KLASIFIKASI BERITA PADA TWITTER MENGGUNAKAN METODE NAIVE BAYES DAN QUERY EXPANSION HIPERNIM-HIPONIM

SKRIPSI

Untuk Memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Fakhruddin Farid Irfani
NIM: 145150200111031

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

PENGESAHAN

Klasifikasi Berita pada Twitter Menggunakan Metode Naive Bayes dan Query Expansion Hipernim-Hiponim

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Fakhruddin Farid Irfani
NIM: 145150200111031

Skripsi ini telah diuji dan dinyat lulus pada:
26 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
M. Ali Fauzi, S.Kom, M.Kom	Yuita Arum Sari, S.Kom., M.Kom
NIK: 201502 890101 001	NIK: 201609 880715 2 001

Mengetahui
Ketua Jurusan Teknik Informatika

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 disitasi 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, 13 Juli 2018

n Farid Irfani

NIM:145150200111031

KATA PENGANTAR

Puji syukur Penulis panjatkan kepada Allah SWT atas limpahan rahmat, hidayah, inayah dan kesmpatan yang telah diberikan kepada Penulis, sehingga Penulis dapat menyelesaikan penelitian dengan judul "Klasifikasi Berita pada Twitter Menggunakan Metode Naive Bayes dan Query Expansion Hipernim-Hiponim". Skripsi ini diajukan untuk memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer (FILKOM) Universitas Brawijaya.

Dalam melakukan penelitian dan penyusunan dokumen skripsi ini, Penulis banyak sekali mendapatkan panduan, dukungan, dan arahan dari berbgai pihak. Dengan tulisan ini Penulis ingin menyampaikan terimakasih yang sebesar-besarnya kepada:

1. Bapak M. Ali Fauzi, S.Kom, M.Kom selaku dosen pembimbing satu yang telah meluangkan waktu serta memberikan arahan dan bimbingan dalam pengerjaan skripsi ini.
2. Ibu Yuita Arum Sari, S.Kom., M.Kom selaku dosen pembimbing dua yang telah meluangkan waktu serta memberikan arahan dan bimbingan dalam pengerjaan skripsi ini.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
6. Seluruh Dosen Fakultas Ilmu Komputer yang telah bersedia membagi ilmunya dan mendidik Penulis.
7. Bapak dan Ibu, serta seluruh keluarga yang selalu mendukung dan mendorong Penulis demi kemudahan dan kelancaraan pengerjaan skripsi.
8. Teman-teman penulis, Tami, Fahcrurozy, Ochi, Amal, Martha, Gita, Hamim, Dwi, Rosi, anggota transkoper, serta teman-teman penghuni kontrakan ceria, kontrakan keramat, dan koocan, yang telah mendukung dan terlibat dalam segala proses pengerjaan skripsi.
9. Untuk seluruh teman-teman yang belum dapat disebut satu-persatu, yang telah membantu dan mendukung penulis dalam pengerjaan skripsi.

Penulis sadar bahwasanya skripsi ini memiliki banyak kekurangan dan jauh dari kata sempurna. Penulis mengharapkan dan menerima kritik maupun saran agar penulis menjadi lebih baik. Akhir kata, Penulis berharap agar skripsi ini dapat

bermaanfaat untuk seluruh pembacanya dan untuk penelitian-penelitian selanjutnya.

Malang, 13 Juli 2018

Penulis

Udinirfan17@gmail.com

ABSTRAK

Fakhruddin Farid Irfani, 2018. Klasifikasi Berita Pada Twitter Menggunakan Metode Naive Bayes Dan query expansion Hipernim-Hiponim

Pembimbing : M. Ali Fauzi, S.Kom, M.Kom dan Yuita Arum Sari, S.Kom., M.Kom

Banyaknya jumlah *tweet* yang di-*post* mengakibatkan *tweet* yang tersebar dan muncul dalam beranda Twitter sangat beragam dan tidak dikelompokkan berdasarkan kategori beritanya seperti kesehatan, olahraga, teknologi, ekonomi, wisata dan lain sebagainya. Tidak adanya pengkategorian menyebabkan pengguna kesulitan jika ingin membaca atau mengambil informasi terkait kategori tertentu yang diinginkan. Salah satu solusi yang dapat dilakukan adalah dengan metode klasifikasi teks, yang dalam proses klasifikasinya mampu mengklasifikasikan secara otomatis terhadap beberapa kategori pada teks tidak terstruktur dengan Bahasa alami. Pada penelitian ini dilakukan proses klasifikasi menggunakan metode Naive Bayes dengan tambahan *query expansion* untuk menambahkan *term* pada dokumen awal. Penambahan *term* bertujuan untuk mengoptimalkan proses klasifikasi dikarenakan *tweet* merupakan *short text* yang dapat menimbulkan ambiguitas kelas klasifikasi. Penambahan yang dilakukan adalah hiponim dan hipernim dari dokumen asli yang diambil dari WordNet. Metode perhitungan akurasi yang digunakan adalah *k-fold* yang bertujuan untuk menguji kehandalan dari sistem. Akurasi yang didapatkan adalah sebesar 72% untuk klasifikasi tanpa *query expansion*, 65,75% untuk penambahan hiponim dan hipernim, 66,3% untuk penambahan hiponim saja, dan 67,5% untuk penambahan hipernim saja. Dapat disimpulkan bahwa penambahan *query* yang dilakukan kurang efektif untuk meningkatkan akurasi proses klasifikasi.

Kata kunci: *Tweet, Naive Bayes, WordNet, Hiponim, Hipernim, K-Fold*

ABSTRACT

Fakhruddin Farid Irfani, 2018. News Classification On Twitter Using Naive Bayes Method And query expansion Hyperniym-Hyphonyim

Advisor :M. Ali Fauzi, S.Kom, M.Kom and Yuita Arum Sari, S.Kom., M.Kom

The large number of posted tweets resulted in scattered tweets and appearing on the Twitter homepage very diverse and not classified by categories such as health, sports, technology, economics, tourism and so on. The absence of categorization causes the user difficulty to read or retrieve information related to certain desired categories. Ssolution that can be done is by the method of text classification, which in the process of classification is able to classify automatically against some categories on unstructured text with natural language. In this research will be done classification process using Naive Bayes method with additional query expansion to add term in initial document. The addition of term aims to optimize the classification process because the tweet is a short text that can lead to ambiguity of classification classi. The additions made are hyponym and hypernym from original documents extracted from WordNet. Accuracy calculation method used is k-fold that aims to test the robustness of system. The accuracy obtained was 72% for the classification without query expansion, 65.75% for hyponym and hypernym addition, 66.3% for hyponym addition, and 67.5% for hypernym addition. It can be concluded that the addition of queries made less effective to improve the accuracy of the classification process.

Keywords: *Tweet, Naive Bayes, WordNet, Hyponym, Hypernym, K-Fold*

DAFTAR ISI

PENGESAHAN ii
PERNYATAAN ORISINALITAS iii
KATA PENGANTAR iv
ABSTRAK vi
ABSTRACT vii
DAFTAR ISI viii
DAFTAR TABEL xi
DAFTAR GAMBAR xii
DAFTAR KODE PROGRAM xiii
DAFTAR LAMPIRAN xiv
BAB 1 PENDAHULUAN ..... **Error! Bookmark not defined.**
1.1 Latar belakang ..... **Error! Bookmark not defined.**
1.2 Rumusan masalah ..... **Error! Bookmark not defined.**
1.3 Tujuan ..... **Error! Bookmark not defined.**
1.4 Manfaat ..... **Error! Bookmark not defined.**
1.5 Batasan masalah ..... **Error! Bookmark not defined.**
1.6 Sistematika pemBahasan ..... **Error! Bookmark not defined.**
BAB 2 LANDASAAN KEPUSTAKAAN ..... **Error! Bookmark not defined.**
2.1 Kajian Pustaka ..... **Error! Bookmark not defined.**
2.2 Twitter ..... **Error! Bookmark not defined.**
2.3 Text Mining ..... **Error! Bookmark not defined.**
2.4 Text Preprocessing ..... **Error! Bookmark not defined.**
2.4.1 *Tokenisasi* ..... **Error! Bookmark not defined.**
2.4.2 *Filtering* ..... **Error! Bookmark not defined.**
2.4.3 *Stemming* ..... **Error! Bookmark not defined.**
2.5 Term Weighting ..... **Error! Bookmark not defined.**
2.5.1 *Term Frequency* ..... **Error! Bookmark not defined.**
2.6 Klasifikasi Teks ..... **Error! Bookmark not defined.**
2.7 Naïve Bayes *Classifier* ..... **Error! Bookmark not defined.**
2.7.1 *Multinomial* Naïve Bayes ..... **Error! Bookmark not defined.**
2.8 *Query Expansion* ..... **Error! Bookmark not defined.**

2.9 Google Translate API.....**Error! Bookmark not defined.**
2.10 Wordnet.....**Error! Bookmark not defined.**
2.10.1 Hiponim.....**Error! Bookmark not defined.**
2.10.2 Hipernim.....**Error! Bookmark not defined.**
2.11 Evaluasi.....**Error! Bookmark not defined.**
BAB 3 METODOLOGI PENELITIAN.....**Error! Bookmark not defined.**
3.1 Tipe Penelitian.....**Error! Bookmark not defined.**
3.2 Strategi Penelitian.....**Error! Bookmark not defined.**
3.3 Partisipan Penelitian.....**Error! Bookmark not defined.**
3.4 Lokasi Penelitian.....**Error! Bookmark not defined.**
3.5 Pengumpulan Data.....**Error! Bookmark not defined.**
3.6 Perancangan Algoritme.....**Error! Bookmark not defined.**
3.7 Teknik Pengujian dan Analisis.....**Error! Bookmark not defined.**
3.8 Penarikan kesimpulan dan saran.....**Error! Bookmark not defined.**
BAB 4 PERANCANGAN ALGORITME.....**Error! Bookmark not defined.**
4.1 Perancangan diagram alir proses.....**Error! Bookmark not defined.**
4.2 Perhitungan Manual.....**Error! Bookmark not defined.**
4.2.1 Manualisasi tanpa*query expansion*.....**Error! Bookmark not defined.**
4.3 Perhitungan Manual Menggunakan *query expansion*..... **Error! Bookmark not defined.**
4.4 Perancangan Pengujian.....**Error! Bookmark not defined.**
4.4.1 Pengujian *k-fold*.....**Error! Bookmark not defined.**
4.4.2 Pengujian variasi *Threshold*.....**Error! Bookmark not defined.**
4.4.3 Pengujian *Query* yang Ditambahkan.....**Error! Bookmark not defined.**
BAB 5 HASIL.....**Error! Bookmark not defined.**
5.1 Batasan Implementasi.....**Error! Bookmark not defined.**
5.2 Implementasi Algoritme *Preprocessing*.....**Error! Bookmark not defined.**
5.2.1 Implementasi Kode Program Tokenisasi.....**Error! Bookmark not defined.**
5.2.2 Implementasi Kode Program *Filtering*.....**Error! Bookmark not defined.**
5.2.3 Implementasi Kode Program *Stemming*.....**Error! Bookmark not defined.**
5.3 Implementasi Algoritme *Query Expansion*.....**Error! Bookmark not defined.**
5.3.1 Implementasi Kode Program *Translate* Dokumen..... **Error! Bookmark not defined.**
5.3.2 Implementasi Kode Program Search*QueryExpansion*..... **Error! Bookmark not defined.**

