa Indonesia dengan menggunakan metode *rocchio*. Berdasarkan latar belakang yang dipaparkan maka skripsi ini berjudul "**Klasifikasi Dokumen Teks Berbahasa Indonesia Menggunakan Metode Rocchio**".

1.2 Rumusan Masalah

Bagaimana mengimplementasikan metode klasifikasi dokumen untuk melakukan klasifikasi pada berita berbahasa Indonesia yang hanya berupa teks menggunakan metode *rocchio*.

1.3 Batasan Masalah

Pada skripsi ini, masalah dibatasi pada :

1. Klasifikasi dokumen dengan menggunakan metode *rocchio*
2. Dokumen yang digunakan hanya merupakan berita berbahasa Indonesia dengan format teks (txt)
3. *Stemming* yang dilakukan adalah *stemming* sederhana.

1.4 Tujuan Penelitian

Tujuan dari pengerjaan skripsi ini adalah melakukan analisa dan mengaplikasikan klasifikasi dokumen berbahasa Indonesia dengan menggunakan metode *rocchio*.

1.5 Manfaat Penelitian

Manfaat yang diharapkan pada skripsi ini adalah menghasilkan *software text mining*.

1.6 Metodologi Penelitian

1. Studi Kepustakaan
   Dilakukan dengan cara mencari pustaka sekaligus mencari artikel mengenai klasifikasi dokumen.
2. Perancangan dan Pembuatan Program
   Diperlukan untuk melakukan percobaan dengan program komputer.

3. Pengujian Perangkat Lunak
   Melakukan pengujian metode klasifikasi dokumen dengan menggunakan berita-berita yang dipilih dari *website* koran tertentu.
4. Pembuatan Laporan
   Membuat laporan tertulis untuk skripsi ini.

1.7 Sistematika Penulisan

BAB I : PENDAHULUAN
Dalam bab ini, dijelaskan pentingnya klasifikasi dokumen yang termuat dalam latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA
Dalam bab ini dijelaskan teori teori beserta pustaka yang digunakan dalam pembuatan klasifikasi dokumen.

BAB III : METODOLOGI DAN PERANCANGAN
Dalam bab ini dijelaskan metode yang digunakan dan tahapan-tahapan klasifikasi dokumen.

BAB IV : IMPLEMENTASI DAN PEMBAHASAN
Dalam bab ini dijelaskan bagaimana mengimplementasikan metode pada bahasa pemrograman, serta uji coba dan analisanya.

BAB V : PENUTUP
Dalam bab ini berisi kesimpulan dan saran.

DAFTAR PUSTAKA
LAMPIRAN

UNIVERSITAS BRAWIJAYA

BAB II
TINJAUAN PUSTAKA

2.1 Konsep Klasifikasi Dokumen

Pengguna *search engine* di Internet sering menganggap *search engine* merupakan salah satu implementasi dari klasifikasi dokumen, walaupun *search engine* dan klasifikasi dokumen mempunyai perbedaan. *Search engine* menyingkirkan teks yang tidak sesuai dengan kata kunci yang dicari, sedangkan klasifikasi dokumen berusaha menemukan kategori suatu dokumen baru dengan metode tertentu. Klasifikasi dokumen merupakan sebuah eksplorasi dan analisa dari suatu informasi untuk mendapatkan sesuatu yang belum diketahui sebelumnya oleh sistem **(Hearst, 2003)**.

Contoh dari aplikasi klasifikasi dokumen adalah *amazon.com* yang mengelompokkan pembeli berpotensial ke dalam sebuah profil teks, atau pada bidang industri yang melakukan identifikasi produk dan harga dari *website* pesaing, atau pengelompokan lowongan pekerjaan sesuai klasifikasi yang dibutuhkan pada *flipdog.com* **(Zohar, 2002)**.

2.1.1 Tipe Klasifikasi

Klasifikasi dokumen dibagi menjadi 2 metode dasar, yaitu *unsupervised document classification* dan *supervised document classification***.**

1. *Unsupervised document classification*
   Unsupervised document classification merupakan metode klasifikasi dokumen yang tidak memiliki pola atau aturan yang dicari untuk dijadikan pembelajaran. Keseluruhan data di kelompokkan secara bersama sama berdasarkan kesamaan isi. *Unsupervised document classification* tidak menggunakan metode pembelajaran yang digunakan pada *supervised document classification* **(Larose, 2005)**. Contoh *unsupervised document classification* adalah *clustering*, yang merepresentasikan dokumen dokumen ke dalam cluster yang mempunyai kesamaan isi.
2. *Supervised document classification*
   Merupakan metode klasifikasi dokumen dengan cara mempelajari pola atau aturan yang tidak tergambar jelas sebelumnya pada dokumen yang telah terklasifikasi oleh sistem ataupun oleh manusia, sehingga dapat dipakai untuk mengklasifikasi dokumen baru. *Supervised*

document classification membangun model atau pembelajaran yang dapat digunakan untuk melakukan klasifikasi dari sekumpulan dokumen yang terklasifikasi sebelumnya. Dokumen yang terklasifikasi sebelumnya ini disebut dengan *training sets* atau data *training*. Model atau hasil pembelajaran yang diperoleh dari data *training* dapat digunakan untuk mengklasifikasi dokumen yang belum terklasifikasi sebelumnya, disebut dengan *test sets* atau data *test* **(Bing, 2005).**

2.1.2 Jenis Data

Data yang dipergunakan dapat berupa *text*, *hypertext* dan *multimedia*. Data teks hanya berisi teks secara keseluruhan. Media *hypertext* merupakan media teks yang mempunyai *link* ke teks yang lain, contohnya adalah *file* pdf, *help* pada windows, *help* pada linux. Media *multimedia* merupakan media yang berupa teks, gambar, suara yang disajikan secara bersama-sama, contohnya adalah halaman web yang interaktif disertai gambar bergerak dan suara.

Pada klasifikasi dokumen dengan media teks menggunakan metode *supervised document classification* biasa disebut dengan *text mining*, salah satu contohnya adalah klasifikasi berita teks berbahasa Indonesia. *Text mining* merupakan proses penemuan akan informasi baru yang sebelumnya tidak terungkap dengan memproses dan menganalisa data dalam jumlah besar. Dalam menganalisa sebagian atau keseluruhan *unstructured text*, *text mining* mengasosiasikan satu bagian teks dengan yang lainnya berdasarkan aturan-aturan tertentu. Hasil yang diperoleh adalah informasi baru yang sebelumnya tidak terungkap dengan jelas **(Adiwijaya, 2006)**.

2.1.3 Tahapan Klasifikasi Dokumen

Ada beberapa tahapan yang dilakukan dalam klasifikasi dokumen antara lain *text preprocessing*, *feature selection*, *learning method* dan menganalisa hasil yang diperoleh dari klasifikasi dokumen **(Zohar, 2002)**.

1. *Text preprocessing*

   Text Preprocessing dilakukan untuk menyiapkan dokumen awal untuk menjadi dokumen teks yang siap diolah pada tahap selanjutnya. Proses yang dilakukan dalam *text preprocessing* adalah menghilangkan semua tanda baca dan *case folding* atau mengubah semua karakter dalam dokumen ke dalam huruf kecil **(Garcia, 2005).**

2. *Features selection*
   Semua kata yang telah melewati tahapan *text preprocessing* dianggap sama, yaitu memiliki peran yang sama dalam klasifikasi dokumen. Peran sama ini disebut dengan *all features*, sedangkan kata yang memiliki peran yang penting dalam klasifikasi disebut juga dengan *good features*. Untuk membedakan antara *all features* dan *good features* dilakukan *features selection.* Banyak metode yang dipakai dalam tahapan *features selection*, satu diantaranya adalah *pruning of high frequency words* **(Staelin, features selection)**.
   prunning of high frequency words merupakan metode *features selection* yang melakukan penghilangan kata yang sering muncul dan tidak mempunyai makna *(stopword)*. Contohnya "jika", "ini", "maka", "itu" **(Gonen, 2004)**.
3. Metode pembelajaran
   Tahapan metode pembelajaran mempelajari pola atau aturan yang menunjukkan model kategori dari data *training* setelah dilakukan *features selection*. Banyak metode pembelajaran *supervised* atau *unsupervised* yang dikembangkan, salah satunya adalah *rocchio classifiers* yang merupakan salah satu dari banyak metode pembelajaran *text mining* **(Manning, 2003)**.

2.1.4 Pengukuran Efektifitas

Pengukuran efektifitas yang digunakan adalah peluang jawaban benar dari setiap kategori. Hal tersebut dikarenakan semua data *training* telah mempunyai kategori yang ditentukan sebelumnya, data *training* tidak dapat bertambah, dan pemakaian pengukuran efektifitas menggunakan nilai peluang lebih informatif. Peluang kebenaran kategori adalah jumlah data *test* yang benar dalam satu kategori dibagi dengan jumlah data *test* keseluruhan dalam satu kategori tersebut. Dapat ditulis sebagai berikut :

$$P(c_j) = \left(\frac{d'_j}{D'_j} \right) \quad \text{.....(2.1)}$$

Keterangan :
- $P(c_j)$ merupakan peluang benar untuk kategori ke *j*
- d'_j merupakan jumlah data test kategori *j* yang benar
- D'_j merupakan jumlah keseluruhan data *test* pada kategori *j*.

2.2 Metode Pembelajaran *Rocchio*

Rocchio classifiers merupakan salah satu metode pembelajaran *supervised document classification*. Metode klasifikasi *rocchio* membandingkan kesamaan isi antara data *training* dan data *test* dengan merepresentasikan semua data ke dalam vektor. Setiap bobot kata merupakan dimensi dalam ruang vektor.

Kedekatan kesamaan isi dihitung dari kedekatan sudut yang terbentuk antara bobot data *training* dan bobot data *test* menggunakan aturan cosine. Untuk menghitung bobot setiap kata dalam dokumen digunakan skema pembobotan *TFIDF (Term Frequency / Invers Document Frequency)*. Karena komponen *heuristic* / utama dari klasifikasi *rocchio* adalah skema pembobotan *TFIDF*, metode pembelajaran *rocchio* disebut juga dengan *TFIDF Classifiers* **(Joachihms, 1997).**

Dalam membandingkan kesamaan isi antara data *training* dan data *test*, *TFIDF classifiers* menggunakan prototipe vektor untuk merepresentasikan kategori yang terbentuk dari data training, dengan kata lain prototipe vektor merupakan vektor yang mewakili seluruh vektor data *training* dalam setiap kategori.

Tiga hal utama yang dipakai pada klasifikasi *TFIDF* adalah menggunakan skema pembobotan *TFIDF* untuk merepresentasikan dokumen ke dalam vektor, merepresentasikan prototipe setiap kategori dengan menjumlahkan vektor-vektor dalam satu kategori dari data *training*, membandingkan kedekatan sudut antara vektor data *test* dengan semua prototipe vektor **(Tomassen, 2007).**

2.2.1 Pembobotan *TFIDF*

Skema pembobotan *TFIDF* merupakan kombinasi dari *Term frequency (TF)* dan *Inverse Document Frequency (IDF)*. *Term frequency* merupakan jumlah kemunculan kata w_i dalam dokumen *d*, yang dapat dituliskan sebagai $TF(w_i, d)$. Dimana w_i merupakan kata dalam dokumen dan *d* merupakan dokumen klasifikasi.

