IMPLEMENTASI METODE *MODIFIED K-NEAREST NEIGHBOR* (MKNN) PADA PENGKLASIFIKASIAN TEKS BERITA BERBAHASA INDONESIA

HALAMAN JUDUL

SKRIPSI

Oleh:
MUH. RAIS RAHIM
0410960036-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG
2011

UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN SKRIPSI

IMPLEMENTASI METODE *MODIFIED K-NEAREST NEIGHBOR* (MKNN) PADA PENGKLASIFIKASIAN TEKS BERITA BERBAHASA INDONESIA

Oleh:
Muh. Rais Rahim
0410960036-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 11 Agustus 2011
dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Dian Eka R., S.Si, M.Kom	**Nanang Yudi Setiawan,ST**
NIP. 197306192002122001	**NIP. 197606192006041001**

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP. 196709071992031001

UNIVERSITAS BRAWIJAYA

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama	: Muh. Rais Rahim
NIM	: 0410960036-96
Jurusan	: Matematika
Penulis Skripsi berjudul	: Implementasi Metode *Modified K-Nearest Neighbor* (MKNN) Pada Pengklasifikasian Teks Berita Berbahasa Indonesia

Dengan ini menyatakan bahwa :

1. 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.
2. Apabila di kemudian 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, 11 Agustus 2011
Yang menyatakan,

Muh. Rais Rahim
NIM. 0410960036-96

UNIVERSITAS BRAWIJAYA

IMPLEMENTASI METODE *MODIFIED K-NEAREST NEIGHBOR* (MKNN) PADA PENGKLASIFIKASIAN TEKS BERITA BERBAHASA INDONESIA

ABSTRAK

Berbagai kemudahan media internet menyebabkan terjadinya "ledakan informasi", yakni produk jurnalistik menjadi tak terhingga jumlahnya. Fenomena ini menjadi alasan perlunya sistem pengklasifikasi teks berita berbahasa Indonesia.

Sistem pengklasifikasi teks berita berbahasa Indonesia ini meliputi tiga tahap. Pertama, tahap pra-pemrosesan mengolah dokumen teks menjadi vektor numerik melalui *case folding* dan *tokenization*, *stop words removal*, *stemming*, *dictionary construction*, serta *feature weighting*. Kedua, tahap pembentukan *classifier* berdasarkan algoritma *Modified K-Nearest Neighbor* (MKNN). MKNN *classifier* dibentuk dengan menghitung nilai *validity* setiap data latih untuk mengukur stabilitas kedekatan dengan tetangganya. Ketiga, tahap pengklasifikasian memprediksi kategori data uji dari *K* data latih terdekat yang diukur dengan *cosine similarity*. Masing-masing dari *K* data latih dihitung bobotnya berdasarkan perkalian nilai *validity* dengan hasil *cosine similarity*. Bobot data latih dijumlahkan untuk setiap kategori yang sama. Sehingga, kategori yang bobotnya terbesar dipilih sebagai kategori untuk data uji.

Evaluasi sistem dilakukan dengan membandingkan efektivitas antara MKNN dengan KNN tradisional. Efektivitas diukur dengan *recall*, *precision*, lalu F_1 *measure*. Parameter *K* yang diujikan yaitu 3, 4, 5, 14, 15, dan 16. Secara keseluruhan, rata-rata F_1 *measure* MKNN berkisar dari 64% hingga 69% yakni secara konsisten lebih rendah dari KNN yang berkisar dari 74% hingga 77%.

UNIVERSITAS BRAWIJAYA

IMPLEMENTATION OF MODIFIED K-NEAREST NEIGHBOR (MKNN) METHOD IN TEXT CLASSIFICATION OF NEWS IN BAHASA INDONESIA

ABSTRACT

The ease of used of the Internet, has caused the "information explosion", by which, journalistic products become infinite in number. This phenomenon is the reason for the need of a system of text classifier of news in Bahasa Indonesia.

This system of text classifier of news in Bahasa Indonesia includes three phases. First, the preprocessing phase processes the text documents to be represented as numerical vector through case folding and tokenization, stop words removal, stemming, dictionary construction, and feature weighting. The second step employs the classifier construction phase, which is based on Modified K-Nearest Neighbor (MKNN) algorithm. MKNN classifier is constructed by calculating the validity value of each training data to measure the closeness stability with their neighbor. Third, the classification phase predicts category for the test data from the K nearest training data which is measured by cosine similarity. Weight of each of the K training data is calculated from multiplication between the validity value with the result of cosine similarity. Weights of K training data are summed for each category. Thus, the category which has the greatest weight is chosen for the test data.

The evaluation of this system is done by comparing the effectiveness between MKNN and traditional KNN. Effectiveness is measured using recall and precision, then F_1 measure. Values of K that are used here are 3, 4, 5, 14, 15, and 16. Overall, average of F_1 measure MKNN range from 64% to 69% which is consistently lower than KNN which range from 74% to 77%.

UNIVERSITAS BRAWIJAYA

KATA PENGANTAR

Assalamuakaikum'alaikum warahmatullahi wabarakatuh.

Teriring salam dan do'a semoga rahmat dan hidayah Allah SWT senantiasa tercurahkan kepada kita dalam melaksanakan tugas sehari-hari mengemban amanah selaku kholifah dimuka bumi ini. Amin.

Alhamdulillahi rabbil 'alamin.

Puji syukur penulis panjatkan kehadirat Allah SWT, karena atas segala rahmat dan limpahan hidayah-Nya, skripsi yang berjudul "Implementasi Metode *Modified K-Nearest Neighbor* (MKNN) Pada Pengklasifikasian Teks Berita Berbahasa Indonesia" ini dapat Penulis selesaikan.

Dalam penyelesaian skripsi ini, Penulis telah mendapat begitu banyak bantuan baik moral maupun materil dari banyak pihak. Atas bantuan yang telah diberikan, Penulis menyampaikan penghargaan dan ucapan terima kasih yang sedalam-dalamnya kepada :

1. Dian Eka Ratnawati, S.Si, M.Kom selaku pembimbing utama penulisan tugas akhir ini.
2. Nanang Yudi Setiawan, ST. selaku pembimbing pendamping dalam penulisan tugas akhir ini.
3. Drs. Marji, M.T selaku Ketua Program Studi S1-Ilmu Komputer Universitas Brawijaya Malang.
4. Dr. Abdul Rouf Alghofari, M.Sc selaku Ketua Jurusan Matematika Fakultas MIPA Universitas Brawijaya.
5. Nurul Hidayat, S.Pd, M.Sc selaku penasehat akademik Penulis.
6. Segenap bapak dan ibu dosen yang telah mendidik dan mengamalkan ilmunya kepada Penulis.
7. Segenap staf dan karyawan di Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya Malang.
8. Kedua orang tua dan saudara-saudari Penulis yang tak pernah berhenti memberikan doa dan dukungannya.
9. Kawan-kawan mahasiswa Program Studi Ilmu Komputer.
10. Sahabat-sahabat HMI Cabang Malang Komisariat MIPA Brawijaya.

Semoga laporan skripsi ini bermanfaat bagi pembaca sekalian.

Dengan tidak lupa kodratnya sebagai manusia, Penulis menyadari bahwa skripsi ini masih jauh dari kesempurnaan. Dengan segala kerendahan hati, Penulis mengharapkan kritik dan saran yang membangun dari pembaca.

Malang, 11 Agustus 2011

Penulis

DAFTAR ISI

UNIVERSITAS BRAWIJAYA

DAFTAR GAMBAR

DAFTAR TABEL

UNIVERSITAS BRAWIJAYA

DAFTAR LAMPIRAN

UNIVERSITAS BRAWIJAYA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penyajian informasi dan fakta dengan menggunakan perantara teknologi internet menjadi tren terkini dalam bidang jurnalistik di Indonesia. Perkembangan teknologi internet menawarkan banyak kemudahan bagi wartawan dalam mencari dan mengumpulkan bahan-bahan berita. Sedangkan masyarakat dimungkinkan untuk tidak hanya menjadi konsumen berita saja, tetapi juga sebagai produsen atas informasi.

Berbagai kemudahan media internet ini menjadi penyebab dari apa yang disebut dengan "ledakan informasi", di mana produk jurnalistik yang dihasilkan menjadi tak terhingga jumlah dan ragamnya. Bentuk informasi atau berita dapat berwujud teks, gambar, suara, video, ataupun gabungan dari antaranya, namun paling umum disajikan dalam bentuk teks. Fenomena ini menjadi alasan perlunya sistem pengklasifikasi teks berita baik bagi produsen maupun konsumen.

Sejumlah algoritma mesin pembelajaran telah diperkenalkan untuk menangani klasifikasi teks seperti *Support Vector Machines*, *K-Nearest Neighbor*, *Neural Network*, *Linear Least- Squares Fit*, dan *Naive Bayes* (Yang dan Liu, 1999). *K-Nearest Neighbor* (KNN) adalah salah satu jenis algoritma mesin pembelajaran yang umum digunakan. KNN telah diterapkan untuk klasifikasi teks dan merupakan salah satu metode dengan performa terbaik pada korpus *Reuters benchmark* (Yang dan Liu, 1999).

Klasifikasi teks dengan pendekatan KNN cukup sederhana. Kategori dokumen uji ditentukan dengan bobot kategori terbaik dari sejumlah *K* dokumen latih terdekat. Bobot kategori dihitung berdasarkan kategori dari *K* dokumen latih terdekat (Manning, dkk., 2009). KNN mudah diterapkan dengan metode yang sederhana, sangat baik hasilnya walau pada data latih yang memiliki banyak *noise*, dan juga efektif apabila data latih-nya banyak dan tidak diketahui jenis distribusinya (Parvin, dkk., 2008).

Banyak penelitian telah mengajukan beberapa perbaikan untuk meningkatkan akurasinya, salah satunya adalah metode *Modified K-*

Nearest Neighbor (MKNN) (Parvin, dkk., 2008). Metode ini meningkatkan kinerja metode KNN dengan menerapkan semacam pra-pemrosesan pada data latih. Ide utama dari metode MKNN adalah memprediksi kategori data uji berdasar pada sejumlah *K* data latih terdekat yang telah divalidkan. Pertama, penghitungan nilai *validity* untuk setiap sampel data latih. Nilai *validity* ini merupakan ukuran stabilitas kedekatan suatu sampel data latih terhadap tetangga-tetangga terdekatnya. Selanjutnya, pembobotan setiap data dalam KNN berdasarkan faktor perkalian nilai *validity* setiap data latih terhadap faktor nilai jarak kedekatannya. Dan terakhir, pembobotan kategori untuk mendapatkan kategori dengan skor terbesar sebagai hasil prediksi atau klasifikasi. Berdasarkan penelitian oleh Parvin, dkk. (2008) pada persoalan data mining, kinerja MKNN lebih efektif bila dibandingkan dengan kinerja metode KNN biasa. Kesimpulan tersebut diambil berdasarkan eksperimen pada lima basis data (*Wine*, *Isodata and Monk's problems*) dari *UCI Repository of machine learning databases*.

Keberhasilan MKNN pada kasus data mining tersebut memungkinkan untuk diterapkan pada *text mining* khususnya persoalan pengklasifikasian teks. Sehingga berdasarkan ulasan ini, penulis melakukan penelitian dengan judul "Implementasi Metode *Modified K-Nearest Neighbor* (MKNN) Pada Pengklasifikasian Teks Berita Berbahasa Indonesia".

1.2 Rumusan Masalah

Rumusan masalah yang diajukan adalah sebagai berikut :

1. Bagaimana implementasi metode MKNN pada sistem pengklasifikasi teks berita berbahasa Indonesia ?
2. Bagaimana tingkat akurasi sistem pengklasifikasi teks berita berbahasa Indonesia berbasis metode MKNN jika dibandingkan dengan metode KNN biasa ?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah :

1. Dokumen teks berita berbahasa Indonesia dalam bentuk teks murni (*plain text*).
2. Model klasifikasi teks dokumen berkategori tunggal.

1.4 Tujuan

Tujuan penelitian ini adalah :

1. Pembuatan sistem pengklasifikasi teks berita berbahasa Indonesia berbasis metode MKNN.
2. Pengujian sistem pengklasifikasi teks berita berbahasa Indonesia berbasis metode MKNN.

1.5 Manfaat

Manfaat penelitian ini adalah :

1. Sistem yang dikembangkan ini dapat membantu media pers untuk mengklasifikasikan berita yang akan dipublikasikannya secara terkomputerisasi.
2. Sistem yang dikembangkan ini juga dapat membantu masyarakat, untuk mengklasifikasikan berita-berita yang telah diterimanya secara terkomputerisasi.

1.6 Sistematika Penulisan

Laporan penelitian ini disusun berdasarkan sistematika penulisan sebagai berikut:

1. BAB I PENDAHULUAN
   Berisi latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.
2. BAB II TINJAUAN PUSTAKA
   Berisi uraian teori-teori yang menjadi dasar rujukan pada penelitian ini.
3. BAB III METODOLOGI DAN PERANCANGAN
   Berisi penjelasan metode yang digunakan dalam penelitian dan bagaimana kerangka dasar solusi yang diusulkan.
4. BAB IV IMPLEMENTASI DAN PEMBAHASAN
   Berisi uraian mengenai implementasi dari rancangan solusi yang diajukan dan pembahasan mengenai hasil pengujian.
5. BAB V PENUTUP
   Berisi kesimpulan dan saran penelitian.

UNIVERSITAS BRAWIJAYA

BAB II
TINJAUAN PUSTAKA

2.1 *Text Mining*

Text mining berkaitan erat dengan *data mining*. *Data mining* dapat didefinisikan sederhana sebagai ekstraksi, "penambangan", atau penemuan pengetahuan dari data yang sangat banyak (Han dan Kamber, 2006). Sedangkan *text mining* merupakan penerapan metodologi *data mining* pada sumber tekstual yang tidak terstruktur (Solka, 2008).

Velickov dan Solomatine (2000) menjelaskan bahwa tujuan *data mining* dalam praktiknya cenderung untuk mendapatkan deskripsi dan prediksi. Tujuan deskripsi yaitu menemukan pola, hubungan, atau korelasi antar data untuk menjelaskan isi data sehingga bisa ditafsirkan oleh manusia. Sedangkan tujuan prediksi yaitu untuk meramal variabel data yang belum diketahui nilainya dengan melibatkan beberapa variabel lain yang telah diketahui nilainya.

Pengelompokan teks termasuk dalam lingkup *text mining*. Pengelompokan teks juga dilakukan untuk mendapatkan deskripsi ataupun prediksi. Pengelompokan teks yang bertujuan deskripsi, disebut juga *text clustering*, mengelompokkan teks dengan cara mempelajari pola khas dari sekumpulan dokumen teks itu sendiri. Sedangkan pengelompokan teks yang bertujuan prediksi, disebut juga *text categorization* atau *text classification*, mengelompokkan teks yang belum berkategori dengan mempelajari contoh sekumpulan dokumen teks yang telah diberi kategori sebelumnya (Solka, 2008).

2.2 Klasifikasi Teks

Klasifikasi teks dirumuskan oleh Sebastiani (2005) sebagai fungsi pendekatan terhadap kategori sasaran yang belum diketahui. $\Phi : D \times C \rightarrow \{T, F\}$ merupakan notasi fungsi yang menjelaskan bagaimana dokumen diklasifikasikan berdasarkan pengklasifikasian sebelumnya oleh ahli yang berwenang. Fungsi $\Phi: D \times C \rightarrow \{T, F\}$ merupakan fungsi *classifier,* di mana $C = \{c_1, \ldots, c_{|C|}\}$ adalah sekumpulan kategori yang telah ditentukan dan D adalah sejumlah

dokumen (mungkin juga tak terbatas jumlahnya). Jika $\Phi (d_j, c_i)=T$, maka d_j dimasukkan sebagai anggota dari kategori c_i, sedangkan jika $\Phi (d_j, c_i)=F$, berarti d_j tidak masuk kategori c_i.

Klasifikasi teks secara rinci oleh Guo, dkk., (2004) dijelaskan sebagai suatu proses pengelompokan sejumlah dokumen teks ke dalam satu atau lebih kelompok yang telah diklasifikasikan sebelumnya berdasarkan isi teks. Pengklasifikasian teks dilakukan dengan merepresentasikan dokumen teks ke dalam bentuk vektor numerik. Dokumen dalam model ruang vektor memungkinkan operasi matematis sehingga dapat dikelompokkan dengan menggunakan *classifier*. *Classifier* menerapkan algoritma pembelajaran terhadap sekumpulan dokumen yang telah berkategori tertentu. Pengklasifikasian teks menerapkan fungsi pengujian *classifier* terhadap dokumen yang belum berkategori. Arsitektur umum sistem pengklasifikasi teks terdiri dari tiga proses utama: (1) pra-pemrosesan, (2) pembentukan *classifier*, dan (3) pengklasifikasian dokumen.

2.3 Pra-pemrosesan Dokumen Teks

Sebastiani (2005) merumuskan pra-pemrosesan dokumen sebagai proses pengindeksan dokumen yang memetakan isi dokumen menjadi representasi ringkas serta padat sehingga bisa langsung ditafsirkan oleh algoritma pembentukan *classifier*, atau juga ketika *classifier* telah terbentuk. Metode pengindeksan dokumen tersebut umumnya menggunakan metode dari bidang *information retrieval*, di mana teks d_j biasanya direpresentasikan sebagai vektor bobot term. Vektor $d_j = <w_{1j}, \ldots, w_{|T|j}>$, di mana T adalah kamus berisi term-term unik (juga disebut sebagai fitur) yang muncul setidaknya sekali dalam sejumlah k dokumen (pada klasifikasi teks: setidaknya sekali dalam k dokumen latih), dan w_k adalah nilai bobot yang mengukur tingkat penting t_k sebagai pembentuk pola ciri d_j.

Guo, dkk., (2004) membagi pra-pemrosesan dokumen teks menjadi enam komponen yang terdiri dari *document conversion*, *stop words removal*, *word stemming*, *feature selection*, *dictionary construction*, dan *feature weighting*.

Selain itu, pada tahap pra-pemrosesan juga dapat dilakukan proses *case folding* dan *tokenization* (Manning, dkk., 2009).

2.3.1 *Case Folding* dan *Tokenization*

Manning, dkk., (2009) menjelaskan proses *case folding* dan *tokenization* dilakukan sebelum pra-pemrosesan lingustik (yaitu *functional word removal* dan *word stemming*). *Case folding* adalah proses menyeragamkan bentuk huruf dalam dokumen. Proses *case folding* mengubah semua huruf dalam dokumen menjadi huruf kecil, yaitu huruf 'a' hingga 'z'. *Tokenization* adalah proses untuk mengambil kata dan istilah sederhana dari sebuah dokumen. Proses *tokenization* mendeteksi suatu kata yang terdiri dari kombinasi karakter huruf ('a' hingga 'z'), sedangkan karakter selain huruf dihilangkan dan dianggap *delimiter*.

2.3.2 *Stop Words Removal*

Proses ini menghapus *stop words* atau kata-kata tidak penting yang bukan merupakan ciri dari suatu dokumen. *Stop words* merupakan term-term yang dianggap tidak penting atau berkategori netral. Umumnya berupa kata sambung, kata tanya, dan kata sapa.

Tala (2003) telah mengajukan daftar *stop words* Bahasa Indonesia. Daftar *stop words* tersebut merupakan hasil analisa frekuensi kemunculan term berdasarkan penelitiannya pada sejumlah teks berita. Penelitian tersebut menunjukkan bahwa penghapusan daftar *stop words* dapat meningkatkan presisi, terutama pada tingkat *recall* rendah walaupun tidak signifikan.

2.3.3 *Word Stemming*

Word stemming adalah proses pemetaan varian morfologi suatu kata yang berbeda dari kata dasarnya. *Stemming* berawal dari asumsi bahwa istilah yang mempunyai bentuk kata dasar yang sama biasanya akan memiliki makna yang sama. Proses *stemming* secara luas digunakan dalam bidang *information retrieval* (begitu pun dalam bidang klasifikasi teks) sebagai cara untuk meningkatkan kinerja. Selain kemampuannya untuk meningkatkan kinerja *information retrieval*, proses *stemming* juga akan mengurangi ukuran indeks atau kamus kata (Tala, 2003).

Algoritma *Porter* merupakan algoritma yang umum digunakan untuk *stemming* teks Bahasa Inggris. Akan tetapi, proses *stemming* pada teks berbahasa Indonesia berbeda dengan *stemming* pada teks

berbahasa Inggris. Pada teks berbahasa Inggris, proses yang diperlukan hanya proses menghilangkan sufiks. Sedangkan pada teks berbahasa Indonesia, selain sufiks, prefiks dan konfiks juga dihilangkan (Tala, 2003).

Tala (2003) melalui penelitiannya telah mengajukan algoritma *Porter Stemmer* untuk teks Bahasa Indonesia. Adapun langkah-langkah algoritma ini adalah sebagai berikut:

1. Hapus partikel (-kah, -lah, -pun)
2. Hapus kata ganti kepemilikan (-ku, -mu, -nya)
3. Hapus awalan pertama (meng-, meny-, men-, mem-, me-, peng-, peny-, pen-, pem-, di-, ter-, ke-). Jika tidak ada lanjutkan ke langkah 4a, jika ada maka lanjutkan ke langkah 4b
4. a. Hapus awalan kedua (ber-, bel-, be-, per-, pel-, pe-), lanjutkan ke langkah 5a
   b. Hapus akhiran (-kan, -an, -i), jika tidak ditemukan maka kata tersebut diasumsikan sebagai kata dasar. Jika ditemukan maka lanjutkan ke langkah 5b
5. a. Hapus akhiran (-kan, -an, -i). Kemudian kata akhir diasumsikan sebagai kata dasar
   b. Hapus awalan kedua (ber-, bel-, be-, per-, pel-, pe-). Kemudian kata akhir diasumsikan sebagai kata dasar.

2.3.4 *Dictionary Construction*

Guo, dkk., (2004) menjelaskan bahwa proses ini membentuk kamus yang berisi term-term unik dari seluruh dokumen. Kamus ini kemudian menjadi acuan untuk mengubah dokumen teks menjadi vektor. Setiap fitur dalam vektor merujuk pada sebuah term dalam kamus.

Inverted index adalah bentuk kamus yang umum digunakan dalam *information retrieval*. *Inverted index* terdiri dari dua bagian yaitu *term list* dan *posting list*. *Term list* merupakan daftar kata unik yang ada dalam suatu koleksi dokumen. Sedangkan *posting list* merupakan data kemunculan setiap kata yang berisi daftar dokumen beserta frekuensi setiap kata dari *term list* (Manning, dkk., 2009).

