KLASIFIKASI BERITA PADA *TWITTER* DENGAN MENGGUNAKAN METODE *NAÏVE BAYES* DAN *FEATURE EXPANSION* BERBASIS *COSINE SIMILARITY*

SKRIPSI

Untuk memenuhi sebagian persyaratan
Memperoleh gelar Sarjana Komputer

Disusun oleh:
Resti Febriana
NIM: 135150218113002

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

KLASIFIKASI BERITA PADA *TWITTER* DENGAN MENGGUNAKAN METODE *NAÏVE BAYES* DAN *FEATURE EXPANSION* BERBASIS *COSINE SIMILARITY*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Resti Febriana
NIM: 135150218113002

Skripsi ini telah diuji dan dinyatakan lulus pada:
31 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
M. Ali Fauzi, S.Kom, M.Kom	Rizal Setya Perdana, S.Kom, M.Kom
NIK: 201502 890101 1 001	NIK: 201603 910118 1 001

Mengetahui
Ketua Jurusan Teknik Infomatika

Tri Astoto Kurniawan, S.T, M.T, Ph. D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis di sitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 31 Juli 2018

Resti Febriana

NIM: 135150218113002

KATA PENGANTAR

Puji syukur kepada Allah SWT yang selalu memberikan pertolongan, ridho, kesehatan, rizki serta karuania-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Berita Pada *Twitter* Dengan Menggunakan Metode *Naïve Bayes* Dan *Feature Expansion* Berbasis *Cosine Similarity*". Penulis mengucapkan terima kasih kepada semua pihak yang sangat membantu penulis dalam menyelesaikan skripsi ini. Ucapan terima kasih ditujukan kepada yang terhormat:

1. Bapak M. Ali Fauzi, S.Kom, M.Kom dan Bapak Rizal Setya Perdana, S.Kom, M.Kom selaku Dosen Pembimbing yang telah sabar dalam memberikan bimbingan, ilmu, arahan, bantuan, dan saran terhadap pelaksanaan dan penulisan skripsi penulis.
2. Bapak Agus Wahyu Widodo, S.T., M. Cs. selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya,
3. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph. D. selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya,
4. Bapak Issa Arwani S.Kom, M. Sc. Dan Bapak Agus Wahyu Widodo, S.T., M. Cs, selaku deosen Penasehat Akademik yang selalu memberikan nasehat kepada penulis seama menempuh studi ,
5. Seluruh dosen dan karyawan Fakultas Ilmu Komputer Universitas Brawijaya yang telah mendidik dan memberikan ilmu kepada penulis selama menempuh masa studi di Fakultas Ilmu Komputer,
6. Seluruh keluarga penulis Bapak Sarkam dan Ibu Misinah selaku kedua orang tua penulis, Desi Rina Sari dan Moh. Isnaini selaku kakak penulis serta seluruh keluarga besar atas segala do'a, nasihat, dukungan baik moril maupun materil yang begitu besar terhadap kelancaran dalam menyelesaikan skripsi ini.
7. Seluruh sahabat dan rekan seperjuangan penulis, Fyma Ardita, Ulva Febriana, Ikrar Amalia, Irma Pujadayanti, Aris Sandy S.A, A. Muhammad Sofwan, Firman Hamzah, M. Khojin, Aditya Septadaya, delapan sahabat dan yang tidak bisa disebutkan satu persatu terima kasih atas dukungan dan informasi yang diberikan demi kelancaran skripsi.

Penulis menyadari dalam penulisan skripsi ini terdapat banyak kekurangan baik dalam penulisan serta isinya, maka dari itu penulis menerima kritik dan saran untuk penyempurnaan skripsi ini.

Malang, 31 Juli 2018

Penulis

raestifebri@gmail.com

ABSTRAK

Resti Febriana, Klasifikasi Berita Pada *Twitter* Dengan Menggunakan Metode *Naïve Bayes* Dan *Feature Expansion* Berbasis *Cosine Similarity*.

Pembimbing: M. Ali Fauzi, S.Kom, M.Kom dan Rizal Setya Perdana, S.Kom, M.Kom

Informasi telah menjadi hal yang sangat dibutuhkan di era modern ini, terlebih dengan adanya berbagai media sosial yang mendukung perbaruan informasi. Twitter sebagai salah satu media sosial yang aktif digunakan untuk memperbarui informasi yang tergolong dalam *short text* atau berita pendek yang memiliki beberapa kesulitan ketika dilakukan klasifikasi, seperti kata yang ambigu, kata yang terdapat dalam data uji tidak pernah muncul dalam data latih dan sebagainya. Penelitian ini dilakukan untuk mengetahui pengaruh penggunaan *feature expansion* atau penambahan kata pada *short text* dalam hasil klasifikasi. Sebelum dilakukan klasifikasi, terlebih dahulu data yang akan diujikan ditambahkan dengan daftar kata yang telah dibuat sebelumnya sebagai sumber eksternal atau kamus dengan batasan tertentu yang telah ditetapkan. Batasan ini bertujuan untuk mengetahui nilai batasan minimal yang paling optimal dalam menghasilkan akurasi tertinggi dalam proses klasifikasi. Dalam proses pembuatan sumber eksternal dilakukan proses *cosine similarity* untuk mencari kedekatan antar kata. Hasil penelitian berupa akurasi yang menunjukkan adanya pengaruh penambahan *feature expansion* dalam hasil klasifikasi, hasil akurasi sebesar 83% pada klasifikasi tanpa penggunaan *feature expansion* dan meningkat menjadi 87% pada penggunaan *feature expansion* dengan nilai *threshold* 0,9.

Kata kunci: *cosine similarity, feature expansion, klasifikasi berita, naïve bayes, short text, threshold*

ABSTRACT

Resti Febriana , ***Classification of News on Twitter Using the Naïve Bayes Method and Feature Expansion Based on Cosine Similarity***

***Supervisors*: M. Ali Fauzi, S.Kom, M.Kom and Rizal Setya Perdana, S.Kom, M.Kom.,**

Information has become indispensable in this modern era, especially with the existence of various social media that support information update. Twitter as one of the most active social media is used to update information belonging to short text or short stories that have some difficulty when done classification, such as ambiguous word, the word contained in the test data never appear in the data train and so on. This research was conducted to determine the effect of using feature expansion or addition of word on short text in the result of classification. Prior to classification, the first data to be tested is added to the list of pre-made words as an external source or dictionary with specified limits. This limitation aims to determine the minimum value of the most optimal limit in generating the highest accuracy in the classification process. In the process of making external sources cosine similarity process is done to find the closeness between words. The result of this research is accurate showing effect of expansion of feature expansion in classification result, 83% accuracy in classification without feature expansion and increased to 87% on feature expansion with threshold value 0.9.

Keywords: *cosine similarity, feature expansion, naïve bayes, short text, text classification, threshold*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Informasi telah menjadi hal yang sangat dibutuhkan di era modern ini. Dalam setiap kesempatan setiap manusia berusaha memperoleh informasi baik dalam bentuk cetak, audio, visual maupun digital. Namun yang lebih digemari untuk saat ini adalah menggunakan media digital atau terhubung dalam jaringan internet. Menurut hasil survei yang dilakukan oleh Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) di tahun 2016, alasan utama pengguna internet di Indonesia adalah untuk memperbarui informasi sebanyak 25,3%, mengenai pekerjaan sebanyak 20,8%, hanya mengisi waktu luang sebanyak 13,5%, untuk sosialisasi sebanyak 10,3%, mengenai pendidikan sebanyak 9,2%, sebagai hiburan sebanyak 8,8%, dan untuk bisnis dagang sebanyak 8,5%. dari survei tersebut menunjukkan bahwa kebutuhan memperoleh informasi mendominasi perilaku pengguna internet.

Mengenai cara memperbarui informasi ada banyak cara yang dapat dilakukan, seperti halnya memperbarui informasi melalui media sosial. Menurut APJII media sosial merupakan jenis konten yang paling populer diakses sebanyak 97,4% dari seluruh pengguna internet. Mengenai media sosial, tentunya banyak sekali jenis media yang digunakan seperti Facebook, Instagram, dan Twitter. Twitter merupakan media sosial berbasis *online* yang terkenal pada pertengahan tahun 2012, Twitter juga termasuk dalam 10 besar media sosial yang paling banyak dikunjungi pada tahun 2014 (Perdana, 2015).Twitter sendiri merupakan media sosial yang berisi teks atau gambar yang disebut kicauan(*tweets*), yang mana kicauan tersebut hanya berisi 140 karakter saja (Twitter, 2016). Twitter merupakan salah satu media sosial yang cukup populer di Indonesia. Menurut Kementerian Komunikasi dan Informasi, Indonesia menempati peringkat ke 5 pengguna Twitter terbanyak di dunia pada tahun 2016 (Kominfo, 2016). Sehingga jenis media sosial ini sangat aktif dalam perbaruan informasi oleh penggunanya, namun dari informasi yang disampaikan dalam media sosial tersebut masih tercampur menjadi satu dan tidak terkelompok menjadi topik-topik tertentu. Hal ini tentu saja menyulitkan pengguna yang ingin mencari informasi terkait satu hal dengan maksimal. Meski terdapat fitur pencarian yang sudah disediakan, namun fitur tersebut masih belum maksimal dari kinerjanya. Misalnya pengguna mencari dengan *keyword* ekonomi, walaupun ada berita atau *tweets* yang berisi ekonomi namun tidak sedikit juga muncul jenis berita lain hal ini dikarenakan pencarian masih dikategorikan berdasarkan akunnya saja. Dan pengguna masih harus mencari dan memilah *tweets-tweets* tersebut. Sehingga dibutuhkan sebuah sistem yang dapat melakukan klasifikasi *tweets* yang sesuai dengan jenis atau kategori yang ada.

Salah satu metode klasifikasi yang populer adalah *naive bayes*. Metode ini memiliki akurasi yang cukup tinggi untuk memecahkan masalah klasifikasi, salah satu penelitian pengklasifikasian *Twitter* berdasarkan kategori pernah dilakukan sebelumnya oleh Perdana, R (Perdana, 2013). Hasil yang diperoleh untuk *recall*

sebesar 79%, sedangkan untuk *precision*-nya sebesar 80%. Dari hasil tersebut menunjukkan bahwa metode naive bayes baik digunakan dalam pengklasifikasian teks.

Secara umum penggunaan metode klasifikasi teks masih memiliki kelemahan untuk diterapkan pada *short-text* seperti Twitter (Sukarno, 2016). Kelemahan yang dimaksud adalah maksud kata yang ambigu karena teks berisikan sedikit kata-kata (Tang, 2017). Misalnya *tweet* yang berisi "melihat film ini membuat saya terbahak-bahak". Dari contoh tersebut, informasi yang disampaikan masih kurang lengkap, yang membuat terbahak-bahak judul film apa dan seperti apa. Belakangan ini penelitian terkait masalah *short-text* terus mengalami perkembangan, dengan tujuan untuk lebih memudahkan informasi yang disampaikan dengan menggunakan *short-text* sesuai dengan yang diharapkan. Salah satu metode yang sedang dalam pengembangan adalah klasifikasi dengan penambahan *feature expansion*. Penelitian yang dilakukan oleh sukarno dan Ali(2016) sudah menerapkan penambahan *feature expansion*, akurasi yang dihasilkan sebesar 82%. Lebih tinggi dari penelitian sebelumnya. *Feature expansion* merupakan teknik perluasan kata yang berasal dari informasi eksternal (*Unlabeled Background Knowledge*) seperti Wikipedia, Wordnet, dan dokumen berita sebelum dilakukan klasifikasi. Hal ini memungkinkan adanya penambahan kata yang seharusnya ada dalam suatu kelompok kategori yang sesuai (Mandala, 2009). Kata yang ditambahkan merupakan kata terdekat secara semantik. Kedekatan semantik dapat dihitung dengan model semantik terdistribusi atau MST, yang mana pada teknik ini memanfaatkan metode lain dalam proses pengumpulannya, salah satunya dengan *matrix* yang sama (Chunxia Jin, 2012). Kemudian hasilnya disimpan dalam sebuah kamus sebagai sumber pengetahuan eksternal untuk sistem.

Salah satu teknik *Feature Expansion* adalah dengan metode *cosine similarity*. *Cosine similarity* merupakan salah satu teknik perhitungan kedekatan kategori berbasis *vector*. Hasil penelitian yang dilakukan oleh Ogie Nurdiana dengan membandingkan metode *cosine similarity*, *jaccard* dan k-*nearest neighbor* (K-NN) yang digunakan pada proses klasifikasi dokumen teks dengan hasil akhir dari percobaan yang dilakukan didapatkan hasil bahwa metode cosine yang nilai kemiripannya paling tinggi yakni mencapai 41%, sedangkan untuk metode *jaccard* sebesar 19% dan untuk metode K-NN sebesar 40% (Nurdiana, 2016). Untuk penelitian lain yang dilakukan oleh Heri dengan pengukuran *cosine similarity* dan *euclidian distance* diperoleh hasil dari segi ketepatan, tentu hasil klusterisasi dengan menggunakan *cosine* memberikan hasil yang lebih baik bila dibandingkan dengan euclidian distance (Kurniawan, 2006).

Berdasarkan kajian yang telah dilakukan dengan melihat permasalahan di atas, maka diusulkan penelitian dengan judul "**Klasifikasi Berita Pada Twitter Dengan Menggunakan Metode *Naïve Bayes* Dan *Feature expansion* Berbasis *Cosine Similarity***". Pada penelitian ini, diharapkan dapat memberikan dampak positif dalam perkembangan media sosial Twitter khususnya di Indonesia agar

pengguna lebih mudah dalam membaca suatu berita berdasarkan kebutuhan informasi.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dipaparkan, dapat diambil beberapa rumusan masalah sebagai berikut:

1. Bagaimana pengaruh *feature expansion* terhadap performa hasil akurasi dalam klasifikasi?
2. Bagaimana pengaruh nilai *Threshold Cosine Similarity* pada hasil akurasi?

1.3 Tujuan

Tujuan dari penelitian ini dibagi menjadi dua bagian, yaitu: tujuan umum dan tujuan khusus yang ditunjukkan sebagai berikut.

Tujuan umum:

Memaksimalkan penggunaan metode klasifikasi pada *Twitter* dengan metode *cosine similarity* dan naive bayes

Tujuan khusus:

1. Mengetahui pengaruh *feature expansion* dalam penilaian akurasi klasifikasi.
2. Mengetahui pengaruh penggunaan *Threshold* dalam penilaian akurasi.

1.4 Manfaat

Hasil dari penelitian ini diharapkan dapat memberikan manfaat untuk berbagai pihak. Utamanya bagi pengguna Twitter, yang nantinya mempermudah dalam pencarian jenis konten informasi yang diinginkan. Mengingat sangat banyak sekali konten informasi yang dituliskan oleh sesama pengguna Twitter.

1.5 Batasan Masalah

Agar tidak memperluas area pembahasan dalam penelitian ini, maka penelitian ini dibatasi dalam hal:

1. Data latih dan data uji yang digunakan adalah *tweets* pada Twitter dengan bahasa Indonesia saja.
2. Informasi eksternal yang digunakan dalam pembuatan kamus berasal dari berita Detik.com dan Kompas.com.
3. Data yang digunakan sebagai data latih dan data uji merupakan tweets dari akun Detikcom dan Kompas.com
4. Klasifikasi yang dihasilkan berupa kategori ekonomi, kesehatan, *entertaiment*, teknologi, dan olahraga.
5. Implementasi yang diterapkan pada penelitian ini tidak dapat menangani teks singkatan dan kata tidak baku.
6. Implementasi yang diterapkan tidak menangani kata pada tagar atau *hashtag.*

1.6 Sistematika pembahasan

Secara garis besar sistematika penulisan penelitian ini terdiri dari tujuh bab, yaitu:

BAB 1 PENDAHULUAN

Pada bab I Memuat latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, dan sistematika penulisan.

BAB 2 LANDASAN KEPUSTAKAAN

Membahas tentang uraian dan pembahasan terkait teori, konsep, model, dan metode dari literatur ilmiah, yang berkaitan dengan tema masalah yang diangkat dalam penelitian ini.

BAB 3 METODOLOGI

Menjelaskan langkah yang digunakan oleh peneliti untuk menyelesaikan penyelesaian dengan metode yang dipilih oleh peneliti.

BAB 4 ANALISIS DAN PERANCANGAN

Menjelaskan analisis dan perancangan yang terdiri dari perancangan sistem untuk pendekatan *cosine similarity* dengan kalsifikasi metode naive bayes, perancangan antar muka, serta perancangan uji coba dan evaluasi.

BAB 5 IMPLEMENTASI

Memuat pembahasan proses implementasi, batasan-batasan implementasi, serta Algoritme yang digunakan dalam sistem.

BAB 6 PENGUJIAN DAN ANALISIS

Memuat pengujian terhadap sistem yang sudah dibuat dan analisis hasil pengujian tentang optimasi pendekatan *cosine similarity* dengan klasifikasi naive bayes.

BAB 7 PENUTUP

Memuat kesimpulan yang diperoleh dari penelitian yang telah dilakukan beserta saran untuk pengembangan penelitian lebih lanjut lagi.

BAB 2 TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai pustaka yang akan digunakan pada penelitian ini. Pustaka meliputi penelitian terdahulu dan metode yang digunakan pada penelitian.

2.1 Tinjauan Penelitian Terdahulu

(Perdana, 2013) dalam penelitiannya membahas tentang "Pengkategorian Pesan Singkat pada Jejaring Sosial Twitter dengan Metode Klasifikasi *Naive Bayes*". Dengan melakukan *recall* dan *pecision* hasil yang diperoleh sebesar 79% dan 80% dengan hasil gabungan dari keduanya sebesar 78%. Dari penelitian tersebut menunjukkan bahwa metode pengklasifikasian dengan menggunakan *naive bayes* menghasilkan akurasi yang tinggi, yakni sebesar 78%. Yang perlu diketahui adalah pada penelitan tersebut memiliki kelemahan pada beberapa hasil klasifikasi, hasil yang seharusnya bernilai *true positif* (sesuai kategori) menjadi *true negative* (tidak sesuai kategori). Hal ini dikarenakan memang pengklasifikasian pesan singkat tidak mudah dilakukan, mengingat dalam pesan singkat terdapat beberapa karakteristik yang harus melekat pada teksnya.

(Sukarno, 2016) pada penelitiannya membahas tentang "Klasifikasi Tweets pada Twitter Manggunakan Metode *Naive Bayes* dan *Query Expansion* Berbasis Apriori". Pada penelitian tersebut dijelaskan mengenai penggunaan *query expansion* dalam klasifikasi teks, hasil akurasi menunjukkan tingkat akurasi penggunaan klasifikasi *naive bayes* dan penambahan *query expansion* mampu meningkatkan akurasi sebesar 82% dengan data latih sebanyak 1600 data latih. Hal ini menunjukkan dengan *query expansion* mampu menambah kualitas klasifikasi *naive bayes*. Namun pada penelitian yang dilakukan oleh sukarno, ternyata memang pengklasifikasian berita pada Twitter cukup sulit dilakukan hal ini ditunjukkan oleh perbandingan hasil akhir yang ditunjukkan. Ada perbedaan hasil yang cukup mencolok dari hasil penelitian tersebut, yakni pada hasil pengujian tanpa *preprocessing* dan tanpa *query expansion* hasil yang didapat sebesar 52%, namun ketika pengujian dilakukan tanpa *preprocessing* dan dengan *query expansion* hasilnya turun menjadi 26%,dan ketika pengujian dilakukan dengan *preprocessing* dan tanpa *query expansion* hasilnya justru meningkat menjadi 69%, serta pengujian yang terakhir yakni dengan *preprocessing* dan dengan *query expansion* hasilnya meningkat menjad 82%.