5.3.3 Implementasi Kode Program Pengolahan Kata Ekspansi... **Error! Bookmark not defined.**

5.3.4 Implementasi Kode Program Penambahan .....**Error! Bookmark not defined.**

5.4 Implementasi Algoritme Naïve Bayes .....**Error! Bookmark not defined.**

5.4.1 Implementasi Kode Program Pencarian *Term* Latih ..... **Error! Bookmark not defined.**

5.4.2 Implementasi Kode Program Pencarian *Term* Setiap Kelas**Error! Bookmark not defined.**

5.4.3 Implementasi Kode Program Menghitung *Likelihood* ..... **Error! Bookmark not defined.**

5.4.4 Implementasi Kode Program Menghitung *Posterior* ..... **Error! Bookmark not defined.**

5.4.5 Implementasi Kode Program Memilih kelas .....**Error! Bookmark not defined.**

5.5 Implentasi Algoritme Menghitung Akurasi .....**Error! Bookmark not defined.**

BAB 6 PEMBAHASAN DAN ANALISIS .....**Error! Bookmark not defined.**

6.2 Pengujian Perhitungan Akurasi dengan *K-Fold* .....**Error! Bookmark not defined.**

6.3 Pengujian Variasi *Threshold* Kata yang Ditambahkan.**Error! Bookmark not defined.**

6.3.1 Permasalahan *Translate* saat Penambahan *Query* ..... **Error! Bookmark not defined.**

6.3.2 Permasalahan Penambahan *Query* Tidak Sesuai Konteks . **Error! Bookmark not defined.**

6.4 Pengujian Jenis Kata yang Ditambahkan .....**Error! Bookmark not defined.**

6.4.1 Penambahan Hiponim .....**Error! Bookmark not defined.**

6.4.2 Penambahan Hipernim .....**Error! Bookmark not defined.**

BAB 7 PENUTUP .....**Error! Bookmark not defined.**

7.1 Kesimpulan.....**Error! Bookmark not defined.**

7.2 Saran .....**Error! Bookmark not defined.**

DAFTAR PUSTAKA .....**Error! Bookmark not defined.**

LAMPIRAN .....**Error! Bookmark not defined.**

DAFTAR TABEL

Tabel 2.1 Aturan Stemmer Sastrawi ..... **Error! Bookmark not defined.**
Tabel 2.2 Contoh Dokumen Perhitungan Naïve Bayes ..... **Error! Bookmark not defined.**
Tabel 2.3Contoh Pembagian k-fold ..... **Error! Bookmark not defined.**
Tabel 2.4 Contoh Perhitungan k-fold ..... **Error! Bookmark not defined.**
Tabel 4.1 Dataset ..... **Error! Bookmark not defined.**
Tabel 4.2 Pembagian Dokumen Tweet sejumlah K ..... **Error! Bookmark not defined.**
Tabel 4.3Contoh Preprocessing pada dokumen 1 ..... **Error! Bookmark not defined.**
Tabel 4.4 Contoh Perhitungan Nilai tf ..... **Error! Bookmark not defined.**
Tabel 4.5 Term Unik dan nilai tf ..... **Error! Bookmark not defined.**
Tabel 4.6 Jumlah Term Unik (|V|) dan Jumlah Kata Pada Kategori ..... **Error! Bookmark not defined.**
Tabel 4.7 Perhitungan Likelihood Kata "Rupiah" ..... **Error! Bookmark not defined.**
Tabel 4.8 Likelihood Term Dokumen Tweet 1 ..... **Error! Bookmark not defined.**
Tabel 4.9 Posterior Dokumen Tweet 1 ..... **Error! Bookmark not defined.**
Tabel 4.10 Likelihood Term Dokumen Tweet 1 ..... **Error! Bookmark not defined.**
Tabel 4.11 Perancangan Pengujian k-fold ..... **Error! Bookmark not defined.**
Tabel 4.12 Perancangan Pengujian Threshold ..... **Error! Bookmark not defined.**
Tabel 4.13Pengujian Jenis Query yang Ditambahkan ..... **Error! Bookmark not defined.**
Tabel 6.1 Hasil Perhitungan Akurasi Dengan K-Fold ..... **Error! Bookmark not defined.**
Tabel 6.2 Dokumen Dengan Kesamaan Term ..... **Error! Bookmark not defined.**
Tabel 6.3 Dokumen Dengan Kemungkinan Multi Kelas ..... **Error! Bookmark not defined.**
Tabel 6.4akurasi Dari Variasi Threshold ..... **Error! Bookmark not defined.**
Tabel 6.5Hasil Klasifikasi K2 saat Threshold = 0,1 ..... **Error! Bookmark not defined.**
Tabel 6.6 Hasil Klasifikasi K2 saat Threshold = 0,2 ..... **Error! Bookmark not defined.**
Tabel 6.7 Perbandingan kata ekspansi yang ditambahkan ..... **Error! Bookmark not defined.**
Tabel 6.8 Perbedaan Translate ..... **Error! Bookmark not defined.**
Tabel 6.9 Perbedaan Translate(Lanjutan) ..... **Error! Bookmark not defined.**
Tabel 6.10Contoh Kata Ekspansi di Luar Konteks ..... **Error! Bookmark not defined.**
Tabel 6.11 Akurasi Penambahan Hiponim ..... **Error! Bookmark not defined.**
Tabel 6.12 Akurasi Penambahan Hipernim ..... **Error! Bookmark not defined.**
Tabel 6.13 Perbandingan Penambahan Query ..... **Error! Bookmark not defined.**

DAFTAR GAMBAR

Gambar 2.1 Halaman Beranda Twitter ..... **Error! Bookmark not defined.**
Gambar 2.2 Tweet Pada Twitter ..... **Error! Bookmark not defined.**
Gambar 2.3 Contoh Proses Tokenisasi ..... **Error! Bookmark not defined.**
Gambar 2.4 Contoh Proses Filtering ..... **Error! Bookmark not defined.**
Gambar 2.5 Contoh proses stemming ..... **Error! Bookmark not defined.**
Gambar 2.6 Alur Klasifikasi Teks ..... **Error! Bookmark not defined.**
Gambar 2.7 Contoh Wordnet ..... **Error! Bookmark not defined.**
Gambar 3.1 Perancangan Algoritme ..... **Error! Bookmark not defined.**
Gambar 4.1 Diagram Alir Sistem ..... **Error! Bookmark not defined.**
Gambar 4.2 Diagram Alir Proses Preprocessing ..... **Error! Bookmark not defined.**
Gambar 4.3 Diagram Alir Proses Tokenisasi ..... **Error! Bookmark not defined.**
Gambar 4.4 Diagram Alir Proses Filtering ..... **Error! Bookmark not defined.**
Gambar 4.5 Diagram Alir Proses Stemming ..... **Error! Bookmark not defined.**
Gambar 4.6 Diagram Alir query expansion ..... **Error! Bookmark not defined.**
Gambar 4.7 Diagram alir Naive Bayes ..... **Error! Bookmark not defined.**
Gambar 4.8 Translate dari Bahasa Indonesia ke Bahasa Inggris ..... **Error! Bookmark not defined.**
Gambar 4.9 Query Expansion ..... **Error! Bookmark not defined.**
Gambar 4.10 Translate ke Bahasa Indonesia ..... **Error! Bookmark not defined.**
Gambar 6.1 Akurasi Variasi Threshold ..... **Error! Bookmark not defined.**

DAFTAR KODE PROGRAM

Kode Program 5.1 Implementasi Proses Tokenisasi ..... **Error! Bookmark not defined.**
Kode Program 5.2 Implementasi Proses Filtering ..... **Error! Bookmark not defined.**
Kode Program 5.3 Implementasi Proses Stemming ..... **Error! Bookmark not defined.**
Kode Program 5.4Implementasi Proses Translate Dokumen ..... **Error! Bookmark not defined.**
Kode Program 5.5 Implementasi Proses SearchQueryExpansion ..... **Error! Bookmark not defined.**
Kode Program 5.6 Implementasi Proses Pengolahan Kata Ekspansi..... **Error! Bookmark not defined.**
Kode Program 5.7 Implementasi Porses Penambahan Kata Ekspansi .. **Error! Bookmark not defined.**
Kode Program 5.8 Implementasi Proses Pencarian Term Unik ..... **Error! Bookmark not defined.**
Kode Program 5.9 Implementasi Proses Pencarian Term..... **Error! Bookmark not defined.**
Kode Program 5.10 Implementasi Proses Menghitung Likelihood ..... **Error! Bookmark not defined.**
Kode Program 5.11 Implementasi Kode Program Menghitung Posterior.....**Error! Bookmark not defined.**
Kode Program 5.12 Implementasi Proses Pemilihan Kelas ..... **Error! Bookmark not defined.**
Kode Program 5.13 Implentasi Proses Menghitung Akurasi ..... **Error! Bookmark not defined.**

DAFTAR LAMPIRAN

LAMPIRAN A Contoh Dataset ..... **Error! Bookmark not defined.**

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pada perkembangan zaman saat ini pengguna internet semakin banyak dan tersebar meluas, salah satu penggunaan internet adalah *electronic news (e-news)* yang merupakan berita yang disebarkan dan diberitakan secara *online*. Seperti halnya berita pada media cetak maupun media-media lain, berita dalam *e-news* sangat beragam seperti olahraga, politik, hiburan dan lain sebagainya. Jika digunakan dengan sesuai dan tepat guna, maka dapat meningkatkan kualitas dan memajukan berbagai bidang seperti pendidikan, bisnis, ilmu pengetahuan, dan sosial.