2.3.5 *Feature Weighting*

Fungsi pembobotan fitur bermula dari bidang *Information Retrieval*. Model paling populer adalah *tf.idf*. Dalam buku "*An Introduction to Information Retrieval"* (Manning, dkk., 2009) diterangkan bahwa skema pembobotan *tf.idf* terdiri dari dua faktor, yaitu bobot lokal dan bobot global. Bobot lokal, disimbolkan *tf*, merupakan bobot term t_i dalam sebuah dokumen tertentu d_j, yang diestimasi berdasarkan frekuensi t_i dalam dokumen. Sedangkan bobot global, dinotasikan *idf*, berdasarkan perhitungan jumlah dokumen yang mengandung term t_i dalam koleksi.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right) \quad (2.1)$$

$w_{i,j}$ adalah bobot term *i* dalam dokumen *j*; $tf_{i,j}$ adalah frekuensi kemunculan term *i* dalam dokumen *j*; *N* adalah total jumlah dokumen; dan df_i adalah jumlah dokumen yang mengandung term *i*.

2.4 Classifier

Classifier menerapkan algoritma *machine learning* yang merupakan komponen kunci dari sistem pengklasifikasi terkomputerisasi. *Classifier* dibentuk dengan memperlajari dokumen yang telah ditetapkan, kemudian mengklasifikasikan dokumen yang belum diketahui kategorinya.

2.4.1 *K-Nearest Neighbor* (KNN)

Songbo Tan (2006) menjelaskan bahwa untuk mengklasifikasikan suatu dokumen uji ($\vec{d}_0$), pengklasifikasi KNN memilih dokumen tetangga berdasarkan peringkat kesamaannya di antara koleksi dokumen latih, dan menggunakan label kategori dari k dokumen latih terdekat untuk memprediksi kelas dari dokumen uji. Untuk mengukur tingkat kesamaan dokumen secara efisien, digunakan pengukuran *cosine similarity* (Rumus 2.9). Kemudian, kategori dari dokumen-dokumen tetangga tersebut dihitung skornya berdasarkan hasil tingkat kesamaannya terhadap dokumen uji, seperti pada Rumus 2.2 berikut ini.

$$score(\vec{d}_0, C_i) = \sum_{\vec{d}_j \in KNN(\vec{d}_0)} Cosine(\vec{d}_0, \vec{d}_j)\,\delta(\vec{d}_j, C_i) \quad (2.2)$$

$KNN(\vec{d}_0)$ merupakan himpunan K tetangga terdekat (*K-nearest neighbor*) dari dokumen uji $\vec{d}_0$. Sedangkan, $\delta(\vec{d}_j, C_i)$ merupakan fungsi klasifikasi dokumen $\vec{d}_j$ terhadap kategori C_i seperti pada Rumus 2.3 berikut ini :

$$\delta(\vec{d}_j, C_i) = \begin{cases} 1 & \vec{d}_j \in C_i \\ 0 & \vec{d}_j \notin C_i \end{cases} \quad (2.3)$$

Dengan demikian, hasil prediksi kategori (C) dokumen uji $\vec{d}_0$ dalam pengklasifikasian KNN dapat didefinisikan seperti pada Rumus 2.4 berikut ini :

$$C = arg\,max_{c_i}(score(\vec{d}_0, C_i)) \quad (2.4)$$

2.4.2 *Modified K-Nearest Neighbor* (MKNN)

Parvin, dkk., (2008) telah mengembangkan algoritma *machine learning* baru berdasarkan algoritma *K Nearest Neighbor* (KNN). Algoritma ini disebutnya *Modified K-Nearest Neighbor* (MKNN). Berdasarkan penelitiannya pada persoalan data mining, hasil evaluasi kinerja MKNN disimpulkan sangat efektif bila dibandingkan dengan kinerja metode KNN biasa. Kesimpulan tersebut diambil berdasarkan eksperimen pada lima basis data (*Wine, Isodata and Monk's problems*) dari *UCI Repository of machine learning databases*.

```
Output_label := MKNN(train_set, test_sample)
Begin
  For i := 1 to train_size
    Validity(i) := Compute Validity of i-th sample;
  End for;
  Output_label := Weighted_KNN(Validity, test_sample);
  Return Output_label ;
End.
```

Gambar 2.1: *Pseudo Code* MKNN

Ide utama dari algoritma MKNN adalah memprediksi kategori

data uji berdasar pada sejumlah *K* referensi data latih yang telah divalidkan. Pertama, penghitungan nilai *validity* setiap sampel data latih. Selanjutnya, menerapkan metode pembobotan KNN dalam proses pengujian data. *Validity* mengukur nilai stabilitas kedekatan dari setiap sampel referensi (data latih) terhadap tetangga-tetangganya (*nearest neighbor*). Sedangkan fungsi pembobotan KNN merupakan perkalian jarak dan nilai *validity* setiap data latih.

Validity setiap data latih dihitung berdasarkan data-data lain yang terdekat. Proses validasi dilakukan sekali untuk semua sampel data latih. Untuk menghitung *validity* suatu sampel data latih, diasumsikan *H* sebagai banyak tetangga terdekat dari data latih *x*. Nilai parameter *H* berdasarkan eksperimen Parvin, dkk., (2008) ditentukan sebanyak 10% dari jumlah dokumen latih. Di antara *H* tetangga terdekat dari data *x*, *Validity(x)* dihitung dari jumlah data yang kategorinya sama dengan label kategori data *x*. Perhitungan nilai *validity* ini dirumuskan seperti Rumus 2.5 berikut ini.

$$Validity(x) = \frac{1}{H} \sum_{i=1}^{H} S(lbl(x), lbl(N_i(x))) \tag{2.5}$$

H adalah banyak tetangga terdekat dari titik data latih *x* dan *lbl(x)* adalah label kategori dari sampel *x*. $N_i(x)$ berarti tetangga terdekat ke-*i* dari titik *x*. Sedangkan, fungsi *S* menghitung kesamaan antara titik *x* dan tetangga terdekat ke-*i*.

$$S(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases} \tag{2.6}$$

a dan *b* adalah label kategori suatu data latih. *S* bernilai 1, jika label kategori *a* sama dengan label kategori *b*. *S* bernilai 0, jika label kategori *a* tidak sama dengan label kategori *b*.

Weighted KNN adalah salah satu variasi metode KNN yang menggunakan pembobotan *K* tetangga terdekat dalam penentuan hasil akhir prediksi. Berbeda dengan aturan kategori mayoritas pada metode KNN biasa, *Weighted* KNN menentukan hasil prediksi kategori sampel data uji dengan perhitungan bobot setiap sampel dari sejumlah *K* data latih yang terdekat. Satu per satu dari *K* data latih terdekat dihitung bobotnya berdasarkan Rumus 2.7 di bawah ini.

$$W(i) = Validity(i) \times \frac{1}{d_e + 0.5} \quad (2.7)$$

W(i) dan *Validity(i)* adalah bobot dan nilai *validity* sampel data latih terdekat ke-*i*. Sedangkan d_e adalah *Euclidean Distance* merupakan fungsi jarak antara dua vektor.

$$Euclidean(\vec{U}, \vec{V}) = \sqrt{\sum_{i=1}^{n} (u_i - v_i)^2} \quad (2.8)$$

Rumus 2.8 di atas merupakan fungsi *Euclidean Distance* yang mengukur jarak kedekatan antara dua vektor. Semakin kecil hasil *euclidean* antara dua vektor, maka semakin dekat jaraknya.

Teknik pembobotan ini memberikan bobot lebih penting pada sampel referensi dengan nilai *validity* yang lebih besar dan berdekatan dengan sampel uji. Nilai *validity* dalam pembobotan ini akan mengurangi pengaruh sampel referensi yang kurang stabil terhadap keputusan penentuan kategori. Dengan kata lain, nilai *validity* ini mengatasi kelemahan pembobotan (yang hanya berdasarkan jarak kesamaan) dalam hal kendala *outlier*.

Nilai bobot-bobot sampel ini kemudian dijumlahkan untuk setiap kategori. Terakhir, kategori yang jumlah bobotnya terbesar dipilih sebagai hasil prediksi.

2.4.3 *Cosine Similarity*

Pada aplikasi seperti *information retrieval* dan pengelompokan teks dokumen, data yang dibandingkan berupa vektor obyek yang kompleks berisi sejumlah besar entitas (yang mewakili isi teks). Untuk mengukur jarak antara obyek yang kompleks, fungsi kesamaan non-metrik lebih umum digunakan dibanding perhitungan jarak metrik tradisional seperti *Euclidean Distance* (Han dan Kamber, 2006)

Ada beberapa cara untuk merumuskan fungsi kesamaan antara dua vektor. Metode yang paling baik untuk membandingkan kesamaan antara dua vektor dalam hal *text mining* adalah fungsi *Cosine Similarity* (Han dan Kamber, 2006).

$$Cosine(\vec{U},\vec{V}) = \frac{\vec{U}\cdot\vec{V}}{|\vec{U}|\times|\vec{V}|} = \frac{\sum_{i=1}^{n} u_i \times v_i}{\left(\sqrt{\sum_{i=1}^{n}(u_i)^2}\right)\times\left(\sqrt{\sum_{i=1}^{n}(v_i)^2}\right)} \quad (2.9)$$

Rumus 2.9 di atas merupakan fungsi *Cosine Similarity* antara vektor $\vec{U}$ dan $\vec{V}$. Penyebutnya merupakan *dot product* (juga disebut *inner product*) dari kedua vektor. Sedangkan pembilangnya merupakan perkalian dari *Euclidean length* masing-masing vektor.

Pada persoalan *text mining*, nilai bobot term tidak boleh negatif sehingga hasil *cosine* dua vektor akan berkisar antara 0 hingga 1. Hal tersebut berarti sudut antara dua vektor dokumen tidak dapat lebih besar dari 90°. Sehingga, semakin besar hasil *cosine* maka semakin besar kesamaan antar dokumen.

2.5 Metode Evaluasi

Evaluasi percobaan pada *classifier* biasanya lebih diutamakan untuk mengukur keefektifannya daripada efisiensinya, yaitu kemampuannya untuk melakukan pengklasifikasian secara benar (Sebastiani, 2005).

Menurut Sebastiani (2005), efektivitas *classifier* pada pengklasifikasian berlabel tunggal diukur berdasarkan tingkat akurasinya, yaitu menggunakan *precision* dan *recall*. Adapun F_1 *measure* merupakan kombinasi dari kedua evaluasi standar tersebut.

Dalam konteks persoalan klasifikasi, perumusan *precision* dan *recall* umumnya menggunakan Tabel 2.1 berikut ini (Sebastiani, 2005) :

Tabel 2.1: Kesesuaian Hasil Kategori

		Hasil klasifikasi dari ahli	
		YES	NO
Hasil klasifikasi dari sistem	YES	*TP* (*True Positive*)	*FP* (*False Positive*)
	NO	*FN* (*False Negative*)	*TN* (*True Negative*)

Recall dirumuskan sebagai berikut.

$$Recall = \frac{TP}{TP + FN} \tag{2.10}$$

Precision dirumuskan sebagai berikut.

$$Precision = \frac{TP}{TP + FP} \tag{2.11}$$

Sedangkan F_1 *Measure* (Rumus 2.12) merupakan gabungan antara *recall* (Rumus 2.10) dan *precision* (Rumus 2.11).

$$F_1 Measure = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{2.12}$$

BAB III
METODOLOGI DAN PERANCANGAN

3.1 Skema Penelitian

Skema penelitian ini dirancang melalui berbagai tahapan sebagai berikut :

1. Analisis dan perancangan sistem pengklasifikasi teks berbasis MKNN *classifier*.
2. Implementasi rancangan sistem ke perangkat lunak.
3. Pengujian perangkat lunak pada dokumen masukan yang telah disiapkan.
4. Evaluasi dan analisa hasil pengujian.
5. Penyimpulan hasil penelitian.

3.2 Analisa Data Masukan

Data masukan adalah dokumen teks berita berbahasa Indonesia. Dokumen masukan terdiri dari dua jenis yaitu dokumen latih dan dokumen uji. Dokumen latih merupakan dokumen yang telah memiliki label kategori tertentu dari ahli yang berotoritas. Dokumen latih digunakan sebagai sumber data pembelajaran oleh sistem. Sedangkan dokumen uji merupakan dokumen yang belum memiliki label kategori. Dokumen uji digunakan sebagai sumber data pengujian untuk diklasifikasikan oleh sistem.

3.3 Analisa Sistem

Analisa sistem terdiri dari analisa kebutuhan sistem, batasan-batasan sistem, serta deskripsi umum sistem. Analisa kebutuhan sistem merinci hal-hal pokok yang harus ada dalam sistem. Batasan sistem menjelaskan ketentuan fungsional sistem. Sedangkan deskripsi umum sistem menggambarkan proses sistem secara bertahap.

3.3.1 Analisa Kebutuhan Sistem

Sistem dibangun berdasarkan kebutuhan mengenai perangkat komputer yang membantu mengklasifikasikan dokumen teks berita

berbahasa Indonesia. Hal-hal pokok yang harus ada dalam sistem ini adalah :

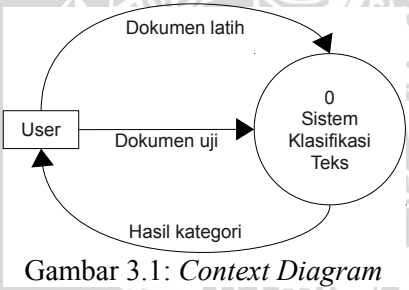
- Sistem mampu membaca pola ciri dokumen latih.
- Sistem mampu membaca pola ciri dokumen uji.
- Sistem mampu memberikan hasil prediksi kategori dokumen uji yang tingkat kebenarannya mendekati hasil klasifikasi manual oleh ahli yang berotoritas.

3.3.2 Batasan Sistem

Sistem dibangun berdasarkan ketentuan-ketentuan berikut :

- Dokumen masukan dalam bentuk berkas teks murni (*plain text*).
- Berkas dokumen masukan berada di dalam media penyimpanan lokal (*local disk*) yang terstruktur pada alamat tertentu.

3.3.3 Deskripsi Umum Sistem



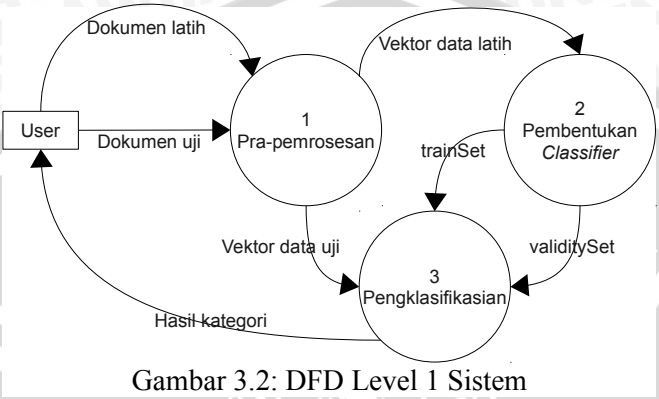
Gambar 3.1: *Context Diagram*

Sistem dibangun untuk persoalan klasifikasi teks berita berbahasa Indonesia. Sistem mengolah teks dokumen masukan, berupa dokumen latih dan dokumen uji, menjadi bentuk representasi vektor numerik. Kemudian, sistem menghasilkan keluaran berupa hasil prediksi kategori untuk dokumen uji.

3.4 Rancangan Proses Sistem

Sistem pengklasifikasi teks berita berbahasa Indonesia ini terdiri dari tiga proses utama yaitu : (1) pra-pemrosesan data, (2) pembentukan *classifier*, dan (3) pengklasifikasian.

Gambar 3.2 berikut merupakan *Data Flow Diagram* (DFD) level 1 sistem.

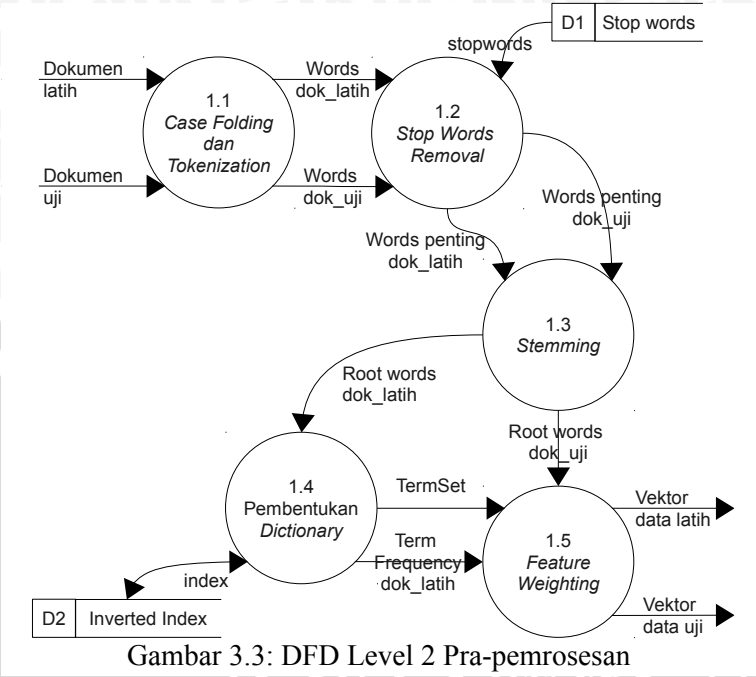


Gambar 3.2: DFD Level 1 Sistem

Pertama, (1) pra-pemrosesan data mengubah dokumen masukan menjadi bentuk vektor data numerik. Kemudian, (2) proses pembentukan MKNN *classifier*. Terakhir, (3) proses pengklasifikasian yang memprediksi kategori berdasarkan MKNN *classifier*.

3.4.1 Rancangan Pra-pemrosesan

Pra-pemrosesan merupakan proses pengolahan dokumen teks masukan menjadi representasi vektor numerik yang ringkas dengan dimensi seragam. Pra-pemrosesan terdiri dari lima proses. (1.1) Proses *case folding* dan *tokenization* memecah teks menjadi bentuk per kata dengan tipe huruf kecil. (1.2) Proses *stop words removal* menghapus term-term yang tidak penting dalam dokumen. (1.3) Proses *stemming* mengurai setiap kata menjadi hanya dalam bentuk kata dasarnya. (1.4) Proses pembentukan *dictionary* menggabungkan fitur-fitur relevan dan unik dari semua dokumen latih menjadi satu vektor fitur. Terakhir, (1.5) proses *feature weighting* yaitu pembobotan setiap fitur untuk mendapatkan representasi vektor numerik setiap dokumen masukan. Gambar 3.3 berikut merupakan *data flow diagram* (DFD) level 2 pra-pemrosesan.

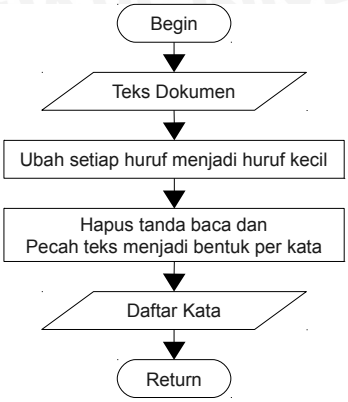


Gambar 3.3: DFD Level 2 Pra-pemrosesan

3.4.1.1 Rancangan Proses *Case Folding* dan *Tokenization*

Proses *case folding* mengubah semua jenis huruf menjadi huruf kecil. Sedangkan proses *tokenization* memecah teks dokumen masukan menjadi bentuk per kata sehingga memudahkan proses selanjutnya.

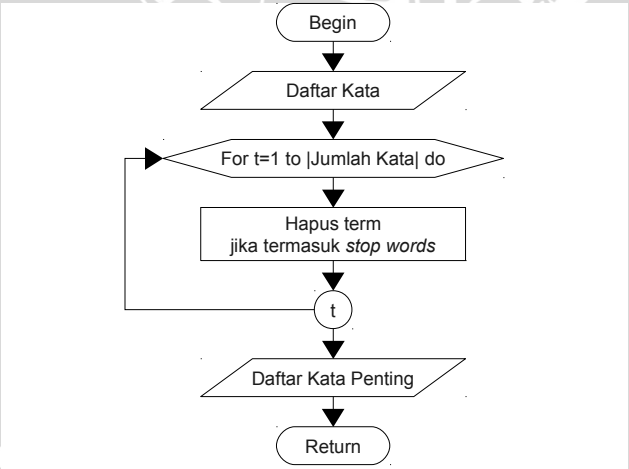
Pertama, teks dokumen masukan diubah bentuk setiap hurufnya menjadi bentuk huruf kecil. Kemudian, semua tanda baca dihilangkan dan teks dokumen dipecah menjadi kata-kata yang terpisah. Gambar 3.4 berikut merupakan diagram *flow chart* proses *tokenization*.



Gambar 3.4: *Flow Chart* Proses *Case Folding* dan *Tokenization*

3.4.1.2 Rancangan Proses *Stop Words Removal*

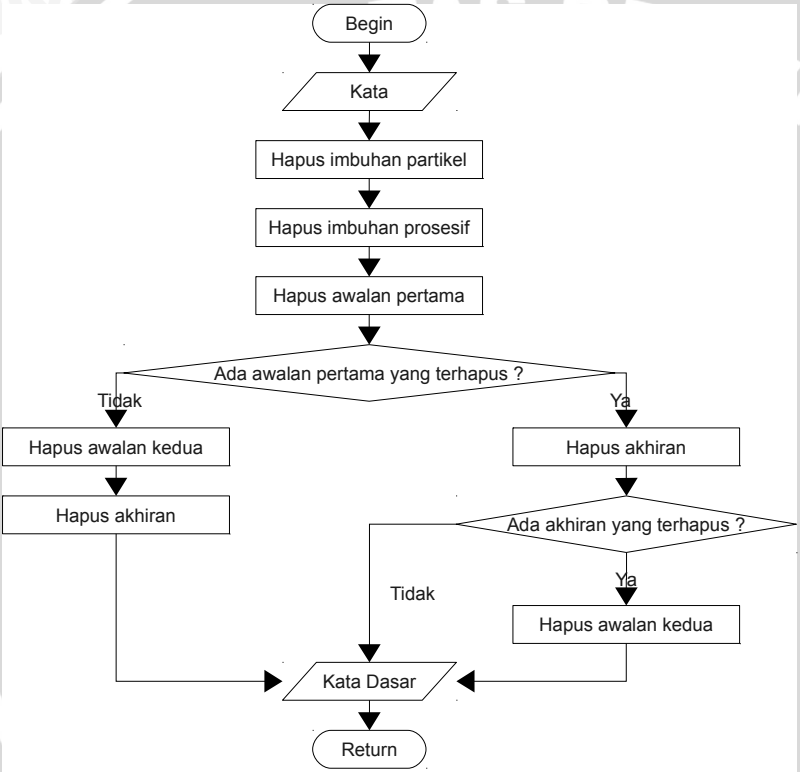
Proses *stop words removal* menyeleksi term yang dianggap penting dari teks dokumen, sedangkan term yang tidak penting akan dihapus. *Stop words* berisi daftar kata-kata yang tidak penting seperti kata sambung dan kata tanya. Daftar *stop words* yang digunakan dalam penelitian ini berdasarkan penelitian oleh Tala (2003). Gambar 3.5 berikut merupakan diagram *flow chart* proses *stop words removal*.



Gambar 3.5: *Flow Chart* Proses *Stop Words Removal*

3.4.1.3 Rancangan Proses *Stemming*

Proses *stemming* mengurai setiap kata menjadi bentuk kata dasarnya saja. Penguraian dilakukan dengan menerapkan berbagai macam aturan tertentu. Acuan algoritma *stemming* yang digunakan adalah algoritma *Porter Stemmer* untuk Bahasa Indonesia berdasarkan penelitian Tala (2003). Gambar 3.6 berikut merupakan diagram *flow chart* proses *stemming* ini.