Dari hasil penelitian yang dilakukan oleh (Nurdiana, 2016) dengan membandingkan metode *cosine similarity*, *jaccard* dan k-*nearest neighbor* (K-NN) yang digunakan pada proses klasifikasi dokumen teks dengan hasil akhir dari percobaan yang dilakukan didapatkan hasil bahwa metode *cosine* yang nilai kemiripannya paling tinggi yakni mencapai 41%, sedangkan untuk metode *jaccard* sebesar 19% dan untuk metode K-NN sebesar 40%. Hal ini dikarenakan dengan metode *cosine similarity* mempunyai konsep normalisasi panjang *vector* data dengan membandingkan N-gram yang sejajar satu sama lain dari 2 pembanding. Sedangkan pada metode *jaccard* hanya membandingkan isi N-gram dengan eksak dan hanya melihat apakah ada satu N-gram tertentu tanpa membandingkan posisi

penulisan. Pada K-NN tidak mempunyai normalisasi panjang vektor data, sehingga nilai akurasi hanya dipengaruhi oleh parameter nilai tetangga terdekat. Dari penelitian tersebut dapat disimpulkan perhitungan kedekatan kata yang paling efektif adalah dengan menggunakan *cosine simlarity*.

Dari hasil penelitian yang dilakukan oleh (Al-Smadi, 2017), mengenai identifikasi frasa dan semantik teks *similarity* pada teks berita Twitter dengan menggunakan tulisan arab dengan menggunakan *lexical*, *sintactic* dan *semantic feature*. Pada penelitian tersebut dijelaskan terkait perbandingan yang dihasilkan ketika menggunakan *lexical* dan *semantic*. Dari hasil yang diperoleh dijelaskan bahwa dengan menggunakan kedua metode tersebut secara bersama-sama hasil akurasi lebih maksimal dari pada menggunakan klasifikasi yang sederhana.

Tabel 2.1 Perbandingan hasil penelitian sebelumnya

No.	judul	Obyek	Metode	Hasil
1	Pengkategorian Pesan Singkat pada Jejaring Sosial Twitter dengan Metode Klasifikasi *Naive Bayes*	Objek : Pesan singkat pada jejaring sosial Twitter Input : kategori jenis pesan singkat	Metode : *Naive Bayes Classfication* Proses : - Pelatihan pengkategorian pesan singkat dari data latih RSS. - *Preprocessing* data yang akan diuji - Pengklasifikasian pesan singkat dengan naive bayes - Perolehan hasil *recall* dan *precision*	Hasil : pengkategorian pesan singkat berdasarkan jenis yang sudah ditentukan.
2	Klasifikasi Tweets pada Twitter Manggunakan Metode *Naive Bayes* dan *Query Expansion*	Objek : klasifikasi teks pada Twitter Input : Kumpulan berita *tweets*	Metode : *Naive Bayes* Classification dan *Query Expansion* berbasis apriori Proses :	Hasil : diketahuinya kategori jenis teks tertentu berdasarkan jenis yang sudah ditentukan.

No.	judul	Obyek	Metode	Hasil
	Berbasis Apriori	yang diperoleh dari detik.com dan kompas.com	- Perhitungan data latih yang dimasukkan ke dalam kamus. - Melakukan *preprocessing* terhadap data uji. - Penambahan kata pada kamus berdasarkan apriori. - Pengklasifikasian *naive bayes*. - Perolehan hasil klasifikasi.	
3	Perbandingan Metode *Cosine Smilarity* dengan Metode *Jaccard Similarty* pada Aplikasi Pencarian Terjemah Al-Qur'an dalam Bahasa Indonesia	Objek : Perbandingan metode dalam pencarian terjemah Al-Qur'an dalam bahasa Indonesia Input : dokumen terjemahan Al-Qur'an	Metode : *Cosine similarity* dan *Jaccard similarity*. Proses : - Dokumen terjemahan disimpan dalam *database*. - Melakukan proses perhitungan *preprocessing*. - Melakukan perhitungan di tiap metode. - Membandingkan hasil metode yang paling maksimal.	Hasil : persentase hasil perhitungan tiap metode. Hasil persentase yang paling tinggi merupakan metode yang paling bagus.

No.	judul	Obyek	Metode	Hasil
4	*Paraphrase identfication and semantic text similarity analisys in Arabic news tweet using lexical, syntactic, and semantic features*	Objek : identifikasi Frasa dan *semantic text similarity* pada berita Twitter dalam tulisan arab Input : berita Twitter dalam tulisan arab.	Metode : lexical, syntatic, dan *semantic feature* Proses : - Pengumpulan data dan anotasi. - Melakukan teks *preprocessing*. - Melakukan feature extraction. - Melakukan perbandingan pada metode yang sesuai. - Melakukan evaluasi terhadap masing-masing metode. - Menentukan penggunaan metode yang paling maksimal.	Hasil : hasil perbandingan tiap metode yang dilakukan.

2.2 Text Mining

Text mining merupakan suatu proses menggali informasi di mana seorang *user* berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis yang merupakan komponen-komponen dalam data *mining* yang salah satunya adalah ketegorisasi. Tujuan utama dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Sumber data yang digunakan pada text mining adalah kumpulan teks yang memiliki format yang tidak terstruktur atau minimal semi terstruktur (Mooney, 2006).

Manfaat dari text mining adalah untuk mempermudah pencarian dan membuat inovasi yang dapa membantu manusia untuk mengerti dan menggunakan informasi dari sebuah repository dokumen (Hand, 2010). Namun, dibalik manfaat yang disajikan oleh teks mining, juga terdapat kendala pada saat

pengimplmentasian teks mining, diantaranya teks mining menganalisis data primer dalam teks berdasarkan frekuensi kemunculan dan hubungan antar kata (Kayser, 2016).

Secara umum proses dalam text mining dilakukan dalam tiga tahapan yakni pemilihan data sebagai sumber informasi, kemudian dilakukan *preprocessing* dan analisis, dan yang terakhir adalah hasil dari proses yang menunjukkan interpretasi dari data yang diuji (Kayser, 2016).

2.3 Klasifikasi

Klasifikasi teks merupakan sebuah teknik dalam *text mining* yang bertujuan untuk menempatkan teks pada kategori yang sesuai dengan karakteristik dari teks tersebut dengan menggunakan aturan-aturan tertentu. Dengan adanya klasifikasi teks, maka dapat memberikan pandangan secara konseptual mengenai cara pengelompokan dokumen yang memiliki peranan penting terhadap dunia nyata (Sriram, 2010).

Tujuan dari pengategorian teks adalah untuk mengklasifikasikan dokumen ke dalam kategori-kategori tertentu. Tiap dokumen dapat diklasifikasikan dalam beberapa kategori, atau tidak sama sekali. Dengan menggunakan *machine learning*, pembelajaran dapat dilakukan dengan aturan kalsifikasi yang sudah ditentukan dan adanya data latih sebagai acuan pembelajaran sehingga dapat melakukan proses pengklasifikasian secara otomatis nantinya.

2.4 *Text preprocessing*

Tahapan *preprocessing* adalah tahapan untuk merepresentasikan dokumen dalam bentuk vektor, yang berarti harus memisahkan teks menjadi kata terpisah (Ramya & Pinakas, 2014). Sebelum melakukan analisis teks, *preprocessing* harusnya dilakukan terlebih dahulu untuk mengeliminasi kata-kata yang tidak diperlukan. Hal ini bertujuan agak maksud dari kalimat lebih jelas lagi (Nokhbeh Zaeem, 2016). Selain itu, teks juga akan lebih terstruktur dan bertransformasi ke dalam mesin pembaca untuk dilakukan proses yang lebih dalam lagi (Kayser, 2016). Tahapan *text prepocessing* bertujuan untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya. Terdapat tiga langkah dalam tahapan ini, yaitu: (Insan, 2013).

2.4.1 *Case Folding*

Tahapan case folding bertujuan untuk mengubah huruf ‘a’ sampai huruf ‘z’ dalam dokumen menjadi huruf kecil (Insan, 2013). Tidak semua dokumen konsisten dengan penggunaan huruf kapital, sehingga case folding diperlukan untuk mengonversi keseluruhan teks dalam dokumen menjadi huruf kecil (Firmansyah, 2016).

2.4.2 *Tokenizing*

Tahapan *tokenizing* merupakan tahapan untuk memecah kalimat menjadi kata atau biasa disebut *token*. Selain itu, tahap ini juga bertujuan untuk membuang beberapa karakter yang dianggap sebagai tanda baca (Insan, 2013). Dalam proses *tokenizing* juga dilakukan pemisahan dokumen menjadi kata dasar, menghapus awalah, penyisipan, akhiran dan duplikasi (Rizqi Lahitani, 2016).

2.4.3 *Filtering*

Tahapan *filtering* adalah tahap pengambilan kata-kata yang penting dari hasil *tokenzing* dengan menggunakan Algoritme *stoplist* (membuang kata yang tidak penting) dan *wordlist* (menyimpan kata yang penting) (Insan, 2013). Tujuan umum dari proses ini adalah untuk mendapatkan representasi dasar dari dokumen yang diujikan (Rizqi Lahitani, 2016).

2.4.4 *Stemming*

Stemming merupakan suatu proses yang mentransformasikan kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (*root word*) dengan menggunakan aturan-aturan tertentu. Sebagai contoh, kata bersama, kebersamaan, menyamai, akan diubah menjadi kata dasar "sama" (Insan, 2013). Pencarian kata dasar pada setiap bahasa berbeda-beda, untuk bahasa Indonesia sendiri *stemming* menghilangkan imbuhan di awal, imbuhan berada di tengah, imbuhan berada di akhir, ataupun imbuhan berada di awal dan di akhir dari kata. Berbeda dengan bahasa lain seperti bahasa inggris pencarian kata dasar hanya dengan menghilangkan imbuhan di akhir kata (Agusta, 2009).

2.5 *Naïve Bayes Classifier*

Naive Bayes adalah salah satu Algoritme pembelajaran induktif yang paling efektif dan efisien untuk *machine learning* dan data *mining*. Performa *naive bayes* yang kompetitif dalam proses klasifikasi walaupun menggunakan asumsi independen atribut (tidak ada kaitan antar atribut). Asumsi independen atribut ini pada data sebenarnya jarang terjadi, namun walaupun asumsi independen atribut tersebut dilanggar performa pengklasifikasian *naive bayes* cukup tinggi, hal in dibuktikan pada berbagai penelitian empiris (Guo, 2010).

Klasifikasi *naïve bayes* termasuk ke dalam Algoritme pembelajaran *bayes* yang dibangun oleh data pelatihan untuk memperkirakan probabilitas dari setiap kategori yang terdapat pada ciri dokumen yang diuji. Sistem akan dilatih dengan menggunakan *dataset* lengkap berupa pasangan nilai atribut dan nilai target. Kemudian sistem akan diberikan data baru (data *training* dan data *testing*) untuk selanjutnya diberi tugas untuk menebak nilai fungsi target dari data tersebut (Destuardi dan Surya, 2009).

Secara umum, proses klasifikasi dengan menggunakan metode *naive bayes* dapat dilihat pada persamaan 2.1 (Desturardi dan Surya, 2009).

$$P(c_j \mid w_i) = \frac{P(c_j) \times P(w_i \mid c_j)}{P(w_i)} \quad (2.1)$$

Keterangan:

$P(C_j \mid W_i)$: Posterior merupakan peluang kategori j ketika terdapat kemunculan kata i.

$P(W_i \mid C_j)$: *Conditional probabilty* merupakan Peluang sebuah kata i masuk ke dalam kategori j.

$P(C_j)$: *Prior* merupakan peluang kemunculan sebuah kategori j.

$P(W_i)$: Peluang kemunculan sebuah kata.

i : indeks kata yang dimulai dari 1 hingga ke-*k*

j : Indeks kategori yang dimulai dari 1 hingga kategori ke-*n*

Peluang kemunculan sebuah kata sebenarnya bisa dihilangkan pada proses perhitungan klasifikasi karena peluang kemunculan kata tidak akan berpengaruh pada perbandingan hasil klasifikasi dari setiap kategori. Sehingga, proses pada klasifikasi dapat disederhanakan dengan persamaan 2.2 (Destuardi dan Surya, 2009).

$$P(c_j \mid w_i) = P(c_j) \times P(w_i \mid c_j) \quad (2.2)$$

Untuk menghitung prior atau peluang kemunculan suatu kategori pada semua dokuen dapat dilakukan dengan menggunakan persamaan 2.3 (Destuardi dan Surya, 2009).

$$P(c) = \frac{N_{cj}}{N} \quad (2.3)$$

Keterangan:

N_{cj} : Dokumen yang masuk kategori c_j.

N : Jumlah keseluruhan dokumen latih yang digunakan.

Pada umumnya data uji memiliki banyak kata yang diproses mulai indeks ke-1 hingga ke-*k*, dalam hal ini *conditional probability* kata w_i pada kategori c_j dilakukan perkalian dari i=1 sampai i=k sehingga untuk mengetahui nilai posterior dapat dihitung dengan menggunakan persamaan 2.4.

$$P(c_j \mid w_i) = P(c_j) \times P(w_1 \mid c_j) \times P(w_2 \mid c_j) \times P(w_3 \mid c_j) \ldots\ldots \times P(w_k \mid c_j) \quad (2.4)$$

2.5.1 *Gaussian Naïve Bayes*

Distribusi *Gaussian* biasanya dipilih untuk merepresentasikan probabilitas bersyarat dari fitur kontinyu pada sebuah kelas P(Xi|Y), sedangkan distribusi *Gaussian* memiliki karakteristik dengan dua parameter: *mean* dan varian.

2.5.2 *Bernoulli Naïve Bayes*

Tipe klasifikasi ini juga sering digunakan untuk pengklasifikasian *short-text*. Pada pengklasifikasian ini menggunakan *binary* (0 dan 1) dalam pembobotan tiap term, berbeda dengan perhitungan term frekuensi yang melakukan pembobotan pada setiap term.

2.5.3 *Multinomial Naïve Bayes*

Metode *multinomial Naïve Bayes* merupakan Algoritme yang *naïve bayes* sebab mengasumsikan independensi di antara kemunculan kata-kata dalam dokumen, tanpa memperhitungkan urutan kata dan informasi konteks kalimat atau dokumen secara umum. Selain itu metode tersebut memperhitungkan jumlah kemunculan kata dalam dokumen (Destuardi dan Surya, 2009).

Pada metode *Multinomial Naïve Bayes*, perhitungan peluang sebuah kata *i* masuk ke dalam kategori *j* dapat dilakukan dengan menggunakan persamaan 2.4 (Destuardi dan Surya, 2009).

$$P(w_1 \mid c_j) = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|} \quad (2.4)$$

Keterangan:

$count(w_i, c) + 1$ merupakan jumlah kemunculan dari kata uji yang muncul dalam kategori c_j ditambah dengan 1 untuk menghindari nilai *zero* atau nol. Sedangkan $count(w, c)$ merupakan jumlah kemunculan seluruh kata yang ada pada kategori c_j. $|V|$ merupakan jumlah seluruh kata unik yang ada pada seluruh kategori.

2.6 *Feature Expansion*

Dalam klasifikasi *short-text* masalah yang sering muncul ialah kata yang muncul dalam dokumen uji tidak ada pada dokumen latih. Tentunya hal tersebut bisa memunculkan permasalahan dalam klasifikasi, yaitu proses klasifikasi tidak berjalan dengan baik karena ada kata yang tidak bisa terdeteksi untuk masuk dalam suatu kategori yang sudah didefinisikan.

Untuk mengatasi hal tersebut dapat digunakan metode *feature expansion*. *Feature expansion* adalah teknik atau proses memformulasikan kembali kata dengan menambahkan kata baru yang sudah disimpan sebelumnya dengan menggunakan teknik tertentu (Tang, 2017).

2.7 *Cosine Similarity*

Cosine similarity merupakan metode yang digunakan untuk menghitung tingkat kesamaan *(Similarity*) antar dua objek (Sugiyamta, 2015). Pembobotan *term* atau kata akan digunakan dalam perhitungan *cosine similarity* dari dokumen latih dan dokumen uji. Sebelum dilakukan pengelompokan dokumen, hendaknya dilakukan pencarian kedekatan antar kata terlebih dahulu, dengan begitu akan lebih mudah dalam pengelompokan (Rizqi Lahitani, 2016). Di mana dalam konteks

ini objek yang dimaksud direpresentasikan ke dalam bentuk vektor-vektor yang nantinya akan dihitung kedekatan jarak antar vektor yang dimaksud. Perhitungan jarak yang dimaksud telah ditetapkan seperti pada persamaan 2.5.

$$Similarity = \cos(A,B) = \frac{\vec{A} \cdot \vec{B}}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (2.5)$$

Keterangan

$\vec{A} dan \vec{B}$ adalah komponen panjang vektor dari A dan B. $\sum_{i=1} A_i B_i$ merupakan jumlah perkalian dari vektor A dan vektor B. |A||B| merupakan nilai mutlak dari tiap-tiap vektor yang di representasikan oleh $\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}$ yang mana nilai tersebut diperoleh dari hasil kuadrat tiap-tiap vektor.

2.8 Evaluasi

Proses evaluasi merupakan kegiatan yang membandingkan antara hasil implementasi dengan kriteria standar yang telah ditetapkan untuk melihat keberhasilannya. Dari hasil evaluasi nantinya akan tersedia informasi mengenai sejauh mana suatu kegiatan tertentu telah tercapai sehingga bisa diketahui bila terdapat selisih antara standar yang telah ditetapkan dengan hasil yang bisa dicapai. Evaluasi dilakukan dengan menghitung nilai persentase akurasi dari sistem dengan membandingkan hasil klasifikasi sistem dengan kriteria yang telah ditetapkan seperti pada persamaan 2.6.

$$akurasi = \frac{Jumlah data Uji Benar}{Banyak Data Uji} \quad (2.6)$$

BAB 3 METODOLOGI

Metodologi penelitian dirancang menganalisis klasifikasi. Alur penelitian yang akan digunakan secara umum meliputi tipe penelitian, strategi penelitian.

3.1 Tipe penelitian

Pada penelitian in menggunakan tipe penelitian nonimplementatif. Penelitian ini berfokus pada investigasi atau penyidikan terhadap fenomena atau situasi tertentu, atau analisis terhadap hubungan antar fenomena yang sedang dikaji untuk kemudian menghasilkan sebuah investigasi atau hasil ilmiah sebagai produk utamanya. Metode yang digunakan untuk menghasilkan produk utama bisa berasal dari survei, eksperimen, studi kasus penelitian tindakan, wawancara, kuisioner, observasi, dan sebagainya.

3.2 Strategi penelitian

Penelitian ini termasuk penelitian yang menggunakan eksperimen. Penelitian eksperimen merupakan penelitian yang melakukan investigasi hubungan sebab akibat dengan menggunakan uji coba yang dikontrol oleh peneliti dengan melibatkan pengembangan dan evaluasi serta pada umumnya dilakukan di laboratorium.

3.3 Partisipan penelitian

Partisipan yang terlibat dalam penelitian adalah mahasiswa atau khalayak umum yang tentunya mengikuti berita atau informasi pada jejaring sosial Twitter.