Bersamaan dengan bertambahnya pengguna internet maka media sosial juga semakin diminati, salah satunya adalah Twitter yang mengalami pertambahan jumlah pengguna setiap tahunnya. Twitter adalah salah satu jenis *microblog* yang unik, di dalamnya para pengguna dapat mengirimkan pesan berisi kumpulan karakter yang disebut *tweet* dengan jumlah karakter maksimal sebanyak 140 (kwak, et al., 2010). Namun pada 7 november 2017 jumlah maksimal karakter Twitter ditambahkan menjadi 280 karakter (The Open University, 2018).

Tweet yang tersebar dan muncul dalam beranda Twitter salah satunya adalah *tweet* berita yang sangat beragam serta tidak dikelompokkan berdasarkan kategori beritanya seperti kesehatan, olahraga, teknologi, ekonomi, wisata dan lain sebagainya. Tidak adanya pengkategorian menyebabkan pengguna kesulitan jika ingin membaca atau mengambil informasi terkait kategori tertentu yang diinginkan. Contohnya jika ada pengguna yang ingin membaca atau mengambil informasi mengenai olahraga, pengguna tersebut harus mencari satu persatu *tweet* yang berkaitan dengan olahraga secara manual, cara lain yang dapat dilakukan adalah dengan mencari atau menjelajah salah satu akun Twitter yang berisi satu konten yang sama. Oleh karena itu, diperlukan sebuah sistem yang mampu mengklasifikasikan *tweet* berdasarkan informasinya, bukan berdasarkan akun Twitter.

Banyaknya jumlah *tweet* yang di-*post* mengakibatkan pertambahan jumlah data teks dalam bentuk elektronik sangat besar, bertambahnya data teks menyebabkan penumpukan data mentah yang belum dimanfaatkan atau dengan kata lain belum diolah menjadi informasi. *Text mining* menawarkan solusi untuk permasalahan tersebut, dengan menerapkan teknik-teknik *information retrieval,natural language processing, information extraction,* dan *data mining* (McDonald & Kelly, 2012). Salah satu teknik yang ada pada *text mining* adalah klasifikasi teks, yang dalam proses klasifikasinya menggunakan *machine learning*. Sebuah sistem yang mampu mengklasifikasikan secara otomatis terhadap beberapa kategori pada teks dengan Bahasa alami.

Salah satu metode klasifikasi adalah Naïve Bayes. Contoh penerapan metode Naïve Bayes adalah pada penelitian yang dilakukan oleh Perdana pada tahun 2013, menunjukkan bahwa Naïve Bayes menghasilkan performa yang baik. Menggunakan metode pengukuran akurasi dengan *precission, recall,* dan *F1 measure* menghasilkan nilai masing-masing yaitu 80%, 79%, dan 78% (Perdana,Suprapto,& Regarsari 2013). Pada penelitian lain yang dilakukan oleh Buzic dan Dobsa pada tahun 2018, menerapkan *Naive Bayes*untuk klasifikasi lirik lagu mendapatkan akurasi sebesar 88,4% dan metode pengukuran *precision, recall,* dan *F1* menghasilkan nilai masing-masing yaitu 86,9%, 95,2% dan 90,9% (Buzic & Dobša, 2018). Dari penelitian-

penelitian sebelumnya, maka dapat dipahami bahwa Naïve Bayes cocok untuk melakukan klasifikasi teks.

Pada proses klasifikasi teks terdapat beberapa permasalahan. Salah satu masalah pada proses klasifikasi teks adalah kata yang menjadi fitur tidak memiliki karakteristik yang cukup untuk menggambarkan sebuah kelas (Sriram, 2010). Pada penelitian kali ini dokumen yang digunakan adalah *tweet* yang merupakan *short text. Short text* memiliki kelemahan yaitu dapat menimbulkan kerancuan informasi (Tang, et al., 2017). Hal ini terjadi karena pendeknya teks memungkinkan ditemukan kesamaan lebih dari satu kategori ketika diklasifikasi. Pendeknya teks juga dapat mempersulit proses klasifikasi dikarenakan kata yang ada pada data uji tidak ada pada data latih, yang pada akhirnya sistem gagal mengklasifikasikan teks tersebut dengan benar. Salah satu cara untuk menangani permasalahan tersebut adalah dengan menggunakan metode *query expansion*, metode ini telah banyak digunakan pada kasus ketidaklengkapan atau ketidakcocokan dalam *information retrival* karena seringnya pengguna tidak lengkap dalam memberikan *query* (Tang, et al., 2017). Cara kerja *Query Expansion* adalah mengoptimalkan teks dengan cara menambahkan *query* atau beberapa kata yang memiliki kedekatan dengan teks misalnya sinonim, hipernim, hiponim , dan lain-lain.

Terdapat beberapa penelitian yang telah menerapkan *query expansion*. Salah satu penggunaan *query expansion* adalah penelitian yang dilakukan oleh Bagaskoro,Fauzi, & Adikara 2017, peningkatan akurasi yang dihasilkan dengan menambahkan *query expansion* yaitu sebesar 2%, sebelum menggunakan *query expansion* akurasi sistem sebesar 90% lalu menjadi 92%. Selain itu pada penelitian lain yang dilakukan oleh Tanjung,Fauzi, & Indriati 2017 menunjukkan peningkatan akurasi yang cukup signifikan dengan penambahan metode *query expansion.* Besaran akurasi tanpa menggunakan *query expansion* adalah 58% sedangkan setelah ditambahkan menjadi 76%. Maka dari itu metode *query expansion* dapat menjadi salah satu metode meningkatkan akurasi dari proses klasifikasi teks.

Ada berbagai macam *source* yang digunakan sebagai sumber *query expansion*. Salah satu *source* adalah dengan menggunakan penambahan *term* dari WordNet, WordNet merupakan sebuah kamus elektronik yang sudah dikembangkan. Dalam WordNet berisi *synonim set* (*synset*) (Li, et al., 016). Penelitian sebelumnya yang dilakukan Elberricchi,Rahmoun, & Bentaalah 2008 menggunakan WordNet sebagai sumber penambahan kata. Penambahan yang dilakukan adalah penambahan hipernim dan hiponim, pada penelitian tersebut didapatkan akurasi sebesar 66,7% untuk *dataset reuters-21578* dan 64,9% untuk *dataset 20Newsgroups*. Penelitian lain yang dilakukan oleh Mansuy dan Hilderman menunjukkan bahwa penambahan hipernim meningkatkan akurasi pada proses klasifikasi teks. Akurasi yang didapatkan setelah penambahan meningkat hingga 3,83% (Mansuy & Hilderman, 2006). Pada penelitian kali ini digunakan hipernim dan hiponim dari kata yang dimaksud dan kemudian ditambahkan di belakang kata awal.

WordNet yang digunakan pada penelitian-penelitian sebelumnya merupakan WordNet berbahasa Inggris. WordNet dengan Bahasa Indonesia sebenarnya telah dikembangkan oleh Lab Fasilkom Universitas Indonesia, WordNet Bahasa Indonesia dibangun dengan memetakan *synset* yang ada pada WordNet Bahasa Inggris ke dalam Bahasa Indonesia (Fasilkom Universitas Indonesia, 2018). Pada proses pengembangan WordNet Bahasa Indonesia bekerja sama dengan user yang telah terdaftar untuk ikut memetakan *synset*, WordNet Bahasa

Indonesia belum sepenuhnya lengkap jika dibandingkan dengan WordNet Bahasa Inggris, maka dari itu dipilih WordNet Bahasa Inggris untuk digunakan sebagai acuan dalam melakukan *query expansion.*

Penggunaan WordNet Bahasa Inggris tentunya memerlukan penanganan lebih lanjut dikarenakan data yang digunakan adalah dokumen *tweet* berbahsa Indonesia. Untuk itu sebelum melakukan proses *query expansion* maka *tweet* terlebih dahulu diterjemahkan ke dalam Bahasa Inggris, kemudian dikembalikan ke dalam Bahasa Indonesia untuk selanjutnya diproses lebih lanjut. Pada proses penerjemahan digunakanakan Google Translate API untuk menjaga konsistensi dan kebenaran hasil terjemahan.

Diharapkan dengan adanya penelitian ini, mampu membantu dan mempermudah pengguna media sosial Twitter untuk mencari dan membaca *tweet* sesuai dengan kategori yang diinginkan.

1.2 Rumusan masalah

Dari latar belakang yang telah dijabarkan sebelumnya, maka rumusan masalah dari penelitian ini adalah bagaimana pengaruh penggunaan *query expansion* terhadap hasil klasifikasi dengan Naïve Bayes.

1.3 Tujuan

Dari rumusan masalah yang telah dijabarkan sebelumnya, maka tujuan yang ingin dicapai dalam penelitian ini adalah menguji serta melakukan analisis terhadap hasil dan tingkat akurasi dari klasifikasi berita pada Twitter menggunakan metode Naïve Bayes dengan *query expansion* berbasis WordNet.

1.4 Manfaat

Manfaat yang didapatkan dari hasil penelitian adalah sebagai berikut:

1. Meningkatkan wawasan pembaca dari permasalahan *query expansion.*
2. Agar mempermudah pembaca untuk mengambil informasi dari *tweet* yang telah diklasifikasikan
3. Agar mempermudah pembaca untuk membaca *tweet* sesuai dengan kategori yang diinginkan.
4. Sebagai referensi atau pembanding untuk penelitian-penelitian *query expansion* selanjutnya dengan substansi penelitiannya.

1.5 Batasan masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Dokumen yang digunakan pada penelitian klasifikasi berita pada Twitter menggunakan Naïve Bayes dan *query expansion* berbasis WordNet adalah *tweet* berbahasa Indonesia.
2. Jumlah data yang digunakan sebagai *dataset* berjumlah 400 *tweet*

3. Tidak menjelaskan perbandingan hasil klasifikasi Naïve Bayes dan metode klasifikasi lainnya.
4. Sumber *query expansion* yang digunakan adalah dengan mengambil *synset* dari WordNet.
5. *Query expansion* hanya dilakukan pada data uji.
6. WordNet yang digunakan sebagai acuan pengambilan hipernim dan hiponim adalah WordNet berbahasa Inggris.
7. Proses penerjemahan yang dilakukan sepenuhnya menggunakan Google Translate API dan tidak dilakukan secara manual.
8. Data yang digunakan adalah data sekunder yang diperoleh dari skripsi Bagaskoro, Fauzi, dan Adikara 2017.
9. Kelas hasil klasifikasi berjumlah delapan kategori, yaitu *travel*, otomotif, olahraga, teknologi, hiburan, kuliner, ekonomi, dan kesehatan.
10. Penelitan yang dilakukan ini tidak dapat menangani teks singkatan dan kata tidak baku.