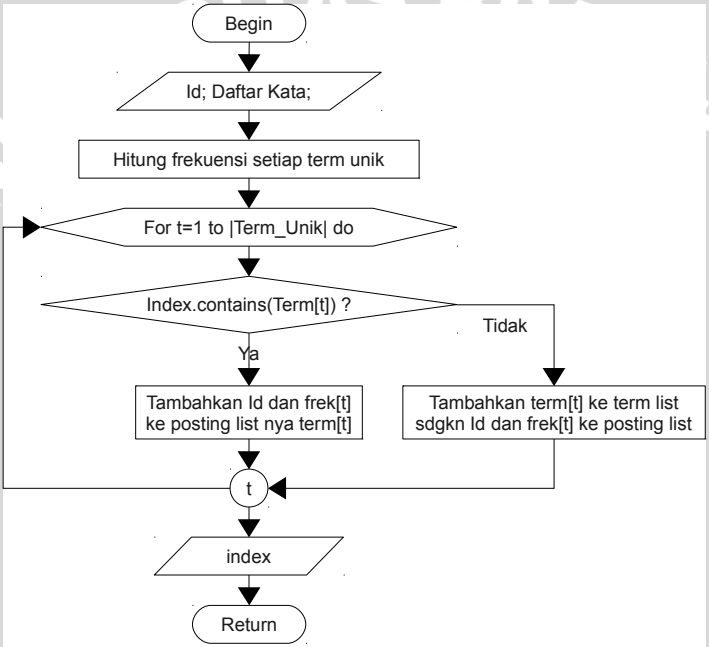


Gambar 3.6: *Flow Chart* Proses *Stemming*

3.4.1.4 Rancangan Proses Pembentukan *Dictionary*

Pembentukan *dictionary* adalah proses membentuk kamus yang seragam sebagai acuan untuk mengubah dokumen teks menjadi vektor fitur. Setiap fitur dalam vektor merujuk pada sebuah kata

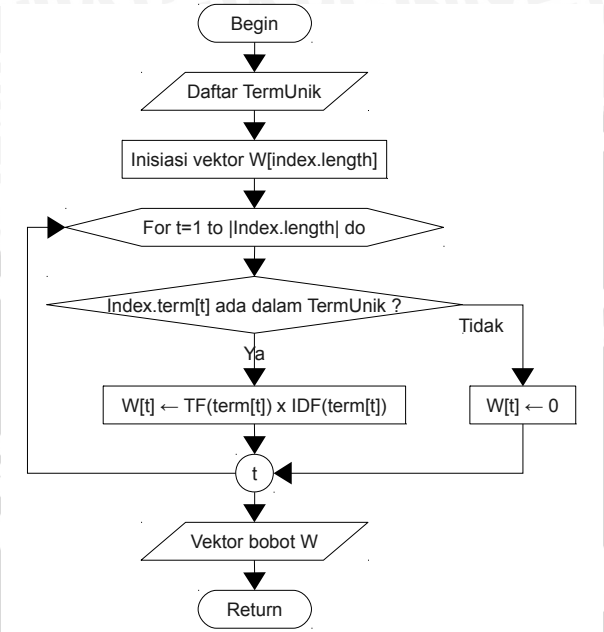
dalam kamus. Kamus ini berisi daftar term unik dari semua dokumen latih setelah melalui pra-pemrosesan sebelumnya. Kamus ini dibentuk berdasarkan struktur data *inverted index*. Struktur *inverted index* terdiri dari dua elemen yaitu *term list* dan *posting list*. *Term list* merupakan daftar kata unik yang ada dalam suatu koleksi dokumen. Sedangkan *posting list* merupakan daftar dokumen beserta frekuensi untuk setiap kata di *term list*. Gambar 3.7 berikut merupakan diagram *flow chart* proses pembentukan kamus *inverted index*.



Gambar 3.7: *Flow Chart* Proses Pembentukan *Dictionary*

3.4.1.5 Rancangan Proses *Feature Weighting*

Feature weighting adalah proses yang merepresentasikan fitur term menjadi bentuk numerik. Skema pembobotan yang digunakan adalah skema pembobotan *tf.idf* (Rumus 2.1). Pembobotan fitur dilakukan berdasarkan *dictionary* yang telah dibentuk dari semua dokumen latih. Gambar 3.8 merupakan diagram *flow chart* proses pembobotan fitur.



Gambar 3.8: *Flow Chart* Proses *Feature Weighting*

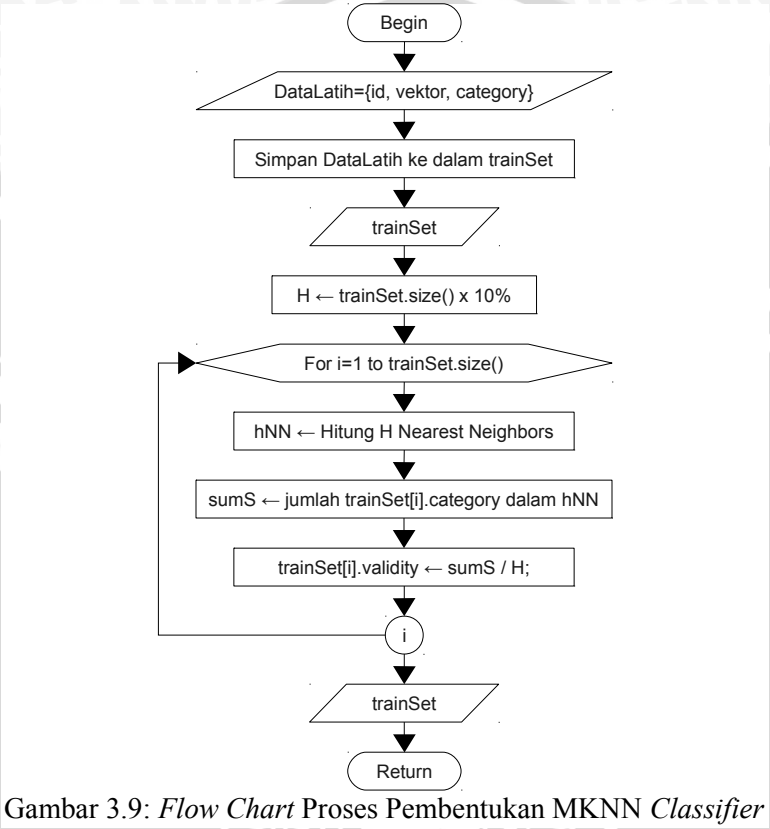
3.4.2 Rancangan Proses Pembentukan *Classifier*

Classifier yang dibentuk adalah *Modified K-Nearest Neighbor* (MKNN) *classifier*. Proses pembentukan *classifier* menyimpan vektor data latih dan menghitung nilai *validity*-nya.

MKNN termasuk jenis *instance-based learning* yang konstruksi pembelajarannya langsung dari sampel data latih. Oleh sebab itu, tujuan utama proses pembentukan MKNN adalah membentuk *trainset* yang merupakan set data latih.

Proses pembentukan MKNN *classifier*, selain menyimpan atribut vektor dan kategori data latih, juga menyimpan atribut tambahan yaitu *validity*. Nilai *validity* untuk setiap data latih dapat dihitung setelah semua data latih tersimpan dalam *trainset*. Untuk menghitung nilai *validity* suatu sampel data latih, mula-mula dicari sebanyak *H* sampel data terdekatnya (atau *h Nearest Neighbor*, hNN). Kemudian, nilai *validity* (Rumus 2.5) dihitung dari jumlah kategori yang sama dengan kategorinya data latih tersebut dalam hNN dibagi dengan nilai *H*.

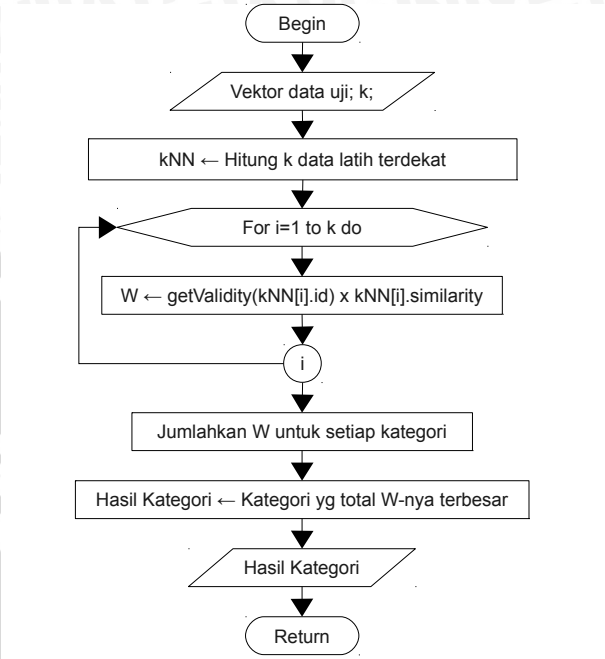
Flow chart proses pembentukan MKNN *classifier* digambarkan seperti pada Gambar 3.9 berikut ini.



Gambar 3.9: *Flow Chart* Proses Pembentukan MKNN *Classifier*

3.4.3 Rancangan Proses Pengklasifikasian

Proses pengklasifikasian ini berdasarkan algoritma MKNN. Tujuan utamanya adalah menghasilkan prediksi kategori untuk vektor data uji.



Gambar 3.10: *Flow Chart* Proses Pengklasifikasian

Pertama-tama, menemukan sebanyak *k* data latih yang terdekat atau *k Nearest Neighbor* (kNN) dari data uji. Tingkat kedekatan data dihitung berdasarkan *cosine similarity* (Rumus 2.9).

Kemudian, masing-masing dari kNN tersebut dihitung bobotnya. Parvin, dkk., (2008) merumuskan fungsi pembobotan kNN ini seperti pada Rumus 2.7. Rumus tersebut pada intinya merupakan perkalian nilai *validity* dan hasil *Euclidean Distance* (Rumus 2.8) antara vektor data latih dan data uji. Namun dalam hal klasifikasi teks, fungsi kesamaaan *Cosine Similarity* (Rumus 2.9) memberikan hasil lebih baik dibandingkan *Euclidean Distance* (Manning, dkk., 2009). Oleh sebab itu, fungsi pembobotan data latih dimodifikasi menggunakan *Cosine Similarity* menjadi Rumus 3.1 berikut ini.

$$W(i) = Validity(i) \times Cosine(\vec{d}_{uji}, \vec{d}_i) \quad (3.1)$$

W(i) dan *Validity(i)* adalah bobot dan nilai *validity* data latih terdekat ke-*i*. $Cosine(\vec{d}_{uji}, \vec{d}_i)$ merupakan fungsi *cosine similarity* antara vektor data uji dan data latih terdekat ke-*i*. Sedangkan *i*

menunjukkan indeks data latih yang berkisar dari i=1 hingga i=K.

Dengan demikian, modifikasi fungsi pembobotan menjadi Rumus 3.1 akan tetap memberikan efek bobot yang besar pada sampel referensi dengan nilai *validity* yang lebih besar dan berdekatan dengan sampel uji.

Selanjutnya, nilai hasil pembobotan kNN tersebut dijumlahkan untuk setiap kategori yang sama. Terakhir, kategori yang jumlah bobotnya terbesar dipilih sebagai hasil prediksi.

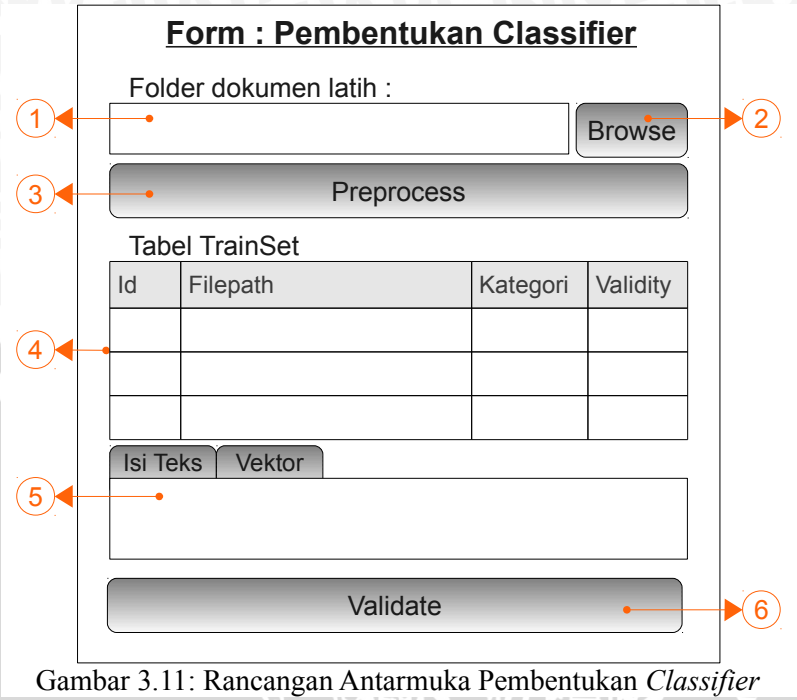
3.5 Rancangan Antarmuka Grafis

Rancangan antarmuka grafis aplikasi terdiri dari tiga *form* : (1) pembentukan *classifier*, (2) dan pengklasifikasian.

Pertama, rancangan antarmuka untuk pembentukan *classifier*. Rancangan antarmuka ini terdiri dari beberapa komponen yaitu:

1. *Edit box*, untuk memasukkan alamat folder yang berisi dokumen latih.
2. Tombol berlabel "Browse", membuka *window* yang berguna untuk menelusuri alamat folder dokumen latih.
3. Tombol berlabel "Pre-process", untuk menerapkan tahap pra-pemrosesan pada semua dokumen latih.
4. Tabel untuk menampilkan TrainSet yang terbentuk dari hasil pra-pemrosesan dokumen latih.
5. Panel *Tab* menampilkan detail data dari tabel (4), berupa isi teks dokumen dan vektor.
6. Tombol berlabel "Validate", untuk menghitung nilai validity semua data latih dalam tabel.

Gambar 3.11 berikut merupakan rancangan antarmuka untuk pembentukan *classifier.*

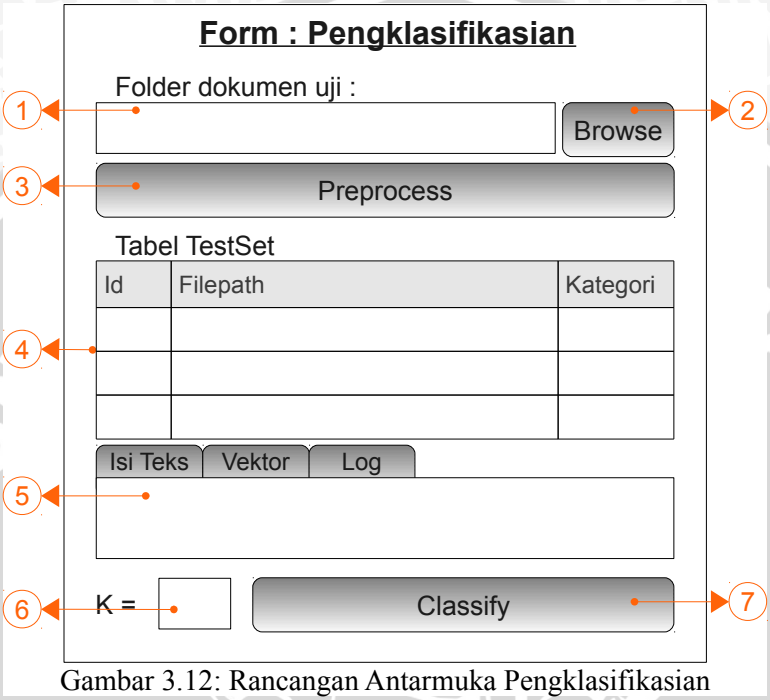


Gambar 3.11: Rancangan Antarmuka Pembentukan *Classifier*

Kedua, rancangan antarmuka untuk pengklasifikasian dokumen uji. Rancangan antarmuka ini terdiri dari beberapa komponen yaitu:

1. *Edit box*, untuk memasukkan alamat folder yang berisi dokumen uji.
2. *Button* berlabel "Browse", membuka *window* yang berguna untuk menelusuri alamat folder dokumen uji.
3. Tombol berlabel "Pre-process", untuk menerapkan tahap pra-pemrosesan pada semua dokumen uji.
4. Tabel untuk menampilkan daftar dokumen setelah melalui pra-pemrosesan (3).
5. Panel *Tab* menampilkan detail data dari tabel (4), berupa isi teks dokumen, vektor, dan rekaman log proses klasifikasi.
6. *Edit box* untuk memasukkan parameter K untuk proses klasifikasi.
7. Tombol berlabel "Classify", untuk mengklasifikasikan semua dokumen uji dalam tabel (4).

Gambar 3.12 berikut merupakan rancangan antarmuka untuk pengklasifikasian dokumen.



Gambar 3.12: Rancangan Antarmuka Pengklasifikasian

3.6 Rancangan Pengujian

Pengujian dilakukan untuk mengetahui keberhasilan dan keefektifan sistem yang dikembangkan. Sistem diuji dengan dokumen masukan yang telah ditentukan dan kemudian dilakukan evaluasi. Selain itu, sebagai perbandingan, hasil pengujian sistem (MKNN) juga akan dibandingkan dengan hasil pengujian sistem yang berbasis KNN.

3.6.1 Sumber Dokumen Masukan

Sumber dokumen latih dan dokumen uji pada penelitian ini diunduh dari situs berita *online* berbahasa Indonesia, http://www.tempointeraktif.com/. Dokumen teks berita diunduh dari tujuh kategori berita berdasarkan rentang waktu terbit selama tanggal

1, 2, dan 3 bulan Desember tahun 2010.

Tabel 3.1: Sebaran Dokumen Masukan

Kategori	Dok. Latih	Dok. Uji	Jumlah
Olahraga	49	48	97
Nasional	113	112	225
Bisnis	60	59	119
Internasional	23	23	46
Teknologi	19	18	37
Nusa	95	94	189
Metro	34	33	67
Jumlah	**393**	**387**	**780**

Total dokumen yang digunakan adalah sejumlah 780 dokumen yang terdiri dari 7 kategori berita. Jumlah sampel dokumen latih (393 dokumen) dan dokumen uji (387 dokumen) dibagi secara acak berdasarkan proporsi yang berimbang untuk setiap kategori.

3.6.2 Skema Pengujian

Skema pengujian ini dilakukan untuk sistem pengklasifikasi teks berbasis MKNN dan (sebagai perbandingan) sistem pengklasifikasi teks berbasis KNN biasa. Tingkat akurasi sistem pengklasifikasi diketahui berdasarkan hasil klasifikasi dibandingkan dengan kategori sebenarnya dari sumber dokumen. Metode evaluasi yang digunakan adalah standar ukuran evaluasi *recall* (Rumus 2.10), *precision* (Rumus 2.11) dan F_1 *measure* (Rumus 2.12).

Skema pengujian sistem ini dilakukan melalui beberapa langkah seperti berikut.

1. Semua dokumen latih diproses untuk membentuk *classifier.*
2. Semua dokumen uji diproses untuk mendapatkan hasil prediksi kategorinya. Langkah ini dilakukan dengan nilai K yang berbeda-beda untuk mendapatkan hasil yang terbaik.
3. Hasil pengujian dievaluasi dengan menggunakan metode evaluasi *recall*, *precision,* dan F_1 *measure*.

Hasil evaluasi pengujian untuk setiap nilai K dirangkum ke dalam Tabel 3.2 berikut ini.

Tabel 3.2: Model Tabel Hasil Evaluasi Pengujian

Kategori	K = ...		
	Recall	Precision	F_1 Measure
Rata-rata	...	...	...

3.7 Contoh Proses Manual

Misalkan korpus dokumen latih pada Tabel 3.3 berikut ini.

Tabel 3.3: Contoh Korpus Dokumen Latih

Id	Teks	Kategori
d0	Bakteri Menggerogoti Bangkai Kapal Titanic	Sains
d1	Bangkai Titanic Habis dalam 20 Tahun	Sains
d2	Gubernur Jawa Tengah Tolak Usulan Pemekaran Wilayah	Nusa
d3	Gubernur NTT Pertanyakan Pengurangan Anggaran	Nusa
d4	Gubernur Jawa Timur Resmikan 21 SMP Satu Atap	Nusa
d5	12 Klub Lokal Tolak PSM Keluar dari Liga Indonesia	Olahraga
d6	Mundur, PSM Bergabung dengan Liga Primer Indonesia	Olahraga

Dokumen uji dimisalkan seperti pada Tabel 3.4 berikut ini.

Tabel 3.4: Contoh Dokumen Uji

Id	Teks	Kategori
dX	Liga Indonesia Tunggu Izin Kepala Polda Jawa Barat Hingga Pukul 8.00 WIB	?

3.7.1 Contoh Pra-pemrosesan

3.7.1.1 Contoh Proses *Case Folding* dan *Tokenization*

Proses ini melalui dua langkah yaitu menyeragamkan bentuk

huruf kemudian mengurai teks. Pertama, *case folding* menyeragamkan semua huruf menjadi huruf kecil. Kedua, *tokenization* mengurai teks menjadi kata yang terpisah-pisah dengan menghapus semua tanda baca dan angka. Contoh hasilnya ditunjukkan di Tabel 3.5.

Tabel 3.5: Contoh Hasil *Case Folding* dan *Tokenization*

Id	**Hasil *Case Folding* dan *Tokenization***
d0	bakteri | menggerogoti | bangkai | kapal | titanic
d1	bangkai | titanic | habis | dalam | tahun
d2	gubernur | jawa | tengah | tolak | usulan | pemekaran | wilayah
d3	gubernur | ntt | pertanyakan | pengurangan | anggaran
d4	gubernur | jawa | timur | resmikan | smp | satu | atap
d5	klub | lokal | tolak | psm | keluar | dari | liga | indonesia
d6	mundur | psm | bergabung | dengan | liga | primer | indonesia
dX	liga | indonesia | tunggu | izin | kepala | polda | jawa | barat | hingga | pukul | wib

3.7.1.2 Contoh Proses *Stop Words Removal*

Misalkan Tabel 3.6 berikut ini adalah sekumpulan kata-kata yang dianggap tidak penting (*stop words*).

Tabel 3.6: Contoh *Stop Words*

dalam	pertanyakan
dari	pukul
dengan	satu
hingga	tahun
keluar	tengah

Kemudian setiap kata dari hasil proses sebelumnya (Tabel 3.5) diseleksi dengan menghapus kata-kata yang tidak penting, maka hasilnya menjadi seperti pada Tabel 3.7 berikut ini.