Sedangkan *Inverse Document Frequency (IDF)* merupakan perhitungan dari jumlah seluruh dokumen *(D)* dibagi dengan *Document Frequency (DF)* dari kata (w_i). Dapat dituliskan sebagai berikut :

$$IDF(w_i) = \log\left(\frac{D}{DF(w_i)}\right) \quad \text{.....(2.2)}$$

Keterangan :

- *IDF* merupakan *Inverse Document Frequency* dari kata w_i
- *D* merupakan jumlah keseluruhan dokumen
- $D = d_1+d_2+d_n+...+d'$
- $DF(w_i)$merupakan jumlah dokumen yang memiliki kata w_i.

Kemudian rumus bobot kata setiap dokumen dihitung menggunakan rumus *TFIDF* sebagai berikut :

$$d^{(i)} = TF(w_i, d) * IDF(w_i) \quad \text{.....(2.3)}$$

Keterangan :
- $TF(w_i, d)$ merupakan frekuensi kata w_i dalam dokumen *d*
- $IDF\ (w_i)$ merupakan *Inverse Document Frequency* dari kata w_i
- $d^{(i)}$ disebut bobot kata (w_i) dalam dokumen *(d).*

2.2.2 Representasi Dokumen

Seluruh dokumen direpresentasikan ke dalam vektor, dengan kata sebagai dimensi vektor dan bobot setiap kata dalam dokumen digunakan untuk menentukan vektor dokumen.

$$\vec{d} = (d^{(1)},...,d^{(i)}) \quad \text{.....(2.4)}$$

Keterangan :
- $\vec{d}$ merupakan vektor dokumen *d*
- $d^{(i)}$ disebut bobot kata (w_i) dalam vektor $\vec{d}$.

Setiap dimensi dari ruang vektor merupakan bobot kata yang telah melewati tahap *features selection*. Bobot kata untuk setiap dokumen dihitung menggunakan skema pembobotan *TFIDF* **(Gonen, 2004)**.

2.2.3 Prototipe Vektor

Klasifikasi *TFIDF* mendefinisikan sebuah prototipe vektor $(\vec{c})$ untuk setiap kelas, dimana setiap kelas berisikan dokumen yang mempunyai kategori yang sama. Prototipe vektor dibangun dengan menambahkan semua vektor dokumen dari data *training* dalam satu kelas atau kategori.

$$\vec{c} = \sum_{d \in C} \vec{d} \quad \text{.....(2.5)}$$

Keterangan :
- $\vec{c}$ merupakan prototipe vektor
- $\vec{d}$ merupakan vektor *data training (d)* dalam satu kelas *C*

Hasil dari prototipe vektor adalah model pembelajaran untuk setiap kelas yang akan digunakan untuk mengklasifikasi data *test* **(Gonen, 2004)**.

2.2.4 Klasifikasi *TFIDF*

Model pembelajaran yang didapat dari prototipe vektor digunakan untuk mengklasifikasi data *test*. Data *test* direpresentasikan ke dalam vektor, kemudian dihitung kedekatan sudut cosine antara vektor data *test* dengan setiap prototipe vektor.

Vektor data *test* dianggap sesuai dengan salah satu kategori, jika sudut cosine yang dihasilkan antara vektor data *test* dan prototipe vektor mempunyai sudut cosine terdekat.

TFIDF classifiers memberi nilai target kepada data baru menggunakan nilai H_{TFIDF}, yaitu nilai tertinggi dari seluruh anggota himpunan set domain H.

$$H_{\text{TFIDF}}(d') = \arg\max_{c_j \in C} \cos(\vec{c}_j, \vec{d}') \quad \text{.....(2.6)}$$

Keterangan :
- $H_{TFIDF}(d')$ adalah algoritma untuk mengkategori dokumen *test* (d') ke dalam kategori tertentu *(c_j)*
- $\vec{c}_j$ merupakan prototipe vektor c_j
- c_j merupakan anggota dari kategori *C*

Algoritma tersebut dapat diringkas sebagai berikut :

$$H_{TFIDF}(d') = \arg\max \frac{\sum_{k=1}^{|F|} c_j^{(k)} \cdot d'^{(k)}}{\sqrt{\sum_{k=1}^{|F|} (c_j^{(k)})^2} \sqrt{\sum_{k=1}^{|F|} (d'^{(k)})^2}} \quad \text{.....(2.7)}$$

Keterangan :
- $c_j^{(k)}$ merupakan kata *(k)* dari prototipe (c_j)
- $d'^{(k)}$ merupakan kata *(k)* dari data test *(d')*
- *(|F|)* merupakan kata yang terakhir
- *(k=1)* merupakan kata yang ke-1

(Sebastiani, 2002)

2.3 Berita

Berita dapat diartikan sebagai laporan tentang suatu peristiwa yang sudah terjadi. Sebuah berita tersusun dari beberapa bagian, sebagai berikut :

1. Judul
   Judul mewakili keseluruhan isi berita yang disampaikan dan memiliki daya tarik yang kuat.
2. Baris tanggal *(dateline)*
   Waktu dan tempat berita tersebut diperoleh atau ditulis.
3. Teras berita *(lead atau intro)*
   Lead terletak sebelum tubuh berita. *Lead* biasanya berisi ringkasan dan berupa kalimat yang menarik. *Lead* bertujuan untuk menarik pembaca untuk melanjutkan ke bagian berita yang selanjutnya. *Lead* juga biasanya berisi fakta paling penting dengan mengedepankan unsur 5W+1H *(what, who, when, where, why, how)*.
4. Tubuh berita *(body)*
   Tubuh berita berisi penjelasan atau uraian dari *Lead.* Biasanya menjelaskan unsur 5W+1H, baik yang sudah ditulis pada *lead* ataupun yang belum ditulis sebelumnya**(Budiman, 2005)**.

2.4 *Stemming* Pada Bahasa Indonesia

2.4.1 Morfologi

Morfologi merupakan bagian dari ilmu bahasa yang mengkaji struktur, bentuk dan penggolongan kata. Struktur kata ialah susunan bentuk bunyi

ujaran atau lambang (tulisan) yang menjadi unit bahasa yang bermakna. Bentuk kata merupakan unit tata bahasa yang berbentuk tunggal atau hasil dari proses pengimbuhan, pemajemukkan dan penggandaan. Penggolongan kata ialah proses mengelompokkan kata berdasarkan kesamaan bentuk dan fungsi ke dalam kumpulan kata. Struktur kata disebut dengan morfem, yang dibedakan antara morfem dasar dan morfem terikat. Morfem dasar merupakan kata yang dapat berdiri sendiri tanpa adanya tambahan, sedangkan morfem terikat merupakan kata yang terdiri dari morfem dasar atau gabungan morfem dasar.

Morfologi kata bahasa Indonesia dibagi ke dalam struktur infleksional dan derivasional, pembagian struktur tersebut digunakan untuk melakukan proses *stemming* pada bahasa Indonesia. Infleksional adalah struktur yang paling sederhana yang dinyatakan dalam penambahan sufiks dimana tidak mempengaruhi arti sebenarnya dari kata dasar yang dilekati. Sufiks infleksional dapat dibagi menjadi 2 jenis. Sufiks infleksional tersebut adalah sebagai berikut :

1. Sufiks *-lah, -kah, -pun, -tah*. Sufiks ini sebenarnya adalah partikel yang tidak mempunyai arti. Keberadaannya pada suatu kata adalah untuk penekanan. Contoh :

dia	+	kah	→	diakah
duduk	+	lah	→	duduklah

2. Sufiks *-ku, -mu, -nya*. Sufiks ini berfungsi sebagai kata ganti kepunyaan. Contoh :

tas	+	ku	→	tasku
buku	+	mu	→	bukumu

Penambahan sufiks infleksional tidak akan merubah bentuk dasar dari kata berimbuhan**(Tala, 2003)**. Dengan kata lain, tidak ada penghilangan atau peleburan kata dasar pada kata berimbuhan. Kata dasar dapat ditentukan dengan mudah pada struktur infleksional.

Sehingga struktur morfologi Infleksional dapat dituliskan sebagai berikut :

Infleksional = (kata dasar +kata ganti) | (kata dasar + partikel) | (kata dasar + kata ganti + partikel)

Struktur derivasional merupakan kata yang terbentuk dari sufiks, prefiks atau gabungan keduanya. Deny Arnos Kwary menyebutnya dengan afiks, dalam analisanya tentang delapan jenis afiks dalam 3 bahasa. Tidak seperti struktur infleksional, pada struktur derivasional pengucapan kata mungkin berubah setelah adanya penambahan prefiks

atau disebut dengan peleburan, tetapi tidak mengalami perubahan bila dilekati oleh sufiks. Pada struktur derivasional juga memungkinkan penambahan sufiks dan prefiks secara bersamaan, meskipun tidak semua sufiks dan prefiks dapat dikombinasikan secara bersama sama.

Prefiks dapat ditambahkan pada suatu kata yang telah terdapat konfiks / prefiks, dan menghasilkan struktur prefiks ganda. Seperti pada pembentukan sebuah konfiks, pada pembentukan prefiks ganda, tidak semua prefiks dapat ditambahkan pada kata yang telah mendapatkan prefiks / konfiks.

Sehingga struktur morfologi pada kata derivasional adalah :

Derivasional = (prefiks + kata dasar) | (kata dasar + sufiks) | (prefiks + kata dasar + sufiks) | (prefiks 1 + prefiks 2 + kata dasar) | (prefiks 1 + prefiks 2 + kata dasar + sufiks).

Contoh dari struktur kata derivasional tersebut adalah sebagai berikut :

1. mem + beri → memberi
2. ambil + kan → ambilkan
3. meng + ambil + kan → mengambilkan
4. mem + per + indah → memperindah
5. mem + per + main + kan → mempermainkan

Struktur lain yang mungkin terjadi dalam morfologi bahasa Indonesia adalah penambahan sufiks infleksional pada struktur derivasional.

Sehingga dapat disimpulkan secara umum struktur morfologi kata bahasa Indonesia adalah :

Struktur morfologi = [prefiks 1] + [prefiks 2] + kata dasar + [sufiks] + [kata ganti] + [partikel].

Macam-macam dan penggolongan prefiks dan sufiks dijelaskan pada sub bab selanjutnya.