1.6 Sistematika pemBahasan

Sistematika Penulisan dalam skripsi ini adalah sebagai berikut:

BAB I Pendahuluan

Bab ini memuat latar belakang permasalahan yang mendasari penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah penelitian, dan sistematika pemBahasan

BAB II Tinjauan Pustaka

Bab ini berisis uraian mengenai dasar teori dan referensi yang menjadi dasar penelitian. Berisi penjabaran Twitter,Naïve Bayes,klasifikasi teks, dan *query expansion.*

BAB III Metodologi

Bab ini memuat metode yang dilakukan pada penelitian ini. Di dalamnya terdapat studi literatur, pegumpulan data, implementasi metode Naïve Bayes dan *query expansion* berupa *flowchart*, dan perhitungan manualisasi.

BAB IV Perancangan Algoritme

Bab ini memuat rancangan dari algoritme yang digunakan, serta berisi implementasi yang menggambarkan tiap – tiap proses dari algoritme.

BAB V Hasil

Bab ini memuat hasil dari penelitian dan Penulisan kode program yang dilakukan pada penelitian.

BAB VI Analisis dan Pembahasan

Bab ini memuat pengujian, pembahasan, dan penjabaran dari hasil yang diperoleh dalam penelitian ini serta memberikan pemahaman baru dari hasil yang didapat.

BAB VII Penutup

Bab ini memuat kesimpulan dan saran dari penelitian.

BAB 2 LANDASAAN KEPUSTAKAAN

Landasan kepustakaan memuat penjabaran dan pembahasan mengenai teori, konsep, model, dan atau sistem dari pustaka ilmiah yang dilakukan sebelumnya. Selain itu pada bab ini juga dibahas terkait dengan Twitter, metode Naïve Bayes, *query expansion,* WordNet, dan klasfisikasi teks.

2.1 Kajian Pustaka

Pada kajian pustaka ini dibahas dan dijelaskan mengenai penelitian-penelitian sebelumnya yang telah dilakukan. Penelitian yang dipaparkan merupakan penelitian dengan objek Twitter berbahasa Indonesia serta ulasan singkat dan dilakukan klasifikasi terhadap dokumen yang ada di dalamnya. Baik itu merupakan sentimen analisis maupun klasifikasi lebih dari dua kategori

Peneltian yang dilakukan sebelumnya adalah penelitian yang dilakukan oleh Perdana,Suprapto, dan Regarsari pada tahun 2013 dengan objek Twitter berbahasa Indonesia dan diklasifikasikan menggunakan metode Naïve Bayes. Pada penelitian tersebut dokumen yang digunakan diambil menggunakan Twitter API, kemudian dilakukan proses *preprocessing* terhadap data latih dan data uji. Selanjutnya dilakukan klasifikasi dengan Naïve Bayes sehingga menghasilkan kelas-kelas berupa kategori. Pada penelitian tersebut pengujian dilakukan menggunakan *precision, recall* dan *F-measure* dengan hasil masing-masing 79% , 80%, dan 78% (Perdana, Suprapto, & Rekyan 2013).

Penelitian lain dilakukan oleh Elberricchi 2008, menggunakan WordNet sebagai kamus penambahan kata. Penambahan yang dilakukan adalah penambahan hipernim dan hiponim. Penelitian tersebut menggunakan data yang telah ada sebelumnya, yaitu *reuters-21578* dan *20Newsgoup* yang merupakan data untuk tujuan penelitian. Pada penelitian tersebut didapatkan akurasi sebesar 66,7% untuk *dataset reuters-21578* dan 64,9% untuk *dataset 20Newsgroups* (Elberrichi,Rahmoun dan Bantalaah, 2008).

2.2Twitter

Twitter merupakan salah satu media sosial yang unik, yang mana di dalamnya para pengguna dapat mengirimkan dan menerima pesan singkat dengan panjang maksimal 140 karakter yang disebut dengan *tweet*(O'Reilly & Milstein, 2009). Namun pada 7 November 2017 jumlah maksimal karakter Twitter ditambahkan menjadi 280 karakter (The Open University, 2018). *Tweet* yang muncul pada halaman beranda tiap pengguna berbeda karena adanya sistem *follow* dan *following*, sistem ini menampilkan *tweet* dari pengguna lain yang diikuti, sedangkan mengikuti terhadap suatu akun memunculkan *tweet* akun tersebut pada *timeline followers*(sebutan untuk orang yang mengikuti suatu akun).

Fitur-fitur yang terdapat pada Twitter, diantaranya adalah (O'Reilly & Milstein, 2009):

1. **Halaman Utama (*Home*)**

   Pada halaman utama pengguna dapat melihat *tweet* yang di-*post*ing atau dikirmkan oleh pengguna lain yang telah kita ikuti. Halaman utama dari tiap pengguna berbeda tergantung akun yang diikuti.

2. **Profil (*Profile*)**

   Pada halaman ini pengguna dapat melihat profil atau data diri pribadi. Selain itu pada halaman ini pengguna dapat melihat *tweet* yang telah dikirim atau dibuat sebelumnya

3. ***Followers***

   Followers atau pengikut adalah orang yang telah menjadikan kita sebagai teman dan berlangganan *tweet* kita. *Tweet* orang yang diikuti ditampilkan pada halaman utama.

4. ***Following***

   Following merupakan kebalikan dari *followers*. Proses *following* adalah dengan mengikuti pengguna lain. *Tweet* dari pengguna yang diikuti muncul pada halaman utama

5. ***Favorite***

   Merupakan fungsi untuk menandai *tweet* sebagai favorit kita sehingga tidak hilang pada halaman sebelumnya.

6. ***Mentions***

   Fungsi *mentions* adalah untuk menandai pengguna yang sedang atau diajak bicara, sehingga *tweet* atau balasan percakapan bisa menuju pengguna yang dimaksud.

7. **Pesan langsung (*Direct Message*)**

   Fungsi ini memungkinkan pengguna untuk melakukan pengiriman pesan secara personal, cara kerjanya hampir sama dengan SMS**.** Pesan hanya dapat dilihat oleh orang yang dialamati untuk dikirim pesan.

8. ***Hashtag***

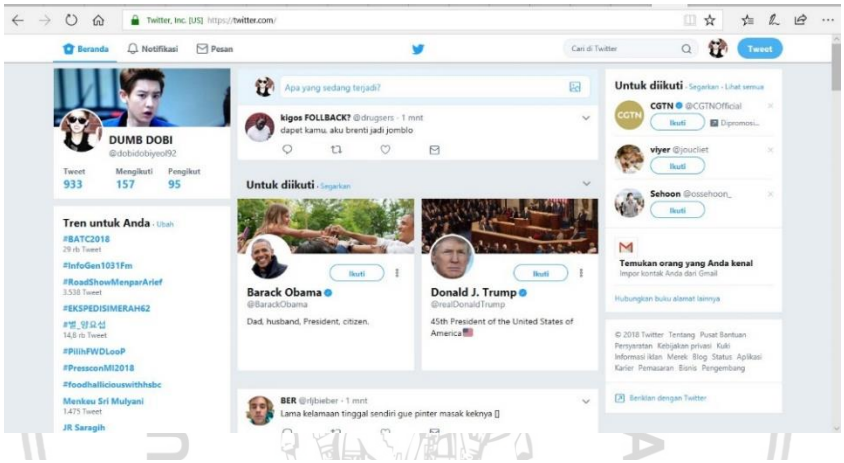
   Hashtag atau yang bisa disimbolkan dengan "#" merupakan fitur yang dituliskan di depan sebuah topik tertentu dalam *tweet.* Bertujuan agar pengguna lain dapat mencari topik yang sejenis yang ditulis oleh pengguna lain. *Hashtag* tidak dapat digunakan sebagai acuan untuk klasifikasi *tweet* karena hanya menampilkan data yang ada pada *hashtag* tersebut saja.

9. **Topik Terkini (*Trending Topic*)**

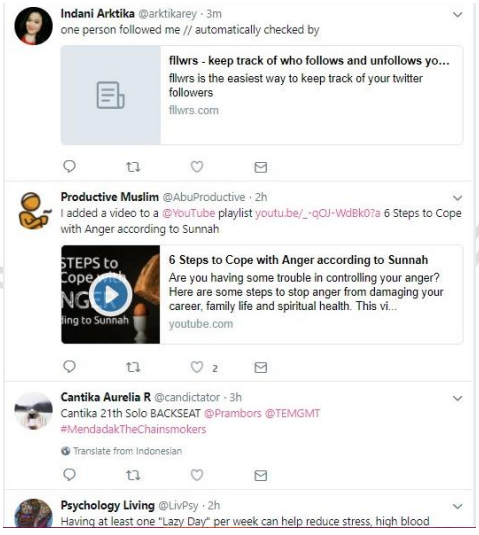
   Merupakan salah satu fitur yang ada pada Twitter yang cara kerjanya menampilkan *tweet* atau topik yang sedang dibicarakan atau viral pada suatu waktu tertentu dan bersamaan.

10. ***List***

    Fitur ini menyediakan kemampuan bagi pengguna untuk mengelompokkan pengguna yang mereka ikuti ke dalam suatu grup atau kelompok. Tujuan dari pengolompokan agar memudahkan pengguna untuk dapat melihat secara keseluruhan *username* yang telah diikuti.



Gambar 0.1 Halaman Beranda Twitter



Gambar 0.2 *Tweet* Pada Twitter

2.3 Text Mining

Secara luas *text mining* dapat diartikan sebagai interaksi antara user dan sekumpulan dokumen dari waktu ke waktu dengan menggunakan *tools* yang tepat. Seperti halnya pada *data mining* tujuan dari *text mining* adalah untuk mencari atau menggali informasi dari sekumpulan data, pada *text mining* data

yang digunakan berupa dokumen text (Feldman & Sanger, 2007). Perbedaan antara *text mining* dan *data mining* juga terletak pada jenis data, text merupakan salah satu data yang tidak terstruktur sehingga memerlukan penanganan atau tahap yang lebih banyak dan kompleks.