Tabel 3.7: Contoh Hasil *Stop Words Removal*

Id	Hasil *Stop Words Removal*
d0	bakteri | menggerogoti | bangkai | kapal | titanic
d1	bangkai | titanic | habis
d2	gubernur | jawa | tolak | usulan | pemekaran | wilayah
d3	gubernur | ntt | pengurangan | anggaran
d4	gubernur | jawa | timur | resmikan | smp | atap
d5	klub | lokal | tolak | psm | liga | indonesia
d6	mundur | psm | bergabung | liga | primer | indonesia
dX	liga | indonesia | tunggu | izin | kepala | polda | jawa | barat | wib

3.7.1.3 Contoh Proses *Stemming*

Proses mengurai suatu kata menjadi bentuk kata dasarnya. Acuan yang digunakan adalah algoritma *Porter Stemmer* untuk Bahasa Indonesia berdasarkan penelitian Tala (2003). Tabel 3.8 berikut ini merupakan contoh hasilnya. Kata yang tercetak miring merupakan kata yang terkena proses *stemming*.

Tabel 3.8: Contoh Hasil *Stemming*

Id	Hasil *Stemming*
d0	*bakter* | *gerogot* | *bangka* | kapal | titanic
d1	*bangka* | titanic | habis
d2	gubernur | jawa | tolak | *usul* | *pekar* | wilayah
d3	gubernur | ntt | *urang* | *anggar*
d4	gubernur | jawa | timur | *resmi* | smp | atap
d5	klub | lokal | tolak | psm | liga | indonesia
d6	mundur | psm | *gabung* | liga | primer | indonesia
dX	liga | indonesia | tunggu | izin | *pala* | polda | jawa | barat | wib

3.7.1.4 Contoh Proses Pembentukan *Dictionary*

Dictionary (*inverted index*) dibentuk dengan mengumpulkan kata unik dari semua dokumen (*term list*) dan menyimpan data

kemunculan setiap kata unik beserta frekuensinya (*posting list*).

Tabel 3.9: Contoh Hasil Pembentukan *Dictionary*

Term List	***Posting List***
atap	{d4=1}
lokal	{d5=1}
gubernur	{d2=1, d3=1, d4=1}
bakter	{d0=1}
klub	{d5=1}
pekar	{d2=1}
tolak	{d2=1, d5=1}
habis	{d1=1}
ntt	{d3=1}
mundur	{d6=1}
urang	{d3=1}
indonesia	{d5=1, d6=1}
psm	{d5=1, d6=1}
resmi	{d4=1}
timur	{d4=1}
jawa	{d2=1, d4=1}
gerogot	{d0=1}
bangka	{d0=1, d1=1}
smp	{d4=1}
liga	{d5=1, d6=1}
gabung	{d6=1}
kapal	{d0=1}
usul	{d2=1}
titanic	{d0=1, d1=1}
primer	{d6=1}
wilayah	{d2=1}
anggar	{d3=1}

3.7.1.5 Contoh Proses *Feature Weighting*

Setiap kata (dalam *dictionary*) dihitung frekuensinya dalam dokumen uji kemudian dikalikan dengan nilai *idf*-nya masing-masing. Sedangkan setiap kata dalam dokumen uji yang tidak ada dalam *dictionary*, tidak dihitung atau diabaikan saja sehingga terbentuk vektor bobot yang sama ukuran dimensinya. Kemudian, bobot kata dalam dokumen dihitung berdasarkan skema pembobotan *tf.idf* (Rumus 2.1). Dengan demikian, bobot setiap kata untuk masing-masing dokumen ditunjukkan seperti pada Tabel 3.10 berikut ini.

Tabel 3.10: Contoh Hasil *Feature Weighting*

Term	IDF	TF.IDF							
		d0	d1	d2	d3	d4	d5	d6	dX
atap	0.845	0.0	0.0	0.0	0.0	0.845	0.0	0.0	0.0
lokal	0.845	0.0	0.0	0.0	0.0	0.0	0.845	0.0	0.0
gubernur	0.368	0.0	0.0	0.368	0.368	0.368	0.0	0.0	0.0
bakter	0.845	0.845	0.0	0.0	0.0	0.0	0.0	0.0	0.0
klub	0.845	0.0	0.0	0.0	0.0	0.0	0.845	0.0	0.0
pekar	0.845	0.0	0.0	0.845	0.0	0.0	0.0	0.0	0.0
tolak	0.544	0.0	0.0	0.544	0.0	0.0	0.544	0.0	0.0
habis	0.845	0.0	0.845	0.0	0.0	0.0	0.0	0.0	0.0
ntt	0.845	0.0	0.0	0.0	0.845	0.0	0.0	0.0	0.0
mundur	0.845	0.0	0.0	0.0	0.0	0.0	0.0	0.845	0.0
urang	0.845	0.0	0.0	0.0	0.845	0.0	0.0	0.0	0.0
indonesia	0.544	0.0	0.0	0.0	0.0	0.0	0.544	0.544	0.544
psm	0.544	0.0	0.0	0.0	0.0	0.0	0.544	0.544	0.0
resmi	0.845	0.0	0.0	0.0	0.0	0.845	0.0	0.0	0.0
timur	0.845	0.0	0.0	0.0	0.0	0.845	0.0	0.0	0.0
jawa	0.544	0.0	0.0	0.544	0.0	0.544	0.0	0.0	0.544
gerogot	0.845	0.845	0.0	0.0	0.0	0.0	0.0	0.0	0.0
bangka	0.544	0.544	0.544	0.0	0.0	0.0	0.0	0.0	0.0
smp	0.845	0.0	0.0	0.0	0.0	0.845	0.0	0.0	0.0

Term	IDF	TF.IDF							
		d0	d1	d2	d3	d4	d5	d6	dX
liga	0.423	0.0	0.0	0.0	0.0	0.0	0.544	0.544	0.544
gabung	0.544	0.0	0.0	0.0	0.0	0.0	0.0	0.845	0.0
kapal	0.845	0.845	0.0	0.0	0.0	0.0	0.0	0.0	0.0
usul	0.845	0.0	0.0	0.845	0.0	0.0	0.0	0.0	0.0
titanic	0.544	0.544	0.544	0.0	0.0	0.0	0.0	0.0	0.0
primer	0.845	0.0	0.0	0.0	0.0	0.0	0.0	0.845	0.0
wilayah	0.845	0.0	0.0	0.845	0.0	0.0	0.0	0.0	0.0
anggar	0.845	0.0	0.0	0.0	0.845	0.0	0.0	0.0	0.0

3.7.2 Contoh Proses Pembentukan *Classifier*

Proses pembentukan *classifier* menyimpan data latih ke dalam *trainset* yang terdiri dari atribut : id, vektor, kategori, dan atribut *validity*. Untuk menghitung nilai validity, nilai *H* ditentukan berdasarkan 10% dari jumlah data latih, namun karena *trainset* hanya berisi 7 data maka untuk contoh ini dimisalkan saja nilai *H*=3. Sebagai contoh proses perhitungan nilai *validity*, dokumen d2 (kategori Nusa) dipilih sebagai sampel.

Pertama, semua data latih (kecuali d2) diurutkan berdasarkan tingkat kesamaannya dengan data d2. Tingkat kesamaan dihitung dengan *Cosine Similarity* (Rumus 2.9). Hasilnya pada Tabel 3.11 berikut.

Tabel 3.11: Contoh Hasil *Cosine*(d0,dn)

Id	Kategori	*Cosine*(d0,dn)
d4	Nusa	0.140
d5	Olahraga	0.108
d3	Nusa	0.053
d6	Olahraga	0.0
d1	Sains	0.0
d0	Sains	0.0

Kedua, misalkan *H*=3 data dengan nilai *Cosine Similarity*

terbesar, yaitu d4, d5, dan d3. Kemudian, hitung nilai ΣS (Rumus 2.5) yaitu banyaknya kategori yang sama dengan kategorinya d2 (Nusa), maka jumlah ΣS=2. Dengan demikian, nilai *validity* untuk d2 adalah sebesar (2/3)=0.67. Adapun hasil nilai *validity* untuk keseluruhan data (dengan nilai H=3) seperti pada Tabel 3.12 berikut.

Tabel 3.12: Contoh Nilai *Validity*

Id	Kategori	*Validity*
d0	Sains	0.33
d1	Sains	0.33
d2	Nusa	0.67
d3	Nusa	0.67
d4	Nusa	0.67
d5	Olahraga	0.33
d6	Olahraga	0.33

3.7.3 Contoh Proses Pengklasifikasian

Berdasarkan vektor-vektor data di Tabel 3.10 dan nilai *validity*-nya di Tabel 3.12. Kemudian misalkan nilai parameter K=3, maka data uji dX dapat diklasifikasikan menggunakan MKNN *classifier*. Apakah termasuk kategori "Sains", "Nusa", atau "Olahraga" ?

Pertama-tama, data uji dibandingkan tingkat kesamaannya dengan masing-masing data latih. Tingkat kesamaan dihitung menggunakan *Cosine Similarity* (Rumus 2.9). Kemudian, hasil *Cosine Similarity* antara vektor data uji dX dengan semua data latih diurutkan dari yang terbesar hingga terkecil. Hasilnya ditunjukkan di Tabel 3.13 berikut.

Tabel 3.13: Contoh Hasil *Cosine*(dX,dn)

Id	Kategori	*Cosine*(dX,dn)
d6	Olahraga	0.361
d5	Olahraga	0.389
d2	Nusa	0.185
d4	Nusa	0.173
d3	Nusa	0.0
d1	Sains	0.0
d0	Sains	0.0

Pembobotan dilakukan untuk setiap *K* (dimisalkan *K*=3) data latih terdekat. Fungsi pembobotan berdasarkan Rumus 3.1. Perhitungan bobot ini menggunakan data nilai *validity* (Tabel 3.12) dan data hasil *Cosine Similarity* (Tabel 3.13). Berikut ini adalah Tabel 3.14 berisi hasil pembobotan *K*=3 data latih terdekat.

Tabel 3.14: Contoh Hasil Pembobotan (K=3) Data Latih Terdekat

Id	Kategori	*Validity*	*Cosine*	*Validity* x *Cosine*
d6	Olahraga	0.33	0.361	0.11913
d5	Olahraga	0.33	0.389	0.12837
d2	Nusa	0.67	0.185	0.12395

Hasil prediksi kategori untuk data dX ditentukan berdasarkan skor kategori yang terbesar. Skor kategori ini dihitung dengan menjumlahkan bobot setiap data latih (dari K yang terdekat) yang berkategori sama. Berdasarkan Tabel 3.14, kandidat kategori yaitu : "Olahraga" dan "Nusa". Hasil skornya pada Tabel 3.15 berikut ini.

Tabel 3.15: Contoh Hasil Skor Kategori

Kategori	Skor
Olahraga	0.11913 + 0.12837 = 0.24750
Nusa	0.12395

Hasil skor kategori pada Tabel 3.15 menunjukkan bahwa skor kategori "Olahraga" lebih besar daripada "Nusa". Dengan demikian, data uji dX diklasifikasikan termasuk kategori "Olahraga".

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Lingkungan Implementasi

Perangkat keras yang digunakan dalam implementasi sistem ini adalah satu set komputer dengan spesifikasi utama sebagai berikut :

1. Prosesor : Intel® Core™ 2 Duo T7100 (2 MB L2 cache, 1.8 GHz, 800 MHz FSB)
2. Memori : 2 GB (DDR2 667 MHz)

Perangkat lunak yang digunakan dalam implementasi sistem ini adalah sebagai berikut :

1. Sistem Operasi : Ubuntu 10.10 (Kernel Linux 2.6.35)
2. Bahasa Pemrograman : Java (OpenJDK 6)
3. IDE Tool : Netbeans IDE 6.9.1

4.2 Implementasi Program

4.2.1 Implementasi Pra-Pemrosesan

Pra-pemrosesan diimplementasikan dalam bentuk *class* Preprocessor. *Class* Preprocessor terdiri dari beberapa variabel dan *method*. Gambar 4.1 berikut ini merupakan spesifikasi dari *class* Preprocessor.

Class : **Preprocessor**
Constructor
• `public Preprocessor()`
Fields
• `HashMap<String, HashSet> CategoryMap;` • `HashMap<String, HashMap<String, Integer>> InvIndex;` • `HashSet<String> StopWordSet;` • `int totalDoc;`
Methods
• `public void initStopWordSet(String StopWords)` • `public Object index(String teks, String kategori)` • `public Double[] buildVektor(Object id)`

```
public Double[] buildVektor(String teks)
private HashMap<String, Integer>
tfMap(ArrayList<String> wordList)
```

Imports

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Iterator;
import java.util.Map.Entry;
import java.util.regex.Pattern;
```

Gambar 4.1: Spesifikasi *Class* Preprocessor

Class Preprocessor memiliki satu *constructor* yaitu :

- Preprocessor()
  Constructor ini menginisiasi variabel-variabel dalam *class*.

```
public Preprocessor() {
  this.CategoryMap = new HashMap();
  this.InvIndex = new HashMap();
  this.totalDoc = 0;
  this.WordPattern = Pattern.compile("[^a-z]+");
  this.StopWordSet = new HashSet();
}
```

Gambar 4.2: *Source Code Constructor* Preprocessor

Class Preprocessor memiliki empat *field* variabel yaitu :

- CategoryMap
  Variabel ini berfungsi untuk menyimpan data kategori dan daftar dokumen kategori tersebut. `CategoryMap` bertipe `HashMap` dengan *keys*-nya (`String`) merupakan data kategori sedangkan *values*-nya (`HashSet`) adalah daftar id dokumen yang termasuk kategori tersebut.
- InvIndex
  Variabel ini berfungsi sebagai *inverted index*. `InvIndex` bertipe `HashMap` dengan *keys*-nya (`String`) merupakan data term unik sedangkan *values*-nya (`HashMap<String, Integer>`) adalah tabel *hash* yang menyimpan data id dan frekuensinya.
- totalDoc

Variabel ini berfungsi untuk menyimpan jumlah dokumen latih.

- StopWordSet
  Variabel ini berfungsi untuk menyimpan *stop words* yaitu daftar kata-kata tidak penting.

Class Preprocessor memiliki empat *public method* yaitu :

- initStopWordSet(String StopWords)
  Method ini berfungsi untuk mengisi `StopWordSet` berdasarkan parameter `StopWords`.
- index(String teks, String kategori)
  Method ini berfungsi untuk melakukan pra-pemrosesan pada dokumen latih. Parameter `teks`, dan `kategori` merupakan atribut dari dokumen latih. Hasil *return* bertipe `Object` yang merupakan kode *identifier* untuk dokumen tersebut.
- buildVektor(Object id)
  Method ini berfungsi untuk membentuk vektor bobot dokumen latih. Parameter `id` merupakan *identifier* dokumen latih yang dimaksud. Hasil *return* berupa vektor bobot yang bertipe *array* (`Double[]`). *Method* ini digunakan jika semua dokumen latih telah diproses oleh *method* `index`.
- buildVektor(String teks)
  Method ini berfungsi untuk membentuk vektor bobot dokumen uji. Parameter `teks` merupakan teks dokumen yang dimaksud. Hasil *return* berupa vektor bobot yang bertipe *array* (`Double[]`). *Method* ini digunakan jika semua dokumen latih telah diproses oleh *method* `index`.

Class Preprocessor memiliki satu *private method* yaitu :

- tfMap(List<String> wordList)
  Method ini berfungsi untuk menghitung frekuensi setiap term dalam `wordList`. Hasil *return* bertipe `HashMap` yang *keys*-nya (`String`) adalah term sedangkan *values*-nya (`Integer`) adalah frekuensi term tersebut.

4.2.1.1 Implementasi *Case Folding* dan *Tokenization*

Case folding dan *tokenization* diimplementasikan seperti pada Gambar 4.3 berikut ini.

```
// case folding
teks = teks.toLowerCase();

// tokenization
ArrayList<String> wordList = new ArrayList();
wordList.addAll(Arrays.asList(WordPattern.split(teks)));
```

Gambar 4.3: *Source Code Case Folding* dan *Tokenization*

Case folding menggunakan *method* `toLowerCase()` untuk mengubah setiap huruf dalam `teks` yang bertipe `String` menjadi huruf kecil.

Proses *tokenization* menggunakan *method* `split` (dari pola `WordPattern`) untuk memecah teks menjadi bentuk per kata. Hasil *return* dari method `split` berbentuk *array* kemudian diubah menjadi bentuk `List` untuk disimpan dalam variabel `wordList` (yang bertipe `ArrayList`). `WordPattern` dibentuk berdasarkan pola *regex* `"[^a-z]+"`, di dalam *constructor* `Preprocessor` (Gambar 4.2). Pola `"[^a-z]+"` merupakan *regular expression* yang berarti himpunan karakter selain 'a' hingga 'z'.

4.2.1.2 Implementasi *Stop Words Removal*

Implementasi *stop words removal* menggunakan variabel `StopWordSet` dalam *class* Preprocessor, seperti pada Gambar 4.4 berikut ini.

```
wordList.removeAll(StopWordSet);
```

Gambar 4.4: *Source Code Stop Words Removal*

Stop words removal diimplementasikan setelah proses *tokenization* yang menyimpan hasilnya pada `wordList` (bertipe `ArrayList`). *Stop words removal* menggunakan *method* `removeAll` yang terdapat dalam *class* `ArrayList` berdasarkan parameter `StopWordSet`.

4.2.1.3 Implementasi *Stemming*

Stemming diimplementasikan dalam bentuk *class* `Stemmer` yang terdiri dari beberapa delapan *static method* yaitu :

- stem

- isVocal
- countSukuKata
- delPartikel
- delPosesif
- delAwalan1
- delAwalan2
- delAkhiran

Method stem berperan sebagai *method* utama yang menerapkan proses *stemming* secara keseluruhan dengan mengembalikan bentuk kata dasar dari `String t`. *Method* `stem` diimplementasikan seperti pada Gambar 4.5 berikut ini.

```
public static String stem(String t) {
  t = delPartikel(t);
  t = delPosesif(t);
  String t0 = t;
  t = delAwalan1(t);
  if (!t.equals(t0)) { //jika ada awalan1 yg terhapus
    String awalan = t0.substring(t0.length()
                  - t.length(), t0.length());
    t0 = t;
    t = delAkhiran(awalan, t);
    if (!t.equals(t0)) { //jika ada akhiran yg terhapus
      t = delAwalan2(t);
    }
  } else { // jika tidak ada awalan1 yg terhapus
    t = delAwalan2(t);
    String awalan = t0.substring(t0.length()
                  - t.length(), t0.length());
    t = delAkhiran(awalan, t);
  }
  return t;
}
```

Gambar 4.5: *Source Code Method* stem

Method isVocal berfungsi untuk mengenali jenis huruf vokal. *Method* mengembalikan nilai `true` jika huruf vokal, `false` jika huruf konsonan.

```
private static boolean isVocal(char c) {
  return (c=='a'||c=='i'||c=='u'||c=='e'||c=='o');
}
```

Gambar 4.6: *Source Code Method* isVocal

Method countSukuKata berfungsi untuk menghitung suku kata. Suku kata dihitung berdasarkan kemunculan huruf vokal dan diftong 'au'.

```
private static int countSukuKata(String t) {
  int r = 0;
  for (int i = 0; i < t.length(); i++) {
    if (isVocal(t.charAt(i))) {
      if ((t.charAt(i)=='a') && (i+1<t.length())) {
        if (t.charAt(i+1)=='u') {
          i++;
        }
      }
      r++;
    }
  }
  return r;
}
```

Gambar 4.7: *Source Code Method* countSukuKata

Method delPartikel berfungsi untuk menghapus imbuhan partikel (-kah, -lah, -pun) dalam kata. Jika jumlah suku kata setelah penghapusan lebih kecil dari dua, maka *method* mengembalikan kata aslinya (kata sebelum penghapusan partikel).

```
private static String delPartikel(String t) {
  String t0 = t;
  if (t.endsWith("kah")||t.endsWith("lah")
        || t.endsWith("pun")) {
    t = t.substring(0, t.length()-3);
  }
  return (countSukuKata(t) < 2) ? t0 : t;
}
```

Gambar 4.8: *Source Code Method* delPartikel

Method delPosesif berfungsi untuk menghapus imbuhan posesif (ku-, kau-, -ku, -mu, -nya) dalam kata. Jika jumlah suku kata setelah penghapusan lebih kecil dari dua, maka *method*

mengembalikan kata aslinya (kata sebelum penghapusan imbuhan posesif).

```
private static String delPosesif(String t) {
  String t0 = t;
  if (t.startsWith("ku")) {
      t = t.substring(2);
  } else if(t.startsWith("kau")) {
      t = t.substring(3);
  } else if(t.endsWith("ku")||t.endsWith("mu")){
      t = t.substring(0, t.length()-2);
  } else if(t.endsWith("nya")) {
      t = t.substring(0, t.length()-3);
  }
  return (countSukuKata(t) < 2) ? t0 : t;
}
```

Gambar 4.9: *Source Code Method* delPosesif

Method delAwalan1 berfungsi untuk menghapus awalan pertama (meng-, meny-, men-, mem-, me-, peng-, peny-, pen-, pem-, di-, ter-, ke-) dalam kata. Jika jumlah suku kata setelah penghapusan lebih kecil dari dua, maka *method* mengembalikan kata aslinya (kata sebelum penghapusan awalan pertama).

```
private static String delAwalan1(String t) {
  String t0 = t;
  if (t.startsWith("meng") || t.startsWith("peng")) {
      t = t.substring(4);
  } else if ((t.startsWith("meny")
          || t.startsWith("peny")) && t.length()>4) {
      if (isVocal(t.charAt(4))) {
        t = 's' + t.substring(4);
      }
  } else if ((t.startsWith("pem")
          || t.startsWith("mem")) && (t.length()>3)) {
      String tmp = t.substring(3);
      t = isVocal(t.charAt(3)) ? ('p' + tmp) : tmp;
  } else if (t.startsWith("pen")
              || t.startsWith("men")
              || t.startsWith("ter")) {
      t = t.substring(3);
  } else if (t.startsWith("me") || t.startsWith("di")
              || t.startsWith("ke")) {
      t = t.substring(2);
  }
```

```
  return (countSukuKata(t) < 2) ? t0 : t;
}
```

Gambar 4.10: *Source Code Method* delAwalan1

Method delAwalan2 berfungsi untuk menghapus awalan kedua (ber-, bel-, be-, per-, pel-, pe-) dalam kata. Jika jumlah suku kata setelah penghapusan lebih kecil dari dua, maka *method* mengembalikan kata aslinya (kata sebelum penghapusan awalan kedua).

```
private static String delAwalan2(String t) {
  String t0 = t;
  if (t.startsWith("ber")
        || t.startsWith("per")
        || t.startsWith("belajar")
        || t.startsWith("pelajar")) {
      t = t.substring(3);
  } else if (t.startsWith("pe")||(t.startsWith("be"))){
      if (t.length() > 2) {
        if (!isVocal(t.charAt(2))) {
          t = t.substring(2);
        }
      }
  }
  return (countSukuKata(t) < 2) ? t0 : t;
}
```

Gambar 4.11: *Source Code Method* delAwalan2

Method delAkhiran berfungsi untuk menghapus akhiran (-kan, -an, -i) dalam kata. Jika jumlah suku kata setelah penghapusan lebih kecil dari dua, maka *method* mengembalikan kata aslinya (kata sebelum penghapusan akhiran).