2.4.2 Proses Stemming Bahasa Indonesia

Terdapat 5 aturan tahapan pada proses *stemming* dalam bahasa Indonesia **(Tala, 2003)**. Aturan-aturan tersebut adalah :

1. Aturan yang menangani partikel infleksional

Tabel 2.1 Aturan partikel infleksional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
kah	NULL	NULL	diakah→ dia
lah	NULL	NULL	adalah → ada

tah	NULL	NULL	apatah → apa
pun	NULL	NULL	bukupun→buku

2. Aturan yang menangani kata ganti infleksional

Tabel 2.2 Aturan kata ganti infleksional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
ku	NULL	NULL	bukuku→buku
mu	NULL	NULL	bukumu→buku
nya	NULL	NULL	bukunya→buku

3. Aturan yang menangani urutan prefiks derivasional pertama.

Tabel 2.3 Aturan prefiks derivasional pertama

Prefiks	Pengganti	Kondisi Tambahan	Contoh
meng	NULL	NULL	mengukur→ukur
meny	S	V...*	menyapu→sapu
men	T	V...*	menuduh→tuduh
men	NULL	NULL	menduga→duga
mem	P	V...*	memukul→pukul
mem	NULL	NULL	membakar→bakar
me	NULL	NULL	merusak→rusak
peng	NULL	NULL	pengukur→ukur
peny	S	V...*	penyelam→selam
pen	T	V...*	penari→tari
pen	NULL	NULL	penduga→duga
pem	P	V...*	pemandu→pandu
pem	NULL	NULL	pembaca→baca
di	NULL	NULL	diukur→ukur
ter	NULL	NULL	tersipu→sipu
ke	NULL	NULL	kekasih→kasih

* kata dasar dimulai huruf vokal

4. Aturan yang menangani urutan prefiks derivasional kedua.

Tabel 2.4 Aturan prefiks derivasional kedua

Prefiks	Pengganti	Kondisi Tambahan	Contoh
ber	NULL	NULL	berlari→lari
bel	NULL	ajar	belajar→ajar
be	NULL	kerja	bekerja→kerja
per	NULL	NULL	perjelas→jelas
pel	NULL	ajar	pelajar→ajar
pe	NULL	NULL	pekerja→kerja

5. Aturan yang menangani sufiks derivasional

Tabel 2.5 Aturan sufiks derivasional

Sufiks	Pengganti	Kondisi Tambahan	Contoh
kan	NULL	prefiks ¢ {ke,peng}	tarikkan→tarik (meng)ambilkan → ambil
an	NULL	prefiks ¢ {di,meng,ter}	makanan→makan (per)janjian→janji
i	NULL	prefiks ¢ {ber,ke,peng}	tandai→tanda (men)dapati→dapat

UNIVERSITAS BRAWIJAYA

BAB III
METODOLOGI DAN PERANCANGAN

Metodologi yang digunakan pada skripsi ini adalah sebagai berikut :

1. Melakukan perancangan sistem untuk hasil yang diperoleh dari studi kepustakaan, dan untuk menguji hasil studi kepustakaan pada program
2. Implementasi pada bahasa pemrograman untuk menguji hasil perancangan yang dilakukan
3. Memasukkan data *training* yang bersumber dari *website* koran tertentu pada program
4. Memasukkan data *test* yang bersumber dari *website* tertentu pada program untuk menguji metode yang digunakan
5. Menganalisa hasil yang diperoleh.

3.1 Perancangan Sistem

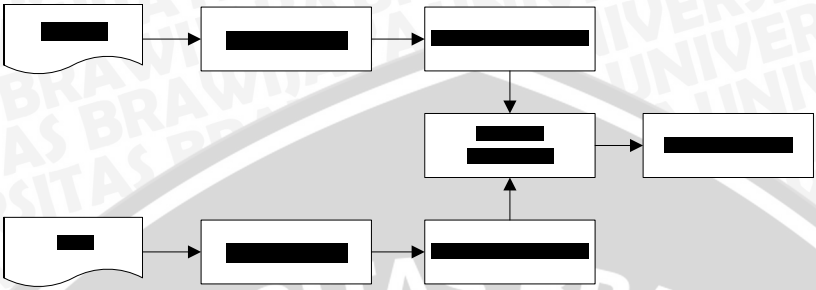
3.1.1 Deskripsi Umum Sistem

Sistem yang akan dibuat merupakan sistem untuk mengklasifikasi suatu berita berbahasa Indonesia dengan menggunakan metode *rocchio*. Hasil yang diperoleh dari sistem adalah kategori dari berita *test* yang diinputkan. Data yang dibutuhkan oleh sistem dibagi menjadi 2 kelompok, yaitu data *training* dan data *test*. Seluruh data diinputkan oleh user. Secara garis besar sistem memiliki beberapa proses, antara lain *preprocessing*, *features selection*, dan metode klasifikasi *rocchio*.

Proses sistem secara umum adalah sebagai berikut :

1. Proses pemasukan data yang dilakukan oleh user. User menentukan data *training* dan data *test*
2. Proses *preprocessing* oleh sistem, yang melakukan pengubahan semua huruf ke dalam huruf kecil (*case folding*) dan penghilangan semua tanda baca. Hasil dari *preprocessing* adalah data yang hanya berisi karakter alfabet (a-z) saja
3. Proses *features selection* yang melakukan *stemming* terhadap data *training* dan *test* dengan pemakaian kamus data
4. Proses klasifikasi dokumen dengan menggunakan metode *rocchio*
5. Hasil akhir adalah data *test* yang terklasifikasi pada kategori data *training* tertentu.

Blok gambar sistem dapat ditunjukkan sebagai berikut :



Gambar 3.1 Blok diagram proses sistem

3.1.2 Batasan Sistem

Batasan dari sistem yang dikembangkan adalah :

1. Sistem hanya mengklasifikasi berita berbahasa Indonesia
2. Data *training* tidak dapat bertambah karena proses penginputan seluruh data *training* dilakukan pada awal proses
3. Tidak dilakukan pemisahan struktur kalimat / *part of speech*
4. Proses *stemming* yang dilakukan adalah proses *stemming* sederhana dan tidak secara mendalam
5. Sistem tidak membedakan bobot judul atau bobot isi terhadap dokumen.

3.2 Proses Klasifikasi

Dalam proses klasifikasi terdapat 3 tahapan proses yang dilakukan, yaitu *preprocessing, features selection,* dan metode pembelajaran yang menggunakan metode klasifikasi *rocchio*.

3.2.1 Proses *Preprocessing*

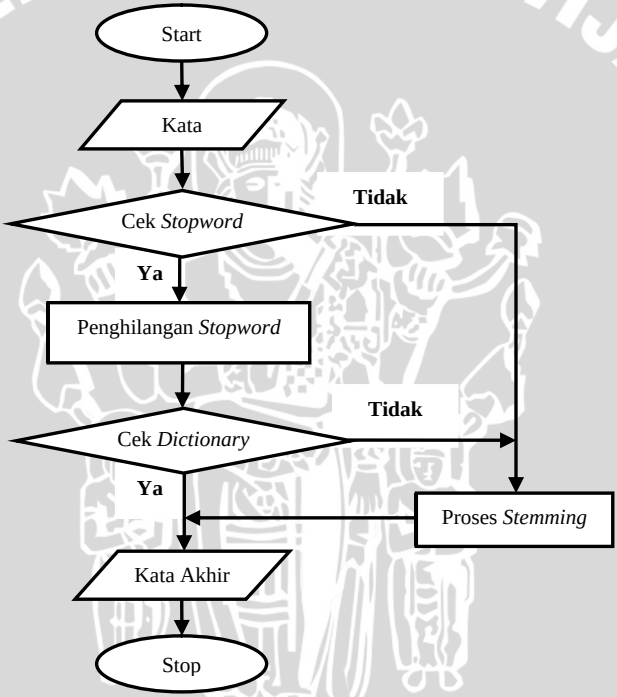
Pada tahapan *preprocessing*, dokumen-dokumen yang diinputkan dijadikan sebagai kumpulan kata dan pemisah kata. Definisi kata menurut Porter Stemmera (1980) adalah kumpulan huruf alfabet, sedangkan tanda baca, angka, dan karakter selain huruf alfabet dianggap sebagai *delimiter* atau pemisah antar kata.

Pada tahapan ini, terdapat proses *case folding* dan parsing sederhana. *Case folding* adalah pengubahan semua huruf ke dalam huruf kecil, sedangkan parsing sederhana memproses dokumen teks tanpa memperhatikan struktur kalimat (penentuan subyek, predikat, obyek, atau kata keterangan pada kalimat / *part of speech*), serta keterkaitan antar

kata, penghilangan kata sambung dan tanda baca. Hasil yang diperoleh dari tahapan ini adalah hanya berupa kumpulan kata.

3.2.2 Proses *Features Selection*

Pada tahapan *features selection* proses yang dilakukan adalah proses penghilangan *stopword* yang merupakan kata-kata yang tidak mempunyai peran penting dalam klasifikasi. Kemudian kata yang bukan merupakan *stopword* akan melewati proses *stemming*. *Stopword* yang digunakan dalam sistem ini merupakan daftar *stop list* yang dipakai oleh Tala (2003).



Gambar 3.2 *Flowchart* proses *features selection*

Proses *stemming* menggunakan sebuah kamus yang berisikan kumpulan kata berimbuhan yang merupakan kata dasar. Kata yang tidak termasuk dalam kamus akan melewati proses *stemming*, sedangkan kata yang termasuk dalam kamus tidak akan melewati proses *stemming*.

Sedangkan aturan *stemming* dilakukan secara sederhana dengan membagi penanganan terhadap partikel infleksional dan derivasional.

Proses-proses yang ada dalam proses *stemming* secara garis besar hanya dibagi menjadi 4 proses, yaitu proses sufiks inflekasional, proses peleburan derivasional, proses prefiks derivasional, dan proses sufiks derivasional.