3.4 Lokasi penelitian

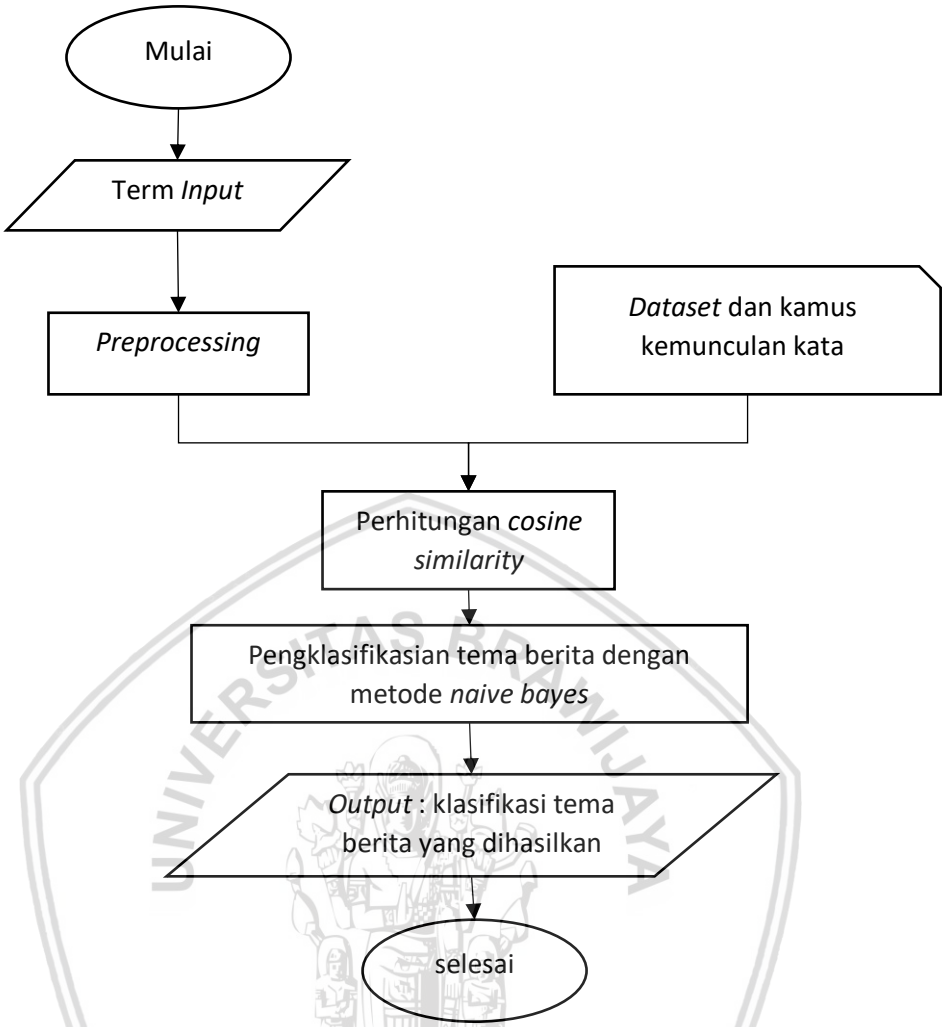
Penelitian dilakukan di Laboratorium Komputasi Cerdas Fakultas Ilmu Komputer Universitas Brawijaya.

3.5 Teknik pengumpulan data

Tahapan ini bertujuan untuk mengetahui cara atau proses dalam pengumpulan data yang digunakan dalam penelitian. Data yang digunakan dalam penelitian terdiri dari data kamus, data latih, dan data uji. Sumber data berasal dari *tweets* dari Twitter dari akun Kompas dan Detik yang disesuaikan dengan kategori yang di labelkan.

3.6 Implementasi algoritme

Perancangan algoritme merupakan penjabaran dari proses dalam pembangunan sebuah sistem. Sistem yang akan dibangun adalah detail dari sistem klasifikasi. Tujuan dari sistem ini adalah memberi jenis kategori dalam teks dokumen tweets berdasar isinya yang diuji cobakan berdasar penambahan kata baru. Gambaran umum dari proses perancangan algoritme ditunjukkan pada Gambar 3.1.



Gambar 3.1 Perancangan sistem

3.7 Teknik analisis data

Pada tahapan ini dilakukan analisis data atau biasa disebut dengan tahapan pengujian data setelah dilakukan pembuatan program. Pengujian dilakukan untuk mengetahui apakah sistem yang dibangun sudah sesuai dengan kebutuhan dan perancangan. Pengujian dilakukan untuk mengetahui hasil dari kerja sistem apakah sudah sesuai dengan hipotetis awal yang sudah dikerjakan. Skenario pengujian yang akan dilakukan adalah sebagai berikut:

1. Skenario pengujian membandingkan hasil yang diperoleh pada saat proses klasifikasi dilakukan dengan menambahkan *feature expansion* dan tidak menggunakan *feature expansion*.
2. Pengujian dengan menambahkan *feature expansion* dilakukan dengan menggunakan variasi *threshold* untuk mengambil kata yang akan di ekspan.
   Setelah proses pengujian selesai, akan dilakukan analisis terhadap hasil pengujian untuk memperoleh akurasi dari penggunaan metode dalam pengolahan data. Tahap analisis ini untuk melihat nilai akurasi yang dihasilkan

oleh *Naïve Bayes* dan melihat pengaruh *Feature Expansion* terhadap akurasi pemberian kategori.

3.8 Jadwal Penelitian

Jadwal penelitian akan dilaksanakan dalam waktu terhitung dari bulan Februari sampai dengan bulan Juni. Berikut ini adalah jadwal penelitian akan ditunjukkan pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1 Jadwal Penelitian

No	Uraian	Februari				Maret				April				Mei				Juni		
		Minggu ke-																		
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3
1	Studi Kepustakaan																			
2	Pengumpulan Data																			
3	Implementasi Algoritme																			
4	Pengujian dan Analisis																			
5	Kesimpulan dan Saran																			

BAB 4 PERANCANGAN SISTEM

Pada bab ini menjelaskan mengenai perancangan sistem, menjelaskan Algoritme dan contoh perhitungan, dan menjelaskan perancangan antarmuka tampilan pada sistem.

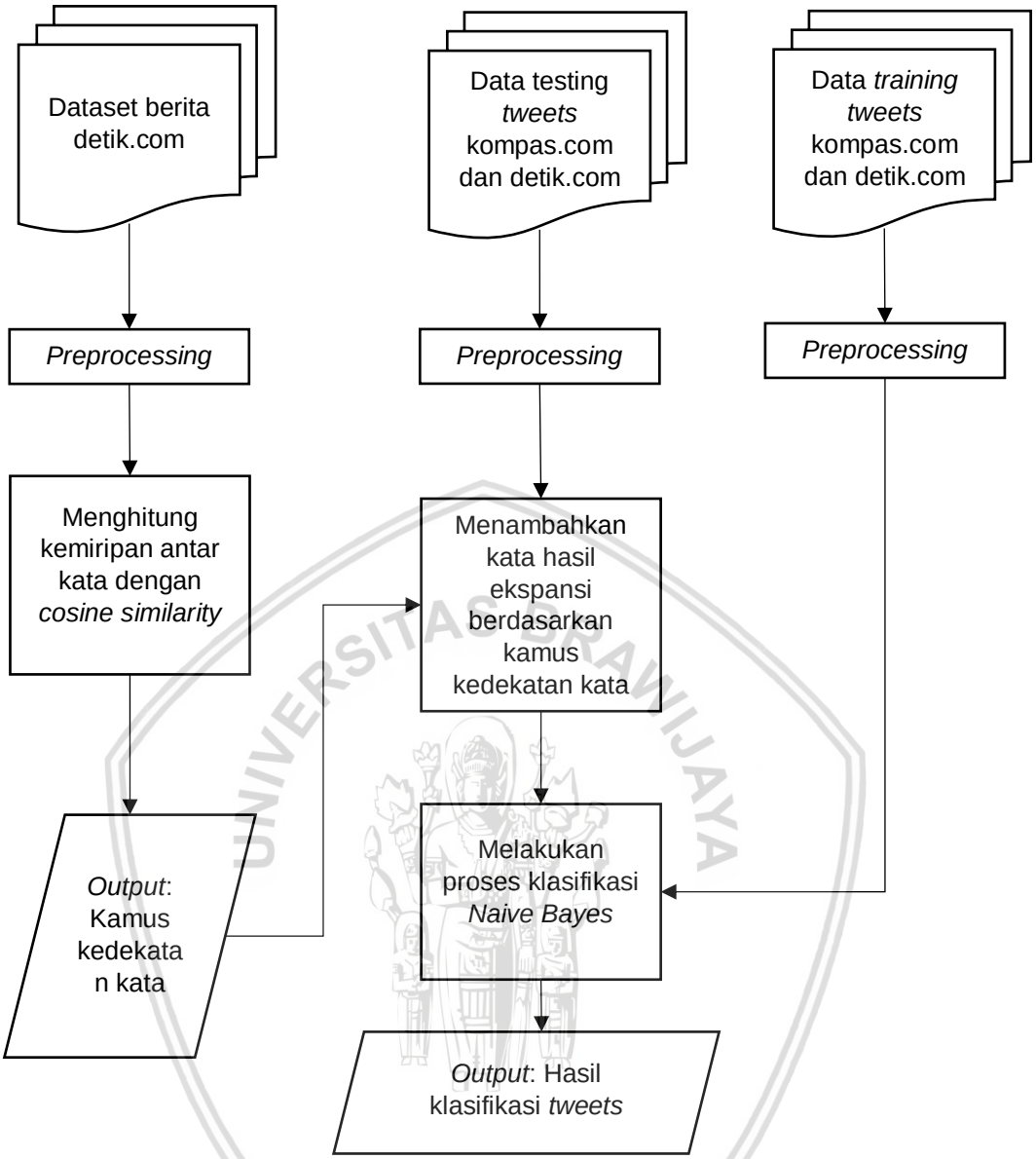
4.1 Analisis Kebutuhan Sistem

Analisis kebutuhan berguna untuk mendapatkan semua keperluan yang dibutuhkan dalam pembuatan sistem. Kebutuhan-kebutuhan tersebut antara lain:

1. Kebutuhan perangkat keras (Hardware):
   - Laptop.
   - RAM 6,00 GB.
   - Monitor 14".
2. Kebutuhan perangkat lunak (*Software*):
   - Sistem operasi Microsoft Windows 10.
   - Aplikasi *Pycharm.*
   - Sistem menggunakan bahasa pemrograman *Python.*
3. Kebutuhan data, meliputi:
   - Berita Detik.com

4.2 Alur Kerja Sistem Secara Umum

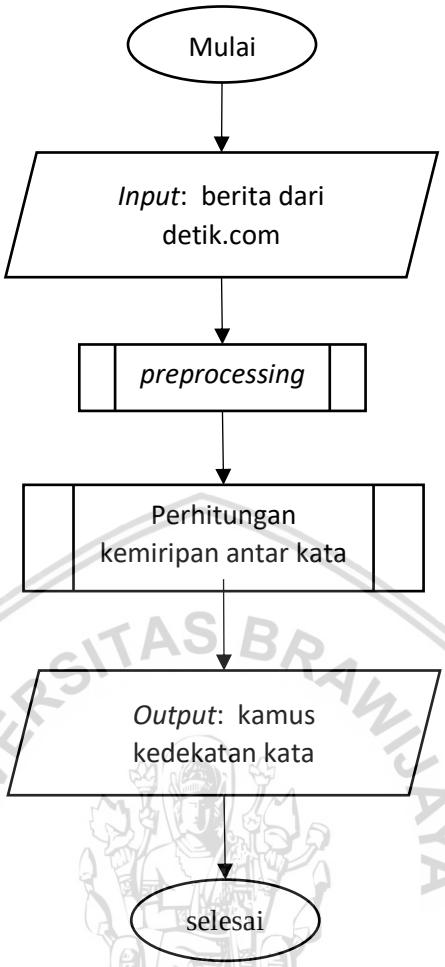
Cara kerja sistem dijelaskan oleh gambar 4.1. secara umum, prinsip kerja sistem akan menghasilkan klasifikasi berita Twitter sebagai hasil akhir. Proses klasifikasi dilakukan dengan menggunakan metode *Naive Bayes*. Sebelum dilakukan proses klasifikasi, sebelumnya sistem akan melakukan ekspansi kata yang didapatkan dengan Algoritme *cosine similarity* yang tersimpan dalam kamus kedekatan kata. Untuk pengambilan kata untuk digunakan sebagai ekspansi kata merupakan kata yang memiliki jarak terdekat dengan kata yang bersangkutan semisal dalam penentuan kata yang di ambil memiliki jarak antara 0,9-0,99. Dengan begitu akan mempersempit daftar kata yang dijadikan sebagai ekspansi kata yang tujuannya untuk mendapatkan hasil yang optimal. Untuk dataset, data latih, dan data uji sebelumnya sudah dilakukan *preprocessing* untuk menghilangkan kata yang dianggap tidak diperlukan oleh sistem. Proses ekspansi kata dilakukan pada data uji atau data testing saja, sebab fokus penelitian pada klasifikasi berita Twiiter berdasarkan hasil klasifikasi yang sudah didapat pada data latih. Setelah data testing mendapatkan daftar kata yang dijadikan sebagai ekspansi kata, dilakukanlah klasifikasi dengan data training. Untuk data set berupa data berita yang diambil dari portal berita Detik.com, serta data latih dan data uji menggunakan *tweets* dari Kompas.com dan Detik.com.



Gambar 4.1 Diagram blok cara kerja sistem

4.2.1 Pembuatan Kamus Kedekatan Kata

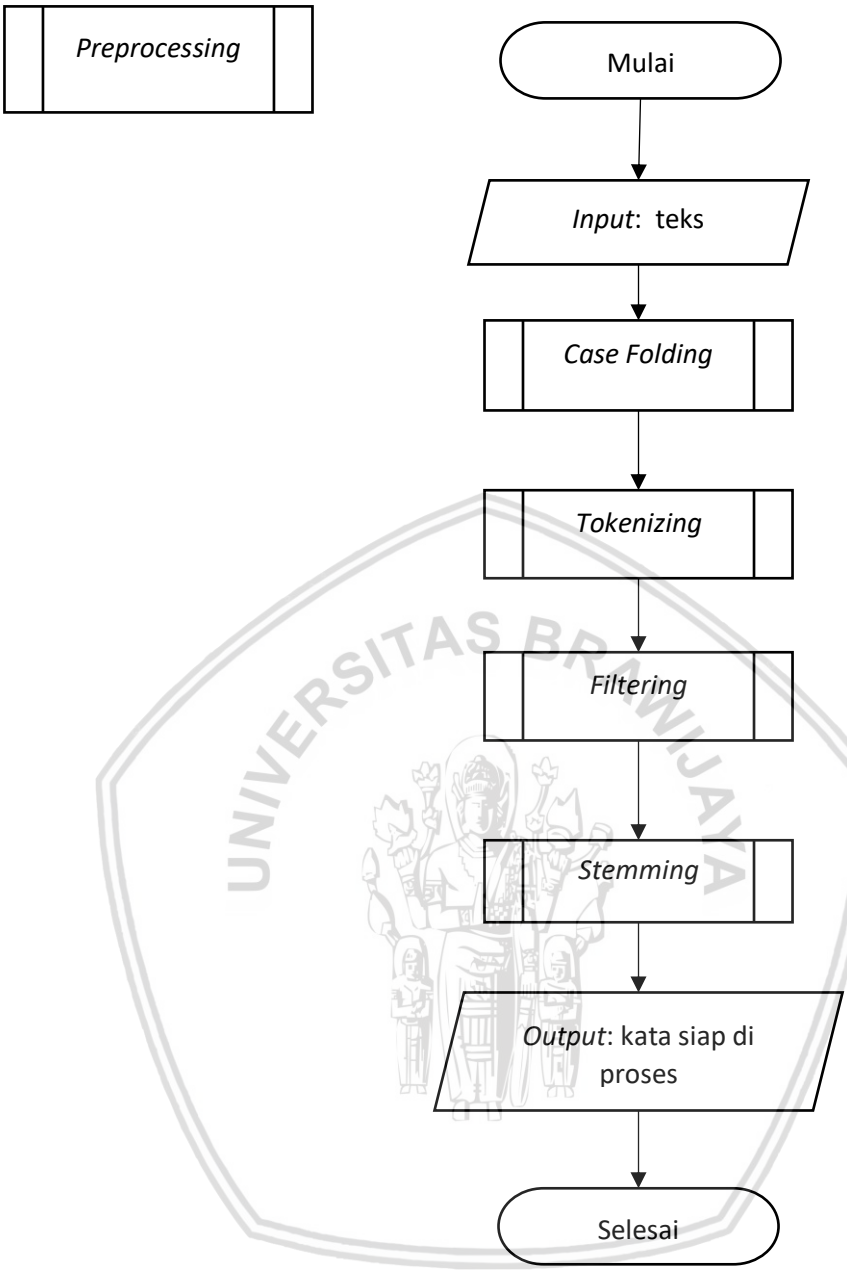
Tahap pertama yang dilakukan adalah proses pembuatan kamus kedekatan kata. Kamus kedekatan kata tersebut merupakan kumpulan kata unik dari dokumen berita detik.com. kumpulan kata in diharapkan adalah kata-kata yang akan muncul dalam data testing nantinya. Pada proses pembuatan kamus kedekatan kata ini hal yang dilakukan adalah melakukan *preprocessing* dan mencari kedekatan kata dengan *cosine similarity*. Untuk alur dari tahap ini dijelaskan pada gambar 4.2.



Gambar 4.2 Diagram alir sistem pembuatan kamus kedekatan

4.2.1.1 Preprocessing

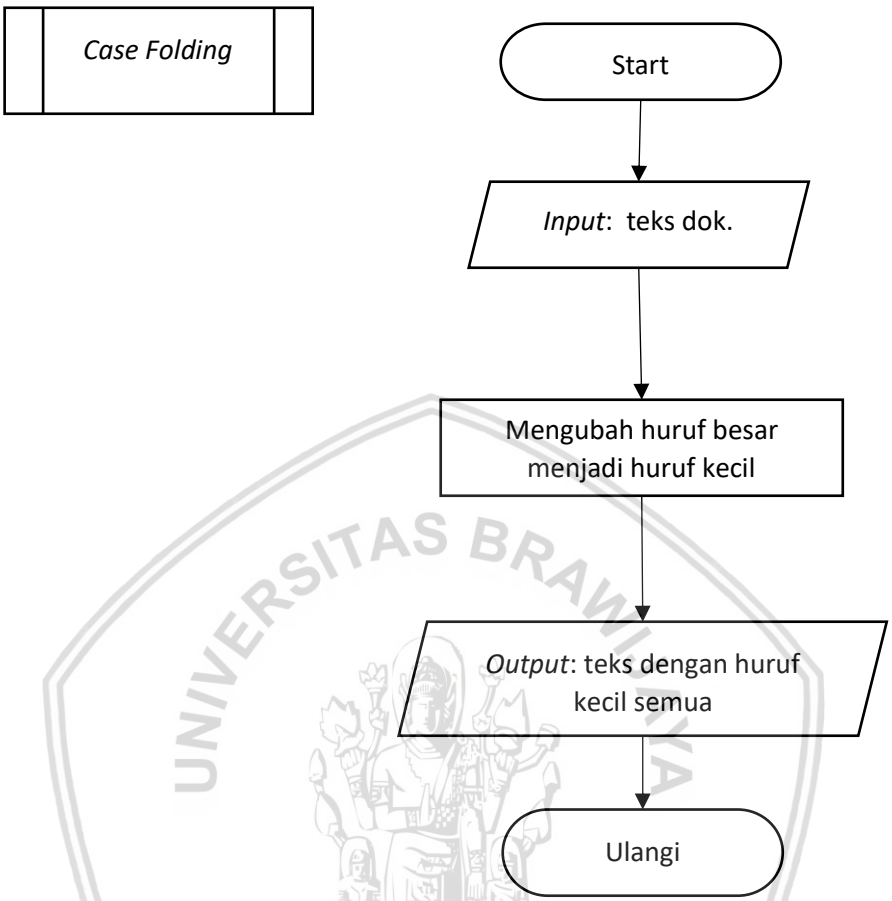
Tahap *preprocessing* dilakukan untuk mendapatkan teks yang siap untuk dilakukan proses analisis. Hasil dari *preprocessing* ini berupa teks yang bersih dari kata yang tidak penting atau *noise* lainnya sehingga hasil akhir berupa term atau kata penting yang siap untuk di proses. Yang dilakukan pada *preprocessing* antara lain *case folding*, *tokenizing*, *filtering*, dan *stemming*. Dimana input berupa dokumen dan outputnya berupa term atau kata. Untuk alur diagram sistem akan dijelaskan pada gambar 4.3