```
private static String delAkhiran(String awalan,
                                 String t) {
  String t0 = t;
  if (t.endsWith("kan")
        && !(awalan.equals("ke")
             || awalan.equals("peng"))) {
      t = t.substring(0, t.length() - 3);
  } else if (t.endsWith("an")
              && !(awalan.equals("di")
                   || awalan.equals("meng")
                   || awalan.equals("ter"))) {
```

```
      t = t.substring(0, t.length() - 2);
  } else if (t.endsWith("i")
              && !(awalan.equals("ber")
                  ||awalan.equals("ke")
                  || awalan.equals("peng"))) {
      t = t.substring(0, t.length() - 1);
  }
  return (countSukuKata(t) < 2) ? t0 : t;
}
```

Gambar 4.12: *SourceCode Method* delAkhiran

4.2.1.4 Implementasi Pembentukan *Dictionary*

Dictionary inverted index diimplementasikan dalam *class* Preprocessor dalam bentuk variabel `InvIndex`. *Method* untuk memasukkan dokumen dalam `InvIndex` diimplementasikan dalam *method* `index` seperti pada Gambar 4.13.

```
public int index(String teks, String kategori) {

  // case folding & tokenization
  teks = teks.toLowerCase();
  ArrayList<String> wordList = new ArrayList();
  wordList.addAll(Arrays.asList(WordPattern.split(teks)));

  // hapus stop words
  wordList.removeAll(StopWordSet);

  // stemming
  for (int i = 0; i < wordList.size(); i++) {
    wordList.set(i, Stemmer.stem(wordList.get(i)));
  }

  // masukkan kategori (jika belum ada) ke CategoryMap
  if (!CategoryMap.containsKey(kategori)) {
    CategoryMap.put(kategori, new HashSet());

  }
  // buat id dan add ke CategoryMap
  Object id = ++totalDoc;
  CategoryMap.get(kategori).add(id);


  // hitung term frequency dan add ke InvIndex
  Iterator i = tfMap(wordList).entrySet().iterator();
  for (; i.hasNext();) {
    Entry<String, Integer> tf = (Entry) i.next();
```

```
    // masukkan term (jika belum ada) ke InvIndex
    if (!InvIndex.containsKey(tf.getKey())) {
      InvIndex.put(tf.getKey(), new HashMap());
    }
    // masukkan id dan frek term ke InvIndex
    InvIndex.get(tf.getKey()).put(id, tf.getValue());
  }
  return id;
}
```

Gambar 4.13: *Source Code Method* index

Method `index` berfungsi untuk melakukan *indexing* pada dokumen latih. Parameter `teks` dan `kategori` merupakan atribut dari dokumen latih.

Method `tfMap` berfungsi untuk menghitung frekuensi setiap term dalam dokumen. Gambar 4.14 berikut ini adalah implementasi *method* `tfMap`.

```
private HashMap<String, Integer> tfMap(
                          ArrayList<String> wordList) {
  HashMap<String, Integer> tfMap = new HashMap();
  for (int i = 0; i < wordList.size(); i++) {
    Integer f = tfMap.get(wordList.get(i));
    f = (f == null) ? 1 : (f + 1);
    tfMap.put(wordList.get(i), f);
  }
  return tfMap;
}
```

Gambar 4.14: *Source Code Method* tfMap

4.2.1.5 Implementasi *Feature Weighting*

Feature weighting diimplementasikan dalam *class* Preprocessor untuk membentuk vektor numerik dari dokumen. *Method* `buildVektor(Object id)` untuk dokumen latih dan *method* `buildVektor(String teks)` untuk dokumen uji. Hasil *return* kedua *method* bertipe *array* (`Double[]`) yang merupakan representasi vektor dokumen.

Method `buildVektor(Object id)` digunakan untuk membentuk vektor bobot dokumen latih. Parameter `id` merupakan

kode *identifier* dokumen latih yang akan dibentuk vektornya. *Source code method* `buildVektor(Object id)` diimplementasikan seperti pada Gambar 4.15.

```
public Double[] buildVektor(Object id) {

  // inisiasi panjang vektor <- ukuran InvIndex
  Double[] vektor = new Double[InvIndex.size()];
  int j = 0; // j sebagai indeksnya vektor

  // iterasi setiap term dalam InvIndex
  for (Iterator i = InvIndex.keySet().iterator();
       i.hasNext();) {

    //  t <- term ke-i dalam InvIndex
    String t = (String) i.next();

    // tf <- frekuensi term t dalam dokumen latih
    Integer tf = null;
    if (InvIndex.containsKey(t)) {
      tf = InvIndex.get(t).get(id);
    }

    // idf <- inverse document frequency-nya term t
    int df = InvIndex.get(t).size();
    double idf = Math.log10((double)totalDoc / df);

    // Vektor[j] <- tf * idf
    vektor[j] = (tf == null) ? 0 : (tf * idf);
    j++;
  }
  return vektor;
}
```

Gambar 4.15: *Source Code Method* buildVektor(Object id)

Method `buildVektor(String teks)` digunakan untuk membentuk vektor bobot dokumen Uji. Parameter `teks` merupakan isi teks dari dokumen uji yang akan dibentuk vektornya. *Source code method* `buildVektor(String teks)` seperti pada Gambar 4.16.

```
public Double[] buildVektor(String teks) {

  // case folding & tokenization
  List<String> wordList = tokenize(teks);

  // stop words removal
  wordList.removeAll(StopWordSet);
```

```
    // stemming
    for (int i = 0; i < wordList.size(); i++) {
      wordList.set(i, Stemmer.stem(wordList.get(i)));
    }

    // hitung frekuensi setiap term unik dari wordList
    HashMap<String, Integer> tfMap = tfMap(wordList);

    // inisiasi panjang vektor <- ukuran InvIndex
    Double[] Vektor = new Double[InvIndex.size()];
    int j = 0; // j sebagai indeks fitur

    // iterasi setiap term dalam InvIndex
    for (Iterator i = InvIndex.keySet().iterator();
         i.hasNext();) {

      //  t <- term ke-i dalam InvIndex
      String t = (String) i.next();

      // tf <- frekuensi term t dalam dokumen uji
      Integer tf = tfMap.get(t);

      // idf <- inverse document frequency-nya term t
      int df = InvIndex.get(t).size();
      double idf = Math.log10((double)totalDoc / df);

      // Vektor[j] <- tf * idf
      Vektor[j] = (tf == null) ? 0 : (tf * idf);
      j++;
    }
    return Vektor;
}
```

Gambar 4.16: *Source Code Method* buildVektor(String teks)

4.2.2 Implementasi Pembentukan *Classifier*

MKNN *classifier* diimplementasikan dalam bentuk *class* `MKNNClassifier`. Spesifikasi *class* `MKNNClassifier` seperti pada Gambar 4.17 berikut ini.

Class : **`MKNNClassifier`**
Constructor
• `public MKNNClassifier()`
Fields
• `private HashMap<Object, DataLatih> trainSet;`

Methods

```
· public void train(Object id, Double[] vektor, String
  category)
· public void validate(int H)
· public String classify(Double[] vektor, int K)
· private HashMap<Object, Double> knnMap(Double[]
  vektor, int k)
· private double cosine(Double[] u, Double[] v)
```

Imports

```
· import java.util.HashMap;
· import java.util.Iterator;
· import java.util.Map.Entry;
```

Gambar 4.17: Spesifikasi *Class* MKNNClassifier

Constructor `MKNNClassifier` berisi inisiasi variabel `trainSet`.

```
public MKNNClassifier() {
  this.trainSet = new HashMap();
}
```

Gambar 4.18: *Source Code Constructor* MKNNClassifier

Variabel `trainSet` bertipe `HashMap`, menyimpan data latih yang terdiri dari atribut id (*identifier*), vektor, kategori, dan nilai *validity*-nya. Atribut id digunakan sebagai variabel kunci (*key*) dalam `HashMap`, sedangkan atribut vektor, kategori, dan nilai *validity* disimpan (dalam bentuk *class* `DataLatih`) sebagai variabel *value*-nya.

```
class DataLatih {
  Double[] vektor;
  String category;
  Double validity;
  public DataLatih(Double[] vektor,
                   String category, Double validity) {
    this.vektor = vektor;
    this.category = category;
    this.validity = validity;
    }
}
```

Gambar 4.19: *Source Code Class* DataLatih

Proses pembentukan *classifier*, yang menyimpan data latih ke dalam *trainset*, diimplementasikan menggunakan *method* `train` dan *method* `validate`. Fungsi untuk menyimpan data latih ke dalam `trainSet` diimplementasikan sebagai *method* `train`. *Method* ini menyimpan atribut data latih ke dalam `trainSet` yang bertipe `HashMap`. Atribut data latih berasal dari parameter `id`, `vektor`, dan `category`), sedangkan untuk *validity*-nya diinisiasikan bernilai `null`.

```
public void train(Object id, Double[] vektor,
                  String category) {
  trainSet.put(id, new DataLatih(vektor,
                                 category, null));
}
```

Gambar 4.20: *Source Code Method* train

Fungsi untuk menghitung nilai *validity* masing-masing data latih dalam `trainSet` diimplementasikan sebagai *method* `validate`. *Method* ini memiliki parameter `H` untuk digunakan dalam perhitungan nilai *validity* sesuai dengan Rumus 2.5.

```
public void validate(int H) {

  // iterasi untuk setiap datalatih dlm trainSet
  for (Iterator i = trainSet.entrySet().iterator();
       i.hasNext();) {

    // x <- datalatih ke-i
    Entry<Object, DataLatih> x = (Entry) i.next();

    // hnnMap <- H Nearest Neighbor-nya x
    HashMap hnnMap = knnMap(x.getValue().vektor, H+1);
    hnnMap.remove(x.getKey());

    // xc <- kategorinya datalatih x
    String xc = x.getValue().category;

    // inisiasi sumS utk menyimpan jumlah xc dalam hnn
    int sumS = 0;

    // iterasi untuk setiap data latih dalam hnnMap
    for (Iterator n = hnnMap.keySet().iterator();
         n.hasNext();) {

      // nc <- kategorinya data latih ke-n dalam hnnMap
      String nc = trainSet.get(n.next()).category;

      // jika xc == nc, maka sumS ditambahkan 1
```

```
      if (xc.equals(nc)) {
        sumS++;
      }
    }

    // nilai validity-nya data latih x <- sumS / H
    x.getValue().validity = (double) sumS / H;
  }
}
```

Gambar 4.21: *Source Code Method* validate

Fungsi untuk menemukan sebanyak *K* data latih terdekat dari suatu vektor diimplementasikan sebagai *method* `knnMap`. *Method* ini digunakan pada *method* `validate` dan *method* `classify` (proses pengklasifikasian). *Method* ini mengukur kedekatan suatu vektor dengan menggunakan *Cosine Similarity*. Sebanyak `K` data latih dengan *similarity* terbesar disimpan dalam variabel bertipe `HashMap` untuk kemudian menjadi hasil *return* dari *method* ini. *Source Code method* ini seperti pada Gambar 4.22 berikut.

```
private HashMap<Object, Double> knnMap(int K,
                                       Double[] vektor) {
  // map menyimpan id datalatih dan hasil cosine
  HashMap<Object, Double> map = new HashMap(K);
  Object lowest_sim_id = null;
  double lowest_sim = Double.MAX_VALUE;

  // iterasi dan isi map sebanyak K data latih
  Iterator i = trainSet.keySet().iterator();
  for (int k = 0; k < K && i.hasNext(); k++) {
    Object id = i.next();
    Double sim =cosine(vektor,trainSet.get(id).vektor);
    map.put(id, sim);
    if (sim < lowest_sim) {
      lowest_sim_id = id;
      lowest_sim = sim;
    }
  }

  // iterasi lanjutan dlm trainset hingga hanya
  // K similarity terbesar yg tersimpan dlm map
  while (i.hasNext()) {
    Object id = i.next();
    Double sim =cosine(vektor,trainSet.get(id).vektor);
```

```
    if (sim > lowest_sim) {
      map.remove(lowest_sim_id);
      map.put(id, sim);

      // reset lowest_sim
      lowest_sim = Double.MAX_VALUE;
      Iterator j = map.entrySet().iterator();
      while (j.hasNext()) {
        Entry<Object, Double> n = (Entry) j.next();
        if (n.getValue() < lowest_sim) {
          lowest_sim_id = n.getKey();
          lowest_sim = n.getValue();
        }
      }
    }
  }
  return map;
}
```

Gambar 4.22: *Source Code Method* knnMap

4.2.3 Implementasi Pengklasifikasian

Fungsi pengklasifikasian MKNN diimplementasikan dalam *class* `MKNNClassifier` sebagai *method* `classify`. *Method* ini memiliki parameter `vektor` sebagai vektor data uji yang akan diklasifikasikan, dan parameter `K` sebagai batasan jumlah data terdekat yang digunakan dalam perhitungan pengklasifikasian. Hasil *return* berupa label kategori yang skornya terbesar. Implementasi *method* `classify` seperti pada Gambar 4.23 berikut ini.

```
public String classify(Double[] vektor, int K) {

  // knnMap <- k nearest neighbor-nya vektor data uji
  HashMap knnMap = knnMap(K, vektor);


  // inisiasi skorMap, max_skor, dan hasil_prediksi
  HashMap<String, Double> skorMap = new HashMap();
  double max_skor = 0;
  String hasil_prediksi = null;

  // iterasi setiap data latih dalam knnMap
  for (Iterator i = knnMap.entrySet().iterator();
       i.hasNext();) {
```

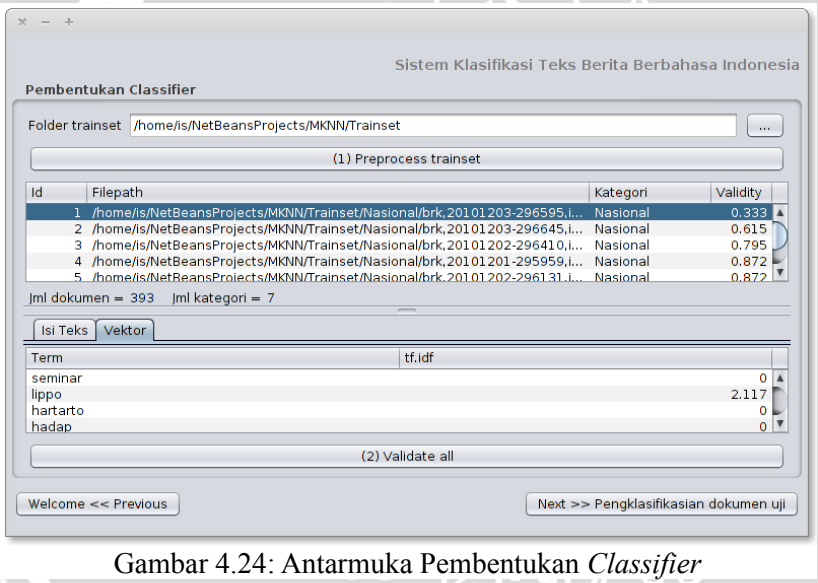
```
    // n <- data latih ke-i dalam knnMap
    Entry<Object, Double> n = (Entry) i.next();

    // category <- label kategorinya n
    String category=trainSet.get(n.getKey()).category;

    // bobot Wn <- validity-nya n * hasil cosine-nya n
    Double Wn = trainSet.get(n.getKey()).validity
                * n.getValue();

    // hitung skor kategorinya n
    Double skor = skorMap.get(category);
    skor = (skor == null) ? Wn : skor + Wn;
    skorMap.put(category, skor);

    // seleksi skor kategori yang terbesar
    if (skor > max_skor) {
      max_skor = skor;

      // hasil_prediksi <- kategori yg skornya terbesar
      hasil_prediksi = category;
    }
  }