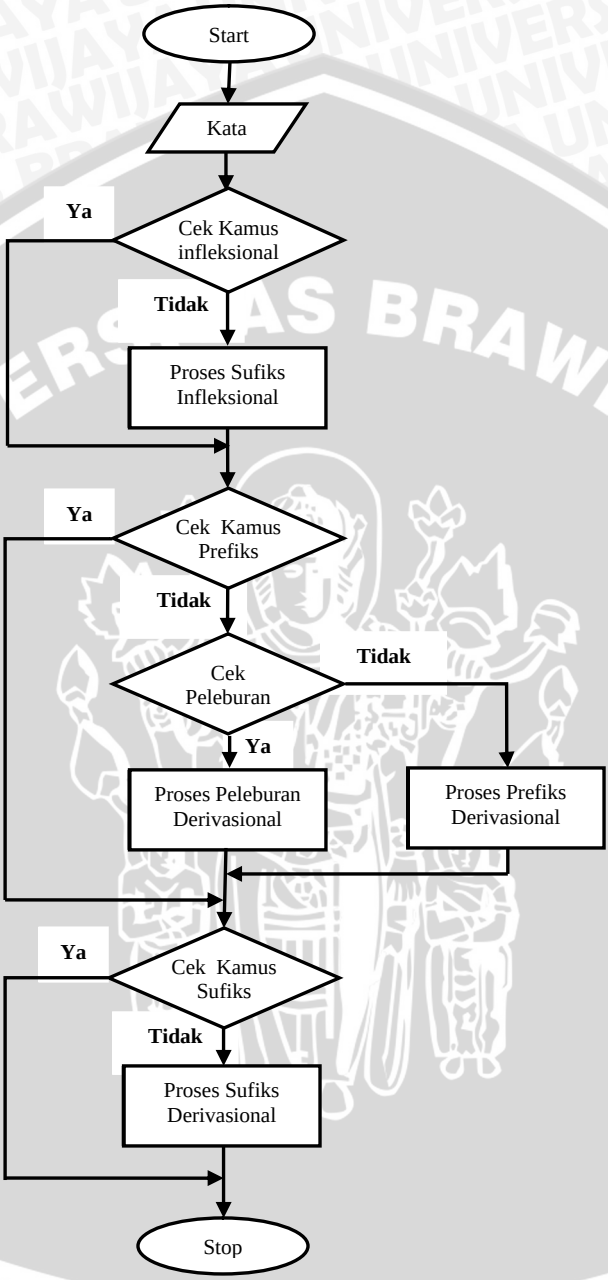
1. Proses sufiks infleksional menangani seluruh sufiks infleksional meliputi “-kah”, “-lah”, “-tah”, “-pun”, “-ku”, “-mu”, “-nya”.
2. Proses peleburan prefiks derivasional menangani prefiks derivasional yang mengalami peleburan, meliputi sebagai berikut:

Tabel 3.1 Aturan peleburan

NO	Prefiks	Kondisi	Lebur	Contoh
1	mem|pem	+ vokal	P	memaku→paku
2	meny|peny	+ vokal	S	menyapu→sapu
3	men|pen	+ vokal	T	menari→tari

3. Proses prefiks derivasional menangani penghilangan prefiks derivasional secara langsung tanpa adanya bentuk lebur.
4. Proses sufiks derivasional menangani penghilangan seluruh sufiks derivasional meliputi “-kan”, “-i” , “-an”.

Secara garis besar proses *stemming* bahasa Indonesia dapat digambarkan dengan diagram alir dibawah ini :



Gambar 3.3 *Flowchart* proses *stemming*

3.2.3 Proses Metode Pembelajaran

Seperti yang telah dijelaskan pada Bab 2, bahwa metode pembelajaran yang digunakan adalah *rocchio classifiers*. Metode klasifikasi *rocchio* mempunyai 3 tahapan pokok, yaitu :

1. Menghitung bobot setiap kata menggunakan skema pembobotan *TFIDF* dan merepresentasikan seluruh dokumen ke dalam vektor dengan kata sebagai dimensi vektor
2. Merepresentasikan prototipe vektor setiap kelas, dengan menjumlahkan seluruh vektor dokumen *training* dalam kelas tersebut sehingga menghasilkan sebuah prototipe vektor
3. Membandingkan kedekatan sudut cosine antara vektor data *test* dengan prototipe vektor. Data *test* dikategorikan pada kelas tertentu jika sudut cosine yang dibentuk oleh vektor dokumen *test* dan salah satu prototipe vektor mempunyai nilai paling besar atau paling mendekati angka 1.

3.3 Perancangan Database

3.3.1 Perancangan Tabel

Tabel-tabel yang dibutuhkan dalam sistem ini adalah tabel news, news_part, news_category, rank, stopword, dan tabel dictionary.

1. Tabel news

Tabel news menyimpan berita yang akan diproses dalam proses klasifikasi yang berisi kode berita, kategori dari dokumen, isi dari dokumen.

Tabel 3.2 Atribut tabel news

Field	Tipe	Null	Default
id	int(10)	No	
cat_id	int(10)	No	
content	text	No	

Keterangan :

id : kode berita
id_cat : kode kategori
content : isi dari berita.

2. Tabel news_category
   Tabel news_category berisi semua kategori atau kelas dari berita yang ada pada data *training*. Tabel ini berisi kode kategori dan nama kategori.

Tabel 3.3 Atribut tabel news_category

Field	Tipe	Null	Default
id	int(10)	No	
cat_name	varchar(50)	No	

Keterangan :
id : kode kategori
cat_name : nama kategori.

3. Tabel news_part
   Tabel news_part merupakan tabel yang menyimpan seluruh kata dari proses *features selection*. Tabel ini mempunyai atribut kode, kode berita, kata, dan jumlah kata.

Tabel 3.4 Atribut tabel news_part

Field	Tipe	Null	Default
id	int(10)	No	
news_id	int(10)	No	
token	varchar(50)	No	
counter	varchar(10)	No	

Keterangan :
id : kode news_part
news_id : kode berita
token : kata
counter : jumlah kata.

4. Tabel stopword
   Tabel stopword berisi kata-kata yang merupakan *stopword*, yang mempunyai atribut kode dan kata.

Tabel 3.5 Atribut tabel stopword

Field	Tipe	Null	Default
id	int(10)	No	
token	varchar(50)	No	

Keterangan :

id : kode stopword
token : kata yang merupakan *stopword*.

5. Tabel dictionary
   Tabel dictionary merupakan kata khusus yang tidak melewati proses *stemming*, yang mempunyai atribut kode dan kata

Tabel 3.6 Atribut tabel dictionary

Field	Tipe	Null	Default
id	int(10)	No	
token	varchar(50)	No	

Keterangan :

id : kode dictionary
token : kata khusus yang tidak melewati proses *stemming*.

6. Tabel rank
   Tabel rank berisikan rangking hasil penghitungan metode klasifikasi data *test* untuk setiap kategori.

Tabel 3.7 Atribut tabel rank

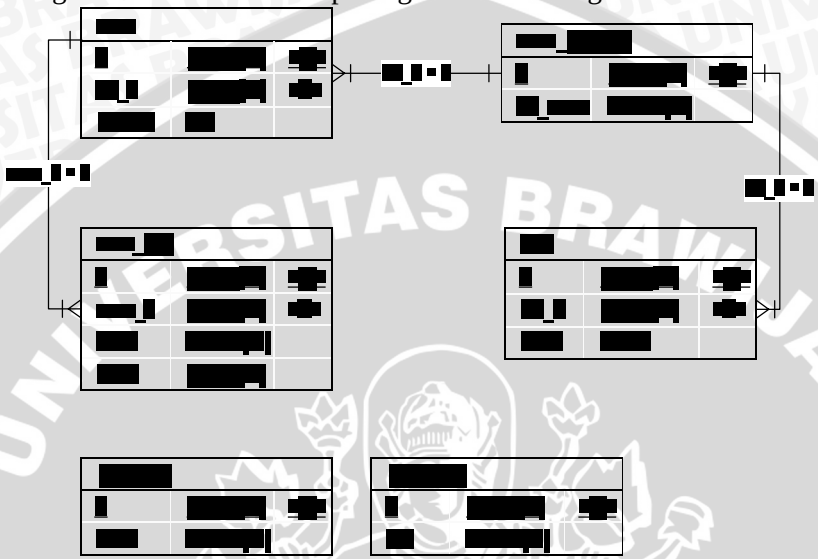
Field	Tipe	Null	Default
id	int(10)	No	
cat_id	int(10)	No	
result	double	No	

Keterangan :

id : kode dari stopword
token : kata yang merupakan stopword.

3.3.2 ERD

Diagram relasi *database* dapat digambarkan sebagai berikut:



Gambar 3.4 ERD

3.4 Implementasi Metode *Rocchio*

Sebagai contoh implementasi terdapat 3 dokumen *training* dalam *database* yang telah terklasifikasi dan setelah dilakukan *text preprocessing* dan *features selection* terdapat 3 *good features* yaitu kata "musik", kata "politik", dan kata "tenis".

Sebagai contoh, dicoba dokumen *(d')* sebagai data *test* yang mempunyai 3 *good features* atau *term frequency (TF)* sebagai berikut:

Tabel 3.8 Daftar kata

Kata *(w_i)*	***TF(w_i, d)***			
	d_1	***d_2***	***d_3***	***d'***
Musik	3	5	0	1
Politik	1	1	3	1
Tenis	0	2	5	3

Keterangan :

1. d_1, d_2 , d_3 merupakan data *training*. Sesuai dengan kategori yang diperoleh pada sumber. d_1 dan d_2 merupakan kategori musik sedangkan d_3 merupakan kategori sport.
2. *d'* merupakan data *test* yang belum memiliki kategori.
3. Kata (w_i) merupakan kata ke-*i* yang ada dalam data *training* atau data *test*. Dalam contoh diatas hanya ditemukan hanya 3 kata berbeda dalam seluruh data *training* dan data *test*, yaitu kata "musik", "politik", dan "tenis".
4. *TF (term frequency)* merupakan jumlah kata (w_i) dalam setiap dokumen *(d)*.

3.4.1 Representasi Dokumen

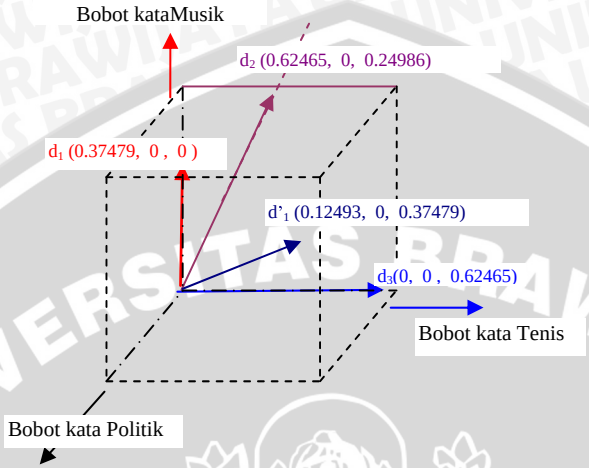
Setelah dilakukan pembobotan *TFIDF*, ketiga data *training* (d_1, d_2, d_3) dan satu data *test (d')* tersebut direpresentasikan ke dalam vektor, seperti yang ditampilkan pada tabel 3.9 dibawah ini :

Tabel 3.9 Daftar bobot kata

Kata (w_i)	$TF(w_i, d)$				$IDF(w_i)$			
	d_1	d_2	d_3	d'	d_1	d_2	d_3	d'
Musik	3	5	0	1	0.12493	0.12493	0.12493	0.12493
Politik	1	1	3	1	0	0	0	0
Tenis	0	2	5	3	0.12493	0.12493	0.12493	0.12493

Kata (w_i)	$TF(w_i, d)$ $IDF(w_i)$			
	d_1	d_2	d_3	d'
Musik	0.37479	0.62465	0	0.12493
Politik	0	0	0	0
Tenis	0	0.24986	0.62465	0.37479

Berdasarkan daftar bobot kata pada tabel diatas, maka representasi dokumen dapat digambarkan sebagai berikut :



Gambar 3.5 Representasi dokumen

Data 1 dan data 2 diklasifikasi sebagai kategori musik, dan data 3 diklasifikasikan sebagai kategori sport.