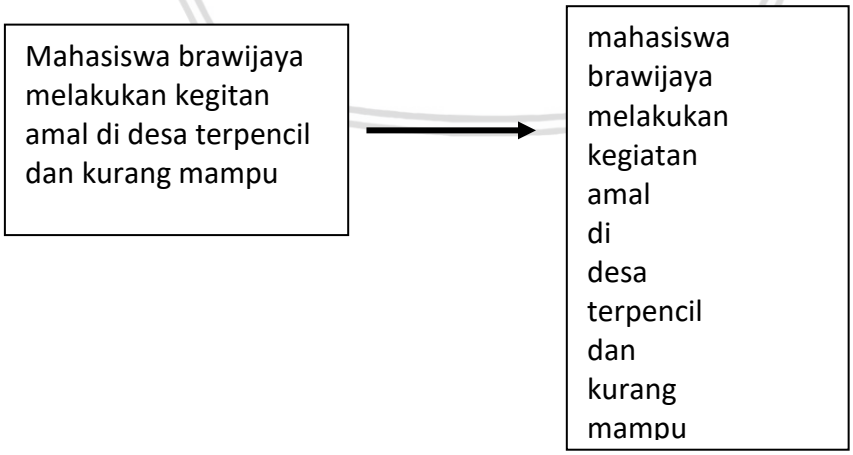
Text Mining terinspirasi dan mengacu pada *data mining*. Maka dari itu seperti pada *data mining, text mining* juga sangat bergantung pada beberapa tahapan. Tahapan-tahapan tersebut yaitu *preprocessing,* algoritme untuk menemukan pola, mesin untuk melakukan pembelajaran (*Machine Learning algorithm*) (Feldman & Sanger, 2007)

2.4 Text Preprocessing

Seperti halnya pada *data mining*, *text mining* juga memerlukan fase *preprocessing*. Walaupun keduanya melakukan *preprocessing* pada data yang diolah, teknik yang digunakan berbeda. Proses *text mining* yang baik didasarkan pada metode *preprocessing* yang baik. Tujuannya adalah mengubah data teks yang sebelumnya tidak terstruktur menjadi data yang siap diolah (Feldman & Sanger, 2007). Ada beberapa tahapan dalam *preprocessing* yaitu *extraction* (tokenisasi), *stopword removal (Filtering),* dan *stemming* (Vijayaran, et al., 2015).

2.4.1 Tokenisasi

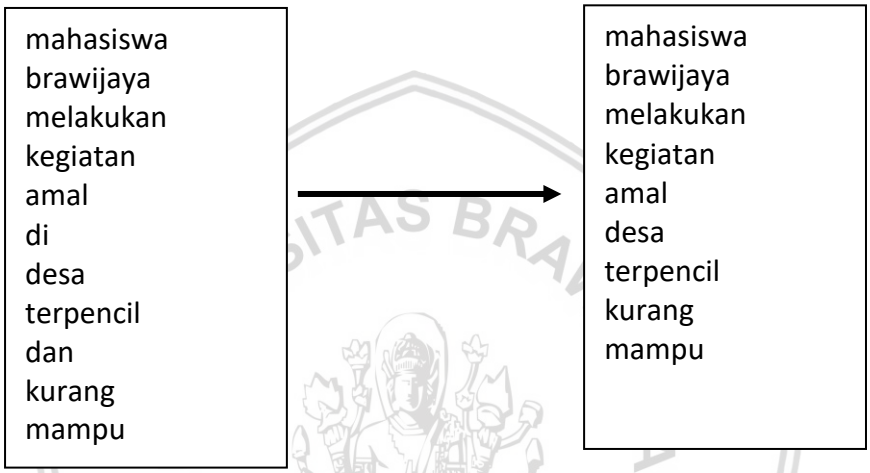
Tahap tokenisasi bertujuan untuk mengubah isi dari dokumen menjadi potongan per kata (Vijayaran,Ilamathi, dan Nithya., 2015). Pada tahapan ini kalimat-kalimat yang ada dipecah menjadi potongan-potongan kata, serta dalam proses ini pula dihapuskan tanda baca dan karakter-karakter unik lainnya. Proses tokenisasi digambarkan pada Gambar 2.3 di bawah ini.



Gambar 0.3 Contoh Proses Tokenisasi

2.4.2 Filtering

Filtering atau yang biasa disebut *stopword removal* bertujuan untuk membuang kata-kata yang kurang penting pada dokumen. Kata yang dimaksud adalah kata yang kemunculannya tidak memengaruhi proses analisis teks. Dengan melakukan pembuangan kata yang kurang penting, maka mengurangi dimensi dari fitur (Vijayaran, et al., 2015). Kata yang dianggap tidak penting biasanya adalah kata sambung dan kata depan. Contoh proses *filtering* digambarkan pada Gambar 2.4.



Gambar 0.4 Contoh Proses *Filtering*

2.4.3 Stemming

Pada tahapan ini adalah tahapan menjadikan kata dasar dari kata yang ada pada dokumenn yang diolah. Contohnya adalah kata "sambungan", "tersambung", dan "menyambungkan" diganti dengan kata dasar yang sama yaitu kata "sambung". Tujuan dari tahap ini adalah untuk menghilangkan macam-macam imbuhan sehingga dapat mengurangi jumlah kata, meningkatkan akurasi kecocokan *term* yang tujuan akhirnya adalah mengurangi penggunaan memori dan waktu komputasi (Vijayaran, et al., 2015).

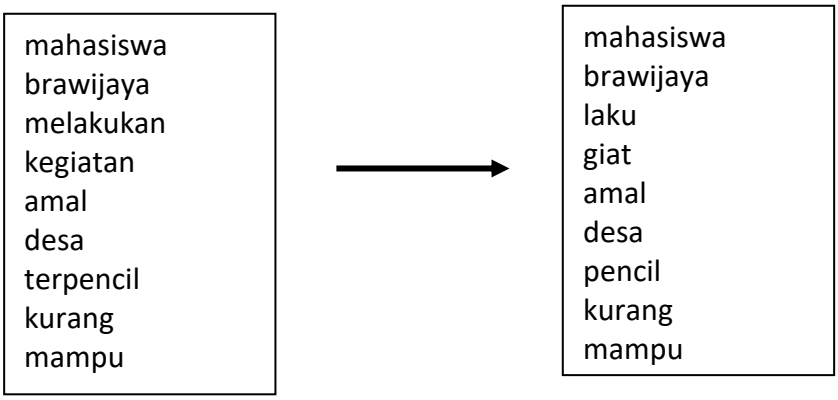
Dalam proses *stemming* ada beberapa permasalahan antara lain (Asian, 2007):

11. *Word-Sense Ambiguity* , maksudnya satu kata dapat memiliki dua makna (misalnya adalah kata homonim), dan berasal dari dasar yang berbeda:
    - Berikan = Ber-ikan
    - Berikan = Beri-kan

12. *Over stemming*, terjadi *stemming* berlebihan sehingga tidak memunculkan makna, dapat dicegah dengan mencocokkan kata pada kamus, jika kata dasar ditemukan maka proses *stemming* dihentikan:
    - Berikan = Ber-i-kan
13. *Under Stemming*, muncul hasil yang salah dikarenakan saat proses pencarian kata dasar pada kamus ditemukan kata dasar lain saat pemenggalan
    - Mengecek = menge- cek (seharusnya)
    - Mengecek = meng-ecek (kata ecek juga ada pada kamus kata dasar)
14. Ketergantungan pada kamus kumpulan kata dasar, kamus digunakan untuk mencegah *over stemming* dan *under stemming*. Sehingga jika kamus yang digunakan memiliki kekurangan maka *over stemming* dan *uder stemming* juga dapat terjadi.
15. Pengguna Bahasa yang inkonsisten dalam menentukan stem,
    - Adalah = ada / Adalah = kata dasar
    - Bagian = bagi / Bagian = kata dasar

    Inkonsistensi dari pengguna Bahasa dapat menjadi masalah dalam proses *stemming*.
16. Permasalahan pada kata bentuk jamak, kesulitan dalam menentukan kata dasar
    - Buku – buku = buku
17. Permasalahan kata serapan dari Bahasa asing, kata serapan seringkali sukar untuk di *stemming.*
    - Mengakomodir = meng-akomodir
18. Kesalahan Penulisan, seringkali pengguna Bahasa melakukan kesalahan dalam melakukan Penulisan kata. Hal ini menyebabkan kata tersebut tidak dapat de*stemming*.
    - Penambahahan = ? , seharusnya kata “penambahan”
19. Nama dari benda, seharusnya tidak di-*stemming*.
    - Abdullah = abdul-lah , seharusnya tetap Abdullah
20. Akronim, seharusnya tidak di-*stemming* namun pada prosesnya tetap di-*stemming*
    - Pemilu = pe-milu, seharusnya kata pemilu tidak di*stemming*.

Berikut digambarkan proses *stemming* pada Gambar 2.5



Gambar 0.5 Contoh proses *stemming*

2.4.3.1 *Stemmer* Sastrawi

Sastrawi *stemmer* merupakan salah satu *library* untuk melakukan proses *stemming*. *Stemmer* ini dibuat sesederhana mungkin agar mempermudah dalam penggunaannya (Librian, 2016). *Stemmer* sastrawi memiliki beberapa aturan dalam melakukan proses *stemming*, berikut dituliskan aturan *stemming* sastrawi pada Tabel 2.1.

Tabel 0.1 Aturan *Stemmer Sastrawi*

No	Aturan	Contoh
1a	berV -> ber + V	Beradu -> adu
1b	berV -> be + rV	Berambut -> rambut
2	berCAP -> ber + CAP	Bersuara -> suara
3	berCAerV -> ber + CAerV (C != "r")	Berdaerah -> daerah
4	Belajar -> bel + ajar	Belajar -> ajar
5	beC1erC2 -> be-C1erC2 (C1 ! = { 'r' | l })	Bekerja -> kerja Beternak -> ternak
6a	terv -> ter + V	Terasing -> asing
6b	terV -> te + rV	Teraup -> raup
7	terCerV -> ter + CerV (C != "r")	Tergerak -> gerak
8	terCP -> ter-CP where C != 'r' and P != 'er'	Terpuruk -> puruk
9	teC1erC2 -> te-C1erC2 where C1 != 'r'	Teterbang -> terbang
10	me{l|r|w|y}V -> me-{l|r|w|y}V	Melipat-> lipat Meringkas-> ringkas Mewarnai -> warna

Tabel 0.1 Aturan *Stemmer* Sastrawi (Lanjutan)

No	Aturan	Contoh
16	meng{g|h|q} -> meng-{g|h|q}	Menggila -> gila Menghajar -> hajar Mengqasar -> qasar
17a	mengV -> meng-V	Mengudara-> udara
17b	mengV -> meng-kV	Mengupas -> kupas
15	men{V} -> me-t{V}	Menangkap –> tangkap
11	mem{b|f|v} -> mem-{b|f|v}	Membangun-> bangun Memfitnah -> fitnah Memvonis -> vonis
12	mempe{r|l} -> mem-pe	Memperbarui -> baru Mempelajari -> ajar
13a	mem{rV|V} -> mem{rV|V}	Meminum -> minum
13b	mem{rV|V} -> me-p{rV|V}	Memukul -> pukul
14	men{c|d|j|z} -> men-{c|d|j|z}	Mencinta -> cinta Mendua -> dua Menjauh –> jauh

Sumber:
https://github.com/sastrawi/sastrawi/blob/master/tests/SastrawiFunctionalTest/Stemmer/StemmerTest.php#L49

Selain aturan–aturan di atas, terdapat juga aturan modifikasi terhadap aturan yang tercantum di tabel dan dapat dilihat langsung pada tautan di atas. Hal ini dikarenakan Bahasa Indonesia lebih kompleks dibandingkan dengan Bahasa lainnya.