  // return (kategori yg skornya terbesar)
  return hasil_prediksi;
}
```

Gambar 4.23: *Source Code Method* classify

4.3 Implementasi Antarmuka Grafis

Implementasi antarmuka grafis aplikasi terdiri dari dua *panel* : (1) pembentukan *classifier*, (2) dan pengklasifikasian.

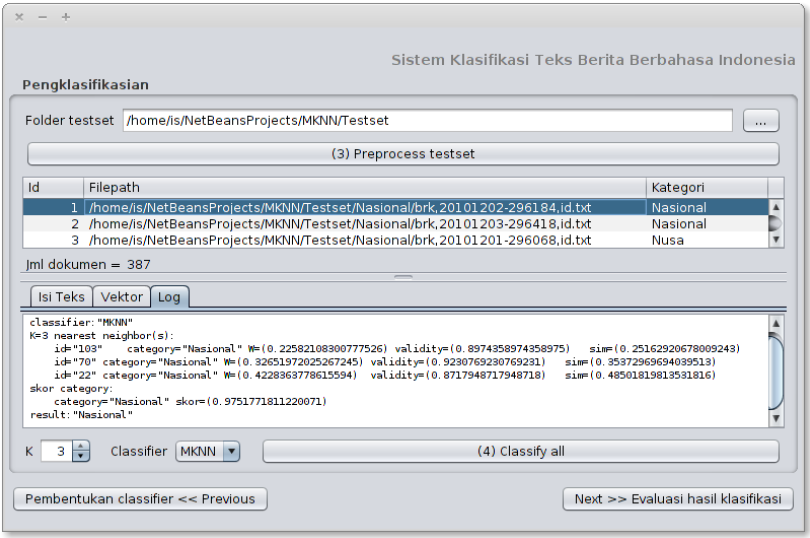
Pertama, antarmuka untuk proses pembentukan *classifier*. Pada antarmuka ini, pengguna memasukkan alamat *folder trainset* untuk kemudian dilakukan prapemrosesan melalui tombol "Preprocess trainset". Sistem akan menampilkan rincian data latih tersebut, baik isi maupun bentuk vektornya, pada tabel. Proses penghitungan nilai *validity* dapat dilakukan dengan menekan tombol "Validate all" dan hasilnya akan muncul pada tabel *trainset*. Gambar 4.24 berikut ini merupakan hasil *screenshot* dari antarmuka pembentukan *classifier*.



Gambar 4.24: Antarmuka Pembentukan *Classifier*

Kedua, antarmuka untuk proses pengklasifikasian. Pada antarmuka ini, pengguna memasukkan alamat *folder testset* (dokumen uji) untuk kemudian dilakukan prapemrosesan melalui tombol "Preprocess testset". Sistem akan menampilkan rincian data uji tersebut, baik isi maupun bentuk vektornya, pada tabel. Proses pengklasifikasian dapat dilakukan dengan menekan tombol "Classify all" setelah mengatur parameter K dan hasil kategorinya akan

muncul pada tabel *testset*. Gambar 4.25 berikut ini merupakan hasil *screenshot* dari antarmuka pengklasifikasian.



Gambar 4.25: Antarmuka Pengklasifikasian

4.4 Hasil Pengujian

Total dokumen yang digunakan adalah sejumlah 780 dokumen yang terdiri dari 7 kategori berita. Sampel dokumen latih (393 dokumen) dan dokumen uji (387 dokumen) dibagi secara acak berdasarkan proporsi yang berimbang untuk setiap kategori (Tabel 3.1). Nilai parameter *H* berdasarkan eksperimen Parvin, dkk., (2008) ditentukan sebanyak 10% dari jumlah dokumen latih, dalam hal ini yaitu sebesar *H*=39. (hasil pengukuran nilai *validity* untuk setiap dokumen latih disajikan pada lampiran)

Pengujian dilakukan dengan nilai parameter K yang berbeda-beda, yaitu pada kisaran K=3 hingga K=20 (terlampir), pada kedua pengklasifikasi (*classifier*) MKNN dan KNN. Namun tingkat efektivitas terbesar didapatkan pada dua kisaran parameter K yaitu pada kisaran 3, 4, 5, dan 14, 15, 16.

4.4.1 Hasil Pengujian MKNN

Hasil uji coba kesesuaian kategori dari pengklasifikasi MKNN disajikan di Tabel 4.1 dan Tabel 4.2 berikut ini.

Tabel 4.1: Hasil Evaluasi Uji Coba MKNN K=3, 4, dan 5

Kategori	K=3			K=4			K=5		
	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	4	48	0	3	48	0	2
Nasional	97	15	33	96	16	33	95	17	39
Bisnis	50	9	12	51	8	13	52	7	13
Internasional	5	18	1	6	17	3	5	18	2
Teknologi	8	10	0	8	10	0	8	10	0
Nusa	74	20	39	71	23	39	70	24	38
Metro	14	19	2	14	19	2	13	20	2

Tabel 4.2: Hasil Evaluasi Uji Coba MKNN K=14, 15, dan 16

Kategori	K=14			K=15			K=16		
	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	6	48	0	6	48	0	5
Nasional	95	17	44	95	17	42	96	16	43
Bisnis	48	11	12	48	11	13	48	11	13
Internasional	6	17	2	6	17	2	6	17	1
Teknologi	7	11	0	6	12	0	6	12	0
Nusa	70	24	38	72	22	38	71	23	42
Metro	10	23	1	10	23	1	8	25	0

Berdasarkan hasil uji coba pada Tabel 4.1 dan Tabel 4.2, hasil perhitungan evaluasi efektivitas (*recall*,*precision*, dan F_1 *measure*) pengklasifikasi MKNN disajikan di Tabel 4.3 dan Tabel 4.4 berikut ini.

Tabel 4.3: Hasil Evaluasi Efektivitas MKNN K=3, 4, dan 5

Kategori	K=3			K=4			K=5			Rata-rata		
	Rec.	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**
Olahraga	1	0,923	0,96	1	0,941	0,97	1	0,96	0,98	**1**	**0,941**	**0,97**
Nasional	0,866	0,746	0,802	0,857	0,744	0,797	0,848	0,709	0,772	**0,857**	**0,733**	**0,79**
Bisnis	0,847	0,806	0,826	0,864	0,797	0,829	0,881	0,8	0,839	**0,864**	**0,801**	**0,831**
Internasional	0,217	0,833	0,345	0,261	0,667	0,375	0,217	0,714	0,333	**0,232**	**0,738**	**0,351**
Teknologi	0,444	1	0,615	0,444	1	0,615	0,444	1	0,615	**0,444**	**1**	**0,615**
Nusa	0,787	0,655	0,715	0,755	0,645	0,696	0,745	0,648	0,693	**0,762**	**0,649**	**0,701**
Metro	0,424	0,875	0,571	0,424	0,875	0,571	0,394	0,867	0,542	**0,414**	**0,872**	**0,562**
Rata-rata	**0,655**	**0,834**	**0,691**	**0,658**	**0,81**	**0,693**	**0,647**	**0,814**	**0,682**			

Tabel 4.4: Hasil Evaluasi Efektivitas MKNN K=14, 15, dan 16

Kategori	K=14			K=15			K=16			Rata-rata		
	Rec.	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**
Olahraga	1	0,889	0,941	1	0,889	0,941	1	0,906	0,95	**1**	**0,894**	**0,944**
Nasional	0,848	0,683	0,757	0,848	0,693	0,763	0,857	0,691	0,765	**0,851**	**0,689**	**0,762**
Bisnis	0,814	0,8	0,807	0,814	0,787	0,8	0,814	0,787	0,8	**0,814**	**0,791**	**0,802**
Internasional	0,261	0,75	0,387	0,261	0,75	0,387	0,261	0,857	0,4	**0,261**	**0,786**	**0,391**
Teknologi	0,389	1	0,56	0,333	1	0,5	0,333	1	0,5	**0,352**	**1**	**0,52**
Nusa	0,745	0,648	0,693	0,766	0,655	0,706	0,755	0,628	0,686	**0,755**	**0,644**	**0,695**
Metro	0,303	0,909	0,455	0,303	0,909	0,455	0,242	1	0,39	**0,283**	**0,939**	**0,433**
Rata-rata	**0,623**	**0,811**	**0,657**	**0,618**	**0,812**	**0,65**	**0,609**	**0,838**	**0,642**			

4.4.2 Hasil Pengujian KNN

Hasil uji coba kesesuaian kategori dari pengklasifikasi KNN disajikan di Tabel 4.5 dan Tabel 4.6 berikut ini.

Tabel 4.5: Hasil Evaluasi Uji Coba KNN K=3, 4, dan 5

Kategori	K=3			K=4			K=5		
	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	2	48	0	0	48	0	0
Nasional	95	17	33	96	16	35	96	16	35
Bisnis	51	8	14	54	5	14	52	7	12
Internasional	12	11	3	12	11	4	11	12	4
Teknologi	11	7	1	9	9	0	9	9	0
Nusa	65	29	28	65	29	25	67	27	31
Metro	18	15	6	19	14	6	17	16	5

Tabel 4.6: Hasil Evaluasi Uji Coba KNN K=14, 15, dan 16

Kategori	K=14			K=15			K=16		
	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	0	48	0	0	48	0	0
Nasional	93	19	29	95	17	27	95	17	31
Bisnis	53	6	14	54	5	13	53	6	13
Internasional	11	12	5	11	12	4	11	12	4
Teknologi	9	9	0	10	8	0	10	8	0
Nusa	72	22	33	73	21	31	72	22	30
Metro	18	15	2	19	14	2	19	14	1

Berdasarkan hasil uji coba pada Tabel 4.5 dan Tabel 4.6, hasil perhitungan evaluasi efektivitas (*recall*, *precision*, dan F_1 *measure*) pengklasifikasi KNN disajikan di Tabel 4.7 dan Tabel 4.8 berikut ini.

Tabel 4.7: Hasil Evaluasi Efektivitas KNN K=3, 4, dan 5

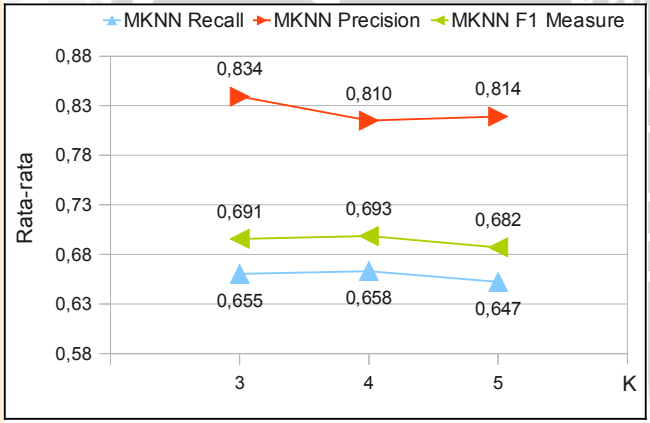
Kategori	**K=3**			**K=4**			**K=5**			**Rata-rata**		
	Rec.	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**
Olahraga	1	0,96	0,98	1	1	1	1	1	1	**1**	**0,987**	**0,993**
Nasional	0,848	0,742	0,792	0,857	0,733	0,79	0,857	0,733	0,79	**0,854**	**0,736**	**0,791**
Bisnis	0,864	0,785	0,823	0,915	0,794	0,85	0,881	0,813	0,846	**0,887**	**0,797**	**0,84**
Internasional	0,522	0,8	0,632	0,522	0,75	0,615	0,478	0,733	0,579	**0,507**	**0,761**	**0,609**
Teknologi	0,611	0,917	0,733	0,5	1	0,667	0,5	1	0,667	**0,537**	**0,972**	**0,689**
Nusa	0,691	0,699	0,695	0,691	0,722	0,707	0,713	0,684	0,698	**0,699**	**0,702**	**0,7**
Metro	0,545	0,75	0,632	0,576	0,76	0,655	0,515	0,773	0,618	**0,545**	**0,761**	**0,635**
Rata-rata	**0,726**	**0,807**	**0,755**	**0,723**	**0,823**	**0,755**	**0,706**	**0,819**	**0,742**			

Tabel 4.8: Hasil Evaluasi Efektivitas KNN K=14, 15, dan 16

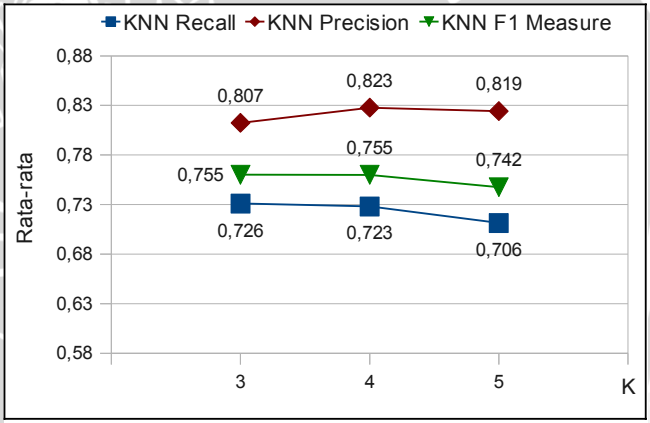
Kategori	K=14			K=15			K=16			Rata-rata		
	Rec.	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**	**Rec.**	**Prec.**	**F_1 M.**
Olahraga	1	1	1	1	1	1	1	1	1	**1**	**1**	**1**
Nasional	0,83	0,762	0,795	0,848	0,779	0,812	0,848	0,754	0,798	**0,842**	**0,765**	**0,802**
Bisnis	0,898	0,791	0,841	0,915	0,806	0,857	0,898	0,803	0,848	**0,904**	**0,8**	**0,849**
Internasional	0,478	0,688	0,564	0,478	0,733	0,579	0,478	0,733	0,579	**0,478**	**0,718**	**0,574**
Teknologi	0,5	1	0,667	0,556	1	0,714	0,556	1	0,714	**0,537**	**1**	**0,698**
Nusa	0,766	0,686	0,724	0,777	0,702	0,737	0,766	0,706	0,735	**0,77**	**0,698**	**0,732**
Metro	0,545	0,9	0,679	0,576	0,905	0,704	0,576	0,95	0,717	**0,566**	**0,918**	**0,7**
Rata-rata	**0,717**	**0,832**	**0,753**	**0,736**	**0,846**	**0,772**	**0,732**	**0,849**	**0,77**			

4.4.3 Grafik Rata-Rata Efektivitas

Berdasarkan hasil evaluasi efektivitas kedua pengklasifikasi (MKNN dan KNN), hasil rata-rata efektivitas (*recall*, *precision*, dan F_1 *measure*) dijadikan sebagai perbandingan. Perbandingan rata-rata efektivitas MKNN dan KNN pada K=3, 4, 5 disajikan di Gambar 4.26 dan Gambar 4.27 berikut ini.

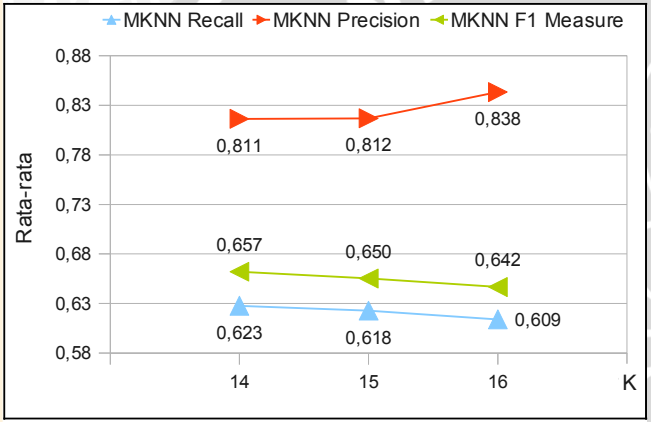


Gambar 4.26: Grafik Rata-Rata Efektivitas MKNN (K=3, 4, dan 5)

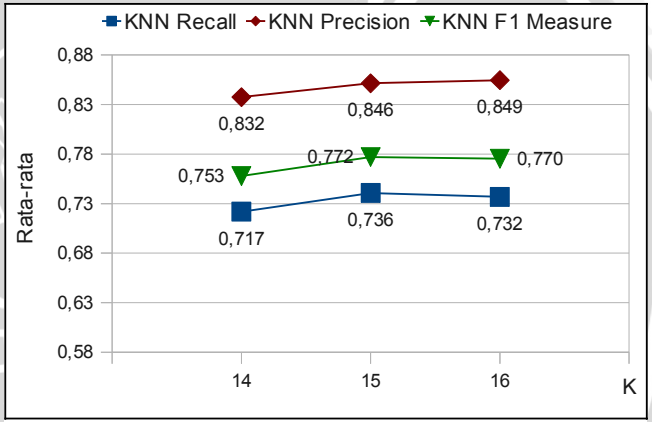


Gambar 4.27: Grafik Rata-Rata Efektivitas KNN (K=3, 4, dan 5)

Sedangkan perbandingan rata-rata efektivitas MKNN dan KNN pada K=14, 15, 16 disajikan di Gambar 4.28 dan Gambar 4.29 berikut ini.



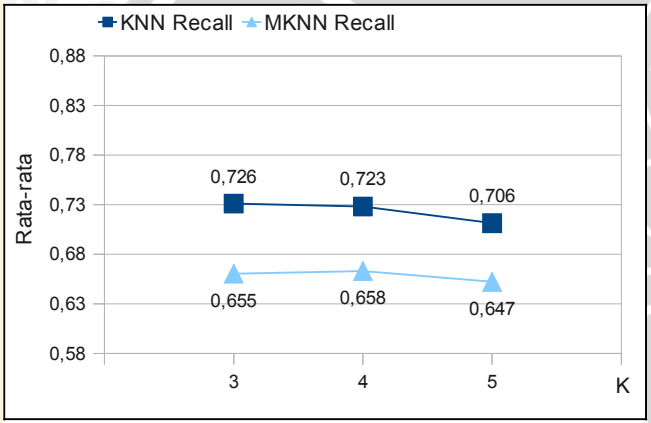
Gambar 4.28: Grafik Rata-Rata Efektivitas MKNN (K=14, 15, dan 16)



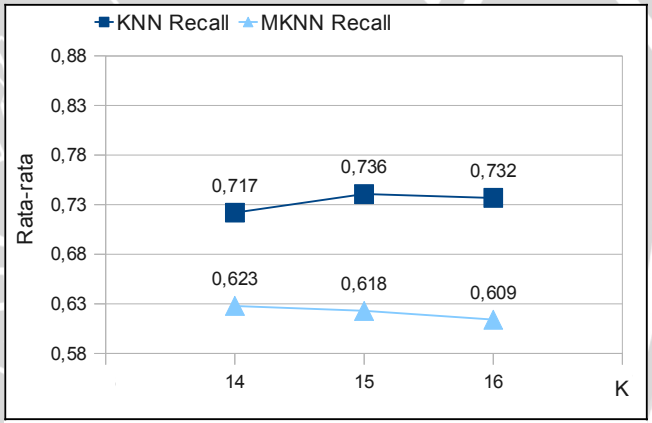
Gambar 4.29: Grafik Rata-Rata Efektivitas KNN (K=14, 15, dan 16)

4.4.3.1 Grafik Rata-Rata *Recall* MKNN dan KNN

Perbandingan rata-rata *recall* MKNN dan KNN pada K=3, 4, 5 disajikan di Gambar 4.30. Sedangkan pada K=14, 15, 16 disajikan di Gambar 4.31 berikut ini.



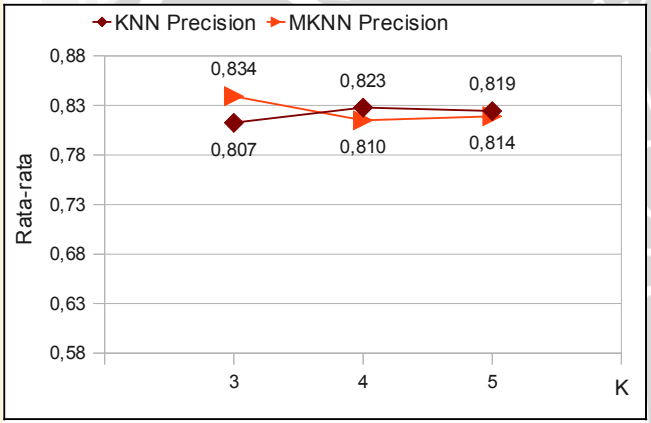
Gambar 4.30: Grafik Rata-Rata *Recall* MKNN dan KNN (K=3, 4, dan 5)



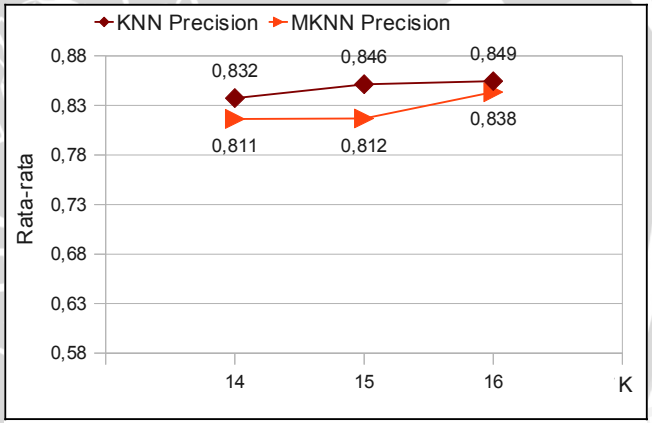
Gambar 4.31: Grafik Rata-Rata *Recall* MKNN dan KNN (K=14, 15, dan 16)

4.4.3.2 Grafik Rata-Rata *Precision* MKNN dan KNN

Perbandingan rata-rata *precision* MKNN dan KNN pada K=3, 4, 5 disajikan di Gambar 4.32. Sedangkan pada K=14, 15, 16 disajikan di Gambar 4.33 berikut ini.



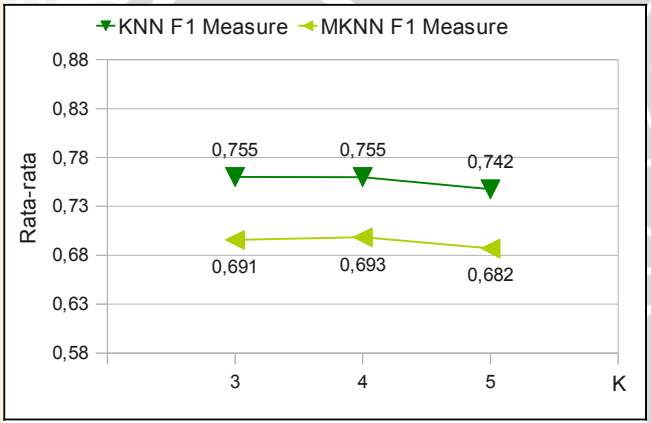
Gambar 4.32: Grafik Rata-Rata *Precision* MKNN dan KNN (K=3, 4, dan 5)



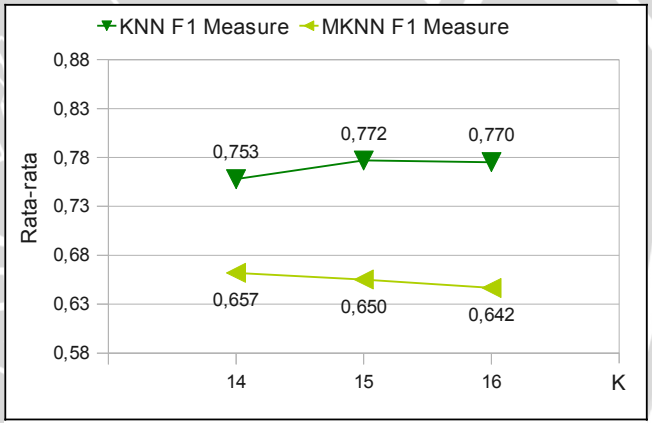
Gambar 4.33: Grafik Rata-Rata *Precision* MKNN dan KNN (K=14, 15, dan 16)

4.4.3.3 Grafik Rata-Rata *F_1 Measure* MKNN dan KNN

Perbandingan rata-rata *F_1 Measure* MKNN dan KNN pada K=3, 4, 5 disajikan di Gambar 4.34. Sedangkan pada K=14, 15, 16 disajikan di Gambar 4.35 berikut ini.



Gambar 4.34: Grafik Rata-Rata *F_1 Measure* MKNN dan KNN (K=3, 4, dan 5)



Gambar 4.35: Grafik Rata-Rata *F_1 Measure* MKNN dan KNN (K=14, 15, dan 16)

4.5 Analisa Hasil Pengujian

Berdasarkan hasil pengujian untuk setiap nilai K yang diujikan yaitu 3, 4, 5, 14, 15, dan 16, maka tingkat efektivitas dari kedua *classifier* (MKNN dan KNN) secara umum dapat diukur dengan nilai rata-rata *recall*, *precision*, dan F_1 *measure*. Hasil perbandingan nilai-nilai rata-rata *recall*, *precision*, dan F_1 *measure* pada setiap nilai K didapatkan bahwa efektivitas MKNN optimal pada kisaran K=3, 4, 5 sedangkan efektivitas KNN optimal pada kisaran K=14, 15, 16.

Pada pengukuran *recall*, nilai rata-rata MKNN terbesar didapatkan pada saat K=4 yaitu sebesar 0.658 (66%), pada saat yang sama ternyata KNN menunjukkan nilai yang lebih besar yaitu 0.723 (72%). Adapun nilai rata-rata *recall* terbesar KNN didapatkan pada saat K=15 yaitu sebesar 0.736 (74%), sedangkan pada saat yang sama MKNN hanya sebesar 0.623 (62%). Secara keseluruhan pada setiap nilai K yang diujikan, nilai rata-rata *recall* MKNN hanya berkisar dari 61% hingga 66% yakni secara konsisten masih berada di bawah KNN yang berkisar dari 72% hingga 74%.

Pada pengukuran nilai *precision*, nilai rata-rata terbesar dari kedua *classifier* sama-sama didapatkan pada saat K=16, namun nilai rata-rata *precision* MKNN yang sebesar 0.838 (84%) sedikit lebih rendah dari KNN yang sebesar 0.849 (85%). Keunggulan nilai rata-rata *precision* MKNN dari KNN hanya terjadi pada saat K=3, yaitu MKNN sebesar 0.834 (83%) sedangkan KNN sebesar 0.807 (81%). Secara keseluruhan pada setiap nilai K yang diujikan, nilai rata-rata *precision* MKNN berkisar dari 81% hingga 84%, sedangkan KNN berkisar dari 81% hingga 85%.

Hasil evaluasi pengujian MKNN menunjukkan bahwa nilai rata-rata F_1 *measure* terbesar didapatkan pada saat K=4 yaitu sebesar 0.693 (69%), namun pada saat yang sama, hasil KNN masih lebih besar yaitu mencapai 0.755 (76%). Adapun nilai rata-rata F_1 *measure* terbesar KNN didapatkan pada saat K=15 yaitu mencapai 0.772 (77%) dan pada saat yang sama MKNN hanya sebesar 0.65 (65%). Secara keseluruhan pada setiap nilai K yang diujikan, nilai rata-rata F_1 *measure* MKNN hanya berkisar dari 64% hingga 69% yakni secara konsisten masih berada di bawah nilai rata-rata F_1 *measure* KNN yang berkisar dari 74% hingga 77%.

Perbedaan utama MKNN dengan KNN terletak pada faktor

nilai *validity*. MKNN menggunakan nilai *validity* dalam proses pengklasifikasiannya, sedangkan KNN tidak. Nilai *validity* merupakan ukuran stabilitas kedekatan antar dokumen latih dalam suatu kategori. Hasil perhitungan nilai *validity* (Lampiran 2) menunjukkan bahwa karakteristik sebaran dokumen latih kurang stabil pada kategori "Internasional", "Teknologi", dan "Metro". Nilai *validity* yang rendah pada kategori-kategori tersebut berpengaruh terhadap rendahnya nilai *recall*-nya masing-masing. *Recall* hanya memperhatikan dokumen yang sesuai untuk satu kategori tertentu yang berhasil diklasifikasikan dengan benar. Sehingga, sebaran dokumen yang kurang stabil dalam suatu kategori menyebabkan semakin rendahnya pula nilai *recall*-nya. Oleh sebab itu, penerapan MKNN pada pengklasifikasian dokumen teks yang sebaran dokumen latih-nya kurang stabil menghasilkan efektivitas yang lebih rendah jika dibandingkan dengan KNN.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka didapatkan kesimpulan sebagai berikut :

1. Metode MKNN dapat diimplementasikan pada sistem pengklasifikasi teks berita berbahasa Indonesia dengan cara merepresentasikan teks menjadi vektor numerik dan menggunakan fungsi *Cosine Similarity* sebagai pengukur tingkat kesamaan antar vektor.
2. Sistem pengklasifikasi teks berita berbahasa Indonesia berbasis metode MKNN memiliki tingkat akurasi yang lebih rendah daripada metode KNN. Secara keseluruhan pada setiap nilai K yang diujikan (3, 4, 5, 14, 15, dan 16), nilai rata-rata F_1 *measure* MKNN berkisar dari 64% hingga 69% yakni secara konsisten masih berada dibawah nilai rata-rata F_1 *measure* KNN yang berkisar dari 74% hingga 77%.

5.2 Saran

Berkaitan dengan penelitian ini, penulis menemukan beberapa hal yang mungkin perlu dikembangkan untuk ke depannya, yaitu :

1. Penerapan metode *feature selection* dalam tahap pra-pemrosesan untuk mengurangi ukuran vektor.
2. Penerapan metode *stemming* yang lebih baik daripada metode *Porter Stemmer* Bahasa Indonesia.
3. Koleksi dokumen teks berbahasa Indonesia untuk penelitian perlu distandarkan dalam lingkup Program Studi Ilmu Komputer Universitas Brawijaya.

UNIVERSITAS BRAWIJAYA

DAFTAR PUSTAKA

Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K. 2004. *an kNN Model-based Approach and Its Application in Text Categorization*. Lecture Notes in Computer Science Volume 2945, 2004.

Han, J., dan Kamber, M. 2006. *Data Mining: Concepts and Techniques, Second Edition*. Morgan Kaufmann Publishers.

Manning, C. D., Raghavan, P., dan Hinrich, S. 2009. *An Introduction to Information Retrieval*. Cambridge University Press.

Parvin, H., Alizadeh, H., dan Minaei-Bidgoli, B. 2008. *MKNN: Modified K-Nearest Neighbor*. World Congress on Engineering and Computer Science 2008, USA.

Sebastiani, F. 2005. *Text Categorization*. In Alessandro Zanasi (ed.), Text Mining and its Applications. WIT Press, Southampton, UK, 2005, pp. 109-129.

Solka, J. L. 2008. *Text Data Mining: Theory and Methods*. Statistics Surveys, Volume 2, 2008. ISSN: 1935-7516 .

Tala, F. Z. 2003. *A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia*. Universiteit van Amsterdam, The Netherlands.

Tan, Songbo. 2006. *An Effective Refinement Strategy for KNN Text Classifier*. Expert System with Applications 30. ELSEVIER.

Velickov, S. dan Solomatine, D. 2000. *Predictive Data Mining: Practical Examples*. Artificial Intelligence in Civil Engineering. Proc. 2nd Joint Workshop, March 2000, Cottbus, Germany. ISBN 3-934934-00-5.