Gambar 4.3 Diagram alir *preprocessing*

4.2.1.2 Case Folding

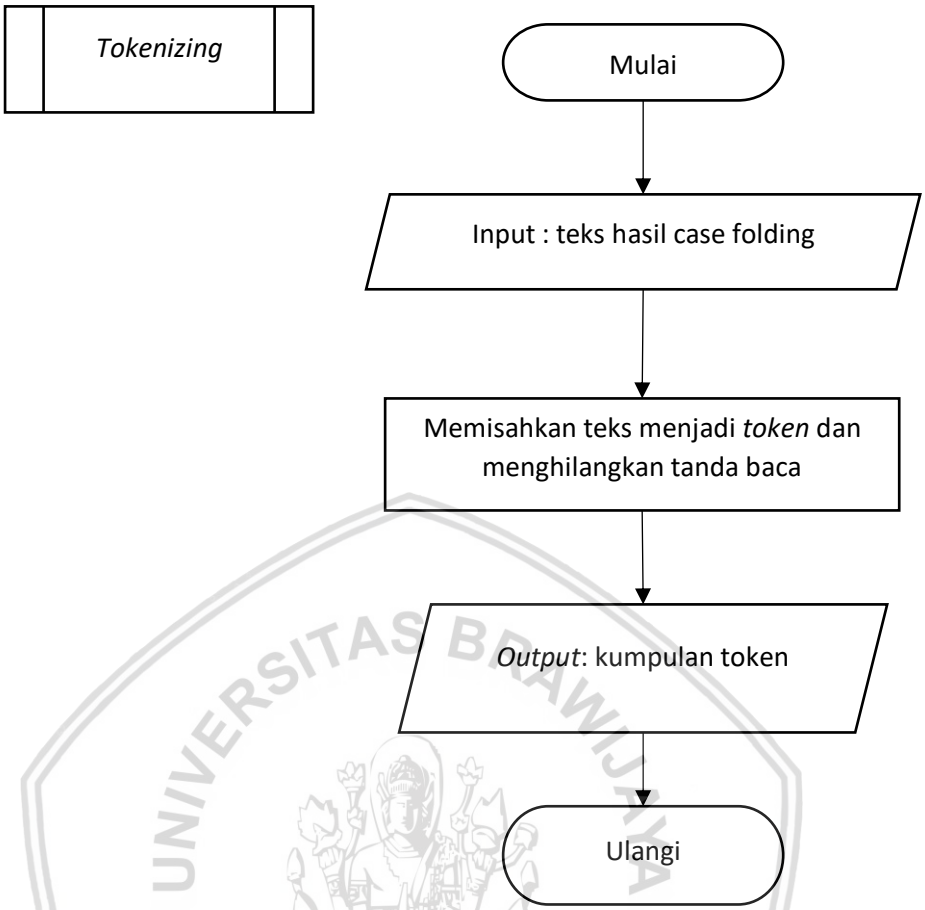
Dalam *preprocessing* terdapat sub-proses *case folding*. Dimana proses ini bertugas untuk mengubah huruf besar menjadi huruf kecil. Tujuannya untuk menyamakan huruf sebagai contoh kata 'menteri' dan 'Menteri' dalam konteks manusia kata tersebut artinya sama. Namun jika kata tersebut tidak disamakan huruf kecil semua, mesin akan menganggap kedua kata tersebut adalah kata yang berbeda. Dari sini proses *case folding* diperlukan. Penjelasan diagram alir proses *case folding* dapat dilihat pada gambar 4.4



Gambar 4.4 Diagram alir *case folding*

4.2.1.3 Tokenizing

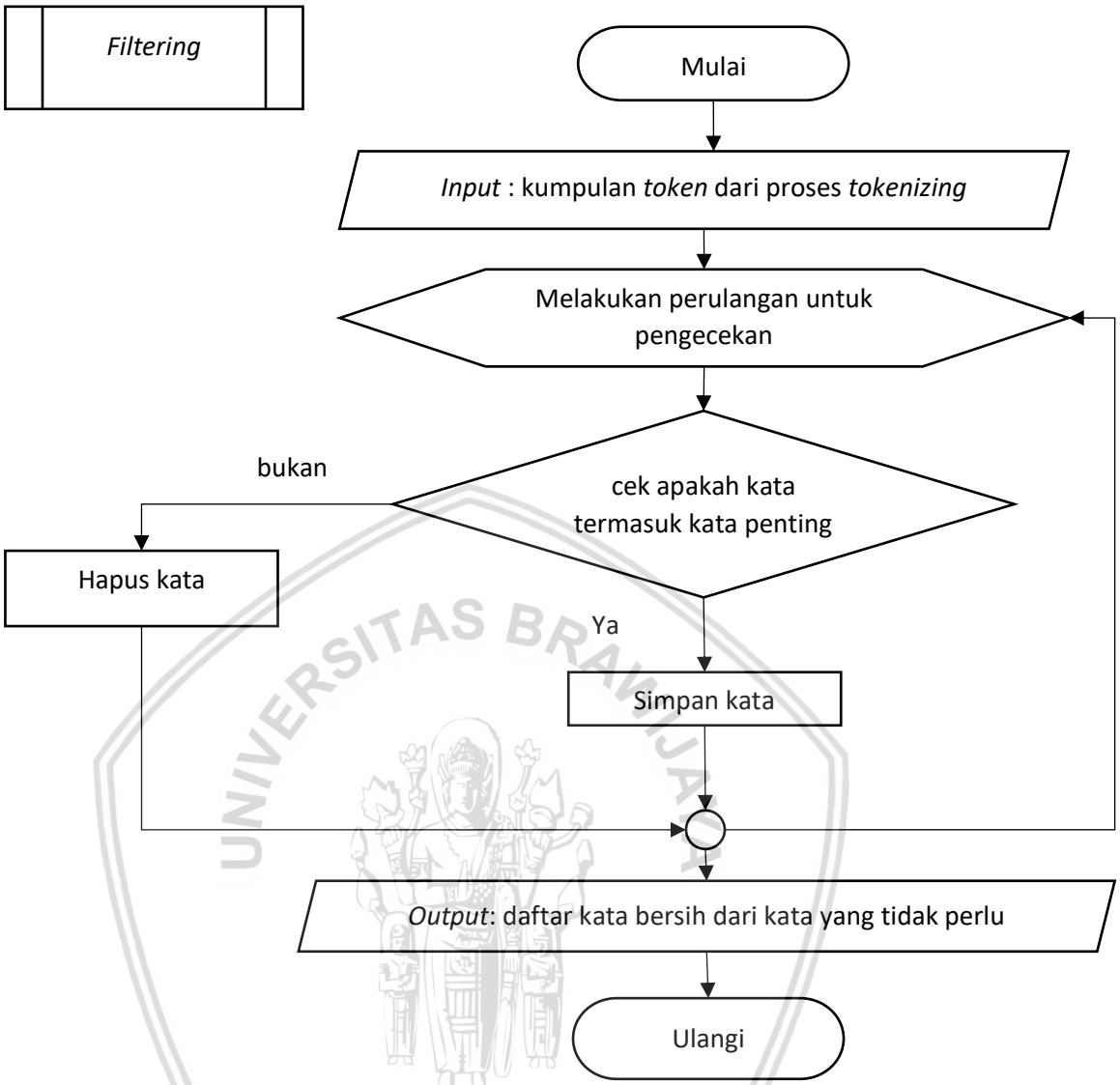
Setelah proses *case folding* terdapat sub-proses *tokenizing*. Dimana proses ini bertugas untuk memisahkan dokumen menjadi token-token. Dan menghilangkan karakter yang dianggap sebagai tanda baca. Dari proses *tokenizing* ini juga bisa didapatkan hasil berupa banyaknya kemunculan suatu kata dalam beberapa dokumen. Dari proses ini dokumen bersih dari tanda baca. Untuk diagram alir dari proses *tokenizing* dapat dilihat pada gambar 4.5



Gambar 4.5Diagram Alir *tokenizing*

4.2.1.4 Filtering

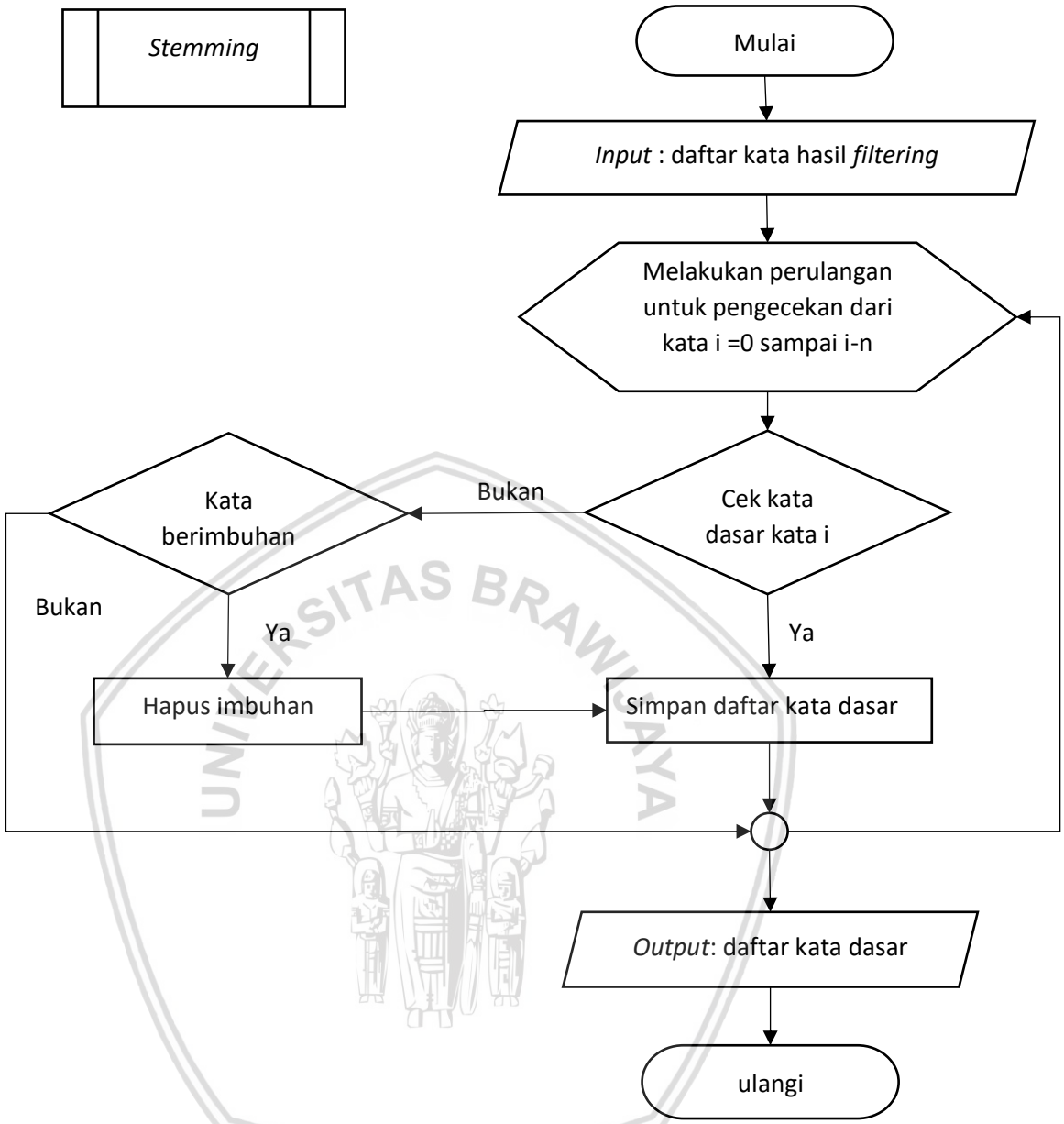
Setelah proses *tokenizing* terdapat sub-proses *filtering*. Dimana pada tahap ini token-token akan dicari mana kata yang dianggap penting. Pada tahap ini dilakukan proses penghilangan kata seperti kata 'yang', 'di', 'dan'. Kata tersebut merupakan kata penghubung untuk menunjukkan maksud dar suatu kalimat, namun tidak memiliki arti yang cukup spesifik. Untuk itu mesin diajarkan hanya mencari kata yang dianggap penting saja untuk disimpan. Dengan begitu daftar kata yang disimpan oleh mesin lebih efisien dalam pembelajaran. Untuk alur kerja diagram alir *filtering* dijelaskan pada gambar 4.6



Gambar 4.6 Diagram Alir *filtering*

4.2.1.5 Stemming

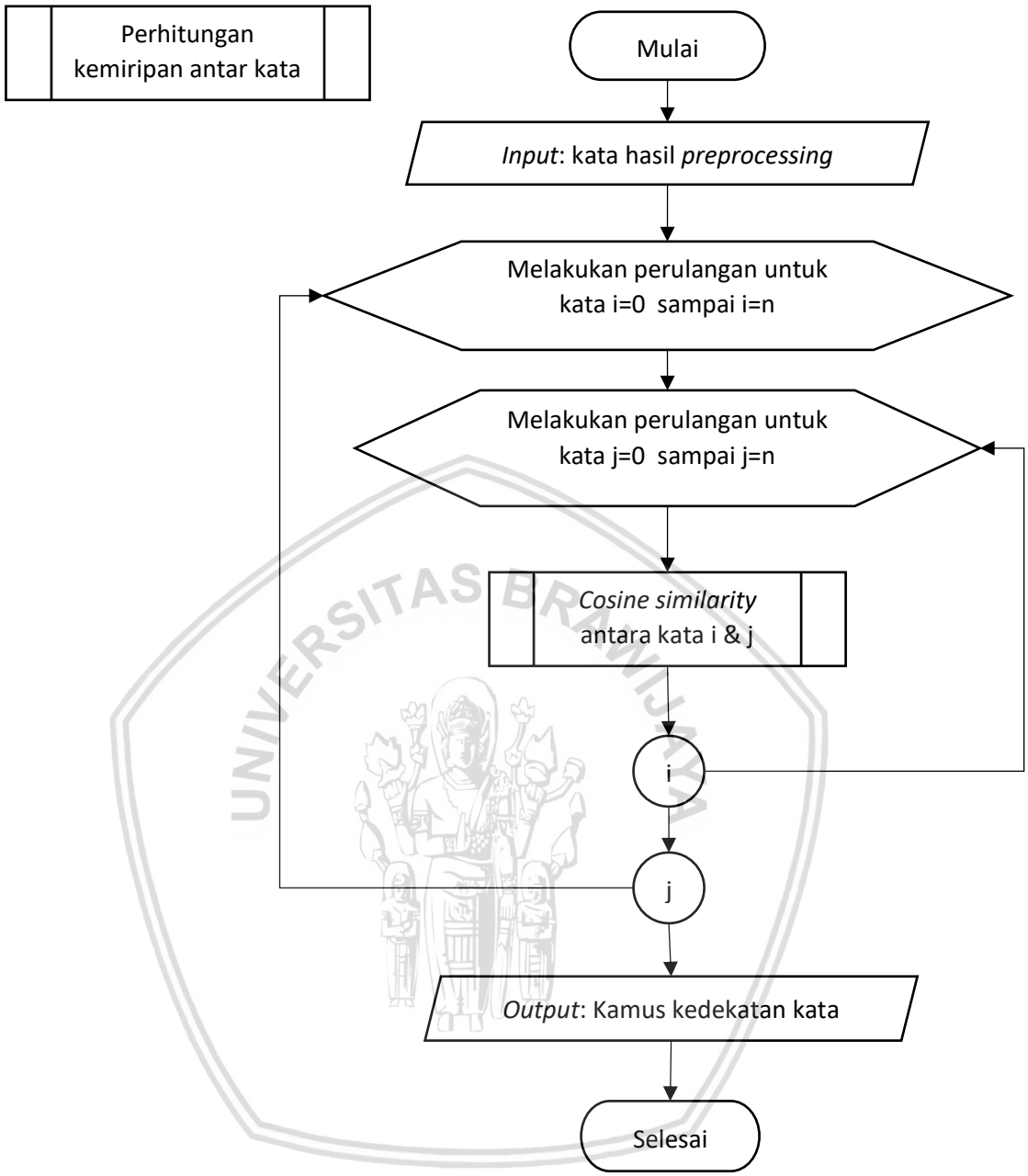
Setelah proses *filtering* terdapat sub-proses *stemming*. Dimana pada proses ini daftar kata penting di olah lagi untuk mengetahui daftar kata tersebut merupakan kata dasar atau bukan. Seperti pada kata 'belajar', 'mengajar', 'diajar' memiliki kata dasar yang sama yakni 'ajar'. Tujuan dari proses *stemming* ini adalah untuk mencari kata dasar dari tiap *token* yang nantinya lebih efisien dalam pembuatan kamus kedekatan ini di mana kata dasar yang sama akan dijadikan satu sehingga kemunculan kata akan semakin banyak lagi. Untuk diagram alir proses *stemming* dapat dilihat pada gambar 4.7.