3.4.2 Membangun Prototipe Vektor

3.4.2.1 Prototipe Vektor Musik ($\overrightarrow{c_{musik}}$)

Menjumlahkan seluruh vektor dokumen yang termasuk kelas musik

$$\overrightarrow{c_{musik}} = \overrightarrow{d_1} + \overrightarrow{d_2}$$

$$= [0.37479 \quad 0 \quad 0] + [0.62465 \quad 0 \quad 0.24986]$$

$$= [0.99944 \quad 0 \quad 0.24986]$$

$$\overrightarrow{c_{musik}} = [0.99944 \quad 0 \quad 0.24986]$$

$$\sqrt{\sum_{k=1}^{|F|} (c_{musik}^{(k)})^2} = (0.99944)^2 + (0) + (0.24986)^2)^{1/2}$$

$$= \sqrt{(0.99944 * 0.99944) + (0.24986 * 0.24986)}$$

$$= \sqrt{0.99888 + 0.06243} = \sqrt{1.06131}$$

$$= 1.03029$$

3.4.2.2 Prototipe Vektor Sport ($\overrightarrow{c\_sport}$)

Menjumlahkan seluruh vektor dokumen yang termasuk kelas sport.

$\overrightarrow{c_{sport}} = \vec{d}_3$

$\overrightarrow{c_{sport}} = [0 \quad 0 \quad 0.62465]$

$$\sqrt{\sum_{k=1}^{|F|}(c_{sport}^{(k)})^2} = \sqrt{0+0+(0.62465)^2}$$
$$= \sqrt{0.62465 * 0.62465}$$
$$= \sqrt{0.39019}$$
$$= 0.62465$$

3.4.3 Data *Test*

$\vec{d}'$ = [0.12493 0 0.37479]

$$\sqrt{\sum_{k=1}^{|F|}(d'^{(k)})^2} = \sqrt{(0.12493)^2+0+(0.37479)^2}$$
$$= \sqrt{(0.12493 * 0.12493)+(0.37479 * 0.37479)}$$
$$= \sqrt{0.0156+0.14047}$$
$$= \sqrt{0.15607} = 0.39506$$

3.4.4 Menghitung Kedekatan Sudut

Setelah penghitungan prototipe vektor, maka dihitung kedekatan sudut antara setiap prototipe vektor dengan data *test*.

$$sim(d', c_j) = \frac{\sum_{k=1}^{|F|} c_j^{(k)} \cdot d'^{(k)}}{\sqrt{\sum_{k=1}^{|F|}(c_j^{(k)})^2}\sqrt{\sum_{k=1}^{|F|}(d'^{(k)})^2}}$$

3.4.4.1 Kedekatan Data *Test* Dan Kategori Musik

$$sim(d', c_{musik}) = \frac{\sum_{k=1}^{|F|} c_{musik}^{(k)} \cdot d'^{(k)}}{\sqrt{\sum_{k=1}^{|F|} (c_{musik}^{(k)})^2} \sqrt{\sum_{k=1}^{|F|} (d'^{(k)})^2}}$$

$$= \frac{(0.99944 * 0.12493) + (0 * 0) + (0.24986 * 0.37479)}{0.03029 * 0.39506}$$

$$= \frac{(0.12486) + (0.09364)}{0.40702}$$

$$= \frac{0.2185}{0.40702} = 0.536828$$

3.4.4.2 Kedekatan Data *Test* Dan Kategori Sport

$$sim(d', c_{sport}) = \frac{\sum_{k=1}^{|F|} c_{sport}^{(k)} \cdot d'^{(k)}}{\sqrt{\sum_{k=1}^{|F|} (c_{sport}^{(k)})^2} \sqrt{\sum_{k=1}^{|F|} (d'^{(k)})^2}}$$

$$= \frac{(0 * 0.12493) + (0 * 0) + (0.62465 * 0.37479)}{0.62465 * 0.39506}$$

$$= \frac{0.234112}{0.24677} = 0.9487$$

3.4.5 Rangking Kedekatan

Rangking 1 : kelas sport = 0.9487
Rangking 2 : kelas musik = 0.536828

Dengan demikian, dokumen *d'* diklasifikasikan ke dalam kategori sport.

UNIVERSITAS BRAWIJAYA

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi

4.1.1 Spesifikasi Komputer

Spesifikasi komputer sebagai alat uji untuk memproses pembuatan aplikasi adalah sebagai berikut:

- Processor Intel Centrino Duo
- Memori 512 MB
- Hardisk 120 GB
- Sistem Operasi Windows XP Professional SP2
- Web Server Apache versi 2.2.3
- Database Server MySQL versi 5.0.27.

4.1.2 Persiapan Data

Data *training* dan data *test* berasal dari *website* koran Kompas (*www.kompas.com*) bulan November 2006 sampai dengan Januari 2007. Jumlah keseluruhan data adalah 945 dengan 9 kategori berita, 679 data *training* dan 315 data *test*. Dengan perincian sebagai berikut:

1. Kategori dikbud
   Kategori dikbud memiliki 81 data *training* dan 34 data *test*. Berisikan berita bidang pendidikan dan kebudayaan.
2. Kategori ekonomi
   Kategori ekonomi memiliki 72 data *training* dan 31 data *test*. Berisikan berita bidang bisnis, keuangan, pertumbuhan ekonomi, perekonomian.
3. Kategori hiburan
   Kategori hiburan memiliki 65 data *training* dan 35 data *test* . Berisikan berita mengenai band, sinetron, film, musik, gosip artis.
4. Kategori internasional
   Kategori internasional memiliki 78 data *training* dan 36 data *test*. Berisikan berita hubungan dua negara, berita internasional.
5. Kategori iptek
   Kategori iptek memiliki 76 data *training* dan 35 data *test*. Kategori inptek berisi berita tentang iptek.
6. Kategori kesehatan

Kategori iptek memiliki 74 data *training* dan 35 data *test*. Kategori kesehatan berisikan berita kesehatan masyarakat, penyakit.

7. Kategori metropolitan
   Kategori metropolitan memiliki 72 data *training* dan 35 data *test*. Kategori metropolitan berisikan berita kriminal, berita seputar kehidupan metropolitan.
8. Kategori nasional
   Kategori nasional memiliki 72 data *training* dan 39 data *test*. Berisikan berita tentang infrastruktur dan pembangunan.
9. Kategori olahraga
   Kategori olahraga memiliki 86 data *training* dan 35 data *test*. Berisikan berita olahraga.

Seluruh kategori di atas berdasarkan kategori yang telah ada pada data *training*.

4.1.3 Implementasi *Database*

Tabel yang dirancang untuk pembuatan klasifikasi adalah tabel news, tabel news_part, tabel news_category, tabel rank, tabel stopword dan tabel dictionary.

Tabel news adalah tabel untuk menyimpan seluruh berita yang digunakan pada klasifikasi. Kode dibawah ini merupakan *query* yang bertujuan membuat tabel news.

```
CREATE TABEL `news` (
  `id` int(10) NOT NULL auto_increment,
  `cat_id` int(10) NOT NULL,
  `text` text collate latin1_general_ci NOT NULL,
  PRIMARY KEY  (`id`)
)
```

Tabel news_part merupakan tabel yang berisi seluruh kata setelah dilakukan tahap *preprocessing* dan *features selection.*

```
CREATE TABEL `news_part` (
  `id` int(10) NOT NULL auto_increment,
  `news_id` int(10) NOT NULL,
  `token` varchar(50) collate latin1_general_ci NOT
NULL,
  `counter` int(10) NOT NULL,
  PRIMARY KEY  (`id`),
  KEY `string` (`token`)
)
```

Tabel news_category berisi kategori dari data *training*.

```
CREATE TABEL `news_category` (
  `id` int(10) NOT NULL auto_increment,
  `cat_name` varchar(50) collate latin1_general_ci NOT
NULL,
  PRIMARY KEY  (`id`)
)
```

Tabel rank merupakan tabel untuk menyimpan hasil dari klasifikasi. Kode dibawah ini merupakan *query* untuk membuat tabel rank.

```
CREATE TABEL `rank` (
  `id` int(10) NOT NULL auto_increment,
  `cat_id` int(10) NOT NULL,
  `result` double NOT NULL,
  PRIMARY KEY  (`id`)
)
```

Tabel stopword merupakan tabel untuk menyimpan kata yang merupakan *stopword*. Kode dibawah ini merupakan *query* untuk membuat tabel stopword.

```
CREATE TABEL `stopwords` (
  `id` int(10) NOT NULL auto_increment,
  `token` varchar(255) collate latin1_general_ci NOT
NULL,
  PRIMARY KEY  (`id`),
  UNIQUE KEY `token` (`token`)
)
```

Tabel dictionary merupakan tabel untuk menyimpan kata yang termasuk *dictionary*. Kode dibawah ini merupakan *query* untuk membuat tabel dictionary.

```
CREATE TABEL `dictionary` (
  `id` int(10) NOT NULL auto_increment,
  `token` varchar(255) collate latin1_general_ci NOT
NULL,
  PRIMARY KEY  (`id`),
  UNIQUE KEY `token` (`token`)
)
```

4.1.4 Deskripsi Program

Proses yang dilakukan antara lain *preprocessing*, *filterstring*, *stemming*, dan *do_rocchio.*

4.1.4.1 *Preprocessing*

Pada proses *preprocessing* berita akan mengalami *case folding* dan penghilangan seluruh tanda baca, yang kemudian dilakukan dengan proses *filterstring*.

```
function preprocessing()
{
    $query = mysql_query("SELECT * FROM news");

    while($result = mysql_fetch_array($query))
    {
           $news_content = $result['text'];
           $news_content = strtolower($news_content);
           $separators = "
1234567890,.:;?!@#$%^&*”“’‘`".'"'."(~-+=–_\/|)\n\t";

           for($token = strtok($news_content,
$separators); $token !== FALSE; $token =
strtok($separators))
           {
                  $token  = trim($token);
                  $token  = filterstring($token);
                  if(strlen($token)>2)
                  {
                         $the_chosen[] =
$token.','.$result['id'];
                  }
           }
    }
    $token_counts = array_count_values($the_chosen);
    foreach ($token_counts as $token => $num)
    {
           $part = explode(",", $token);
           $query = mysql_query("
                  INSERT INTO news_part (news_id, token,
counter)
                  VALUES ('$part[1]','$part[0]','$num')
           ");
    }
}
```

4.1.4.2 *Filterstring*

Pada proses *filterstring* dilakukan penghilangan *stopword* dan pengecekan kata pada *dictionary*. Kata yang termasuk *stopword* tidak dipakai, kemudian kata yang bukan termasuk *dictionary* dilakukan proses *stemming*.

```
function filterstring($string)
{
    global $stopword;
    global $dict;

    if(!in_array($string,
$stopword)||!check_in_array($stopword, $string))
    {
          if(in_array($string, $dict))
          {
                $final_token = $string;
          }
          elseif($token = check_in_array($dict,
$string))
          {
                $final_token = $token;
          }
          else
          {
                $string = stemming($string, $dict);


                if (!in_array($string,
$stopword)||!check_in_array($stopword, $string))
                      $final_token = $string;
          }
    }
    return $final_token;
}
```