2.5 Term Weighting

Term Weighting merupakan metode untuk melakukan pembobotan kata yang ada pada dokumen. Bobot tiap kata menyatakan kepentingan kata tersebut dalam dokumen, dengan kata lain kata tersebut penting atau tidaknya dapat direpresentasikan dengan bobot yang diberikan. Salah satu penanda bahwa kata tersebut penting dapat dilihat dari kemuncuan kata atau *frequency* (Fauzi, Arifin, & Yuniarti., 2014).

2.5.1 Term Frequency

Term Frequecy merupakan salah satu metode untuk menentukan bobot dari sebuah kata. Metode ini tergolong sederhana karena bobot kepentingan sebuah kata direpresentasikan dari jumlah kemunculan kata pada dokumen tersebut (Fauzi, Arifin, & Yuniarti., 2014). Semakin sering kata tersebut muncul, maka kata tersebut dianggap penting. Persamaan dari *term frequency* dapat dituliskan dengan Persamaan 2.1.

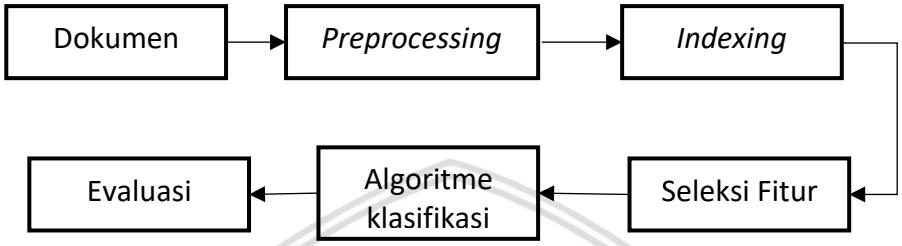
$$TF(d\,,t) = f(d\,,t) \tag{2.1}$$

Keterangan:

$f(d\,,t)$ = frekuensi munculnya kata *t* dalam dokumen *d*.

2.6 Klasifikasi Teks

Klasifikasi teks merupakan salah satu bagian penting dari *text mining*. Klasfikasi adalah mengelompokkan teks yang memiliki karakteristik yang sama atau mirip ke dalam sebuah kelas tertentu. Biasanya klasifikasi teks dilakukan secara manual, namun terus dikembangkan dengan pengetahuan pakar agar dapat melakukan klasifikasi otomatis dengan pengelompokan beberapa kategori (Korde, 2012).



Gambar 0.6 Alur Klasifikasi Teks

Sumber: Korde (2012)

Pada Gambar 2.6 digambarkan proses klasifikasi teks. Langkah pertama yang dilakukan adalah mempersiapkan dokumen yang diolah, dokumen mentah tersebut terlebih dahulu dilakukan proses *preprocessing.* Proses selanjutnya yaitu pembobotan pada setiap kata hasil *preprocessing*. Setelah melakukan *prepocessing* dan pembobotan kemudian dipilih fitur mana saja yang digunakan. Tahap selanjutnya adalah proses kalsifikasi dengan algoritme yang telah ditentukan, setelah seluruh proses selesai dikerjakan maka dilakukan pengukuran performa dari sistem klasifikasi yang dibuat.

2.7 Naïve Bayes *Classifier*

Naïve Bayes merupakan salah satu metode klasifikasi yang popular dan banyak digunakan, algoritme ini banyak digunakan karena mudah diterapkan dan memiliki waktu pemrosesan yang cepat sehingga lebih efektif dan efisien (Taheri & Mammadov, 2013). Metode Naïve Bayes didasarkan pada teorema Bayes namun setiap fitur berdiri sendiri atau bersifat independen.

Naïve Bayes menggunakan seluruh *probalitas prior* (probabilitas dokumen terhadap kategori) dan melakukan klasifikasi berdasarkan *probabilitas posterior.* Dengan menerapkan teorema Bayes maka ada atau tidak adanya fitur tertentu dari kelas tidak menimbulkan masalah dalam proses klasifikasi karena setiap fitur bersifat independen. Berikut adalah persamaan Naïve Bayes yang dituliskan dalam Persamaan 2.2 (Vidhya & Aghila, 2010)

$$P(c|d) = \frac{P(d|c)\,P(c)}{P(d)} \qquad (2.2)$$

Keterangan:

$P(c|d)$ = Peluang *poste*rior (c adalah kelas dan d adalah dokumen)

$P(c)$ = Peluar *prior* dari kelas

$P(d|c)$ = Peluang *likelihood*

$P(d)$ = Peluang *prior* dokumen

2.7.1 Multinomial Naïve Bayes

Multinomial Naïve Bayes merupakan salah satu representasi Naïve Bayes yang menggunakan *contional probability.* Untuk memenuhi *conditional propability* maka digunakan frekuensi dari setiap kemunculan kata pada kelas. Berikut adalah persamaan Multinomial Naïve Bayes yang dituliskan dalam Persamaan 2.3 (Rahman, et al., 2017)

$$P(w_i|c_j) = \frac{count(w_i,c_j)+1}{(\sum_{w \in V} count(w_i,c_j)+|V|)} \quad (2.3)$$

Keterangan:

$Count(W_i,C_j)$ = Jumlah kata atau *query* dalam sebuah kelas, diberi nilai +1 untuk menghindari nilai 0

$\sum_{w \epsilon V} count(w,c_j)$ = Jumlah kata (*w*) dari keseluruhan *term* yang ada pada kelas *c*

V = Jumlah kata unik

Setelah didapatkan nilai *probabilitas* setiap kata terhadap setiap kelas, maka selanjutnya adalah menghitung *probabilitas* dokumen. Nilai probabilitas dokumen terhadap sebuah kelas (*posterior*) didapatkan dengan mengalikan nilai probabilitas setiap kata dalam satu kelas dengan nilai *prior*. Rumus *posterior* dapat dilihat pada Persamaan 2.4 (Rahman, et al., 2017).

$$P(c_i|d_j) = P(c)\ x\ P(w_1|c_j)\ x\ P(w_2|c_j) \ldots x\ P(w_n|c_j) \quad (2.4)$$

Keterangan:

$P(c)$ = probabilitas *prior* kelas *c*

$P(c_i|d_j)$ = probabilitas dokumen *d* terhadap kelas *c*

$P(w_1|c_j)$ =probabilitas kata *w* terhadap kelas *c*

Mengenai contoh perhitungan Naïve Bayesdapat dilihat pada Tabel 2.2, dengan menggunakan dokumen sembarang berjumlah lima dokumen.

Tabel 0.2 Contoh Dokumen Perhitungan Naïve Bayes

	No	Isi Dokumen	Kelas
Training	1	Malang Kediri Malang	C
	2	Malang Malang Blitar	C
	3	Malang Brazil	C
	4	Tulungagung Singapura Malang	X
Test	5	Malang Malang Malang Bandung Singapura	?

Langkah pertama yang dilakukan adalah menghitung nilai *prior*:

$$P(c) = \frac{N_c}{N}$$

- $P(C) = \frac{3}{4}$
- $P(X) = \frac{1}{4}$

Selanjutnya yaitu menghitung *conditional probability/Likelihood*:

- $P(Malang|C) = \frac{(5)+1}{(8)+(6)} = \frac{3}{7}$
- $P(Singapura|C) = \frac{(0)+1}{(8)+(6)} = \frac{1}{14}$
- $P(Malang|X) = \frac{(1)+1}{(3)+6} = \frac{2}{9}$
- $P(Singapura|X) = \frac{(1)+1}{(3)+(6)} = \frac{2}{9}$

Setelah didapatkan nilai *likelihoodnya*, selanjutnya adalah menghitung nilai *posterior*-nya dengan cara mengalikan nilai *prior* dengan setiap nilai *likelihood*-nya. Rumus perhitungan dapat dilihat pada Persamaan 2.4:

$$P(c|d5) = Nilai\ Prior\ kelas\ x\ nilai\ likelihood$$

$$P(C|d5) = \frac{3}{4}\ x\ \left(\frac{3}{7}\right)^3\ x\ \left(\frac{1}{14}\right) = 0{,}009$$

$$P(X|d5) = \frac{1}{4}\ x\ \left(\frac{2}{9}\right)^4 = 0{,}0005$$

Dari perhitungan *posterior* di atas kemudian dipilih kelas dengan *probabilitas* terbesar yaitu kelas *C*, sehingga dokumen 5 (d5) diklasifikasikan sebagai kelas *C.*

2.8 *Query Expansion*

Query expansion merupakan sebuah metode untuk melakukan perluasan *query* . Perluasan *query* dilakukan dengan menambahkan kata lain yang memiliki keterkaitan dengan *query* (Zhang, Deng& Li., 2009). Perluasan atau penambahan *query* bertujuan untuk menangani permasalahan yang sering muncul dalam klasifikasi teks pendek, yang mana kata-kata di dalamnya tidak ada dalam data latih. Hal itu menyebabkan proses klasifikasi tidak berjalan dengan baik atau bahkan tidak terklasifikasi karena tidak ada kata yang cocok dalam data latih.

Banyak metode yang dapat diterapkan untuk menentukan kata yang ditambahkan dalam *query*, salah satunya adalah dengan penambahan sinonim dari kata yang terdapat pada *query*. Sinonim ini dapat diambil dari kamus (*Thesaurus*), *word books,* dan kamus lainnya. Hal ini dapat membantu proses klasifikasi karena perbendaharaan kata dalam *query* bertambah.

2.9 Google Translate API

Google Translate API merupakan salah satu dari *Cloud Machine Learning* yang dapat digunakan secara gratis (*freeware*). Google Translate API bekerja secara dinamis, mampu melakukan translasi dari ratusan Bahasa (Google Cloud Platform, 2018) .

2.10 Wordnet

WordNet adalah sebuah kamus yang merupakan pengembangan dari kamus kata *thesaurus*, dikembangkan oleh Universitas Princeton dan berisi pemodelan leksikal Bahasa Inggris. Di dalam WordNet terdapat *synset (synonym set)*, di dalamnya terdapat kata-kata yang memiliki keterkaitan dan relasi *semantic* dalam berbagai kategori kata seperti kata benda, kata kerja, kata sifat, dan kata keterangan (Zhang,Deng& Li., 2009). Dalam *synset*, satu kata dengan kata lainnya dapat saling menggantikan tanpa merusak atau mengubah makna dari kalimat dalam satu konteks yang sama.