Yang, Y., dan Liu, X. 1999. *A re-examination of text categorization methods*. The 22nd Annual International ACM SIGIR Conference on Research and Development in the Information Retrieval, ACM Press, USA.

UNIVERSITAS BRAWIJAYA

LAMPIRAN

Lampiran 1. Daftar *Stop Words*

ada	awal	benar
adalah	awalnya	benarkah
adanya	bagai	benarlah
adapun	bagaikan	berada
agak	bagaimana	berakhir
agaknya	bagaimanakah	berakhirlah
agar	bagaimanapun	berakhirnya
akan	bagi	berapa
akankah	bagian	berapakah
akhir	bahkan	berapalah
akhiri	bahwa	berapapun
akhirnya	bahwasanya	berarti
aku	baik	berawal
akulah	bakal	berbagai
amat	bakalan	berdatangan
amatlah	balik	beri
anda	banyak	berikan
andalah	bapak	berikut
antar	baru	berikutnya
antara	bawah	berjumlah
antaranya	beberapa	berkali-kali
apa	begini	berkata
apaan	beginian	berkehendak
apabila	beginikah	berkeinginan
apakah	beginilah	berkenaan
apalagi	begitu	berlainan
apatah	begitukah	berlalu
artinya	begitulah	berlangsung
asal	begitupun	berlebihan
asalkan	bekerja	bermacam
atas	belakang	bermacam-macam
atau	belakangan	bermaksud
ataukah	belum	bermula
ataupun	belumlah	bersama

bersama-sama
bersiap
bersiap-siap
bertanya
bertanya-tanya
berturut
berturut-turut
bertutur
berujar
berupa
besar
betul
betulkah
biasa
biasanya
bila
bilakah
bisa
bisakah
boleh
bolehkah
bolehlah
buat
bukan
bukankah
bukanlah
bukannya
bulan
bung
cara
caranya
cukup
cukupkah
cukuplah
cuma
dahulu
dalam
dan
dapat
dari
daripada
datang
dekat
demi
demikian
demikianlah
dengan
depan
di
dia
diakhiri
diakhirinya
dialah
diantara
diantaranya
diberi
diberikan
diberikannya
dibuat
dibuatnya
didapat
didatangkan
digunakan
diibaratkan
diibaratkannya
diingat
diingatkan
diinginkan
dijawab
dijelaskan
dijelaskannya
dikarenakan
dikatakan
dikatakannya
dikerjakan
diketahui
diketahuinya
dikira
dilakukan
dilalui
dilihat
dimaksud
dimaksudkan
dimaksudkannya
dimaksudnya
diminta
dimintai
dimisalkan
dimulai
dimulailah
dimulainya
dimungkinkan
dini
dipastikan
diperbuat
diperbuatnya
dipergunakan
diperkirakan
diperlihatkan
diperlukan
diperlukannya
dipersoalkan
dipertanyakan
dipunyai
diri
dirinya
disampaikan
disebut
disebutkan
disebutkannya
disini
disinilah
ditambahkan
ditandaskan

ditanya
ditanyai
ditanyakan
ditegaskan
ditujukan
ditunjuk
ditunjuki
ditunjukkan
ditunjukkannya
ditunjuknya
dituturkan
dituturkannya
diucapkan
diucapkannya
diungkapkan
dong
dua
dulu
empat
enggak
enggaknya
entah
entahlah
guna
gunakan
hal
hampir
hanya
hanyalah
hari
harus
haruslah
harusnya
hendak
hendaklah
hendaknya
hingga
ia
ialah
ibarat
ibaratkan
ibaratnya
ibu
ikut
ingat
ingat-ingat
ingin
inginkah
inginkan
ini
inikah
inilah
itu
itukah
itulah
jadi
jadilah
jadinya
jangan
jangankan
janganlah
jauh
jawab
jawaban
jawabnya
jelas
jelaskan
jelaslah
jelasnya
jika
jikalau
juga
jumlah
jumlahnya
justru
kala
kalau
kalaulah
kalaupun
kalian
kami
kamilah
kamu
kamulah
kan
kapan
kapankah
kapanpun
karena
karenanya
kasus
kata
katakan
katakanlah
katanya
ke
keadaan
kebetulan
kecil
kedua
keduanya
keinginan
kelamaan
kelihatan
kelihatannya
kelima
keluar
kembali
kemudian
kemungkinan
kemungkinannya
kenapa
kepada
kepadanya

kesampaian
keseluruhan
keseluruhannya
keterlaluan
ketika
khususnya
kini
kinilah
kira
kira-kira
kiranya
kita
kitalah
kok
kurang
lagi
lagian
lah
lain
lainnya
lalu
lama
lamanya
lanjut
lanjutnya
lebih
lewat
lima
luar
macam
maka
makanya
makin
malah
malahan
mampu
mampukah
mana
manakala
manalagi
masa
masalah
masalahnya
masih
masihkah
masing
masing-masing
mau
maupun
melainkan
melakukan
melalui
melihat
melihatnya
memang
memastikan
memberi
memberikan
membuat
memerlukan
memihak
meminta
memintakan
memisalkan
memperbuat
mempergunakan
memperkirakan
memperlihatkan
mempersiapkan
mempersoalkan
mempertanyakan
mempunyai
memulai
memungkinkan
menaiki
menambahkan
menandaskan
menanti
menanti-nanti
menantikan
menanya
menanyai
menanyakan
mendapat
mendapatkan
mendatang
mendatangi
mendatangkan
menegaskan
mengakhiri
mengapa
mengatakan
mengatakannya
mengenai
mengerjakan
mengetahui
menggunakan
menghendaki
mengibaratkan
mengibaratkannya
mengingat
mengingatkan
menginginkan
mengira
mengucapkan
mengucapkannya
mengungkapkan
menjadi
menjawab
menjelaskan
menuju
menunjuk
menunjuki
menunjukkan

menunjuknya
menurut
menuturkan
menyampaikan
menyangkut
menyatakan
menyebutkan
menyeluruh
menyiapkan
merasa
mereka
merekalah
merupakan
meski
meskipun
meyakini
meyakinkan
minta
mirip
misal
misalkan
misalnya
mula
mulai
mulailah
mulanya
mungkin
mungkinkah
nah
naik
namun
nanti
nantinya
nyaris
nyatanya
oleh
olehnya
pada
padahal
padanya
pak
paling
panjang
pantas
para
pasti
pastilah
penting
pentingnya
per
percuma
perlu
perlukah
perlunya
pernah
persoalan
pertama
pertama-tama
pertanyaan
pertanyakan
pihak
pihaknya
pukul
pula
pun
punya
rasa
rasanya
rata
rupanya
saat
saatnya
saja
sajalah
saling
sama
sama-sama
sambil
sampai
sampai-sampai
sampaikan
sana
sangat
sangatlah
satu
saya
sayalah
se
sebab
sebabnya
sebagai
sebagaimana
sebagainya
sebagian
sebaik
sebaik-baiknya
sebaiknya
sebaliknya
sebanyak
sebegini
sebegitu
sebelum
sebelumnya
sebenarnya
seberapa
sebesar
sebetulnya
sebisanya
sebuah
sebut
sebutlah
sebutnya
secara
secukupnya

sedang
sedangkan
sedemikian
sedikit
sedikitnya
seenaknya
segala
segalanya
segera
seharusnya
sehingga
seingat
sejak
sejauh
sejenak
sejumlah
sekadar
sekadarnya
sekali
sekali-kali
sekalian
sekaligus
sekalipun
sekarang
sekarang
sekecil
seketika
sekiranya
sekitar
sekitarnya
sekurang-kurangnya
sekurangnya
sela
selain
selaku
selalu
selama
selama-lamanya
selamanya
selanjutnya
seluruh
seluruhnya
semacam
semakin
semampu
semampunya
semasa
semasih
semata
semata-mata
semaunya
sementara
semisal
semisalnya
sempat
semua
semuanya
semula
sendiri
sendirian
sendirinya
seolah
seolah-olah
seorang
sepanjang
sepantasnya
sepantasnyalah
seperlunya
seperti
sepertinya
sepihak
sering
seringnya
serta
serupa
sesaat
sesama
sesampai
sesegera
sesekali
seseorang
sesuatu
sesuatunya
sesudah
sesudahnya
setelah
setempat
setengah
seterusnya
setiap
setiba
setibanya
setidak-tidaknya
setidaknya
setinggi
seusai
sewaktu
siap
siapa
siapakah
siapapun
sini
sinilah
soal
soalnya
suatu
sudah
sudahkah
sudahlah
supaya
tadi
tadinya
tahu
tahun

tak	terhadapnya	toh
tambah	teringat	tunjuk
tambahnya	teringat-ingat	turut
tampak	terjadi	tutur
tampaknya	terjadilah	tuturnya
tandas	terjadinya	ucap
tandasnya	terkira	ucapnya
tanpa	terlalu	ujar
tanya	terlebih	ujarnya
tanyakan	terlihat	umum
tanyanya	termasuk	umumnya
tapi	ternyata	ungkap
tegas	tersampaikan	ungkapnya
tegasnya	tersebut	untuk
telah	tersebutlah	usah
tempat	tertentu	usai
tengah	tertuju	waduh
tentang	terus	wah
tentu	terutama	wahai
tentulah	tetap	waktu
tentunya	tetapi	waktunya
tepat	tiap	walau
terakhir	tiba	walaupun
terasa	tiba-tiba	wong
terbanyak	tidak	yaitu
terdahulu	tidakkah	yakin
terdapat	tidaklah	yakni
terdiri	tiga	yang
terhadap	tinggi	

Lampiran 2. Nilai *Validity* (H=39)

Tabel Rata-rata *Validity* per-Kategori

Kategori	Rata-rata *Validity*
Nasional	0.672
Internasional	0.158
Metro	0.188
Olahraga	0.691
Bisnis	0.423
Nusa	0.435
Teknologi	0.132

Tabel *Validity* Dokumen

Id	Nama File	Kategori	Validity
1	brk,20101203-296595,id.txt	Nasional	0.333
2	brk,20101203-296645,id.txt	Nasional	0.615
3	brk,20101202-296410,id.txt	Nasional	0.795
4	brk,20101201-295959,id.txt	Nasional	0.872
5	brk,20101202-296131,id.txt	Nasional	0.872
6	brk,20101203-296441,id.txt	Nasional	0.385
7	brk,20101203-296456,id.txt	Nasional	0.231
8	brk,20101201-295913,id.txt	Nasional	0.308
9	brk,20101203-296615,id.txt	Nasional	0.897
10	brk,20101203-296670,id.txt	Nasional	0.282
11	brk,20101202-296412,id.txt	Nasional	0.667
12	brk,20101201-296067,id.txt	Nasional	0.744
13	brk,20101202-296140,id.txt	Nasional	0.846
14	brk,20101202-296117,id.txt	Nasional	0.846
15	brk,20101202-296164,id.txt	Nasional	0.897
16	brk,20101202-296165,id.txt	Nasional	0.897
17	brk,20101201-295938,id.txt	Nasional	0.821
18	brk,20101201-295995,id.txt	Nasional	0.333
19	brk,20101202-296367,id.txt	Nasional	0.923
20	brk,20101203-296598,id.txt	Nasional	0.744
21	brk,20101202-296225,id.txt	Nasional	0.41
22	brk,20101201-296060,id.txt	Nasional	0.872
23	brk,20101202-296169,id.txt	Nasional	0.59
24	brk,20101203-296602,id.txt	Nasional	0.795

25	brk,20101203-296417,id.txt	Nasional	0.923
26	brk,20101202-296325,id.txt	Nasional	0.615
27	brk,20101203-296434,id.txt	Nasional	0.897
28	brk,20101202-296337,id.txt	Nasional	0.538
29	brk,20101201-295873,id.txt	Nasional	0.641
30	brk,20101202-296156,id.txt	Nasional	0.718
31	brk,20101201-295961,id.txt	Nasional	0.795
32	brk,20101202-296137,id.txt	Nasional	0.795
33	brk,20101201-295836,id.txt	Nasional	0.718
34	brk,20101201-296018,id.txt	Nasional	0.692
35	brk,20101202-296186,id.txt	Nasional	0.872
36	brk,20101201-295939,id.txt	Nasional	0.872
37	brk,20101203-296511,id.txt	Nasional	0.692
38	brk,20101203-296419,id.txt	Nasional	0.923
39	brk,20101203-296515,id.txt	Nasional	0.769
40	brk,20101202-296314,id.txt	Nasional	0.667
41	brk,20101201-295921,id.txt	Nasional	0.897
42	brk,20101201-295890,id.txt	Nasional	0.154
43	brk,20101203-296534,id.txt	Nasional	0.615
44	brk,20101201-295808,id.txt	Nasional	0.872
45	brk,20101203-296443,id.txt	Nasional	0.692
46	brk,20101203-296541,id.txt	Nasional	0.641
47	brk,20101202-296226,id.txt	Nasional	0.667
48	brk,20101202-296148,id.txt	Nasional	0.846
49	brk,20101203-296479,id.txt	Nasional	0.718
50	brk,20101201-296031,id.txt	Nasional	0.333
51	brk,20101202-296213,id.txt	Nasional	0.641
52	brk,20101203-296516,id.txt	Nasional	0.744
53	brk,20101202-296091,id.txt	Nasional	0.564
54	brk,20101201-295889,id.txt	Nasional	0.897
55	brk,20101202-296376,id.txt	Nasional	0.897
56	brk,20101203-296459,id.txt	Nasional	0.897
57	brk,20101202-296394,id.txt	Nasional	0.846
58	brk,20101202-296136,id.txt	Nasional	0.231
59	brk,20101202-296176,id.txt	Nasional	0.667
60	brk,20101201-296036,id.txt	Nasional	0.385
61	brk,20101203-296636,id.txt	Nasional	0.564
62	brk,20101201-295802,id.txt	Nasional	0.769

63	brk,20101202-296095,id.txt	Nasional	0.41
64	brk,20101201-296063,id.txt	Nasional	0.744
65	brk,20101203-296490,id.txt	Nasional	0.744
66	brk,20101202-296408,id.txt	Nasional	0.179
67	brk,20101202-296092,id.txt	Nasional	0.256
68	brk,20101201-295789,id.txt	Nasional	0.795
69	brk,20101202-296341,id.txt	Nasional	0.538
70	brk,20101203-296416,id.txt	Nasional	0.923
71	brk,20101203-296480,id.txt	Nasional	0.872
72	brk,20101202-296368,id.txt	Nasional	0.923
73	brk,20101201-296062,id.txt	Nasional	0.615
74	brk,20101201-295783,id.txt	Nasional	0.846
75	brk,20101201-295776,id.txt	Nasional	0.308
76	brk,20101203-296619,id.txt	Nasional	0.487
77	brk,20101202-296099,id.txt	Nasional	0.333
78	brk,20101202-296126,id.txt	Nasional	0.641
79	brk,20101201-295870,id.txt	Nasional	0.41
80	brk,20101203-296643,id.txt	Nasional	0.744
81	brk,20101203-296467,id.txt	Nasional	0.256
82	brk,20101202-296318,id.txt	Nasional	0.769
83	brk,20101203-296612,id.txt	Nasional	0.692
84	brk,20101202-296242,id.txt	Nasional	0.923
85	brk,20101203-296464,id.txt	Nasional	0.846
86	brk,20101202-296166,id.txt	Nasional	0.897
87	brk,20101202-296381,id.txt	Nasional	0.821
88	brk,20101201-295814,id.txt	Nasional	0.718
89	brk,20101203-296647,id.txt	Nasional	0.333
90	brk,20101203-296544,id.txt	Nasional	0.692
91	brk,20101202-296134,id.txt	Nasional	0.487
92	brk,20101202-296263,id.txt	Nasional	0.564
93	brk,20101201-295985,id.txt	Nasional	0.564
94	brk,20101201-295900,id.txt	Nasional	0.59
95	brk,20101201-296049,id.txt	Nasional	0.923
96	brk,20101203-296444,id.txt	Nasional	0.59
97	brk,20101203-296606,id.txt	Nasional	0.692
98	brk,20101203-296439,id.txt	Nasional	0.769
99	brk,20101202-296251,id.txt	Nasional	0.769
100	brk,20101201-295797,id.txt	Nasional	0.436

101	brk,20101202-296257,id.txt	Nasional	0.615
102	brk,20101202-296294,id.txt	Nasional	0.641
103	brk,20101201-295975,id.txt	Nasional	0.897
104	brk,20101201-295853,id.txt	Nasional	0.923
105	brk,20101201-296041,id.txt	Nasional	0.692
106	brk,20101202-296411,id.txt	Nasional	0.949
107	brk,20101202-296331,id.txt	Nasional	0.872
108	brk,20101201-295855,id.txt	Nasional	0.436
109	brk,20101202-296089,id.txt	Nasional	0.308
110	brk,20101203-296639,id.txt	Nasional	0.256
111	brk,20101203-296549,id.txt	Nasional	0.692
112	brk,20101201-295810,id.txt	Nasional	0.949
113	brk,20101201-295933,id.txt	Nasional	0.923
114	brk,20101201-295761,id.txt	Internasional	0.231
115	brk,20101202-296159,id.txt	Internasional	0.077
116	brk,20101201-295778,id.txt	Internasional	0.103
117	brk,20101202-296199,id.txt	Internasional	0.154
118	brk,20101202-296349,id.txt	Internasional	0.051
119	brk,20101202-296326,id.txt	Internasional	0.179
120	brk,20101203-296597,id.txt	Internasional	0.256
121	brk,20101201-296048,id.txt	Internasional	0.103
122	brk,20101202-296373,id.txt	Internasional	0.103
123	brk,20101201-295828,id.txt	Internasional	0.333
124	brk,20101202-296276,id.txt	Internasional	0.051
125	brk,20101203-296470,id.txt	Internasional	0.231
126	brk,20101202-296285,id.txt	Internasional	0.051
127	brk,20101203-296433,id.txt	Internasional	0.077
128	brk,20101202-296101,id.txt	Internasional	0.051
129	brk,20101201-295859,id.txt	Internasional	0.103
130	brk,20101202-296207,id.txt	Internasional	0.256
131	brk,20101202-296236,id.txt	Internasional	0.282
132	brk,20101201-295872,id.txt	Internasional	0.154
133	brk,20101201-295979,id.txt	Internasional	0.026
134	brk,20101201-295940,id.txt	Internasional	0.282
135	brk,20101201-295941,id.txt	Internasional	0.231
136	brk,20101203-296508,id.txt	Internasional	0.256
137	brk,20101202-296158,id.txt	Metro	0.128
138	brk,20101202-296274,id.txt	Metro	0.205

139	brk,20101201-295817,id.txt	Metro	0.154
140	brk,20101201-295775,id.txt	Metro	0.282
141	brk,20101201-295849,id.txt	Metro	0.256
142	brk,20101203-296425,id.txt	Metro	0.077
143	brk,20101201-295856,id.txt	Metro	0.179
144	brk,20101201-296058,id.txt	Metro	0.282
145	brk,20101202-296248,id.txt	Metro	0.256
146	brk,20101202-296130,id.txt	Metro	0.205
147	brk,20101203-296512,id.txt	Metro	0.179
148	brk,20101202-296288,id.txt	Metro	0.103
149	brk,20101202-296232,id.txt	Metro	0.051
150	brk,20101203-296424,id.txt	Metro	0.359
151	brk,20101201-295835,id.txt	Metro	0.179
152	brk,20101202-296320,id.txt	Metro	0.256
153	brk,20101203-296574,id.txt	Metro	0.026
154	brk,20101203-296591,id.txt	Metro	0.128
155	brk,20101203-296588,id.txt	Metro	0.256
156	brk,20101203-296524,id.txt	Metro	0.128
157	brk,20101203-296566,id.txt	Metro	0.308
158	brk,20101202-296202,id.txt	Metro	0.205
159	brk,20101203-296614,id.txt	Metro	0.231
160	brk,20101203-296533,id.txt	Metro	0.154
161	brk,20101202-296179,id.txt	Metro	0.179
162	brk,20101201-295935,id.txt	Metro	0.205
163	brk,20101202-296180,id.txt	Metro	0.154
164	brk,20101203-296641,id.txt	Metro	0.128
165	brk,20101202-296217,id.txt	Metro	0.179
166	brk,20101203-296483,id.txt	Metro	0.051
167	brk,20101203-296618,id.txt	Metro	0.308
168	brk,20101201-295769,id.txt	Metro	0.256
169	brk,20101201-295912,id.txt	Metro	0.179
170	brk,20101203-296506,id.txt	Metro	0.179
171	brk,20101201-295830,id.txt	Olahraga	0.769
172	brk,20101201-295764,id.txt	Olahraga	0.385
173	brk,20101202-296162,id.txt	Olahraga	0.256
174	brk,20101202-296358,id.txt	Olahraga	0.821
175	brk,20101202-296218,id.txt	Olahraga	0.846
176	brk,20101203-296431,id.txt	Olahraga	0.564

177	brk,20101202-296108,id.txt	Olahraga	0.436
178	brk,20101202-296201,id.txt	Olahraga	0.795
179	brk,20101203-296558,id.txt	Olahraga	0.897
180	brk,20101201-295918,id.txt	Olahraga	0.333
181	brk,20101201-296055,id.txt	Olahraga	0.692
182	brk,20101202-296152,id.txt	Olahraga	0.923
183	brk,20101202-296312,id.txt	Olahraga	0.846
184	brk,20101201-295784,id.txt	Olahraga	0.923
185	brk,20101201-296061,id.txt	Olahraga	0.821
186	brk,20101202-296258,id.txt	Olahraga	0.821
187	brk,20101203-296637,id.txt	Olahraga	0.718
188	brk,20101202-296175,id.txt	Olahraga	0.436
189	brk,20101201-295812,id.txt	Olahraga	0.795
190	brk,20101202-296240,id.txt	Olahraga	0.974
191	brk,20101201-295971,id.txt	Olahraga	0.846