Gambar 4.7 Diagram Alir *Stemming*

4.2.1.6 Perhitungan Kemiripan Kata

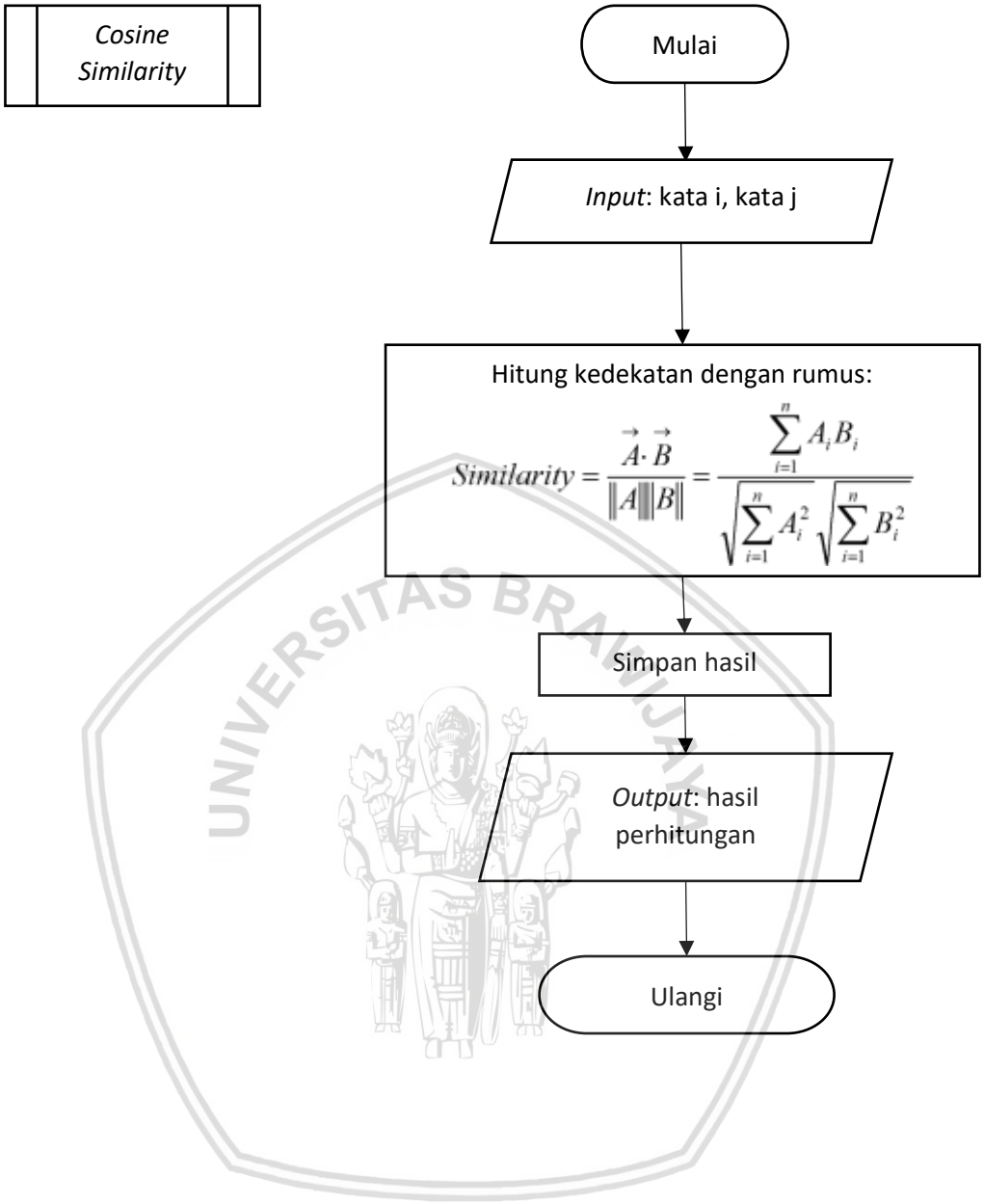
Setelah proses *preprocessing* selesai dikerjakan dan telah didapatkan hasilnya. Langkah selanjutnya adalah proses perhitungan kemiripan kata di mana token-token yang sudah didapatkan setelah proses *stemming* dimasukkan ke dalam sebuah ruang vektor untuk dicari kedekatan katanya dengan menggunakan metode *cosine similarity*. Hasil yang diperoleh pada tahapan ini adalah kumpulan kata yang memiliki atribut berupa nilai kedekatan dengan kata lainnya di mana kata tersebut yang nantinya akan dijadikan sebagai ekspansi kata dalam proses klasifikasi. Untuk diagram alir proses perhitungan kemiripan kata dapat dilihat pada gambar 4.8



Gambar 4.8 Diagram Alir perhitungan kemiripan kata

4.2.1.7 Cosine Similarity

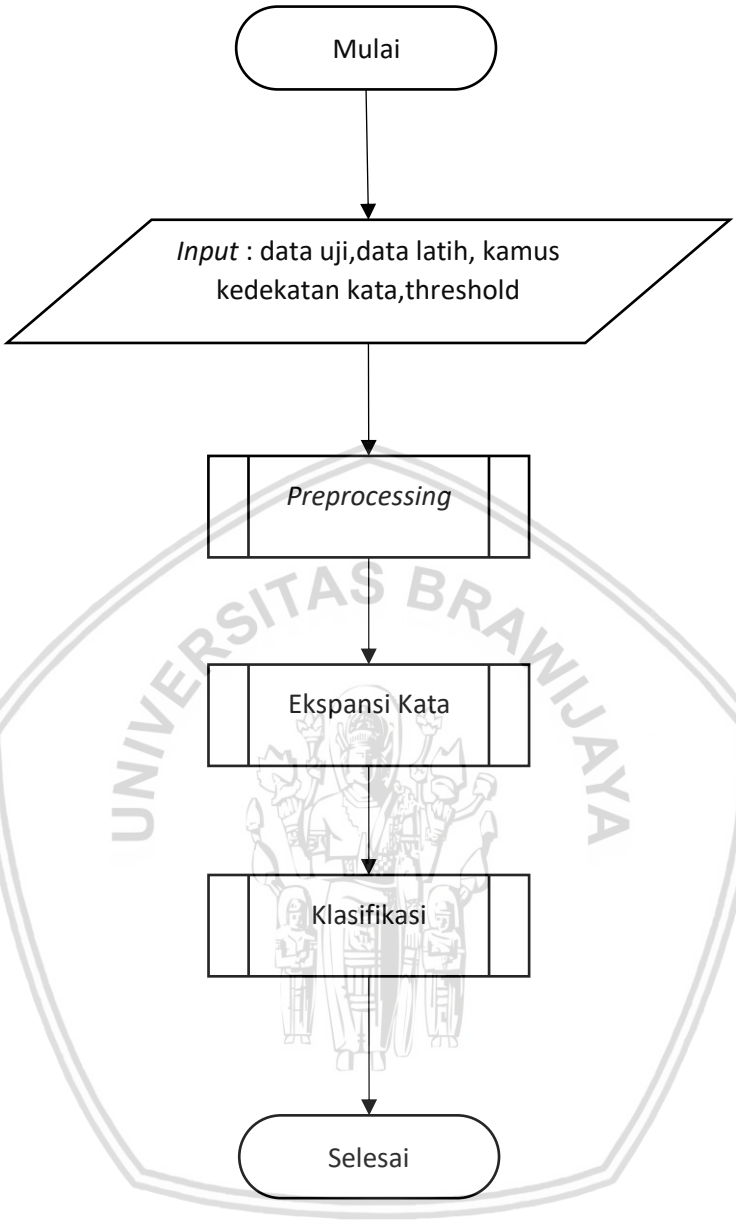
Pada tahapan ini *token* atau kata yang sudah di *preprocessing* dihitung kedekatan antar katanya dengan metode *cosine similarity*. Pencarian kedekatan kata dihitung dengan menyimpan semua kata ke dalam sebuah vektor dan melakukan perhitungan antar kata (*Vector Space Model*). Dari perhitungan ini didapatlah jarak antar kata yang mana akan diperoleh nilai 0 sampai dengan 1, semakin mendekati angka 1 maka jarak kata tersebut semakin dekat. Untuk diagram alirnya dapat dilihat pada gambar 4.9



Gambar 4.9 Diagram Alir proses *cosine similarity*

4.2.2 Klasifikasi

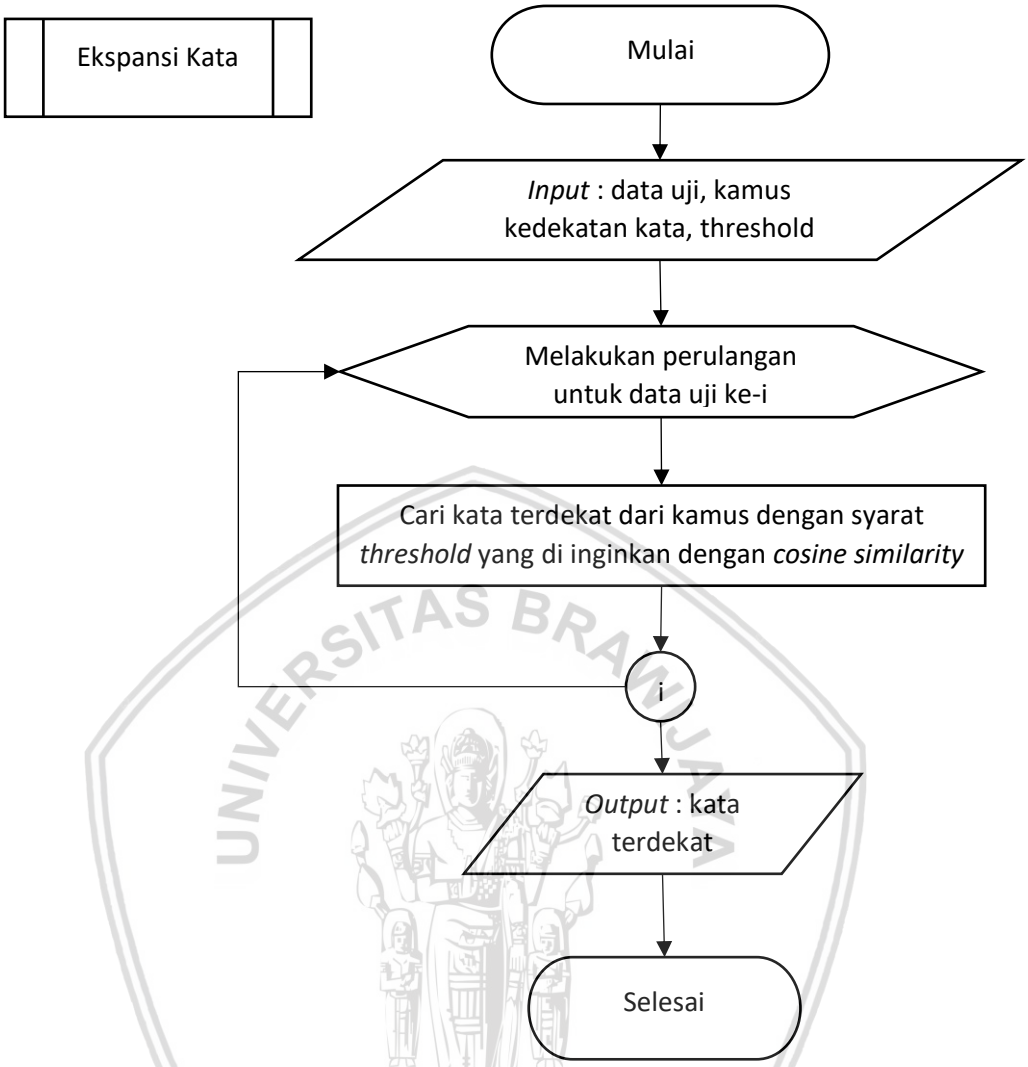
Setelah kamus kedekatan selesai dibuat, langkah selanjutnya adalah proses klasifikasi. Proses ini diawali dengan masukan berupa data latih, data uji data kamus dan *threshold* yang diinginkan. Setiap data latih dan data uji dilakukan *preprocessing* seperti pada proses pembuatan kamus. Setelah kata dari data uji selesai dilakukan *preprocessing*, pencarian ekspansi kata ditambahkan ke dalam data uji kemudian dilakukan klasifikasi untuk memperoleh hasil kategori. Untuk diagram alir proses klasifikasi secara umum dapat dilihat pada gambar 4.10



Gambar 4.10 Diagram alir proses klasifikasi

4.2.2.1 Ekspansi kata

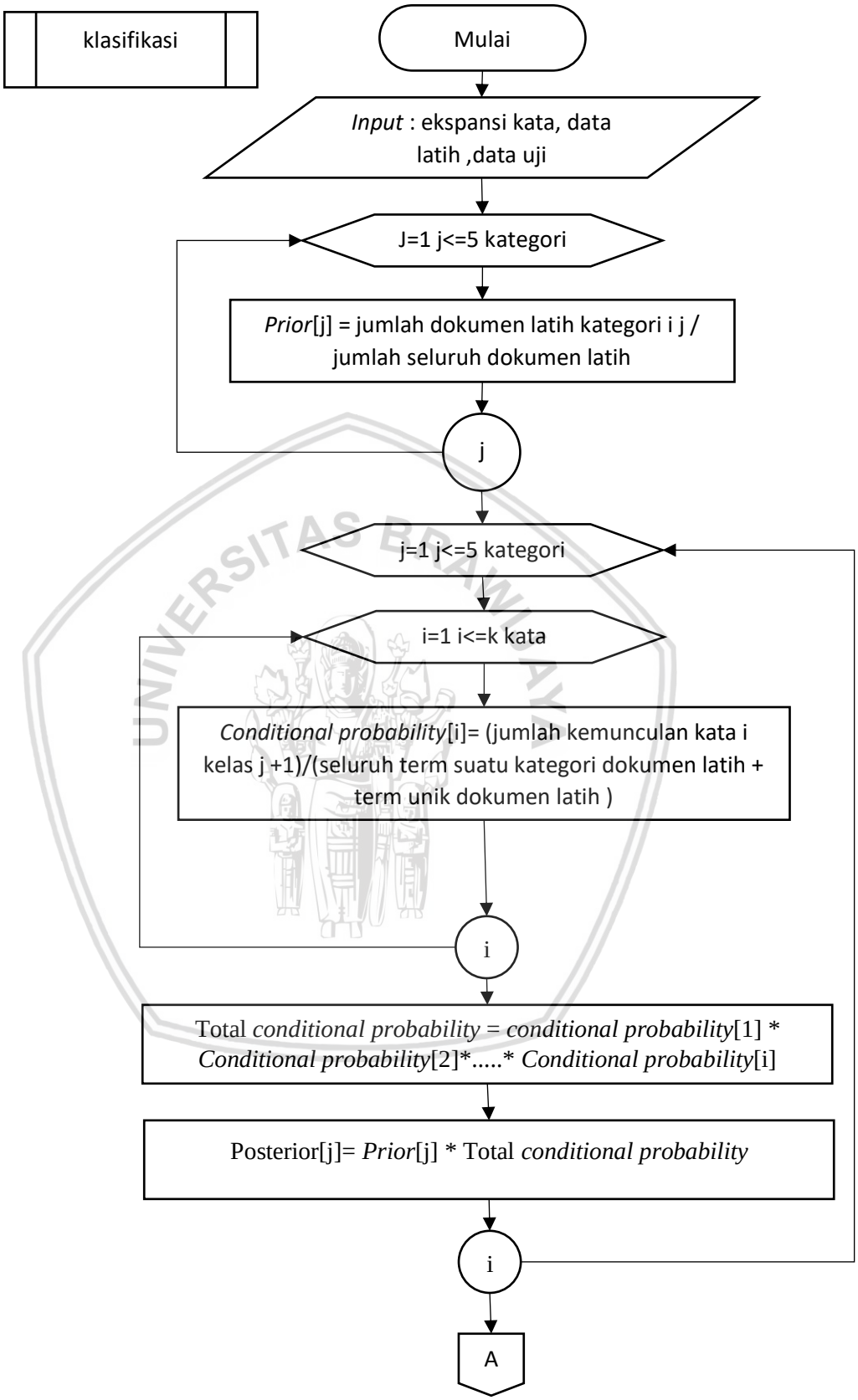
Tahapan ini merupakan cara mencari kata yang akan dimasukkan kedalam data uji sebagai ekspansi kata. Dimana pada tahapan ini masukan berupa data uji dan kamus yang sudah dibuat sebelumnya. Pencarian kedekatan kata dihitung dengan cosine similarity dan syarat threshold yang diinginkan. Setelah muncul hasil dari perhitungan, akan didapat kata yang akan diekspan kedalam data uji yang sesuai. Untuk diagram alir proses dapat dilihat pada gambar 4.11

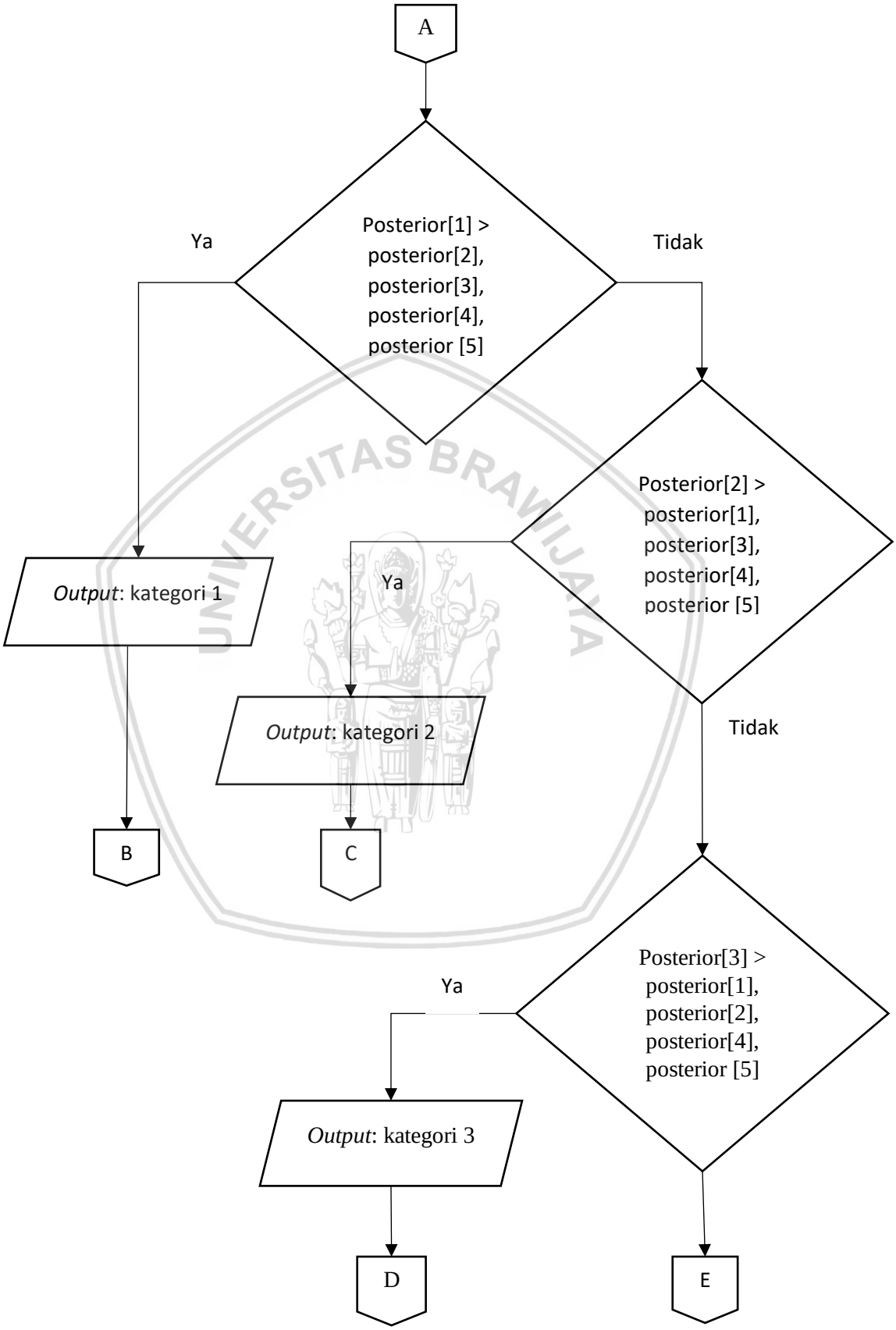


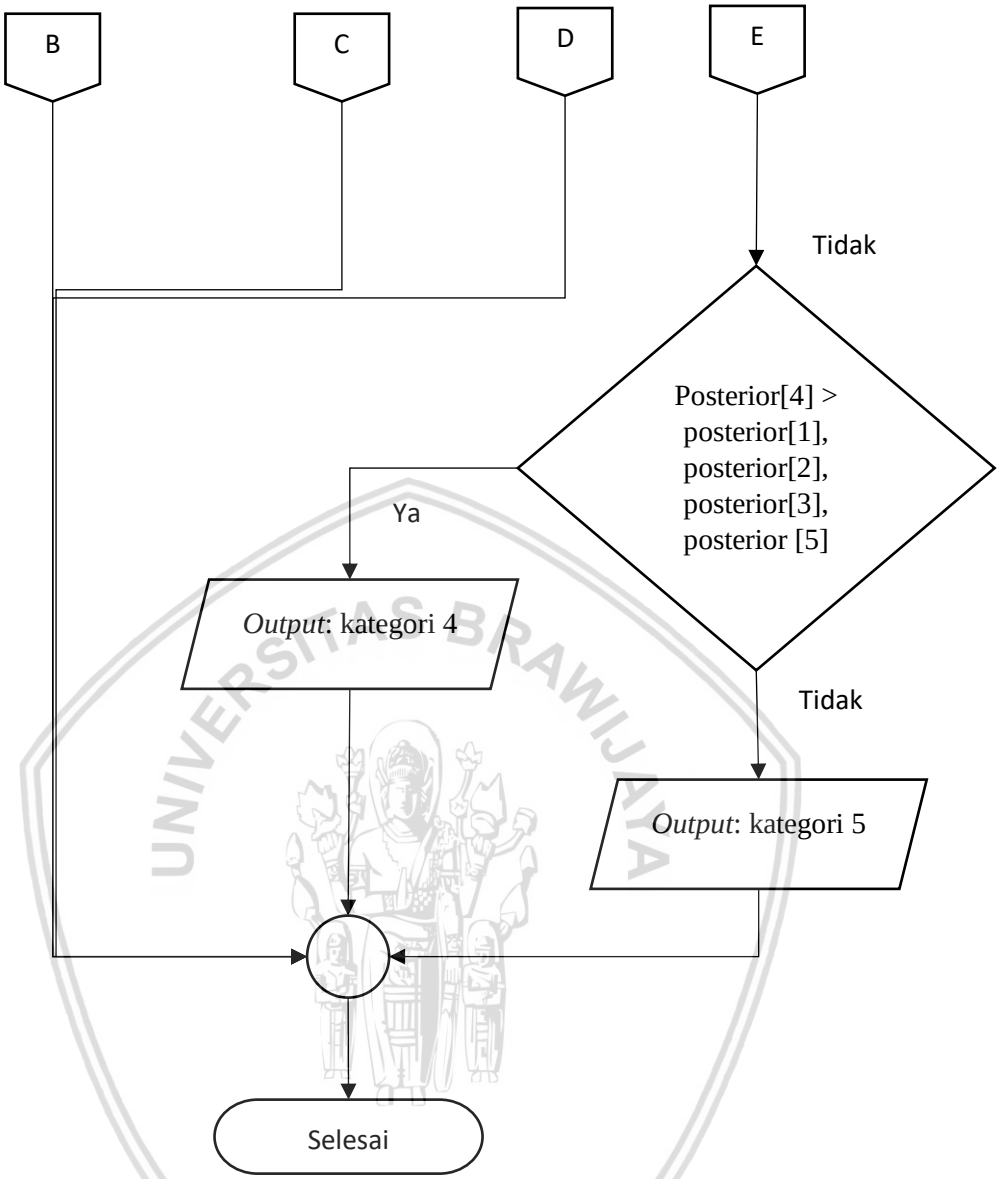
Gambar 4.11 Diagram Alir proses ekspansi