Dengan menggunakan WordNet yang merupakan sebuah kamus kata yang telah dikembangkan maka dapat dilakukan penambahan *query* untuk memperbaiki sebuah teks tertentu sehingga sesuai dengan konsep kalimat tertentu (Zhang,Deng& Li., 2009) . Pada penelitian ini digunakan *synset* berupa relasi kata hipernim dan hiponim untuk ditambahkan ke dalam *query*. Berikut adalah contoh penggunaan WordNet yang digambarkan pada Gambar2.8.

```
Wn good –synsn
Sense 1
Good => advantage, vantage
Sense 2
good, goodness => morality
Sense 3
Good,goodness => quality
Sense 4
Commodity, trade good,good => artifact, artifact
```

Gambar 0.7 Contoh Wordnet

Sumber: Zhang (2009)

Pada Gambar 2.8 di atas merupakan contoh penerapan WordNet. Kata "*good*" memiliki empat makna jika digunakan sebagai kata benda yaitu "*Advantage,vantage*" , "*morality*" , "*quality*" , "*artifact, artifact*". Maksudnya kata "*good*" dapat digantikan oleh kata-kata dalam *synset* dengan syarat tetap dalam konteks kalimat yang sama.

2.10.1 Hiponim

Hiponim merupakan salah satu relasi antar kata yang mana satu kata merupakan bentuk khusus dari kata umum (*subordinate*) (Miller, et al., 1993). Contohnya adalah kata cemara merupakan hiponim dari pohon, dan kata pohon merupakan hiponim dari tumbuhan. Maksudnya adalah kata cemara merupakan salah satu dari jenis pepohonan dan pohon sendiri merupakan salah satu jenis tanaman.

2.10.2 Hipernim

Hipernim merupakan kebalikan dari hiponim, hipernim merupakan relasi antar kata yang mana satu kata merupakan bentuk umum dari kata khusus (Miller, et al., 1993). Contohnya adalah kata kendaraan merupakan hipernim dari mobil.

2.11 Evaluasi

Proses evaluasi merupakan tahapan yang bertujuan untuk mengukur seberapa besar akurasi dari penelitian yang dilakukan. Ada berbagai macam cara untuk mengukur akurasi dari sebuah metode yang diterapkan dalam penelitian.

Pada penelitian kali ini menggunakan metode *cross validation / K-Fold.* Metode ini merupakan metode yang sering digunakan untuk melakukan penghitungan akurasi sebuah proses klasifikasi. *Dataset* yang ada dibagi menjadi *k fold* dengan pembagian data yang sama. Selanjutnya dilakukan perulangan atau iterasi terhadap semua *fold*, masing masing *fold* sejumlah *k* digunakan sebagai data uji dan sisanya dijadikan data latih (Anguita, et al., 2012). Ilustrasi metode evaluasi *cross validation* dapat dilihat pada Tabel 2.3 berikut.

Tabel 0.3Contoh Pembagian *k-fold*

No	Dokumen	Kelas	*K*
1	Dokumen1	A	1
2	Dokumen2	B	
3	Dokumen3	A	2
4	Dokumen4	B	
5	Dokumen5	A	3
6	Dokumen6	B	
7	Dokumen7	A	4
8	Dokumen8	B	
9	Dokumen9	A	5
10	Dokumen10	B	

Pada setiap *fold* diusahakan memiliki pembagian jumlah dokumen yang merata. Setelah didapatkan pembagian tersebut maka *K1* menjadi data uji sedangkan dokumen yang ada pada *K*-lainnya menjadi data latih. Perhitungan evaluasi untuk setiap *K* menggunakan akurasi. Nilai akurasi yang sebelumnya telah didapatkan selanjutnya dijumlah dan dibagi dengan jumlah *k* untuk mencari rata-rata akurasi, yang menjadi akurasi akhir dari sistem. Contoh perhitungan dapat dilihat pada Tabel 2.4 dengan data sembarang.

Tabel 0.4 Contoh Perhitungan *k-fold*

K data uji	Nilai akurasi setiap K
K1	70%
K2	82%
K3	67%
K4	56%
K5	85%
Akurasi	$\frac{360}{5} = 72\%$

BAB 3 METODOLOGI PENELITIAN

Pada bab ini menjelaskan langkah-langkah dalam menyusun dan melakukan penelitian. Tahapan-tahapan yang dilakukan diantaranya studi pustaka, pengumpulan data, perancangan algoritme, serta pengujian dan analisis, yang terakhir adalah kesimpulan dan saran.

3.1 Tipe Penelitian

Pada penelitian ini, dilakukan penelitian dengan tipe *non-implementatif*. Penelitian tipe *non-implementatif* berfokus kepada pengkajian terhadap sebuah kasus atau kondisi tertentu, yang pada akhirnya menghasilkan analisis ilmiah. Metode yang dapat digunakan untuk menghasilkan analisis ilmiah dapat berupa eksperimen, kuesioner, observasi, survei, studi kasus, dan metode-metode yang lainnya. Jika dilihat dari kegiatan yang dilakukan saat penelitian, maka dapat dikatakan bahwa pendekatan yang dilakukan pada penelitian ini adalah penelitian *non-implementatif analitik*. Penelitian dengan pendekatan tersebut memiliki tujuan untuk menjelaskan relasi atau hubungan antara komponen dalam penelitian dengan kondisi tertentu yang sedang diteliti.

3.2 Strategi Penelitian

Pada penelitian ini, strategi yang digunakan adalah eksperimen. Eksperimen merupakan jenis penelitian yang bertujuan untuk mencari hubungan sebab akibat. Eksperimen juga sepenuhnya dikontrol oleh peneliti serta pengembang yang bertanggung jawab, biasanya eksperimen dilakukan di dalam laboratorium. Metode eksperimen yang digunakan dalam penelitian ini adalah Naïve Bayes dan *query expansion.*

3.3 Partisipan Penelitian

Pada penelitian kali ini, partisipan yang terlibat adalah akun-akun Twitter yang sudah resmi (*Official*) lebih spesifiknya dalah akun Twitter miliki Kompas dan Detik.com. Alasan memilih partisipan adalah, kedua akun tersebut merupakan akun resmi yang telah diverifikasi oleh pihak Twitter. Selain itu *tweet* yang diberikan berisi berbagai macam kategori sehingga sangat cocok untuk dijadikan sebagai partisipan dalam penelitian ini.

3.4 Lokasi Penelitian

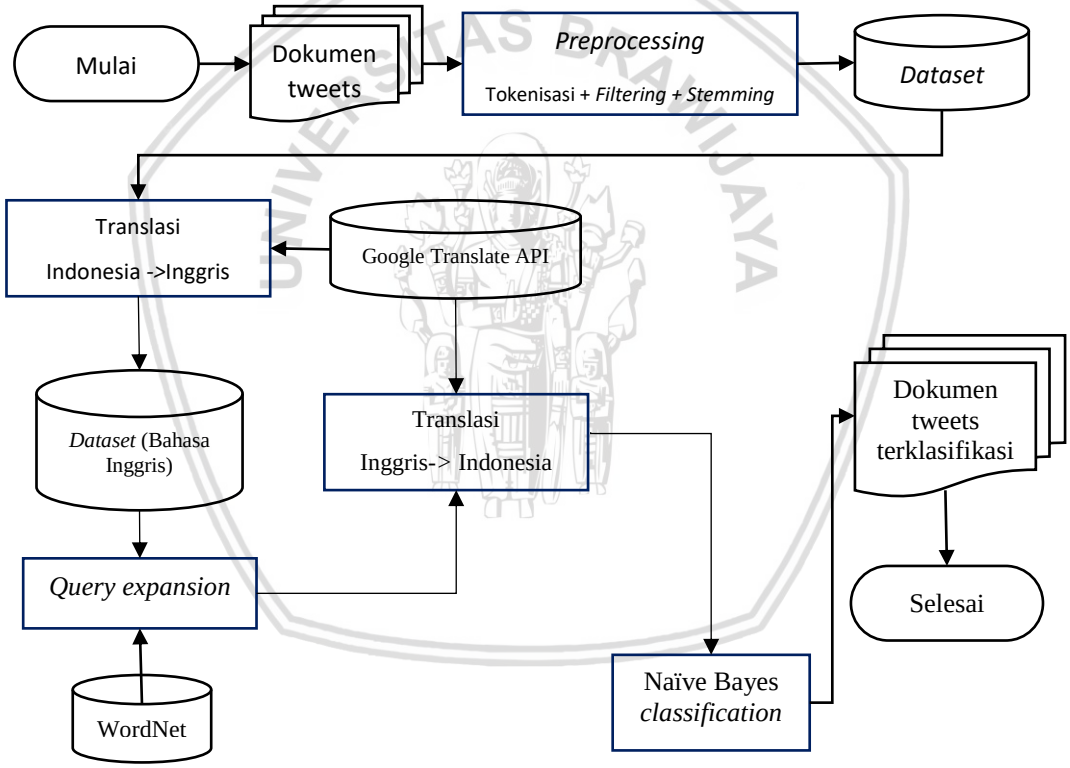
Penelitian ini dilakukan di Laboratorium Komputasi Cerdas yang bertempat di Fakultas Ilmu Komputer (Filkom) Universitas Brawijaya. Lokasi dipilih dikarenakan di dalamnya dapat dilakukan eksperimen terkait dengan klasifikasi *tweet* berita yang ada pada Twitter dan *query expansion* berbasis WordNet.

3.5 Pengumpulan Data

Metode pengumpulan data yang digunakan pada penelitian ini adalah dengan pengumpulan data sekunder. *Dataset* yang digunakan dalam penelitian ini adalah data yang telah ada sebelumnya dan telah digunakan dalam penelitian oleh Bagaskoro (2017) dengan jumlah *dataset* sebanyak 400 data. Data *tweet* tersebut didapatkan dari akun Twitter Kompas dan Detik.com (Bagaskoro, Fauzi, dan Adikara 2017)

3.6 Perancangan Algoritme

Pada tahap ini merupakan tahap perancangan algoritme yang digunakan dalam penelitian ini. Selain itu pada tahap perancangan algoritme menggambarkan jalannya algoritme yang digunakan.