4.1.4.3 *Stemming*

Proses *stemming* bertujuan untuk mendapatkan kata dasar dengan menghilangkan prefiks dan sufiks. Proses *stemming* secara garis besar dibagi menjadi 4 proses yaitu penanganan sufiks infleksional, penanganan prefiks derivasional lebur, penanganan prefiks derivasional dan penanganan sufiks derivasional.

```
function stemming($value, $dictionary)
{
   if(check_in_array($dictionary, $value))
   {
       $token1 = $value ;
   }
   else if(preg_match("/^(.*)(kah|lah|tah|pun|ku|mu|nya)$/",
$value, $match))
   {
       $token1 = $match[1];
   }
   else
   {
       $token1 = $value;
   }
   if(check_in_array($dictionary, $token1))
   {
       $token2 = $tok;
   }
   else if
(preg_match("/^(mem|pem|meny|peny|men|pen|pel)([aiueo])(.*)$
/",$token1))
   {
       if (preg_match("/^(mem|pem)([aiueo])(.*)$/",$token1))
       {
   $token2=ereg_replace("^(mem|pem)([aiueo])(.*)$","p\\2\\3",
$token1);
       }
       else if
(preg_match("/^(meny|peny)([aiueo])(.*)$/",$token1))
       {
   $token2=ereg_replace("^(meny|peny)([aiueo])(.*)$","s\\2\\3
",$token1);
       }
       else if
(preg_match("/^(men|pen)([aiueo])(.*)$/",$token1))
       {
   $token2=ereg_replace("^(men|pen)([aiueo])(.*)$","t\\2\\3",
$token1);
       }
       else if (preg_match("/^(pel)([aiueo])(.*)$/",$token1))
       {
   $token2=ereg_replace("^(pel)([aiueo])(.*)$","l\\2\\3",$tok
en1);
       }
       else
       {
              return $token2=$token1;
```

```
        }
    }
    else if
(preg_match("/^(meng|peng|memper|diper)(.*)$/",$token1,$matc
h))
    {
        $token2 = $match[2];
    }
    else if
(preg_match("/^(men|mem|pen|pem|pel|per|ter|ber|bel|di)(.*)$
/",$token1,$match))
    {
        $token2 = $match[2];
    }
    else if
(preg_match("/^(pe|me|be|ke)(.*)$/",$token1,$match))
    {
        $token2 = $match[2];
    }
    else
    {
        $token2 = $token1;
    }

    if(check_in_array($dictionary, $token2))
    {
        $token3 = $token2;
    }
    elseif(preg_match('/(kan|i)$/', trim($token2)))
    {
        $token = ereg_replace("^(.*)(kan|i)", "\\1", $token2);
        $token3 = $token;
    }
    elseif(preg_match('/(an)$/', trim($token2)))
    {
        $token = ereg_replace("^(.*)(an)", "\\1", $token2);
        $token3 = $token;
    }
    else
        $token3 = $token2;
    return $token3;
}
```

4.1.4.5 Metode Pembelajaran

Implementasi *TFIDF classifiers* ditunjukkan dengan kode program sebagai berikut :

```
$query_d = mysql_query("
          SELECT SQRT(SUM(POW(t,2)))
          FROM
                    (SELECT
                              SUM((LOG10(((SELECT COUNT(*)
FROM news) / (SELECT COUNT(*) FROM news_part b WHERE
(a.token = b.token)))) * a.counter)) AS t
                    FROM news_part a, news c
                    WHERE a.news_id=c.id AND c.cat_id=0
                    GROUP BY a.token) w
          ");

   $d = mysql_fetch_row($query_d);
```

Pada kode diatas, dihitung panjang vektor untuk data *test* seperti yang telah ditunjukkan pada sub bab 3.4.2.3. Untuk setiap kategori dihitung perkalian antara vektor data *test* dan prototipe vektor yang ditunjukkan pada kode program di bawah ini :

```
$query1 = mysql_query("SELECT id FROM news_category");

while($result1 = mysql_fetch_row($query1))
{
    $query2 = mysql_query("
          SELECT SUM(t1*t2)
          FROM
(SELECT
                    token, SUM((LOG10(((SELECT COUNT(*)
FROM news) /
                    (SELECT COUNT(*) FROM news_part b WHERE
(a.token = b.token)))) * a.counter)) as t1
                    FROM news_part a, news c
                    WHERE a.news_id=c.id AND
c.cat_id=$result1[0]
                    GROUP BY a.token) w1
                    ,
                    (SELECT
                    token, SUM((LOG10(((SELECT COUNT(*)
FROM news) /
                    (SELECT COUNT(*) FROM news_part b WHERE
(a.token = b.token)))) * a.counter)) as t2
                    FROM news_part a, news c
                    WHERE a.news_id=c.id AND c.cat_id=0
                    GROUP BY a.token) w2
          WHERE w1.token=w2.token
");
```

```
$value2 = mysql_fetch_row($query2);
```

Selanjutnya dihitung panjang vektor untuk setiap kategori atau disebut dengan prototipe vektor melalui kode program di bawah ini :

```
$query3 = mysql_query("
    SELECT SQRT(SUM(POW(t,2)))
    FROM
          (SELECT
                SUM((LOG10(((SELECT COUNT(*) FROM news)
/ (SELECT COUNT(*) FROM news_part b WHERE (a.token =
b.token)))) * a.counter)) AS t
          FROM news_part a, news c
          WHERE a.news_id=c.id AND c.cat_id=$result1[0]
          GROUP BY a.token) w
");
$value3 = mysql_fetch_row($query3);
```

Secara garis besar, *TFIDF classifiers* ditunjukkan pada kode program sebagai berikut :

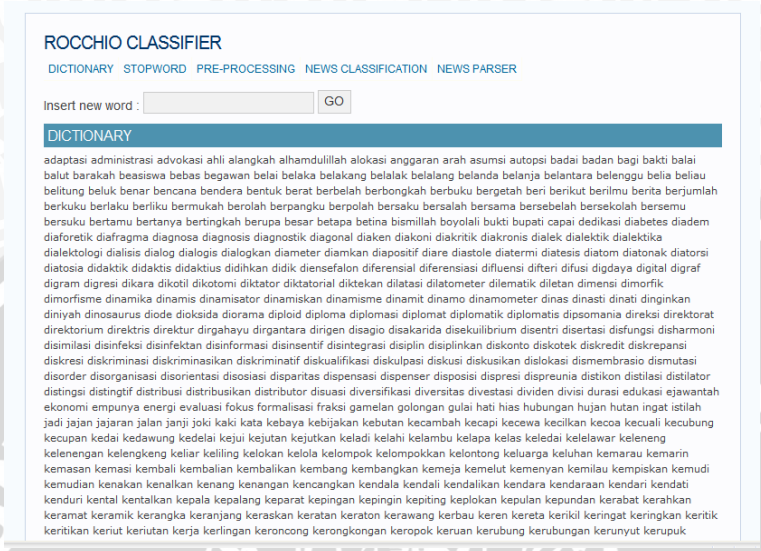
```
$cos[]    =    $value2[0]/($value3[0]*$d[0])    .    ","
.$result1[0];
```

4.2 Penerapan Aplikasi

Tampilan aplikasi yang dibuat terdiri dari halaman dictionary, halaman stopword, halaman preprocessing, halaman news parser yang merupakan pengecekan terhadap data training, dan halaman news classification yang merupakan halaman proses klasifikasi.

4.2.1 Halaman Dictionary

Halaman dictionary merupakan halaman untuk menampilkan, menghapus serta menambah tabel dictionary. Pada gambar 4.1 ditampilkan halaman dictionary dengan jumlah kata 1106 kata. Pada halaman dictionary terdapat form input untuk memasukkan kata ke dalam tabel dictionary, penghapusan kata dilakukan dengan cara memilih kata yang akan dihapus secara langsung.



Gambar 4.1 Halaman dictionary

4.2.2 Halaman Stopword

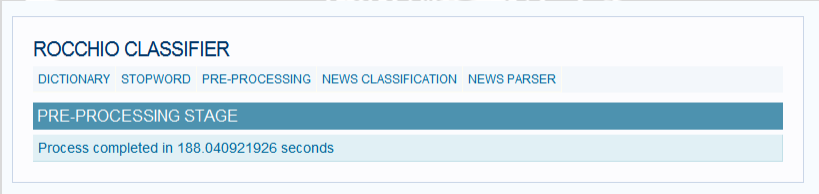
Halaman stopword merupakan halaman untuk menampilkan, menghapus serta menambah tabel stopword.

Gambar 4.2 Halaman stopword

Pada gambar 4.2 ditampilkan halaman stopword dengan jumlah kata 455 kata. Pada halaman stopword terdapat form input untuk memasukkan kata ke dalam tabel stopword, penghapusan kata dilakukan dengan cara memilih kata yang akan dihapus secara langsung.

4.2.3 Halaman Preprocessing

Halaman preprocessing merupakan tahapan awal untuk melakukan klasifikasi, yaitu dengan menjalankan proses *scan_dir, preprocessing, filterstring,* dan *stemming* terhadap data *training*. Proses *scan_dir* bertujuan membaca file teks data *training* dan memasukkan seluruh file teks data *training* ke dalam *database* beserta kategorinya. Halaman preprocessing hanya menyediakan satu tombol yaitu ”insert & parse all news”, yang menjalankan proses *preprocessing* terhadap seluruh data *training* secara keseluruhan.



Gambar 4.3 Halaman preprocessing

Hasil yang diperoleh dari *preprocessing* data *training* adalah waktu komputasi dari proses *preprocessing* data *training* secara keseluruhan.

4.2.4 Halaman News Parser

Halaman news parser dibuat untuk melakukan pengecekan pada data *training* secara individual. Pengecekan dilakukan dengan memasukkan kode berita data *training* pada *form* yang tersedia.

Gambar 4.4 Halaman news parser

Setelah kode berita telah diisi dan dijalankan, contoh hasil yang ditunjukkan adalah sebagai berikut :

Gambar 4.5 Hasil parsing

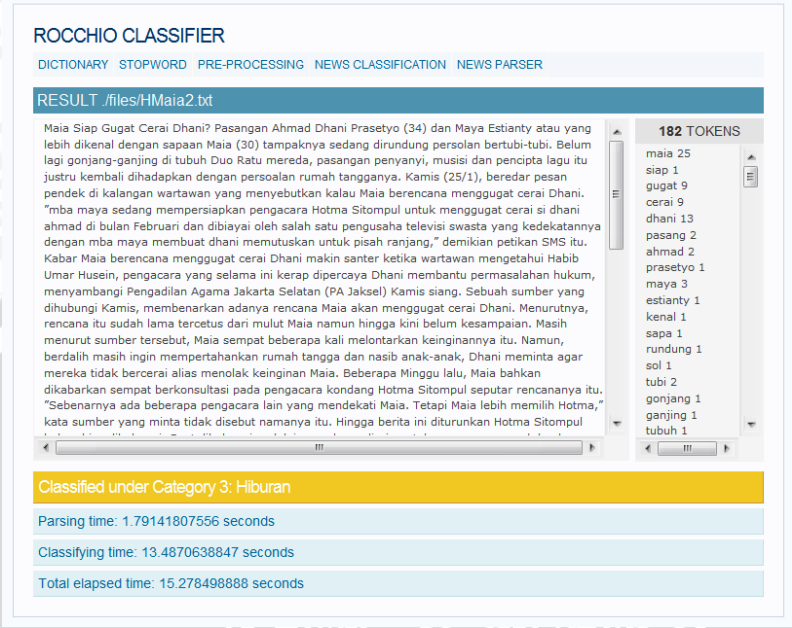
Hasil yang diperoleh adalah berupa daftar kata yang telah melewati proses *preprocessing*, *filterstring*, dan *stemming* dengan menunjukkan waktu komputasi dalam melakukan ketiga proses tersebut.