4.2.2.2 Klasifikasi Naive Bayes

Tahapan berikut merupakan tahapan akhir, yakni proses pengklasifikasian data uji. Setelah didapat data uji yang sudah ditambahkan dengan kata yang diperoleh dari kamus, dicari kategorinya dengan menggunakan rumus *naive bayes*. Hasil perhitungan tertinggi merupakan hasil akhir dari kategori yang didapat. Untuk diagram alir dari proses ini dapat dilihat pada gambar 4.12


klasifikasi
Mulai
Input : ekspansi kata, data latih ,data uji
J=1 j<=5 kategori
Prior[j] = jumlah dokumen latih kategori i j / jumlah seluruh dokumen latih
j
j=1 j<=5 kategori
i=1 i<=k kata
Conditional probability[i]= (jumlah kemunculan kata i kelas j +1)/(seluruh term suatu kategori dokumen latih + term unik dokumen latih)
i
Total *conditional probability* = *conditional probability*[1] * *Conditional probability*[2]*.....* *Conditional probability*[i]
Posterior[j]= *Prior*[j] * Total *conditional probability*
i
A


A
Posterior[1] > posterior[2], posterior[3], posterior[4], posterior [5]
Ya
Tidak
Output: kategori 1
B
Posterior[2] > posterior[1], posterior[3], posterior[4], posterior [5]
Ya
Output: kategori 2
C
Tidak
Posterior[3] > posterior[1], posterior[2], posterior[4], posterior [5]
Ya
Output: kategori 3
D
E



Gambar 4.12 Diagram alir naive bayes

4.3 Manualisasi

4.3.1 Manualisasi Klasifikasi Tanpa Feature expansion

Perhitungan manualisasi berikut akan menunjukkan perhitungan pengklasifikasian sebuah *tweets* yang dilakukan tanpa menggunakan ekspansi kata. Proses perhitungan klasifikasi menggunakan data latih pada tabel 4.1

Tabel 4.1 Contoh Data Latih

Dok.	Isi	Kategori
1	Pesan Jokowi ke Menteri: APBN itu Terbatas belanjakan dengan wise	Ekonomi
2	Melihat potret kehidupan John Lennon	Enteritaiment
3	Ini mengapa alasan Wanita sering Alami Kerontokan rambut	Kesehatan
4	Dortmund berharap 'Monster' Reus bebas cedera	Olahraga
5	Membedah Kotak 'Si Lucu' Robot BB-8	Teknologi

Data latih yang digunakan pada manualisasi, dari masing-masing kategori diambil satu contoh dari *tweets.* Untuk data yang akan diuji hasil klasifikasinya, ditampilkan pada tabel 4.2

Tabel 4.2 Contoh Data Uji

Dok.	Isi	Kategori
1	Kereta Cepat Jakarta-Bandung Tuai Kritik, Pemerintah Beri Penjelasan	?

Proses utama yang dilakukan untuk mengklasifikasi pada data uji dengan menghitung nila *prior* pada masing-masing kategori yang digunakan pada data latih. Rumus untuk menghitung nilai *prior* menggunakan persamaan 2.3. berikut contoh perhitungan untuk mencari nilai *prior* dan hasil perhitungan *prior* ditunjukkan pada tabel 4.3.

Contoh perhitungan *prior*:

$$P(ekonomi) = \frac{1}{5} = 0{,}2$$

Tabel 4.3 Hasil Perhitungan Prior

No.	Kategori	prior	
		Perhitungan	Hasil Akhir
1	Ekonomi	1/5	0.2

No.	Kategori	prior	
		Perhitungan	Hasil Akhir
2	Entertaiment	1/5	0.2
3	Kesehatan	1/5	0.2
4	Olahraga	1/5	0.2
5	Teknologi	1/5	0.2

Setelah nilai *prior* didapatkan untuk masing-masing kategori, kemudian menghitung nilai *likelihood*. Untuk nilai dari kata unik tiap dokumen dan total seluruh kata unik dapat dilihat pada tabel 4.4.

Tabel 4.4 Jumlah Kata Dan Kata Unik Dalam Dokumen

Dok	**Isi**	**Kategori**	**Kata Unik Tiap Dokumen**	**|V|**
1	Pesan Jokowi ke Menteri APBN itu terbatas belanjakan dengan Wise	Ekonomi	7	7
2	Melihat Potret kehidupan John Lennon	*Entertaiment*	5	5
3	Ini alasan mengaoa wanta sering alami kerontokan rambut	Kesehatan	8	8
4	Dortmund berharap monster reus bebas cedera	Olahraga	6	6
5	Membedah kotak si lucu bb-b	Teknologi	6	6
Total kata unik seluruh dokumen				**32**

Likelihood merupakan perhitungan untuk mencari probabilitas kemunculan suatu kata pada kategori yang sudah tersimpan pada data latih. Cara perhitungan dilakukan dengan menjumlahkan kemunculan kata dalam dokumen *training* dan ditambah dengan 1 kemudian dibagi dengan jumlah kata dalam dokumen *training* yang ditambahkan dengan total kata unik dalam seluruh dokumen *training* seperti pada persamaan 2..contoh perhitungan *likelihood* seperti di bawah ini dan hasil dari perhitungan dapat di lihat pada tabel 4.5 :

Contoh perhitungan likelihood:

$$P(kereta|ekonomi) = \frac{1+0}{7+32} = \frac{1}{39} = 0{,}026$$

Tabel 4.5 Hasil Perhitungan Likelihood

	Ekonomi	*Entertaiment*	Kesehatan	Olahraga	Teknologi
Kereta	0,026	0,027	0,025	0,026	0,026
Cepat	0,026	0,027	0,025	0,026	0,026
Jakarta	0,027	0,027	0,025	0,026	0,026
bandung	0,026	0,027	0,025	0,026	0,026
Tuai	0,026	0,027	0,025	0,026	0,026
Kritik	0,026	0,027	0,025	0,026	0,026
pemerintah	0,026	0,027	0,025	0,026	0,026
beri	0,026	0,027	0,025	0,026	0,026
penjelasan	0,026	0,027	0,025	0,026	0,026

setelah nilai *likelihood* didapatkan, kemudian menghitung nilai posterior untuk setiap kategori. Nilai tersebut diperoleh dari perkalian prior dan *likelihood*-nya. Untuk hasil dari perhitungan posterior dapat dilihat pada tabel 4.6 :

$$Posterior\ (ekonomi)$$
$$= 0{,}2 \times 0{,}26 \times 0{,}27 \times 0{,}26 \times 0{,}26 \times 0{,}26 \times 0{,}26 \times 0{,}26 \times 0{,}26 \times 0{,}26$$
$$= 1{,}00998\text{E} - 15$$

Tabel 4.6 Hasil Perhitungan Posterior

Kategori	Nilai
Ekonomi	1,00998E-15
Entertaiment	1,53891E-15
Kesehatan	7,62939E-16
Olahraga	1,21054E-15
Teknologi	1,21054E-15

Setelah menentukan nilai *posterior* pada masing-masing kategori, proses klasifikasi sudah bisa dilakukan. Untuk menentukan hasil dari proses klasifikasi dilakukan dengan mencari nilai *posterior* yang tertinggi dari semua nilai yang didapatkan. Dari hasil perhitungan pada tabel 4.6, nilai *posterior* tertinggi adalah kategori *entertaiment*. Sehingga, dokumen uji termasuk ke dalam kategori *entertaiment.*

Dari perhitungan yang telah dilakukan dengan menggunakan klasifikasi tanpa query expansion menghasilkan nilai yang salah, karena hasil klasifikasi seharusnya menunjukkan hasil perhitungan ke dalam kategori ekonomi sesuai kategori aslinya.

4.3.2 Manualisasi Feature expansion

Perhitungan manualisasi *feature expansion* menunjukkan perhitungan yang digunakan dalam pembuatan kamus kedekatan kata dan menentukan kata yang memenuhi kriteria untuk dijadikan sebagai ekspansi kata. Perhitungan *feature expansion* dengan menghitung kemunculan kata dalam dokumen yang disimpan dalam sebuah vektor. Dari kumpulan vektor yang telah didapat dihitung kedekatan jaraknya dengan menggunakan metode *cosine similarity*. Untuk daftar kata ekspansi yang akan digunakan dapat dilihat pada tabel 4.7

Tabel 4.7 Id Kata Ekspansi

Id	Kata
1	Jakarta
2	Menteri
3	Laut
4	Ikan
5	Susi
6	Pudjiastuti
7	Tenggelam
8	Kapal
9	Nelayan
10	Asing
11	Tangkap
12	Bukti
13	Curi
14	Air
15	Indonesia
16	Aparat
17	Tegak
18	Hukum
19	Ledak

Untuk perhitungan kemunculan kada dalam tiap dokumen dapat dilihat pada tabel 4.8. di mana pada perhitungan ini di cari kemunculan sejumlah kata yang sudah di sebutkan terhadap sepuluh dokumen sebagai sampel. Cara pengisian tabel : Pada baris ke-2 kolom ke-2 berisi nilai 10, nilai tersebut didapat dari id kata ke-1 yaitu Jakarta bertemu dengan id kata ke-1 sebanyak 10 kali. Sama halnya dengan baris ke-2 kolom ke-3 berisi 7, berarti kata pada id kata ke-1 dan ke-2 bertemu sebanyak 7 kali dalam 10 dokumen, begitu seterusnya. Untuk hasil lengkapnya dapat dilihat pada tabel 4.8

Tabel 4.8 Kemunculan Kata Pada Tiap Dokumen

Id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	10	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	7	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	2	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	2	5	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	2	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	1	2	1	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	3	1	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1	9	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	9	1	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	1	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Setelah dilakukan pencarian kemunculan kata dalam tiap dokumen, hal yang dilakukan adalah mencari kedekatan antar kata dengan menggunakan *cosine similarity* seperti persamaan 2.5. Untuk hasil perhitungan dapat dilihat pada tabel 4.9. dan contoh perhitungan sebagai berikut:

Contoh perhitungan mencari nilai *cosine similarity*

$$\text{Cosine Similarity}(\text{jakarta, menteri}) = \frac{(10\times 7)+(7\times 10)+(0\times 2)+(0\times 0)+(0\times 0).....+(0\times 0)}{\sqrt{10^2+7^2+0^2+0^2....+0^2}\times\sqrt{7^2+10^2+0^2+0^2....+0^2}}$$

$$= \frac{140}{\sqrt{149}\times\sqrt{149}}$$

$$= \frac{140}{149}$$

$$= 0.9396 = 0{,}9$$

Tabel 4.9 Hasil Kedekatan Cosine Similarity

Id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	1,0	0,9	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
2	0,9	1,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
3	0,0	0,5	1,0	0,7	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
4	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
5	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
6	0,0	0,0	0,0	0,0	0,7	1,0	0,5	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
7	0,0	0,0	0,0	0,0	0,0	0,5	1,0	0,4	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
8	0,0	0,0	0,0	0,0	0,0	0,0	0,4	1,0	0,6	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
9	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,6	1,0	0,6	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
10	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,6	1,0	0,6	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
11	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,6	1,0	0,7	0,0	0,0	0,0	0,0	0,0	0,0	0,0
12	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0	0,0	0,0	0,0
13	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0	0,0	0,0
14	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0	0,0
15	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0	0,0
16	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0	0,0
17	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,7	0,0
18	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	1,0	0,8
19	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,8	1,0

Dari kumpulan kata yang sudah dicari kedekatan katanya, kata yang memenuhi ekspansi kata dalam data uji adalah 'menteri'. Pemilihan ekspansi kata ini bisa dibatasi dengan banyaknya kata yang memiliki jarak mendekati 1. Untuk hasil perhitungan *likelihood* bisa dilihat pada tabel 4.10. dan cara menghitungnya sama seperti sebelumnya.

Tabel 4.10 Hasil Llikelihood Hasil Dengan Ekspansi Kata

	Ekonomi	*Entertaiment*	Kesehatan	Olahraga	Teknologi
Kereta	0,026	0,027	0,025	0,026	0,026

	Ekonomi	*Entertaiment*	Kesehatan	Olahraga	Teknologi
Cepat	0,026	0,027	0,025	0,026	0,026
Jakarta	0,027	0,027	0,025	0,026	0,026
Bandung	0,026	0,027	0,025	0,026	0,026
Tuai	0,026	0,027	0,025	0,026	0,026
Kritik	0,026	0,027	0,025	0,026	0,026
Pemerintah	0,026	0,027	0,025	0,026	0,026
Beri	0,026	0,027	0,025	0,026	0,026
Penjelasan	0,026	0,027	0,025	0,026	0,026
Menteri	0,051	0,027	0,025	0,026	0,026

Sama seperti proses sebelumnya, setelah didapat nilai *likelihood*-nya kemudian dihitung nilai posteriornya. Hasil perhitungan hasil posterior dengan penambahan ekspansi kata dapat dilihat pada tabel 4.11

Tabel 4.11 Hasil Posterior

kategori	Nilai
ekonomi	5,17937E-17
entertaiment	4,15923E-17
kesehatan	1,90735E-17
olahraga	3,18562E-17
teknologi	3,18562E-17

Dari nilai yang telah didapat, ternyata nilai tertinggi terdapat pada kategori ekonomi. Dan hasil tersebut sudah benar dengan data yang sebenarnya. Hal ini menunjukkan jika penambahan *feature expansion* mampu menambahkan akurasi dalam mengategorikan berita.

4.4 Perancangan Skenario Pengujian

Perancangan pengujian yang akan dilakukan adalah dengan melakukan 2 pengujian, yaitu pengujian dengan tidak menggunakan ekspansi kata atau *feature expansion* dan pengujian dengan menggunakan *feature expansion* dengan variasi nilai *threshold* tertentu.

4.4.1 Pengujian Tanpa Penggunaan *Feature Expansion* dan Penggunaan *Feature Expansion*

Pengujian ini dilakukan untuk mendapatkan nilai akurasi ketika tidak menggunakan *feature expansion* atau ekspansi kata, dengan tujuan untuk membandingkan hasil akhir ketika tidak menggunakan ekspansi kata dan ketika

menggunakan ekspansi kata. Untuk perancangan pengujian dapat dilihat pada tabel 4.12

Tabel 4.12 Perancangan Pengujian Penggunaan Feature Expansion Dan Tanpa Penggunaan Feature Expansion

Akurasi Tanpa Menggunakan Ekspansi Kata	Dengan Menggunakan Ekspansi Kata	
	Nilai *Threshold*	Akurasi
**%%	0,1	**%
	0,2	**%
	0,3	**%
	0,4	**%
	0,5	**%
	0,6	**%
	0,7	**%
	0,8	**%
	0,9	**%
	1,0	**%

4.5 Penarikan Kesimpulan

Penarikan kesimpulan dilakukan pada saat semua proses telah selesai dijalankan. Kesimpulan ini diambil dari rumusan masalah dan hasil pengujian yang telah dilakukan. Dari kesimpulan ini juga didukung saran dan masukan untuk pengembangan selanjutnya.

BAB 5 IMPLEMENTASI

Pada bab ini menjelaskan implementasi sistem berdasarkan metodologi dan perancangan yang telah dijelaskan pada bab sebelumnya.

5.1 Batasan Implementasi

Batasan implementasi yang dimaksud adalah batasan proses yang dapat dilakukan oleh sistem berdasarkan perancangan yang telah diuraikan. Tujuan dari batasan implementasi adalah untuk membuat sistem sesuai dengan ruang lingkup yang sudah ditentukan secara jelas dan tidak keluar dari tujuan utama pengembangan sistem. Batasan implementasi ini meliputi:

1. Klasifikasi *tweets* pada Twitter menggunakan metode *Naive Bayes* dan *feature expansion* yang menggunakan metode *cosine similarity* yang dirancang dan dijalankan pada aplikasi *dekstop* dengan bahasa pemrograman *python*.
2. Metode dalam penyelesaian masalah menggunakan *Multinomial Naive bayes* dan *feature expansion*.
3. Data uji dan data latih merupakan *tweets* yang berasal dari kompas dan detik.
4. Keluaran yang dihasilkan berupa klasifikasi pada kategori teknologi, olahraga, *entertaiment*, kesehatan, dan ekonomi.

5.2 Implementasi Algoritme

Implementasi Algoritme secara umum terdiri dari tiga Algoritme utama yaitu, *text presprocessing*, *feature expansion*, dan klasifikasi.

5.2.1 Implementasi Algoritme *Text Preprocessing*

Algoritme *Text Preprocessing* merupakan tahapan awal yang dilakukan oleh sistem sebelum melakukan tahapan *feature expansion* dan klasifikasi. *Text preprocessing* terdiri dari tahapan *case folding*, *filtering*, *steming*, dan *tokenizing*. Pada program yang dikembangkan proses ini disimpan dalam satu *class* yang dinamakan *class* proses. Proses ini dapat dilihat pada Kode program 5.1.