192	brk,20101201-296082,id.txt	Olahraga	0.769
193	brk,20101201-295794,id.txt	Olahraga	0.436
194	brk,20101202-296241,id.txt	Olahraga	0.692
195	brk,20101203-296660,id.txt	Olahraga	0.179
196	brk,20101201-295837,id.txt	Olahraga	0.923
197	brk,20101201-296015,id.txt	Olahraga	0.231
198	brk,20101203-296631,id.txt	Olahraga	0.846
199	brk,20101203-296650,id.txt	Olahraga	0.846
200	brk,20101203-296529,id.txt	Olahraga	0.974
201	brk,20101202-296107,id.txt	Olahraga	0.897
202	brk,20101201-295910,id.txt	Olahraga	0.667
203	brk,20101203-296497,id.txt	Olahraga	0.795
204	brk,20101201-296034,id.txt	Olahraga	0.359
205	brk,20101201-295771,id.txt	Olahraga	0.923
206	brk,20101202-296110,id.txt	Olahraga	0.615
207	brk,20101201-295954,id.txt	Olahraga	0.795
208	brk,20101203-296513,id.txt	Olahraga	0.692
209	brk,20101201-295851,id.txt	Olahraga	0.385
210	brk,20101201-295867,id.txt	Olahraga	0.974
211	brk,20101201-296057,id.txt	Olahraga	0.769
212	brk,20101201-295893,id.txt	Olahraga	0.769
213	brk,20101203-296657,id.txt	Olahraga	0.205
214	brk,20101201-295763,id.txt	Olahraga	0.821

215	brk,20101202-296389,id.txt	Olahraga	0.462
216	brk,20101201-295767,id.txt	Olahraga	0.949
217	brk,20101201-295804,id.txt	Olahraga	0.231
218	brk,20101201-296029,id.txt	Olahraga	0.923
219	brk,20101202-296145,id.txt	Olahraga	0.846
220	brk,20101202-296144,id.txt	Bisnis	0.769
221	brk,20101202-296178,id.txt	Bisnis	0.41
222	brk,20101202-296383,id.txt	Bisnis	0.359
223	brk,20101201-295832,id.txt	Bisnis	0.282
224	brk,20101202-296244,id.txt	Bisnis	0.282
225	brk,20101202-296397,id.txt	Bisnis	0.385
226	brk,20101201-296071,id.txt	Bisnis	0.436
227	brk,20101202-296379,id.txt	Bisnis	0.462
228	brk,20101202-296336,id.txt	Bisnis	0.41
229	brk,20101202-296399,id.txt	Bisnis	0.308
230	brk,20101202-296398,id.txt	Bisnis	0.564
231	brk,20101201-295874,id.txt	Bisnis	0.513
232	brk,20101202-296338,id.txt	Bisnis	0.077
233	brk,20101201-295787,id.txt	Bisnis	0.615
234	brk,20101201-295806,id.txt	Bisnis	0.641
235	brk,20101202-296209,id.txt	Bisnis	0.308
236	brk,20101202-296233,id.txt	Bisnis	0.179
237	brk,20101202-296313,id.txt	Bisnis	0.641
238	brk,20101203-296561,id.txt	Bisnis	0.231
239	brk,20101203-296626,id.txt	Bisnis	0.41
240	brk,20101201-295816,id.txt	Bisnis	0.667
241	brk,20101202-296268,id.txt	Bisnis	0.59
242	brk,20101202-296214,id.txt	Bisnis	0.59
243	brk,20101203-296522,id.txt	Bisnis	0.59
244	brk,20101202-296121,id.txt	Bisnis	0.692
245	brk,20101201-295964,id.txt	Bisnis	0.462
246	brk,20101203-296674,id.txt	Bisnis	0.308
247	brk,20101202-296351,id.txt	Bisnis	0.205
248	brk,20101201-296077,id.txt	Bisnis	0.308
249	brk,20101201-296011,id.txt	Bisnis	0.692
250	brk,20101202-296150,id.txt	Bisnis	0.436
251	brk,20101203-296454,id.txt	Bisnis	0.41
252	brk,20101202-296329,id.txt	Bisnis	0.615

253	brk,20101201-295916,id.txt	Bisnis	0.487
254	brk,20101201-295865,id.txt	Bisnis	0.692
255	brk,20101201-296076,id.txt	Bisnis	0.564
256	brk,20101202-296400,id.txt	Bisnis	0.359
257	brk,20101203-296475,id.txt	Bisnis	0.564
258	brk,20101203-296542,id.txt	Bisnis	0.641
259	brk,20101201-296038,id.txt	Bisnis	0.026
260	brk,20101201-295779,id.txt	Bisnis	0.205
261	brk,20101203-296551,id.txt	Bisnis	0.308
262	brk,20101202-296266,id.txt	Bisnis	0.359
263	brk,20101203-296675,id.txt	Bisnis	0.41
264	brk,20101201-296014,id.txt	Bisnis	0.538
265	brk,20101203-296501,id.txt	Bisnis	0.385
266	brk,20101203-296548,id.txt	Bisnis	0.256
267	brk,20101201-295963,id.txt	Bisnis	0.538
268	brk,20101202-296362,id.txt	Bisnis	0.333
269	brk,20101201-296079,id.txt	Bisnis	0.59
270	brk,20101202-296345,id.txt	Bisnis	0.564
271	brk,20101202-296181,id.txt	Bisnis	0.718
272	brk,20101201-295857,id.txt	Bisnis	0.179
273	brk,20101202-296282,id.txt	Bisnis	0.256
274	brk,20101202-296155,id.txt	Bisnis	0.333
275	brk,20101203-296649,id.txt	Bisnis	0.333
276	brk,20101202-296138,id.txt	Bisnis	0.256
277	brk,20101202-296216,id.txt	Bisnis	0.077
278	brk,20101202-296315,id.txt	Bisnis	0.179
279	brk,20101201-296075,id.txt	Bisnis	0.385
280	brk,20101202-296118,id.txt	Nusa	0.41
281	brk,20101201-295821,id.txt	Nusa	0.59
282	brk,20101201-296072,id.txt	Nusa	0.436
283	brk,20101202-296163,id.txt	Nusa	0.513
284	brk,20101203-296517,id.txt	Nusa	0.282
285	brk,20101202-296377,id.txt	Nusa	0.308
286	brk,20101201-296059,id.txt	Nusa	0.564
287	brk,20101201-295854,id.txt	Nusa	0.333
288	brk,20101202-296342,id.txt	Nusa	0.359
289	brk,20101202-296270,id.txt	Nusa	0.615
290	brk,20101202-296097,id.txt	Nusa	0.538

291	brk,20101203-296463,id.txt	Nusa	0.205
292	brk,20101203-296498,id.txt	Nusa	0.333
293	brk,20101202-296122,id.txt	Nusa	0.462
294	brk,20101202-296096,id.txt	Nusa	0.59
295	brk,20101201-296003,id.txt	Nusa	0.487
296	brk,20101201-295807,id.txt	Nusa	0.513
297	brk,20101202-296221,id.txt	Nusa	0.282
298	brk,20101201-296017,id.txt	Nusa	0.282
299	brk,20101202-296323,id.txt	Nusa	0.308
300	brk,20101201-295947,id.txt	Nusa	0.385
301	brk,20101203-296540,id.txt	Nusa	0.436
302	brk,20101202-296111,id.txt	Nusa	0.205
303	brk,20101202-296380,id.txt	Nusa	0.41
304	brk,20101202-296390,id.txt	Nusa	0.282
305	brk,20101202-296146,id.txt	Nusa	0.385
306	brk,20101203-296484,id.txt	Nusa	0.564
307	brk,20101203-296624,id.txt	Nusa	0.41
308	brk,20101203-296496,id.txt	Nusa	0.692
309	brk,20101203-296620,id.txt	Nusa	0.564
310	brk,20101203-296604,id.txt	Nusa	0.615
311	brk,20101203-296568,id.txt	Nusa	0.333
312	brk,20101203-296599,id.txt	Nusa	0.487
313	brk,20101201-295868,id.txt	Nusa	0.615
314	brk,20101202-296154,id.txt	Nusa	0.487
315	brk,20101203-296677,id.txt	Nusa	0.256
316	brk,20101203-296471,id.txt	Nusa	0.564
317	brk,20101202-296296,id.txt	Nusa	0.436
318	brk,20101202-296305,id.txt	Nusa	0.744
319	brk,20101202-296357,id.txt	Nusa	0.385
320	brk,20101202-296109,id.txt	Nusa	0.513
321	brk,20101201-295949,id.txt	Nusa	0.436
322	brk,20101202-296306,id.txt	Nusa	0.692
323	brk,20101201-295885,id.txt	Nusa	0.385
324	brk,20101201-295948,id.txt	Nusa	0.256
325	brk,20101201-295986,id.txt	Nusa	0.487
326	brk,20101203-296633,id.txt	Nusa	0.513
327	brk,20101202-296182,id.txt	Nusa	0.154
328	brk,20101203-296552,id.txt	Nusa	0.308

329	brk,20101202-296239,id.txt	Nusa	0.41
330	brk,20101203-296596,id.txt	Nusa	0.179
331	brk,20101201-295871,id.txt	Nusa	0.513
332	brk,20101203-296507,id.txt	Nusa	0.359
333	brk,20101201-295834,id.txt	Nusa	0.538
334	brk,20101201-295981,id.txt	Nusa	0.436
335	brk,20101201-295880,id.txt	Nusa	0.385
336	brk,20101201-295891,id.txt	Nusa	0.462
337	brk,20101202-296171,id.txt	Nusa	0.359
338	brk,20101203-296607,id.txt	Nusa	0.205
339	brk,20101202-296193,id.txt	Nusa	0.231
340	brk,20101203-296446,id.txt	Nusa	0.513
341	brk,20101202-296167,id.txt	Nusa	0.308
342	brk,20101203-296613,id.txt	Nusa	0.564
343	brk,20101203-296567,id.txt	Nusa	0.282
344	brk,20101202-296347,id.txt	Nusa	0.462
345	brk,20101203-296478,id.txt	Nusa	0.308
346	brk,20101203-296518,id.txt	Nusa	0.308
347	brk,20101203-296460,id.txt	Nusa	0.564
348	brk,20101201-296025,id.txt	Nusa	0.359
349	brk,20101201-295983,id.txt	Nusa	0.436
350	brk,20101201-296080,id.txt	Nusa	0.359
351	brk,20101202-296372,id.txt	Nusa	0.692
352	brk,20101202-296385,id.txt	Nusa	0.513
353	brk,20101201-296020,id.txt	Nusa	0.333
354	brk,20101203-296562,id.txt	Nusa	0.513
355	brk,20101201-296081,id.txt	Nusa	0.513
356	brk,20101203-296560,id.txt	Nusa	0.359
357	brk,20101203-296600,id.txt	Nusa	0.641
358	brk,20101203-296573,id.txt	Nusa	0.718
359	brk,20101202-296147,id.txt	Nusa	0.41
360	brk,20101202-296353,id.txt	Nusa	0.41
361	brk,20101203-296656,id.txt	Nusa	0.615
362	brk,20101201-295987,id.txt	Nusa	0.487
363	brk,20101202-296235,id.txt	Nusa	0.667
364	brk,20101202-296255,id.txt	Nusa	0.513
365	brk,20101201-295992,id.txt	Nusa	0.256
366	brk,20101201-295908,id.txt	Nusa	0.59

367	brk,20101202-296352,id.txt	Nusa	0.205
368	brk,20101203-296531,id.txt	Nusa	0.308
369	brk,20101201-295903,id.txt	Nusa	0.513
370	brk,20101201-295957,id.txt	Nusa	0.308
371	brk,20101203-296638,id.txt	Nusa	0.308
372	brk,20101202-296316,id.txt	Nusa	0.564
373	brk,20101201-295822,id.txt	Nusa	0.538
374	brk,20101201-295998,id.txt	Nusa	0.436
375	brk,20101201-296030,id.txt	Teknologi	0.103
376	brk,20101202-296113,id.txt	Teknologi	0.256
377	brk,20101201-295973,id.txt	Teknologi	0.103
378	brk,20101201-295955,id.txt	Teknologi	0.154
379	brk,20101203-296570,id.txt	Teknologi	0.077
380	brk,20101202-296102,id.txt	Teknologi	0.154
381	brk,20101201-296027,id.txt	Teknologi	0.154
382	brk,20101201-295877,id.txt	Teknologi	0.051
383	brk,20101201-295768,id.txt	Teknologi	0.128
384	brk,20101201-295829,id.txt	Teknologi	0.077
385	brk,20101203-296525,id.txt	Teknologi	0.154
386	brk,20101201-295965,id.txt	Teknologi	0.231
387	brk,20101203-296629,id.txt	Teknologi	0.051
388	brk,20101202-296129,id.txt	Teknologi	0.179
389	brk,20101201-295788,id.txt	Teknologi	0.077
390	brk,20101201-295791,id.txt	Teknologi	0.231
391	brk,20101203-296514,id.txt	Teknologi	0.179
392	brk,20101201-295846,id.txt	Teknologi	0.103
393	brk,20101202-296185,id.txt	Teknologi	0.051

Lampiran 3. Hasil Pengujian MKNN dan KNN (K=6, 7, 8, 9, 10, 11, 12, 13, 17, 18, 19, dan 20)

Hasil Evaluasi Uji Coba MKNN K=6, 7, 8, 9, 10, dan 11

Kategori	K=6			K=7			K=8			K=9			K=10			K=11		
	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	47	1	4	47	1	6	47	1	6	48	0	5	47	1	5	48	0	5
Nasional	94	18	34	94	18	38	94	18	37	93	19	38	94	18	36	94	18	39
Bisnis	52	7	16	50	9	15	50	9	15	49	10	14	48	11	14	48	11	13
Internasional	6	17	2	6	17	2	6	17	2	6	17	2	6	17	2	6	17	2
Teknologi	8	10	0	8	10	0	8	10	0	8	10	0	8	10	0	8	10	0
Nusa	71	23	37	69	25	39	69	25	38	71	23	38	73	21	38	72	22	39
Metro	14	19	2	11	22	2	13	20	2	13	20	2	14	19	2	12	21	1

Hasil Evaluasi Uji Coba MKNN K=12, 13, 17, 18, 19, dan 20

Kategori	K=12			K=13			K=17			K=18			K=19			K=20		
	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	5	48	0	5	48	0	4	48	0	4	48	0	5	48	0	5
Nasional	95	17	41	95	17	44	96	16	44	96	16	47	97	15	48	97	15	46
Bisnis	48	11	12	48	11	12	48	11	13	48	11	14	48	11	14	48	11	13
Internasional	5	18	2	6	17	2	6	17	1	5	18	1	5	18	1	5	18	1
Teknologi	8	10	0	8	10	0	6	12	0	6	12	0	6	12	0	6	12	0
Nusa	73	21	37	71	23	36	70	24	45	69	25	44	69	25	41	69	25	43
Metro	12	21	1	11	22	1	6	27	0	5	28	0	5	28	0	6	27	0

Hasil Evaluasi Efektivitas MKNN K=6, 7, 8, dan 9

Kategori	K=6			K=7			K=8			K=9		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	0.979	0.922	0.949	0.979	0.887	0.931	0.979	0.887	0.931	1	0.906	0.95
Nasional	0.839	0.734	0.783	0.839	0.712	0.77	0.839	0.718	0.774	0.83	0.71	0.765
Bisnis	0.881	0.765	0.819	0.847	0.769	0.806	0.847	0.769	0.806	0.831	0.778	0.803
Internasional	0.261	0.75	0.387	0.261	0.75	0.387	0.261	0.75	0.387	0.261	0.75	0.387
Teknologi	0.444	1	0.615	0.444	1	0.615	0.444	1	0.615	0.444	1	0.615
Nusa	0.755	0.657	0.703	0.734	0.639	0.683	0.734	0.645	0.687	0.755	0.651	0.7
Metro	0.424	0.875	0.571	0.333	0.846	0.478	0.394	0.867	0.542	0.394	0.867	0.542
Rata-rata	**0.655**	**0.815**	**0.69**	**0.634**	**0.8**	**0.667**	**0.643**	**0.805**	**0.677**	**0.645**	**0.809**	**0.68**

Hasil Evaluasi Efektivitas MKNN K=10, 11, 12, dan 13

Kategori	K=10			K=11			K=12			K=13		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	0.979	0.904	0.94	1	0.906	0.95	1	0.906	0.95	1	0.906	0.95
Nasional	0.839	0.723	0.777	0.839	0.707	0.767	0.848	0.699	0.766	0.848	0.683	0.757
Bisnis	0.814	0.774	0.793	0.814	0.787	0.8	0.814	0.8	0.807	0.814	0.8	0.807
Internasional	0.261	0.75	0.387	0.261	0.75	0.387	0.217	0.714	0.333	0.261	0.75	0.387
Teknologi	0.444	1	0.615	0.444	1	0.615	0.444	1	0.615	0.444	1	0.615
Nusa	0.777	0.658	0.712	0.766	0.649	0.702	0.777	0.664	0.716	0.755	0.664	0.706
Metro	0.424	0.875	0.571	0.364	0.923	0.522	0.364	0.923	0.522	0.333	0.917	0.489
Rata-rata	**0.648**	**0.812**	**0.685**	**0.641**	**0.817**	**0.678**	**0.638**	**0.815**	**0.673**	**0.637**	**0.817**	**0.673**

Hasil Evaluasi Efektivitas MKNN K=17, 18, 19, dan 20

Kategori	K=17			K=18			K=19			K=20		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	1	0.923	0.96	1	0.923	0.96	1	0.906	0.95	1	0.906	0.95
Nasional	0.857	0.686	0.762	0.857	0.671	0.753	0.866	0.669	0.755	0.866	0.678	0.761
Bisnis	0.814	0.787	0.8	0.814	0.774	0.793	0.814	0.774	0.793	0.814	0.787	0.8
Internasional	0.261	0.857	0.4	0.217	0.833	0.345	0.217	0.833	0.345	0.217	0.833	0.345
Teknologi	0.333	1	0.5	0.333	1	0.5	0.333	1	0.5	0.333	1	0.5
Nusa	0.745	0.609	0.67	0.734	0.611	0.667	0.734	0.627	0.676	0.734	0.616	0.67
Metro	0.182	1	0.308	0.152	1	0.263	0.152	1	0.263	0.182	1	0.308
Rata-rata	**0.599**	**0.837**	**0.628**	**0.587**	**0.83**	**0.612**	**0.588**	**0.83**	**0.612**	**0.592**	**0.831**	**0.619**

Hasil Evaluasi Uji Coba KNN K=6, 7, 8, 9, 10, dan 11

Kategori	K=6			K=7			K=8			K=9			K=10			K=11		
	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	47	1	2	47	1	2	47	1	0	47	1	1	46	2	0	48	0	0
Nasional	94	18	29	94	18	34	94	18	34	93	19	31	95	17	26	94	18	30
Bisnis	52	7	15	52	7	15	52	7	14	53	6	14	53	6	15	53	6	14
Internasional	11	12	4	11	12	4	12	11	4	11	12	4	11	12	5	11	12	5
Teknologi	9	9	0	9	9	0	9	9	0	9	9	0	9	9	0	9	9	0
Nusa	70	24	32	64	30	34	67	27	32	71	23	31	76	18	29	72	22	29
Metro	18	15	4	18	15	3	19	14	3	19	14	3	19	14	3	19	14	3

Hasil Evaluasi Uji Coba KNN K=12, 13, 17, 18, 19, dan 20

Kategori	K=12			K=13			K=17			K=18			K=19			K=20		
	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP	TP	FN	FP
Olahraga	48	0	0	48	0	0	48	0	0	48	0	0	48	0	0	48	0	0
Nasional	94	18	28	93	19	30	93	19	32	95	17	32	95	17	34	95	17	33
Bisnis	53	6	15	54	5	16	52	7	12	52	7	14	51	8	14	51	8	14
Internasional	11	12	5	11	12	5	11	12	4	11	12	4	11	12	5	11	12	5
Teknologi	9	9	0	9	9	0	10	8	0	9	9	0	8	10	0	8	10	0
Nusa	72	22	31	70	24	32	72	22	32	70	24	31	71	23	31	71	23	33
Metro	18	15	3	18	15	1	20	13	1	20	13	1	18	15	1	17	16	1

Hasil Evaluasi Efektivitas KNN K=6, 7, 8, dan 9

Kategori	K=6			K=7			K=8			K=9		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	0.979	0.959	0.969	0.979	0.959	0.969	0.979	1	0.989	0.979	0.979	0.979
Nasional	0.839	0.764	0.8	0.839	0.734	0.783	0.839	0.734	0.783	0.83	0.75	0.788
Bisnis	0.881	0.776	0.825	0.881	0.776	0.825	0.881	0.788	0.832	0.898	0.791	0.841
Internasional	0.478	0.733	0.579	0.478	0.733	0.579	0.522	0.75	0.615	0.478	0.733	0.579
Teknologi	0.5	1	0.667	0.5	1	0.667	0.5	1	0.667	0.5	1	0.667
Nusa	0.745	0.686	0.714	0.681	0.653	0.667	0.713	0.677	0.694	0.755	0.696	0.724
Metro	0.545	0.818	0.655	0.545	0.857	0.667	0.576	0.864	0.691	0.576	0.864	0.691
Rata-rata	**0.71**	**0.82**	**0.744**	**0.701**	**0.816**	**0.737**	**0.716**	**0.83**	**0.753**	**0.717**	**0.83**	**0.753**

Hasil Evaluasi Efektivitas KNN K=10, 11, 12, dan 13

Kategori	K=10			K=11			K=12			K=13		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	0.958	1	0.979	1	1	1	1	1	1	1	1	1
Nasional	0.848	0.785	0.815	0.839	0.758	0.797	0.839	0.77	0.803	0.83	0.756	0.791
Bisnis	0.898	0.779	0.835	0.898	0.791	0.841	0.898	0.779	0.835	0.915	0.771	0.837
Internasional	0.478	0.688	0.564	0.478	0.688	0.564	0.478	0.688	0.564	0.478	0.688	0.564
Teknologi	0.5	1	0.667	0.5	1	0.667	0.5	1	0.667	0.5	1	0.667
Nusa	0.809	0.724	0.764	0.766	0.713	0.738	0.766	0.699	0.731	0.745	0.686	0.714
Metro	0.576	0.864	0.691	0.576	0.864	0.691	0.545	0.857	0.667	0.545	0.947	0.692
Rata-rata	**0.724**	**0.834**	**0.759**	**0.723**	**0.83**	**0.757**	**0.718**	**0.828**	**0.752**	**0.716**	**0.836**	**0.752**

Hasil Evaluasi Efektivitas KNN K=17, 18, 19, dan 20

Kategori	K=17			K=18			K=19			K=20		
	Rec.	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**	**Rec.**	**Prec.**	**F1 M.**
Olahraga	1	1	1	1	1	1	1	1	1	1	1	1
Nasional	0.83	0.744	0.785	0.848	0.748	0.795	0.848	0.736	0.788	0.848	0.742	0.792
Bisnis	0.881	0.813	0.846	0.881	0.788	0.832	0.864	0.785	0.823	0.864	0.785	0.823
Internasional	0.478	0.733	0.579	0.478	0.733	0.579	0.478	0.688	0.564	0.478	0.688	0.564
Teknologi	0.556	1	0.714	0.5	1	0.667	0.444	1	0.615	0.444	1	0.615
Nusa	0.766	0.692	0.727	0.745	0.693	0.718	0.755	0.696	0.724	0.755	0.683	0.717
Metro	0.606	0.952	0.741	0.606	0.952	0.741	0.545	0.947	0.692	0.515	0.944	0.667
Rata-rata	**0.731**	**0.848**	**0.77**	**0.723**	**0.845**	**0.762**	**0.705**	**0.836**	**0.744**	**0.701**	**0.834**	**0.74**