4.2.4 Halaman News Classification

Halaman news classification merupakan halaman untuk melakukan klasifikasi pada data *test* yang ditentukan oleh *user*. Proses yang dilakukan merupakan implementasi metode klasifikasi *rocchio*. Pada halaman ini disediakan *form* untuk memilih *file* yang akan diklasifikasi.

Gambar 4.6 Halaman news classification

Setelah data *test* telah dipilih dari *file* teks dan dijalankan, maka hasil yang diperoleh adalah sebagai berikut :



Gambar 4.7 Hasil klasifikasi

Hasil yang diperoleh adalah kategori data *test* berserta waktu yang dibutuhkan beserta rangking data *test* pada tiap kategori.

4.3 Analisa Hasil

4.3.1 Efektifitas

Hasil yang diperoleh dari keseluruhan proses klasifikasi adalah data *test* yang terklasifikasi pada kategori tertentu. Setiap kategori dihitung peluang nilai kebenaran. Pengukuran efektifitas dilakukan sebanyak 2 kali dengan jumlah data *training* yang berbeda. Jumlah data *training* yang digunakan pada pengujian efektifitas pertama adalah 100 persen, dan 50 persen pada pengujian efektifitas kedua dari jumlah total keseluruhan data *training*.

Berikut ini merupakan tabel hasil pengujian efektifitas pertama dengan menggunakan 679 data *training* dan 315 data *test*.

Tabel 4.1 Hasil uji efektivitas 1

No	Kategori	Jumlah	Benar	$P(c_i)$
1	Dikbud	34	28	0,8235
2	Ekonomi	31	30	0,9677
3	Hiburan	35	33	0,9429
4	Internasional	36	25	0,6945
5	IPTEK	35	29	0,8286
6	Kesehatan	35	30	0,8572
7	Metropolitan	35	29	0,8286
8	Nasional	39	36	0,9231
9	Olah raga	35	34	0,9714
Rata – rata				**0,8703**

Nilai kesalahan yang terbanyak dimiliki oleh kategori internasional dan nilai kesalahan terkecil dimiliki oleh kategori olah raga. Pengujian efektifitas kedua menggunakan 333 data *training* dan 315 data *test*. Dibawah ini merupakan tabel hasil pengujian efektifitas kedua :

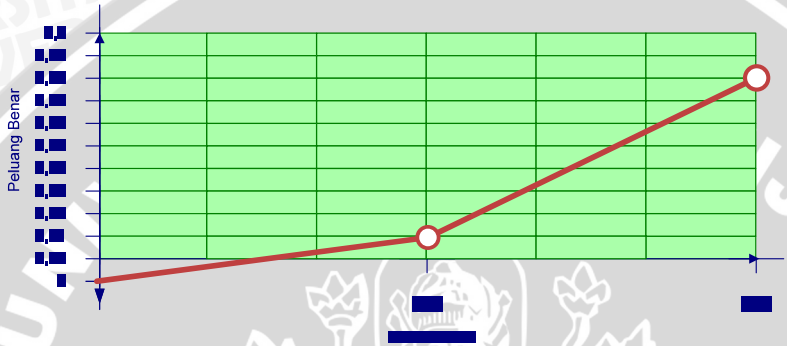
Tabel 4.2 Hasil uji efektivitas 2

No	Kategori	Jumlah	Benar	$P(c_i)$
1	Dikbud	34	28	0,8235
2	Ekonomi	31	30	0,9677
3	Hiburan	35	30	0,8572
4	Internasional	36	25	0,6945
5	IPTEK	35	27	0,7714
6	Kesehatan	35	29	0,8286
7	Metropolitan	35	24	0,6857
8	Nasional	39	20	0,6897
9	Olah raga	35	34	0,9714
Rata - rata				**0,81**

Dari kedua hasil pengujian efektifitas tersebut, maka dapat dibuktikan bahwa jumlah data *training* mempengaruhi efektifitas, salah satunya dibuktikan dengan semakin banyaknya kesalahan saat jumlah data training dikurangi menjadi 333 data training. Isi dari data training juga sangat mempengaruhi efektifitas, hal tersebut ditunjukkan pada kategori metropolitan dan nasional yang mempunyai perbedaan penurunan yang

jauh dari pada kategori yang lain. Penurunan ini disebabkan isi dari data training kategori internasional, metropolitan dan nasional mempunyai banyak kesamaan isi. Secara keseluruhan hubungan antara efektifitas dan jumlah data training dapat digambarkan sebagai berikut :

Gambar 4.8 Rata-rata Efektifitas



4.3.2 Efisiensi

Pengujian efisiensi dilakukan sebanyak 2 kali dengan jumlah data *training* yang berbeda. Jumlah data *training* yang digunakan pada pengujian efisiensi pertama adalah 100 persen, dan 50 persen pada pengujian efisiensi yang kedua dari jumlah total keseluruhan data *training*.

Waktu yang dibutuhkan untuk memproses 679 data training adalah 175,056 detik, sedangkan waktu yang dibutuhkan untuk memproses 333 data training adalah 94,449 detik.

Berikut ini merupakan tabel hasil pengujian efisiensi pertama dengan menggunakan 679 data *training*, dengan data test yang dipilih secara.

Tabel 4.3 Hasil uji efisiensi 1

Uji Coba	Nama File	Kategori	Waktu Komputasi
1	DBuku	Dikbud	9,694 detik
2	Eadidas	Ekonomi	9,889 detik
3	HAktor	Hiburan	9,901 detik
4	InAs	Internasional	10,553 detik
5	IKura	IPTEK	10,618 detik
6	KAlkohol	Kesehatan	10,358 detik

7	MUnggas2	Metropolitan	10,018 detik
8	NKisah	Nasional	10,529 detik
9	ODua	Olahraga	10,361detik
Rata-rata			**10,213 detik**

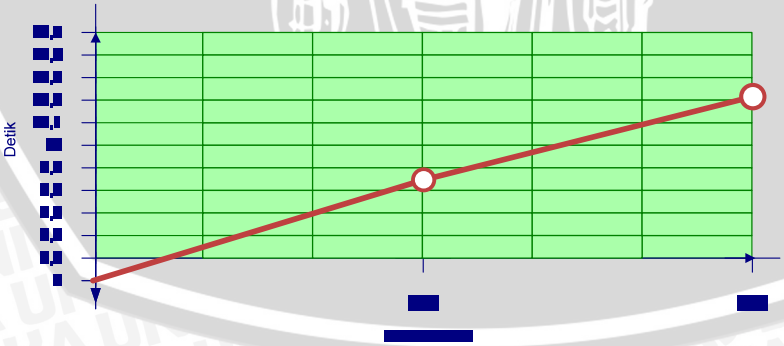
Pengujian efisiensi kedua menggunakan 333 data *training*, dengan data *test* yang dipilih secara acak. Hasil pengujian efisiensi kedua tersebut dituliskan dalam tabel sebagai berikut :

Tabel 4.4 Hasil uji efisiensi 2

Uji Coba	Nama File	Kategori	Waktu Komputasi
1	DBuku	Dikbud	9,808 detik
2	Eadidas	Ekonomi	9,854 detik
3	HAktor	Hiburan	9,578 detik
4	InAs	Internasional	10,150 detik
5	IKura	IPTEK	9,829 detik
6	KAlkohol	Kesehatan	9,921 detik
7	MUnggas2	Metropolitan	9,668 detik
8	NKisah	Nasional	10,065 detik
9	ODua	Olahraga	9,812 detik
Rata-rata			**9,854 detik**

Dari hasil uji efisiensi diatas maka dapat disimpulkan bahwa semakin besar jumlah data *training* maka semakin lama waktu yang dibutuhkan untuk memproses data *training* begitu pula dengan waktu untuk mengklasifikasi data *test*.

Gambar 4.9 Rata-rata Efisiensi



BAB V
PENUTUP

5.1 Kesimpulan

Kesimpulan yang didapat selama pengerjaan skripsi ini adalah sebagai berikut :

1. Klasifikasi dokumen teks berbahasa Indonesia menggunakan metode *rocchio* dengan menggunakan 679 data *training* dan 315 data *test* menghasilkan rata-rata efektifitas sebesar 0,8703 (87%), dan rata-rata efisiensi sebesar 10,231 detik.
2. Efektifitas sistem semakin meningkat dengan meningkatnya jumlah data *training* yang digunakan dalam pembelajaran, sebaliknya efisiensi sistem semakin menurun dengan meningkatnya jumlah data *training* dalam pembelajaran.
3. Isi dari setiap data *training* mempengaruhi efektifitas sistem.

5.2 Saran

Beberapa saran pengembangan lebih lanjut yang dapat diberikan oleh penulis adalah sebagai berikut :

1. Perlu dilakukan proses *stemming* yang lebih mendetail atau kompleks untuk menghindari kata yang tidak mempunyai makna.
2. Tidak digunakannya kamus atau *dictionary* dalam proses *stemming*, untuk meningkatkan efisiensi sistem.
3. Digunakan data *training* dan data *test* yang lebih besar, sebagai analisa lebih dalam mengenai efektifitas dan efisien sistem.
4. Jumlah kategori yang beragam yang meliputi seluruh kategori berita yang memungkinkan dalam kehidupan sehari-hari.
5. Adanya perbedaan bobot kata berdasarkan posisi kata tersebut dalam teks berita (judul, tanggal, teras berita, isi berita).
6. Perlu adanya perbandingan dengan metode klasifikasi yang lain.
7. Memberikan batasan untuk kemunculan kata sebagai cara mengurangi dimensi vektor.

UNIVERSITAS BRAWIJAYA

DAFTAR PUSTAKA

Adiwijaya, Igg. 2006. Text Mining and Knowledge Discovery. Kolokium bersama komunitas datamining Indonesia & soft-computing Indonesia, Sep'06.

Bing, Liu. 2005. Partially Supervised Classification of Text Documents, Computer Science, UIC. www.cs.uic.edu/~liub/teach/cs583-spring-05/CS583-semi-supervised-learning.ppt.

Deny Arnos Kwary. Afiksasi dalam 3 Bahasa. http://www.kwary.net/linguistics/gl/afiksasi.doc. Tanggal akses : 4 juni 2007.