Kode Program *5.1* Implementasi Algoritme *Text Preprocessing*

```
1   def proses(self,liskata):
2
3       # penghilangan karakter tidak penting
4       punctuations = '''!()-
5   []{};:'"\,<>./?@#$%^&*_~0123456789'''
6       for kata in liskata:
7           no_punct = ""
8           for char in kata[1]:
9               if char not in punctuations:
10                  no_punct = no_punct + char
11              elif char in punctuations:
12                  no_punct = no_punct + ' '
13          kata[1] = no_punct
14
```

```
15      # cek stemmer
16      stemmer = factory.create_stemmer()
17      for bb in liskata:
18          dokumen = re.split(r'\s', bb[1])
19          gabung = ''
20          for a in range(len(dokumen)):
21              hasil = stemmer.stem(str(dokumen[a]))
22              gabung += ' ' + hasil
23          bb[1] = gabung
24
25      # filtering
26      stopword =
27  open('D:\SKRIPSI\Program\data\stopwordID.csv',
28  'r').read().lower()
29      stopwordlist = re.split(r'\n', stopword)
30      for kolom in liskata:
31          kaka = re.split(r'\s', kolom[1])
32          fil = ''
33          for b in kaka:
34              if b not in stopwordlist:
35                  fil += ' ' + b
36          kolom[1] = fil
37
38      # tokenisasi
39      for value in liskata:
40          baris = re.split(r'\s',value[1])
41          value[1]=baris
41          del value[1][0]
42
43      return liskata
```

Keterangan kode program :

1. Pada baris ke-1 sampai dengan baris ke-43 merupakan method yang digunakan untuk melakukan presprocessing.
2. Pada baris ke-4 sampai dengan baris ke-13 merupakan baris program untuk melakukan proses penghilangan karakter yang tidak diperlukan. Pertama dengan membuat daftar karakter yang ingin dihilangkan, kemudian setiap karakter pada teks yang diproses dicocokkan dengan daftar karakter yang sudah dibuat, jika karakter yang di cek tidak terdapat pada daftar kata maka karakter akan dikembalikan dan jika karakter terdapat dalam daftar, maka karakter akan dihapus atau dihilangan.
3. Pada baris ke-16 sampai dengan baris ke-23 merupakan baris program untuk melakukan proses stemming, dimana kata yang di cek akan dicari kata dasarnya. Stemmer yang digunakan adalah stemmer Jsastrawi.
4. Pada baris ke-26 sampai dengan baris ke-36 merupakan baris program untuk melakukan proses filtering, yakni proses untuk menghilangkan kata yang dianggap tidak penting. Pada proses ini terlebih dahulu dibuat daftar kata yang tidak diperlukan, seperti kata hubung dan bahasa asing. Kemudian kata tersebut dicocokkan pada kata yang diproses. Jika kata yang dimaksud terdapat dalam daftar maka kata akan dihilangkan.
5. Pada baris ke-39 sampai dengan baris ke-41 merupakan baris proses prpogram untuk melakukan proses tokenizing, yakni proses untuk memecah kalimat menjadi token-token.

5.2.2 Implementasi Algoritme Pembuatan Kamus

Pada tahapan ini dilakukan proses pembuatan kamus kedekatan kata yang terbuat dari data eksternal, yang mana pada proses ini menggunakan metode atau penghitungan *cosine similarity* untuk mendapatkan hasil kamus. Keluaran dari tahapan ini berupa pasangan kata dan nilai *cosine similarity* tiap pasangan kata yang disimpan dalam file.csv. untuk kode program dapat dilihat pada Kode program 5.2.

Kode Program 5.2 Implementasi Pembuatan Kamus

```
1   def frekuensi(self,matrik,kataunik):
2       for ada in kataunik:
3           wew = []
4           for aha in kataunik:
5               jmlditemukan=0
6               if ada +','+ aha in matrik:
7                   jmlditemukan = matrik.get(ada + ',' + aha)
8               if aha + ',' + ada in matrik:
9                   jmlditemukan = matrik.get(aha + ',' + ada)
10              wew.append(jmlditemukan)
11          frekuen[ada] = wew
12      return frekuen
13
14  def cosinesimilarity(self,frekuensi):
15      hasil=0
16      for q,w in frekuensi.items():
17          for e,r in frekuensi.items():
18              for y in range(len(w)):
19                  if q != e:
20                      hasil += w[y]*r[y]
21              perkalian[q+','+e]=hasil
22              hasil=0
23
24      for u,i in perkalian.items():
25          for o,p in frekuensi.items():
26              for z,x in frekuensi.items():
27                  if o+','+z == u and i != 0:
28                      kuadrat1 = sum([c*c for c in p])
29                      kuadrat2 = sum([v*v for v in x])
30                      akar = math.sqrt(kuadrat1) *
31  math.sqrt(kuadrat2)
32                      cosine = i/akar
33                      akhir[u] = cosine
34              kuadrat1=0
35              kuadrat2=0
36              akar=0
37              cosine=0
38      return akhir
```

Keterangan kode program :

1. Pada baris ke-1 sampai dengan baris ke-38 merupakan baris program untuk proses pembuatan kamus kedekatan kata.
2. Pada baris ke-1 sampai dengan baris ke-12 merupakan method untuk menyiapkan kata yang akan dihitung. Sebelumnya, kata yang sudah

dilakukan proses *pre-processing* sebelumnya disimpan dalam sebuah matrik berukuran tak hingga. Pada baris tersebut merupakan proses menghitungan kemunculan pasangan kata pada matrik ke samping dan ke bawah.

3. Pada baris ke-14 sampai dengan baris ke-38 merupakan proses penghitungan untuk pembuatan kamus untuk mencari kedekatan antar kata. Pada proses ini dilakukan penghitungan untuk mencari kedekatan kata dengan *cosine similarity*. Penghitungan dilakukan dengan mengoperasikan antar matrik, sehingga didapat hasil berupa pasangan kata beserta nilai *cosine similarity*. Setiap kata memiliki nilai 0 sampai dengan 1, jika pasangan kata nilai *cosine similarity*-nya mendekati 1, maka kata tersebut sangat dekat atau bisa diartikan kedua kata tersebut sering muncul pada dokumen yang sama.

5.2.3 Implementasi Algoritme *Feature Expansion*

Pada tahapan ini merupakan proses penambahan *feature expansion* pada data uji ketika dilakukan pengujian penggunaan *feature expansion*. Setiap *token* atau kata pada data uji di cari kedekatan katanya pada data yang sebelumnya telah di proses, kemudian tambahan kata tersebut disisipkan dalam data uji, sehingga terbentuklah data uji baru yang berisikan data uji yang telah di *preprocessing* dan ditambahkan dengan kata dari kamus kedekatan kata. Kode program dapat dilihat pada Kode Program 5.3.

Kode Program 5.3 Implementasi Algoritme *Feature Expansion*

```
1   def query(self,angka,kata):
2       nilaitreshold = float(angka)
3       dokum = open('D:\SKRIPSI\Program\data\cosine.csv')
4       dat = csv.reader(dokum, delimiter=',')
5       kaka = []
6       for row in dat:
7           row[1] = float(row[1])
8           kaka.append(row)
9       hasil = []
10      treshol = nilaitreshold
11      daftarkata = []
12      katabaru = []
13      rey = []
14          if treshol <= 1.0 :
15              for data in kata:
16                  for doko in data[1]:
17                      for a in kaka:
18                          if a[1] > treshol and a[1] <= 1.0
19  and doko in a[0]:
20                              isi = a[0], a[1]
21                              if isi not in hasil:
22                                  hasil.append(isi)
23                  for b in hasil:
24                      baru = re.split(r'-', b[0])
25                      rey.append(baru)
26                  for da in data[1]:
```

```
27                          katabaru.append(da)
28                          for rere in rey:
29                              if da == rere[0]:
30                                  if rere[1] not in katabaru:
31                                      katabaru.append(rere[1])
32                  isi = data[0],katabaru, treshol
33                  if isi not in daftarkata:
34                      daftarkata.append(isi)
35                  # print treshol, daftarkata
36                  coba =
37  open('D:\SKRIPSI\Program\data\cobaan.csv', 'a')
38                  csv_coba = csv.writer(coba)
39                  csv_coba.writerows(daftarkata)
40                  del hasil[0:]
41                  del rey[0:]
42                  del daftarkata[0:]
43                  del katabaru[0:]
44              treshol += 0.1
45      return kata
```

Keterangan kode program :

1. Pada baris ke-1 sampai dengan baris ke-45 merupakan proses pembentukan *feature expansion*.
2. Pada baris ke-2 merupakan penyimpanan variabel *threshold* yang sudah di input pada menu tampilan. Nilai *threshold* ini digunakan sebagai batas nilai terendah *cosine similarity* pada pasangan kata yang sesuai.
3. Pada baris ke-3 sampai dengan baris ke-8 merupakan proses membaca dan mengambil data pada hasil *cosine similarity* yang sebelumnya sudah disiapkan dan disimpan pada file.csv.
4. Pada baris ke-4 sampai dengan baris ke-45 merupakan proses pencarian kata yang bersangkutan sesuai dengan data yang ingin dicari *feature expansion*-nya. Proses dimulai pada baris 14, melakukan pengecekan terlebih dahulu nilai *threshold* yang akan digunakan sesuai syarat atau tidak, jika sesuai syarat akan melakukan proses pencarian nilai *cosine similarity* yang sesuai dengan nilai *threshold*. Jika pada pasangan kata terdapat kata yang dicari, maka akan dibuat daftar kata baru sesuai dengan kata uji dan ditambah dengan kata baru. Proses ini dilakukan oleh baris ke-27 sampai dengan baris ke-25. Kemudian hasil dari proses ini disimpan dalam file.csv yang dilakukan oleh baris ke-37 sampai dengan baris ke-40.

5.2.4 Implementasi Algoritme Klasifikasi

Pada tahapan ini merupakan proses klasifikasi dari data yang akan di ujikan. Setelah semua data dilakukan *preprocessing*, kemudian data dilakukan klasifikasi. Klasifikasi dilakukan pada data yang belum mengalami *feature expansion* dan data yang sudah mengalami *feature expasnion*. Untuk kode program dapat dilihat pada Kode Program 5.4.

Kode Program 5.4 Implementasi Algoritme Klasifikasi

```
1  def
2  klasifikasi(self,ujites,datatesting,klasifikasi,kataunik):
```

```
3       probabilitas = {}
4       # hasilklasifikasi = []
5       nilaiakhir = {}
6       likelihood=0
7
8       for key, value in klasifikasi.items():
9           # likelihood = 0
10          for key1, value1 in datatesting.items():
11              for kata in ujites:
12                  kk = key+','+kata
13                  if key+','+kata == key1 and kk not in
14  probabilitas:
15                      likelihood = (value1 + 1.0) /
16  float((len(value) + len(kataunik)))
17                  elif kata in key1 and kk not in
18  probabilitas:
19                      likelihood =  1.0 / float((len(value) +
20  len(kataunik)))
21                  elif key+','+kata == key1 and kk in
22  probabilitas:
23                      likelihood = float((value1 + 1.0)) /
24  float((len(value) + len(kataunik)))
25                  elif kata in key1 and kk in probabilitas:
26                      likelihood =  float(1.0) /
27  float((len(value) + len(kataunik)))
28                  probabilitas[kk] = likelihood
29      prior = 100.0 / 500
30      for key3,value3 in klasifikasi.items() :
31          for key2,value2 in probabilitas.items():
32              if key3+',' in key2:
33                  prior *= value2
34              nilaiakhir[key3]=prior
35          prior=100.0/500
36      akhir = 0
37      for ini, nilai in nilaiakhir.items():
38          if akhir < nilai:
39              akhir = nilai
40          else:
41              akhir = akhir
42      for isi, hasil in nilaiakhir.items():
43          if hasil == akhir:
44              hasilklasifikasi=isi
45
46      return hasilklasifikasi
```

Keterangan kode program :

1. Pada baris ke-1 sampai dengan baris ke-46 merupakan proses penghitungan klasifikasi *naive bayes multinomial*.
2. Pada baris ke-8 sampai dengan baris ke-28 merupakan proses penghitungan *likelihood* atau probabilitas setiap kata terhadap keseluruhan kata pada data latih.
3. Pada baris ke-29 sampai dengan baris ke-35 merupakan proses perkalian *prior* dengan *likelihood*. Dan disimpan dalam variabel nilai akhir pada baris ke-34.

4. Pada baris ke-37 sampai dengan baris ke-44 merupakan proses pencarian klasifikasi. yakni melakukan pengecekan nilai tertinggi untuk memperoleh hasil klasifikasi dari data yang diuji.

5.2.5 Implementasi Algoritme Pengujian

Tahapan ini bertujuan untuk mengetahui hasil akurasi dari klasifikasi yang telah dilakukan, dengan membandingkan kategori sebenarnya dengan kategori hasil klasifikasi dan didapat hasil akurasinya. Untuk kode program dapat dilihat pada Kode Program 5.5

Kode Program 5.5 Implementasi Algoritme Pengujian

```
1   def akurasi(self,datalati):
2       databenar=0
3       datasalah=0
4       for data in datalati:
5           if data[0] == data[3]:
6               databenar += 1
7           else:
8               datasalah += 1
9
10      print ' data benar sekarang : ',databenar
11      print ' data salah sekarang : ',datasalah
12      akurasiakhir =
13  float(databenar)/float((datasalah+databenar))
14      print akurasiakhir
15      return  akurasiakhir
```

Keterangan kode program :

1. Pada baris ke-1 sampai dengan baris ke-15 merupakan proses penghitungan akurasi dari hasil program.
2. Pada baris ke-4 sampai dengan baris ke-8 merupakan pengecekan kesesuaian label kategori awal dengan hasil klasifikasi.
3. Pada baris ke-13 melakukan perhitungan akurasi dengan cara jumlah data benar dibagi dengan seluruh data.

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini menerangkan tentang pengujian yang dilakukan terhadap sistem yang telah dibangun. Selain itu juga pada bab ini menjelaskan analisis dari hasil implementasi dan pengujian yang telah dilakukan.

6.1 Data yang digunakan

Penelitian ini dilakukan berdasarkan *dataset* atau himpunan data yang diperoleh dari sumber penelitian sebelumnya yang dilakukan oleh sukarno,dkk (Sukarno, 2016). *Dataset* terbagi menjadi dua yakni data yang digunakan sebagai data eksternal sebagai bahan untuk kamus kedekatan kata atau *feature expansion* dan data yang digunakan sebagai penelitian. Data eksternal berasal dari *website* berita Kompas.com dan Detik.com, data tersebut diolah dengan menggunakan *method cosine similarity* dan didapat hasil akhir berupa pasangan kata beserta nilai *cosine similarity* yang digunakan sebagai *feature expansion*. *Dataset* yang digunakan untuk penelitian merupakan *tweets* yang berasal dari akun Detik.com dan Kompas, pemilihan *tweets* setiap kategori diambil dari sumber tersebut, disesuaikan dengan kategori yang dipilih oleh peneliti untuk klasifikasi yang dilakukan. Jumlah data yang digunakan yaitu 20 dokumen berita, 500 data latih berupa *tweets*, dan 100 data uji berupa *tweets*. Dokumen berita yang digunakan terbagi rata untuk 5 jenis kategori konten, sama halnya dengan data latih dan data uji yang masing-masing terbagi dalam 5 kategori berita *tweets*.

6.2 Pengujian Penggunaan *Feature Expansion* dan Tanpa Penggunaan *Feature Expansion*

Pengujian penggunaan *feature expansion* dan tanpa penggunaan *feature expansion* dilakukan untuk mengetahui pengaruh penggunaan *feature expansion* terhadap hasil klasifikasi. Pada pengujian tanpa penggunaan feature expansion data uji yang sudah dilakukan *presprocessing* langsung dilakukan proses klasifikasi dan didapat nilai akurasinya. Sedangkan pada pengujian penggunaan *feature expansion*, data uji yang sudah dilakukan *presprocessing* ditambahkan kata atau fitur baru yang sesuai nilai *cosine similarity*-nya pada data eksternal atau kamus dengan menggunakan *threshold* (batas bawah), dari sini didapat nilai *threshold* pada *cosine similarity* yang paling optimal dan menghasilkan akurasi klasifikasi tertinggi.

6.2.1 Skenario Pengujian Penggunaan *Feature Expansion* dan Tanpa Penggunaan *Feature Expansion*

Pengujian dilakukan pada data uji dengan membandingkan hasil akurasi klasifikasi jika dalam data uji terdapat penggunaan *feature expansion* dan tanpa penggunaan *feature expansion*. Keseluruhan data dilakukan proses *preprocessing* untuk mendapatkan data yang siap dilakukan proses klasifikasi. Dan untuk melakukan *feature expansion* terlebih dahulu dibuat kamus kedekatan kata yang

berasal dari dokumen eksternal. Proses pembuatan kamus ini menggunakan perhitungan *cosine similarity* untuk mencari kedekatan antar kata dengan nilai tertentu. Setelah tersimpan pasangan kata beserta nilai *cosine similarity*-nya dilakukan proses pengklasifikasian berita, pada data latih dan data uji sudah dilakukan *text preprocessing*.

Proses klasifikasi dilakukan dengan dua cara, yakni dengan menggunakan *feature expansion* dan tanpa menggunakan *feature expansion*. Pada klasifikasi tanpa menggunakan *feature expansion* dokumen yang sudah dilakukan *text preprpocessing* langsung diklasifikasikan, untuk klasifikasi menggunakan *feature expansion* dokumen uji yang sudah dilakukan *text preprocessing* ditambahkan kata dari data kamus sesuai nilai *threshold* 0,1 sampai dengan 1,0. Untuk mengetahui hasil perubahan perbandingan penggunaan *feature expansion* dan tanpa menggunakan *feature expansion*, pengujian dilakukan dengan mengubah nilai *threshold* dari 0,1 sampai dengan 1,0. Untuk hasil akurasi pengujian dapat dilihat pada tabel 6.1.

Tabel 6.1 Hasil Pengujian Penggunaan Feature Expansion Dan Tanpa Feature Expansion

Akurasi Tanpa Menggunakan Feature Expansion	Menggunakan Feature Expansion	
	Threshold	Akurasi
83%	0,1	58%
	0,2	59%
	0,3	62%
	0,4	65%
	0,5	67%
	0,6	67%
	0,7	73%
	0,8	76%
	0,9	**87%**
	1,0	83%

6.2.2 Analisis Pengujian Penggunaan *Feature Expansion* dan Tanpa Menggunakan *Feature Expansion*

Hasil pengujian penggunaan *feature expansion* dan tanpa penggunaan *feature expansion* menunjukkan hasil yang bervariasi, penggunaan *feature*

expansion terbukti dapat meningkatkan akurasi. Pengujian tanpa penggunaan *feature expansion* menghasilkan akurasi sebesar 83%, sedangkan pada pengujian penggunaan *feature expansion* akurasi meningkat menjadi **87%**. Hasil tersebut didapat pada nilai *threshold* **0,9**.

Dokumen latih dan dokumen uji yang didapat berupa *tweets* yang merupakan jenis dokumen *short text*. Telah dijelaskan sebelumnya bahwa jenis dokumen ini memiliki beberapa kelemahan seperti dalam proses klasifikasi, fitur atau kata yang muncul pada data uji sering kali tidak terdapat dalam data latih, sehingga hasil klasifikasi cenderung tidak sesuai dengan kategori sebenarnya. Dengan penggunaan *feature expansion*, pada data uji ditambahkan kata yang membuat kemungkinan adanya kata tersebut menjadi lebih besar dalam data latih sehingga hasil klasifikasi sesuai dengan kategori sebenarnya. Selain itu, biasanya kata kunci yang terdapat dalam data uji sebelumnya tidak muncul pada data latih, dengan penambahan *feature expansion* kemungkinan besar akan ditambahkan pada data uji baik melalui kata kunci maupun bukan kata kunci. Semakin banyak kata relevan yang ditambahkan dalam data uji hasil klasifikasi kemungkinan besar mengarah pada hasil kategori yang sebenarnya. Sebagai contoh dapat dilihat pada tabel 6.2

Pada penggunaan *feature expansion* dapat dilihat bahwa dengan penambahan kata dapat membuat hasil klasifikasi sesuai dengan data sebenarnya. Pada data sebenarnya hasil klasifikasi tidak sesuai dengan data sebenarnya, hal ini dikarenakan kata-kata pada data yang diujikan tidak muncul sama sekali dalam dokumen latih. Pada proses penambahan *feature expansion* kata **'tumbuh'** yang merupakan kata kunci dari data tersebut mendapat tambahan kata yang sesuai dengan data latih (hasil ekspansi adalah kata dengan cetakan *bold*) seperti kata **'gedung', 'kisar', 'nasional', 'bisnis', 'bbm'**, dan '**bumn'** kata tambahan ini ternyata terdapat dalam data latih sehingga hasil klasifikasi mengarah pada kategori yang sesuai dengan kategori sebenarnya.

Penggunaan *feature expansion* ini juga dipengaruhi oleh nilai *threshold* yang diberikan. Karena pembuatan data kamus kedekatan kata menggunakan *cosine similarity,* dapat dilihat bahwa data dengan nilai *cosine* rendah kata yang di tambahkan akan sangat banyak dan tidak relevan dengan data latih, sehingga hasil klasifikasi kurang optimal. Namun, pada data dengan nilai *cosine similarity* tinggi (mendekati 1,0) kata yang ditambahkan adalah kata yang relevan dan kemungkinan kata tersebut sering muncul dalam data latih, sehingga bisa memaksimalkan hasil klasifikasi dan meningkatkan akurasi.

Pada pengujian penggunaan *feature expansion* dengan nilai *threshold* rendah mendapatkan akurasi yang kecil, yakni 58%. Hal ini dikarenakan pada nilai *threshold* kecil misal 0,1 jumlah daftar kata yang dimasukkan atau ditambahkan dalam dokumen uji sangat banyak, dan tentunya hal ini mempengaruhi hasil klasifikasi. Baik kata yang relevan maupun kata yang tidak relevan semua ditambahkan ke data uji. Sehingga dalam proses penghitungan klasifikasi data tambahan tersebut menghasilkan kategori yang tidak sesuai dengan kategori

sebenarnya karena jarak kedekatan kata yang dimasukkan jauh. Sebagai contoh dapat dilihat pada table 6.3.

Pada penggunaan feature expasnion dengan nilai threshold 0,1 terlihat bahwa hasil klasifikasi dengan menggunakan feature expasnion tidak sesuai dengan kategori sebenarnya. Pada dokumen tersebut yang mendapat tambahan kata adalah kata '**australia'** dan '**air'**. Daftar tambahan kata pada kata '**australia'** lebih cenderung mengarah pada dokumen dengan kategori yang tidak sesuai dengan kata sebenarnya seperti kata **'daerah', 'perintah', 'proyek', 'Indonesia', 'dunia', 'daya'** dan **'tekno'**. Sedangkan pada kata **'air'** mendapat tambahan kata **'kartu'** dan '**harga'**. Meskipun kedua kata tersebut merupakan kata kunci dari data uji, namun kata tambahannya tidak mengarah pada kategori yang sebenarnya dan justru tambahan kata tersebut tersebar pada beberapa kategori. Hal ini dikarenakan *range* atau batasan nilai *threshold* yang diberikan masih terlalu kecil, sehingga semua kata masuk dalam daftar kata tambahan yang menghasilkan klasifikasi tidak maksimal.

Pada nilai *threshold* tinggi 0,9 didapat akurasi tertinggi yakni 87%, hal ini terjadi karena pada nilai *theshold* ini daftar kata yang dimasukkan atau ditambahkan ke dalam data uji adalah kata yang nilai *cosine similarity* nya 0,9 dan daftar kata tersebut bisa dianggap penting oleh sistem. Sehingga pada penghitungan klasifikasi akan menjadi optimal, sebab data yang terdapat pada data uji baru kebanyakan sesuai dengan data latih pada kategori tersebut, sehingga akurasi bisa meningkat. Hal ini tentu saja akan membantu proses klasifikasi dan pada nilai *theshold* ini menunjukkan nilai *threshold* yang paling optimal. Sebagai contoh hasil feature expansion dapat dilihat pada table 6.4.

Pada penggunaan feature expansion dengan threshold 0,9 dapat dilihat bahwa dengan penambahan kata dapat mengarahkan dokumen pada kategori sebenarnya, meski kata kunci **'Microsoft'** tidak pernah muncul dalam data kamus, namun pada kata lain seperti kata **'sedia'** memiliki penambahan kata '**dorong', 'ukur', dan 'listrik**' yang kemungkinan besar kata tersebut terdapat pada data latih teknologi, terlebih lagi pada kata 'Indonesia' yang bisa dijadikan sebagai kata kunci mendapat tambahan kata yang sesuai dengan kategori teknologi seperti kata **'timor', 'leste', 'dunia', 'daya',** dan **'tekno'** sehingga hasil klasifikasi menjadi kategori teknologi dan sesuai dengan kategori aslinya.

Namun, pada nilai *threshold* 1,0 pada pengujian ke sepuluh, akurasi justru menurun menjadi 83%. Hal ini dikarenakan pada nilai *threshold* 1,0 kebanyakan daftar kata dengan nilai *threshold* tersebut merupakan kata pada dirinya sendiri, sangat sedikit kata yang bukan dirinya sendiri bernilai 1,0. sehingga hasil klasifikasi akan sama dengan hasil klasifikasi tanpa menambahkan *feature expansion*. Sebagai contoh dapat dilihat pada table 6.5.

Dari contoh tersebut terlihat bahwa tidak ada penambahan kata dari dokumen awal dengan penambahan *feature expansion* dengan *threshold* 1,0. Dan hasil klasifikasi juga menunjukkan perbedaan dengan label awal dokumen. Hal ini dikarenakan pada daftar kata tersebut seperti kata '**pelosok'** dan '**Indonesia'**

sering muncul pada data latih dengan kategori ekonomi. Itulah mengapa hasil klasifikasi mengarah ke dokumen dengan kategori ekonomi.

Tabel 6.2 Contoh Penggunaan Feature Expansion Pada Klasifikasi

dokumen	isi	Kategori sebenarnya	Hasil klasifikasi
Tanpa feature expansion	'btpn', 'wow', 'tumbuh', 'tabung', 'masyarakat'	Ekonomi	**Kesehatan**
Dengan feature expasnion	'btpn', 'wow', 'tumbuh', **'gedung', 'kisar', 'merauke', 'perum', 'nusantara', 'unggul', 'manfaat', 'stakeholder', 'fisheries', 'olah', 'dak', 'sarana', 'konsen', 'nasional', 'ajak', 'investment', 'pal', 'incorporated', 'dermaga', 'natuna', 'lloyd', 'morotai', 'pasar', 'tumpu', 'sarmi', 'acara', 'patroli', 'single', 'rangka', 'dukung', 'sus', 'konservasi', 'instalasi', 'tambat', 'numfor', 'perban', 'rote', 'garam', 'pabrik', 'ikan', 'kerjasama', 'saumlaki', 'gudang', 'bisnis', 'speedboat', 'bbm', 'integrasi', 'tual', 'pokmaswas', 'simeleue', 'balroom', 'business', 'sektor', 'sambut', 'sumberdaya', 'pembudidaya', 'tawa', 'tahuna', 'harap', 'bas', 'tangkap', 'bahari', 'pesisir', 'rakyat', 'alokasi', 'undang', 'marine', 'pelno', 'perahu', 'komoditas', 'sangihe', 'mina', 'pinggir', 'kkp', 'kodja', 'acu', 'laut', 'bumn', 'pusat', 'gt', 'cold', 'padu', 'prasarana', 'wilayah', 'timika', 'tunjang', 'peta', 'djakarta', 'peluang', 'nunukan', 'apbn', 'ndao', 'es', 'lokasi', 'biak'**, 'tabung', 'masyarakat'	Ekonomi	**Ekonomi**

Tabel 6.3 Contoh Dokumen Yang Dilakukan Feature Expansion Dengan Nilai Threshold 0.1

Dokumen	Isi	Kategori Sebenarnya	Hasil Klasifikasi
Dokumen Asli	'vaksin', 'anak', 'australia', 'wabah', 'cacar', 'air'	Kesehatan	**Entertaiment**

Dokumen	Isi	Kategori Sebenarnya	Hasil Klasifikasi
Dokumen dengan Feature Expansion dengan Threshold 0,1	'vaksin', 'anak', 'australia', **'daerah', 'capai', 'november', 'timor', 'gelap', 'presiden', 'ppn', 'selenggara', 'sofyan', 'malaysia', 'ali', 'alfredo', 'perintah', 'contoh', 'sumber', 'convention', 'johnity', 'bantu', 'nusa', 'ben', 'jumat', 'mou', 'total', 'bndcc', 'al', 'eksekutif', 'kongkrit', 'emisi', 'ribu', 'gelar', 'ministerial', 'pers', 'minyak', 'ongkili', 'arab', 'ciobo', 'hon', 'proyek', 'iklim', 'internasional', 'ahli', 'mw', 'pakat', 'iea', 'konferensi', 'komitmen', 'hadir', 'bcef', 'tindak', 'provinsi', 'energi', 'desa', 'international', 'rang', 'wakil', 'micah', 'triliun', 'manusia', 'jusuf', 'energy', 'baur', 'proyek', 'pencil', 'situasi', 'sdm', 'fatih', 'karbon', 'menandatangani', 'daya', 'sadar', 'indonesia', 'fosil', 'esdm', 'maximus', 'jabat', 'dunia', 'pires', 'leste', 'sudirman', 'saudi', 'djalil', 'paris', 'gantung', 'canang', 'papua', 'strategi', 'ken', 'meeting', 'hijau', 'tekno', 'nugini', 'langkah', 'mineral', 'haru', 'steven', 'agency', 'birol', 'libat', 'ebt', 'kalla', 'rendah', 'senang', 'hati', 'komit', 'center', 'butuh', 'naim',** 'wabah', 'cacar', 'air'**, 'pengki', 'tampung', 'liter', 'rumah', 'mop', 'kerap', 'mega', 'swing', 'bulu', 'sapu', 'enviro', 'cocok', 'promo', 'kepel', 'peras', 'ember', 'tutup', 'lion', 'max', 'ukur', 'jaga', 'square', 'carrefour', 'grill', 'potong', 'star', 'perabot', 'nagata', 'wadah', 'sisa', 'diskon', 'sisih', 'sampah', 'karet', 'minggu', 'jendela', 'lini', 'lengkap', 'meja', 'bundar', 'warnawarni', 'shinpo', 'alat', 'volume', 'ceria', 'rumit', 'tinggal', 'maspion', 'orchid', 'round', 'pel', 'lantai', 'swash', 'laplap', 'kredit', 'pegang', 'kartu', 'forte', 'livina', 'taiwan', 'plastik', 'cotton', 'transmart', 'asw', 'bank', 'bersih', 'enyah', 'mati', 'harga', 'dapur'**	Kesehatan	**Teknologi**

Tabel 6.4 Contoh Dokumen Yang Dilakukan Feature Expansion Dengan Nilai Threshold 0.9

Dokumen	Isi	Kategori Sebenarnya	Hasil Klasifikasi
Dokumen Asli	'microsoft', 'sedia', 'internet', 'pelosok', 'indonesia'	Teknologi	**Ekonomi**
Dokumen dengan Feature Expansion dengan Threshold 0,9	'microsoft', 'sedia', **'dorong', 'ukur', 'listrik',** 'internet', 'pelosok', 'indonesia', **'strategi', 'baur', 'provinsi', 'senang', 'internasional', 'minyak', 'capai', 'hati', 'proyek', 'australia', 'contoh', 'sdm', 'jusuf', 'jumat', 'ministerial', 'wakil', 'alfredo', 'selenggara', 'paris', 'manusia', 'sadar', 'sudirman', 'november', 'canang', 'nugini', 'ebt', 'mineral', 'hadir', 'agency', 'haru', 'kalla', 'ribu', 'sumber', 'ben', 'gantung', 'desa', 'iea', 'maximus', 'pencil', 'tekno', 'negara', 'malaysia', 'saudi', 'mou', 'esdm', 'kembang', 'mw', 'pers', 'birol', 'djalil', 'sofyan', 'ahli', 'total', 'rang', 'ciobo', 'timor', 'meeting', 'pires', 'jabat', 'komit', 'international', 'gelap', 'gelar', 'menteri', 'situasi', 'ppn', 'energy', 'nilai', 'bantu', 'hon', 'daerah', 'papua', 'fosil', 'triliun', 'bcef', 'ali', 'bndcc', 'steven', 'kongkrit', 'butuh', 'eksekutif', 'tindak', 'emisi', 'iklim', 'energi', 'micah', 'nusa', 'arab', 'pakat', 'leste', 'libat', 'komitmen', 'johnity', 'naim', 'karbon', 'daya', 'ongkili', 'langkah', 'hijau', 'konferensi', 'dunia', 'fatih'**	Tekonologi	**Teknologi**

Tabel 6.5 Contoh Dokumen Yang Dilakukan Feature Expansion Dengan Nilai Threshold 1.0

Dokumen	Isi	Kategori Sebenarnya	Hasil Klasifikasi
Dokumen Asli	'microsoft', 'sedia', 'internet', 'pelosok', 'indonesia'	Teknologi	**Ekonomi**
Dokumen dengan Feature Expansion dengan Threshold 1,0	'microsoft', 'sedia', 'internet', 'pelosok', 'indonesia'	Tekonologi	**Ekonomi**

BAB 7 PENUTUP DAN KESIMPULAN

Pada bab ini berisi penjelasan terkait kesimpulan yang didapat pada penelitian ini serta saran yang dapat dijadikan acuan dalam penelitian selanjutnya.

7.1 Kesimpulan

Berdasarkan hasil pembahasan dan penelitian yang telah dilakukan pada bab sebelumnya didapatkan kesimpulan sebagai berikut :

1. Pada penggunaan penambahan *feature expansion* pada klasifikasi dengan menggunakan *naive bayes* menunjukkan adanya peningkatan akurasi dalam proses klasifikasi. Hal ini menunjukkan adanya pengaruh penambahan *feature expansion* dalam klasifikasi *short text*.
2. Pada pengujian tanpa menggunakan *feature expansion* didapat hasil akurasi sebesar 83%. Pada pengujian menggunakan *feature expansion* dengan nilai *threshold* sebagai acuan untuk menambahkan banyaknya kata. Pada nilai *threshold* kecil hasil akurasi menunjukkan persentase yang kecil yakni 58%, hal ini dikarenakan pada penambahan threshold kecil kata yang ditambahkan sangat banyak dan justru bukan kata yang relevan dalam kategori sebenarnya sehingga akurasi yang didapat juga kecil. pada *threshold* sedang hasil akurasi meningkat menjadi 59% sampai dengan 76%, dan pada nilai *threshold* 0,9 akurasi paling tinggi yakni 87%, hal ini dikarenakan pad nilai threshold ini kata yang di tambahkan cenderung memiliki kedekatan kata secara semantik dan pada data latih kata tersebut sesuai atau relevan sehingga hasil klasifikasi sesuai dengan data sebenarnya dan akurasi meningkat. Sedangkan pada nilai *threshold* 1,0 akurasi menurun menjadi 83% hal ini dikarenakan kata yang diambil adalah kata pada data itu sendiri sehingga tidak mengubah data. Penggunaan *feature expansion* dapat meningkatkan akurasi pada proses klasifikasi, namun yang perlu diperhatikan adalah nilai batas bawah atau *threshold* yang dijadikan sebagai acuan untuk penambahan kata. Semakin *threshold* yang di berikan mendekati angka nol (0) hasil akurasi akan semakin menurun, semakin mendekati angka satu (1) nilai *threshold* yang di berikan hasil akurasi akan semakin meningkat.

7.2 Saran

1. Pada tahapan pembuatan kamus berita membutuhkan waktu yang lama, sebab melakukan penghitungan *cosine similarity* dari semua dokumen yang dimasukkan dan disimpan dalam sebuah matrik, sehingga perlu algoritme untuk bisa mengefisien kan proses pembuatan kamus salah satunya menggunakan *multi thread*.

DAFTAR PUSTAKA

Agusta, L., 2009. Perbandngan algortma stemming Porter dengan algoritma Nazief & Adriani untuk Stemming dokumen teks bahasa indonesia. *konferensi Nasional Sistem dan Informatika,* pp. 196-201.

Al-Smadi, M. Z. J. M. A.-a. y. J., 2017. Paraphare identification and semantic text similarity analysis in Arabic tweets using lexical, systactic, dan semantic features. *Elsevier,* Issue Information Processing and Management, pp. 640-452.

Chunxia Jin, H. Z. Q. B., 2012. Short Text Clustering Algorithm eith Feature Keyword Expansion. *Scientific,* Volume 532-533, pp. 1716-1720.

Destuardi dan Surya, S., 2009. *Klasifikasi Emosi Untuk Teks Bahasa Indonesia Menggunakan Metode Naive Bayes,* Surabaya: Teknik Elektro, Institut Teknologi Sepuluh Nopember.

Firmansyah, R. F. N., 2016. Sentiment Anaysis pada Review Aplikasi Moble Menggunakan Metode Naive Bayes dan Query Expansion.

Guo, Q., 2010. *An Affective Algorithm for Improving the performanceif Naive Bayes for Text Classification,* s.l.: Cambridge University Press.

Hand, D., 2010. Text Mining: Classification, Clustering, dan Application edited by Ashor Srivastava, Mehran Sahami. In: M. Sahami, ed. s.l.:International Statistical review, pp. pp.134-135.

Insan, P. P., 2013. *Kalsisikasi Emosi untuk Teks Bahasa Indonesia dengan Menggunakan Algoritma C5.0,* Malang: Program Studi Informatika/Ilmu Komputer PTIIK Universitas Brawijaya.

Kayser, V., 2016. Extending the knowledge base of foresight: The contribution of text mining. *Elsevier,* Issue Technological forecasting & Social Change, pp. 208-215.

Kominfo, 2016. *Kementerian Iinformasi dan Informatika Republik Indonesia.* [Online] Available at: www.kominfo.go.id [Diakses 2018].

Kominfo, 2016. *Kementerian Komunikasi dan Informatika.* [Online] Available at: www.kominfo.go.id [Diakses 2018].

Kurniawan, H., 2006. Otomatisasi Pengelompokkan Koleksi Perpustakaan dengan pengukuran Cosine Similarity dan Euclidian Distance. *Seminar Nasional Aplikasi teknologi Informasi 2006,* pp. J19 - J22.

Mandala, R., 2009. *Relevance Feedback And Query Expansion,* s.l.: Cambridge University Press.

Mooney, R. J., 2006. *Machine Learning Text Categorization,* United State: University of Texas, Austin.

Nokhbeh Zaeem, R. M. M. Y. Y., 2016. Modelling and analysis of identify threat behaviors through text mining of identity theft stories. *Elsevier,* pp. 50-63.

Nurdiana, O., 2016. Perbandingan metode Cosine Similarity dengan Metode JaccardSimilarity pada Aplikasi Pencarian TerjemahAl-Qur'an dalam Bahasa Indonesia. *JOIN,* Volume 1, pp. 59- 63.

Perdana, R., 2013. *Pengkategorian Pesan Singkat Berbahasa Indonesia pada Jejarinag Sosial Twitter dengan Metode Klasifikasi Naive Bayes,* Malang: Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.

Perdana, R. S., 2015. Bot Spammer Detection in Twitter Using Tweet SIimilarity and Time. *Jurnal Ilmu Komputer dan Informasi,* 8(1), pp. 19-25.

Ramya, M. & Pinakas, J., 2014. Different Type of Feature Selection for Text Classification. *Internatioan Journal of Computer Trends and Technology (IJCTT),* Volume 10(2), pp. pp. 102-107.

Rizqi Lahitani, A. E. P. A. A. S. N., 2016. Cosine Similarity to Determine Similarity Measure: Study Case in online Essay Assessment.

Sriram, B., 2010. Short Text classification in twitter to improve information filtering. *Preceeding of 33rd international ACM SIGIR conference on Research and development in information retrieval,* Volume SIGIR'10.

Sugiyamta, 2015. Sistem Deteksi Kemiripan Dokumen dengan Algoritma Cosine Similarity dan Single Pass Clustering. *Dinamika Informatika,* Volume Vol.7 No.2, pp. 85 - 91.

Sukarno, A. K., 2016. Klasifikasi Tweets pada Twitter Menggunakan Metode Naive Bayes dan Query Expansion Berbasis Apriori.

Tang, J., 2017. End-to-end Learning for Short Text Expansion.

Twitter, 2016. *Tweets on Twitter.* [Online] Available at: http://twitter.com [Diakses 2017].