Document Classification. http://eivind.imm.dtu.dk/thor/projects/multimedia /textmining/index.html. Tanggal akses : 22 mei 2007

Even,Yahir dan Zohar. 2002. *Introduction to Text Mining*. Automated Learning Group National Center For Supercomputing Applications. University of illionis. http://algdocs.ncsa.uiuc.edu/PR-20021116-2.ppt.

Gonen, Bilal. 2004. Text Categorization. http://lsdis.cs.uga.edu/~bilal/courses/fall2004/8350/presentations/text_cat_bilal.ppt

Hearst, Marti. 2003. *What is text mining?*. SIMS,UC Berkeley.http://www.sims.berkeley.edu/~hearst/text-mining.html.

Joachims, Thorsten. 1997. A Probalistic Analysis of the Rocchio Algoritm with TFIDF for Text Categorization. Universitas Dortmund, Fachbereich Informatik. Jerman, 1997.

Kantardzic, Mehmed. 2005. Data Mining "Concepts, Models, Methods, and Algorithms". IEEE Press. John Wiley & Sons, Inc. 2005.

Larose, Daniel T, 2005. Discovering Knowledge In Data ”An Introduction to Data Mining”. Wiley – Interscience, John Wiley and Sons, Inc.New Jersey, 2005

Larson, Ray. Vektor, Principles of Information Retrieval. University of California, School of Information. Berkeley. 2007 courses.ischool.berkeley.edu/i240/s07/Lectures/SIMS_240_Spr_2007 _Lecture_09.ppt

Lewis D. 1995. Evaluating and Optimizing Autonomous Text Classification Systems. AT&T Bell Laboratories Murray Hill, NJ 07974 USA Proceedings of the Eighteenth Annual Internatioal ACM-SIGIR Conference on Research and Development in Information Retrieval, July, 1995, pp. 246-254.

Manning, Cristopher. 2003. Text Information Retrieval, Mining and Exploitation. CS276B: Information Retrieval and Web Search. Lecture 13: Classifiers: kNN, Rocchio, etc. [Borrows slides from Ray Mooney and Barbara Rosario]. http://nlp.stanford.edu/IR-book/ppt/lecture14-vclassify.ppt

Sebastiani, Fabrizio. 2002. Machine Learning In Automated Text Categorization. Italy, 2002. www.isti.cnr.it/People/F.Sebastiani/Publications/ACMCS02.pdf Tanggal akses : 22 mei 2007

Staelin, Carl. Features Selection. www.hpl.hp.com/personal/Carl_Staelin/cs236601/Lecture06.ppt

Tala, Fadillah. 2003. *A Study of Stemming E?ects on Information Retrieval in Bahasa Indonesia*. Master of Logic Project - Institute for Logic, Language and Computation Universiteit van Amsterdam, The Netherlands

Yang Yiming dan Liu. 1999. Are-examination of text categorization methods. In Proceedings of ACM SIGIR Conference on Research and development in Information Retrieval (SIGIR), pp 42-49. 1999.

Lampiran 1
Daftar *Stopword*

1	ada	36	bagaimana	71	biasanya
2	adanya	37	bagaimanapun	72	bila
3	adalah	38	sebagaimana	73	bilakah
4	adapun	39	bagaimanakah	74	bisa
5	agak	40	bagi	75	bisakah
6	agaknya	41	bahkan	76	sebisanya
7	agar	42	bahwa	77	boleh
8	akan	43	bahwasanya	78	bolehkah
9	akankah	44	sebaliknya	79	bolehlah
10	akhirnya	45	banyak	80	buat
11	aku	46	sebanyak	81	bukan
12	akulah	47	beberapa	82	bukankah
13	amat	48	seberapa	83	bukanlah
14	amatlah	49	begini	84	bukannya
15	anda	50	beginian	85	cuma
16	andalah	51	beginikah	86	percuma
17	antar	52	beginilah	87	dahulu
18	diantaranya	53	sebegini	88	dalam
19	antara	54	begitu	89	dan
20	antaranya	55	begitukah	90	dapat
21	diantara	56	begitulah	91	dari
22	apa	57	begitupun	92	daripada
23	apaan	58	sebegitu	93	dekat
24	mengapa	59	belum	94	demi
25	apabila	60	belumlah	95	demikian
26	apakah	61	sebelum	96	demikianlah
27	apalagi	62	sebelumnya	97	sedemikian
28	apatah	63	sebenarnya	98	dengan
29	atau	64	berapa	99	depan
30	ataukah	65	berapakah	100	di
31	ataupun	66	berapalah	101	dia
32	bagai	67	berapapun	102	dialah
33	bagaikan	68	betulkah	103	dini
34	sebagai	69	sebetulnya	104	diri
35	sebagainya	70	biasa	105	dirinya

106	terdiri	141	jangankan	176	kini
107	dong	142	janganlah	177	kinilah
108	dulu	143	jika	178	kiranya
109	enggak	144	jikalau	179	sekiranya
110	enggaknya	145	juga	180	kita
111	entah	146	justru	181	kitalah
112	entahlah	147	kala	182	kok
113	terhadap	148	kalau	183	lagi
114	terhadapnya	149	kalaulah	184	lagian
115	hal	150	kalaupun	185	lah
116	hampir	151	berkali	186	lain
117	hanya	152	kali	187	lainnya
118	hanyalah	153	kalian	188	melainkan
119	harus	154	kami	189	selaku
120	haruslah	155	kamilah	190	lalu
121	harusnya	156	kamu	191	melalui
122	seharusnya	157	kamulah	192	terlalu
123	hendak	158	sebagian	193	lama
124	hendaklah	159	kapan	194	lamanya
125	hendaknya	160	kapankah	195	selama
126	hingga	161	kapanpun	196	selamanya
127	sehingga	162	dikarenakan	197	lebih
128	ia	163	karena	198	terlebih
129	ialah	164	karenanya	199	macam
130	ibarat	165	ke	200	semacam
131	ingin	166	kecil	201	maka
132	inginkah	167	kemudian	202	makanya
133	inginkan	168	kenapa	203	makin
134	ini	169	kepada	204	malah
135	inikah	170	kepadanya	205	malahan
136	inilah	171	ketika	206	mampu
137	itu	172	seketika	207	mampukah
138	itukah	173	khususnya	208	mana
139	itulah	174	jangankan	209	manakala
140	jangan	175	janganlah	210	manalagi

211	masih	246	per	281	sedikit
212	masihkah	247	pernah	282	sedikitnya
213	semasih	248	pula	283	segala
214	masing	249	pun	284	segalanya
215	mau	250	merupakan	285	segera
216	maupun	251	rupanya	286	sesegera
217	semaunya	252	serupa	287	sejak
218	memang	253	saat	288	sejenak
219	mereka	254	saatnya	289	sekali
220	merekalah	255	sesaat	290	sekalian
221	meski	256	saja	291	sekalipun
222	meskipun	257	sajalah	292	sesekali
223	semula	258	saling	293	sekaligus
224	mungkin	259	bersama	294	sekarang
225	mungkinkah	260	sama	295	sekitar
226	nah	261	sesama	296	sekitarnya
227	namun	262	sambil	297	sela
228	nanti	263	sampai	298	selain
229	nantinya	264	sana	299	selalu
230	nyaris	265	sangat	300	seluruh
231	oleh	266	sangatlah	301	seluruhnya
232	olehnya	267	saya	302	semakin
233	seorang	268	sayalah	303	sementara
234	seseorang	269	se	304	sempat
235	pada	270	sebab	305	semua
236	padanya	271	sebabnya	306	semuanya
237	padahal	272	sebuah	307	sendiri
238	paling	273	tersebut	308	sendirinya
239	sepanjang	274	tersebutlah	309	seolah
240	pantas	275	sedang	310	olah
241	sepantasnya	276	sedangkan	311	seperti
242	sepantasnyalah	277	per	312	sepertinya
243	para	278	pernah	313	sering
244	pasti	279	pula	314	sedikit
245	pastilah	280	pun	315	sedikitnya

316	seringnya	351	tidaknya	386	belas
317	serta	352	setidaknya	387	puluh
318	siapa	353	tidak	388	ratus
319	siapakah	354	tidakkah	389	ribu
320	siapapun	355	tidaklah	390	juta
321	disini	356	toh	391	milyar
322	disinilah	357	waduh	392	triliun
323	sini	358	wah	393	satuan
324	sinilah	359	wahai	394	puluhan
325	sesuatu	360	sewaktu	395	ribuan
326	sesuatunya	361	walau	396	jutaan
327	suatu	362	walaupun	397	milyaran
328	sesudah	363	wong	398	triliunan
329	sesudahnya	364	yaitu	399	senin
330	sudah	365	yakni	400	selasa
331	sudahkah	366	yang	401	rabu
332	sudahlah	367	wildan	402	kamis
333	supaya	368	untuk	403	jumat
334	tadi	369	mondar	404	sabtu
335	tadinya	370	mandir	405	minggu
336	tak	371	sedia	406	pagi
337	tanpa	372	butuh	407	siang
338	setelah	373	satu	408	sore
339	telah	374	dua	409	malam
340	tentang	375	tiga	410	subuh
341	tentu	376	empat	411	dhuhur
342	tentulah	377	lima	412	ashar
343	tentunya	378	enam	413	magrib
344	tertentu	379	tujuh	414	isya
345	seterusnya	380	delapan	415	dhuha
346	tapi	381	sembilan	416	hari
347	tetapi	382	sepuluh	417	tanggal
348	setiap	383	sebelas	418	bulan
349	tiap	384	tidaknya	419	tahun
350	setidak	385	setidaknya	420	jam

421	menit	456	november	491	pertama
422	detik	457	desember	492	pertanyaan
423	perbulan	458	usai	493	pertanyakan
424	perhari	459	cari	494	pihak
425	perjam	460	merah	495	pihaknya
426	pertahun	461	henti	496	pukul
427	perminggu	462	berbagai	497	pula
428	tinggi	463	rinci	498	pun
429	rendah	464	miliar	499	punya
430	besar	465	selesai	500	rasa
431	atas	466	yakin	501	rasanya
432	bawah	467	secara	502	rata
433	miring	468	februari	503	rupanya
434	kanan	469	jumlah	504	saat
435	kiri	470	jadi	505	saatnya
436	utara	471	olehnya	506	saja
437	selatan	472	pada	507	sajalah
438	barat	473	padahal	508	saling
439	timur	474	padanya	509	sama
440	febuari	475	pak	510	sambil
441	januari	476	paling	511	sampai
442	sang	477	panjang	512	sampaikan
443	maret	478	pantas	513	sana
444	april	479	para	514	sangat
445	mei	480	pasti	515	sangatlah
446	juni	481	pastilah	516	satu
447	juli	482	penting	517	saya
448	agustus	483	pentingnya	518	sayalah
449	september	484	per	519	se
450	oktober	485	percuma	520	sebab
451	menit	486	perlu	521	sebabnya
452	detik	487	perlukah	522	sebagai
453	perbulan	488	perlunya	523	sebagaimana
454	perhari	489	pernah	524	sebagainya
455	perjam	490	persoalan	525	sebagian

UNIVERSITAS BRAWIJAYA