Gambar 0.1 Perancangan Algoritme

3.7 Teknik Pengujian dan Analisis

Pada tahap ini adalah menguji algoritme yang telah dirancang untuk penelitian ini. Algortime klasifikasi diterapkan ke dalam data uji yang telah ada sebelumnya. Tujuan dari pengujian adalah menemukan kesalahan yang mungkin terjadi dalam proses klasifikasi. Setelah didapatkan hasil pengujian kemudian dilanjutkan

dengan analisis terhadapa hasil yang didapat, bagaimana sebuah kondisi terjadi dan bagaimana cara mengatasi kesalahan yang muncul. Tujuan analisis juga melihat apakah penelitian sudah mencapai tujuan yang diinginkan.

Pengujian yang dilakukan pada penelitian ini ada tiga jenis pegujian:

1. Pengujian akurasi sistem klasfikasi dengan Naïve Bayes tanpa melakukan *query expansion*.
2. Pengujian Treshold pada jumlah kata yang diekspansi terhadap akurasi sistem.
3. Pengujian jenis kata ekspansi yang ditambahkan, yaitu hipernim atau hiponim

3.8 Penarikan kesimpulan dan saran

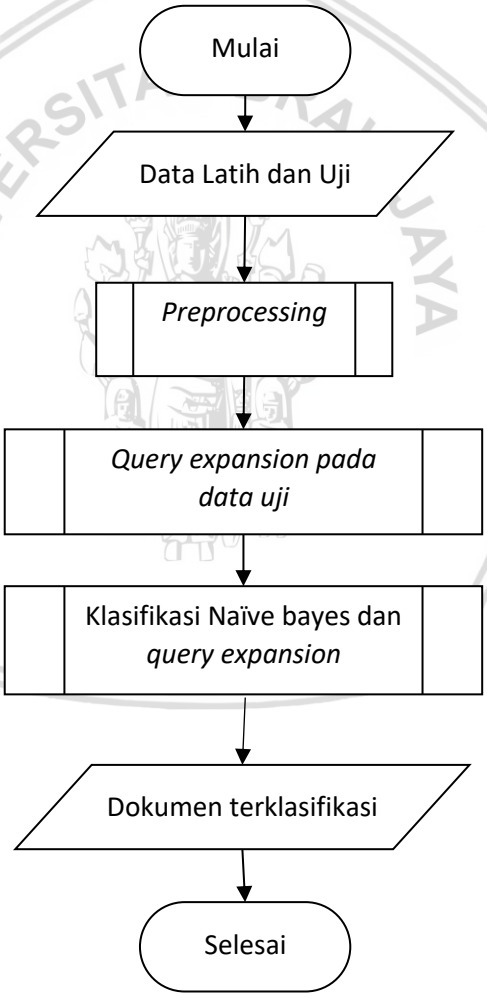
Kesimpulan dari penelitian dapat diambil setelah semua tahap dan langkah penelitian terselesaikan, langkah tersebut adalah perancangan algoritme, pengujian dan analisis. Kesimpulan berisi hasil pengujian dan analisis hasil. Selanjutnya adalah saran, saran bertujuan untuk memberikan masukan kepada peneliti lain yang ingin melakukan penelitian dengan tema yang sama.

BAB 4 PERANCANGAN ALGORITME

Pada bab ini membahas dan menjelaskan mengenai perancangan algoritme yang diterapkan dalam penelitian berjudul "Klasifikasi Berita Pada Twitter Menggunakan Metode Naive Bayes Dan Query Expansion Hipernim-Hiponim". Pada bab ini berisi diagram alir proses setiap langkah algoritme.

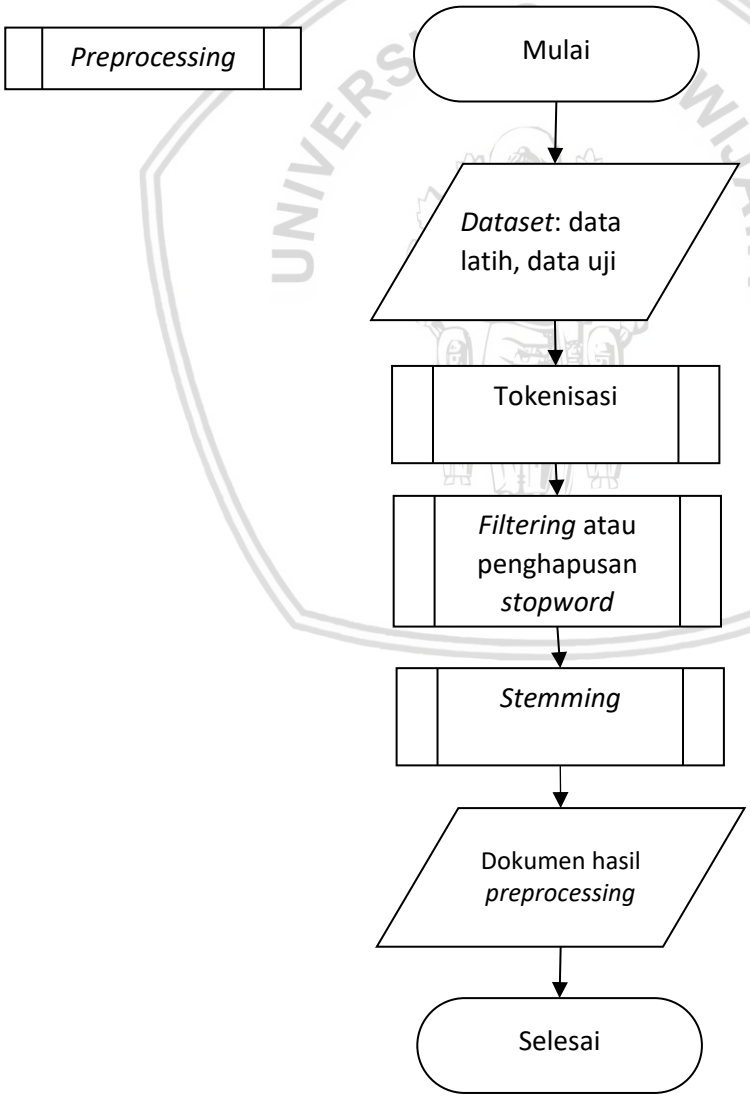
4.1 Perancangan diagram alir proses

Pada diagram alir yang menggambarkan perancangan sistem, terdapat tiga proses utama yaitu *preprocessing,* proses penambahan *query (query expansion)*, dan proses klasifikasi menggunakan metode Naïve Bayes. Dari ketiga proses utama tersebut terdapat proses-proses di dalamnya yang dijabarkan lebih lanjut. Secara keseluruhan proses yang berjalan dapat digambarkan sebagaimana Gambar 4.1.



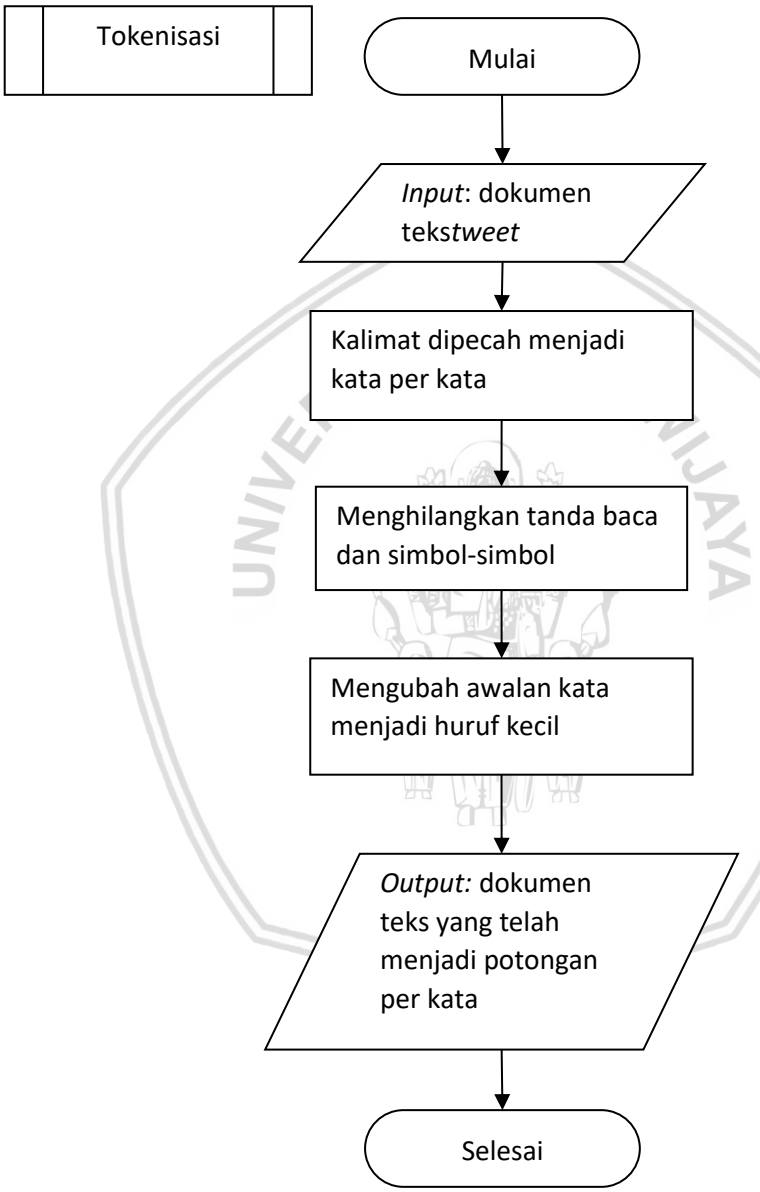
Gambar 0.1 Diagram Alir Sistem

Langkah pertama ada mengambil data yang digunakan sebagai data uji dalam proses klasifikasi. Selanjutya semua data mengalami *preprocessing* (data latih dan data uji), tahap ini bertujuan untuk menyiapkan teks agar dapat diproses lebih lanjut. Di dalam *preprocessing dataset* yang berupa teks, dihilangkan kata yang tidak penting (*stopword*) kemudian juga dilakukan proses penghilangan kata imbuhan agar menjadi bentuk kata dasarnya, selain itu juga menghilangkan tanda baca yang tidak digunakan sebagai fitur pada penelitian ini. Proses selanjutnya adalah melakukan *query expansion* untuk menambahkan *query* asli dengan kata yang merupakan hiponim dan hipernimnya menggunakan metode *query expansion*. Setelah *query* berhasil ditambahkan selanjutnya dilakukan proses klasifikasi terhadap data uji berdasarkan data latih yang telah disiapkan sebelumnya.Berikut ini adalah diagram alir dari *preprocessing*yang digambarkan pada Gambar 4.2.



Gambar 0.2 Diagram Alir Proses Preprocessing

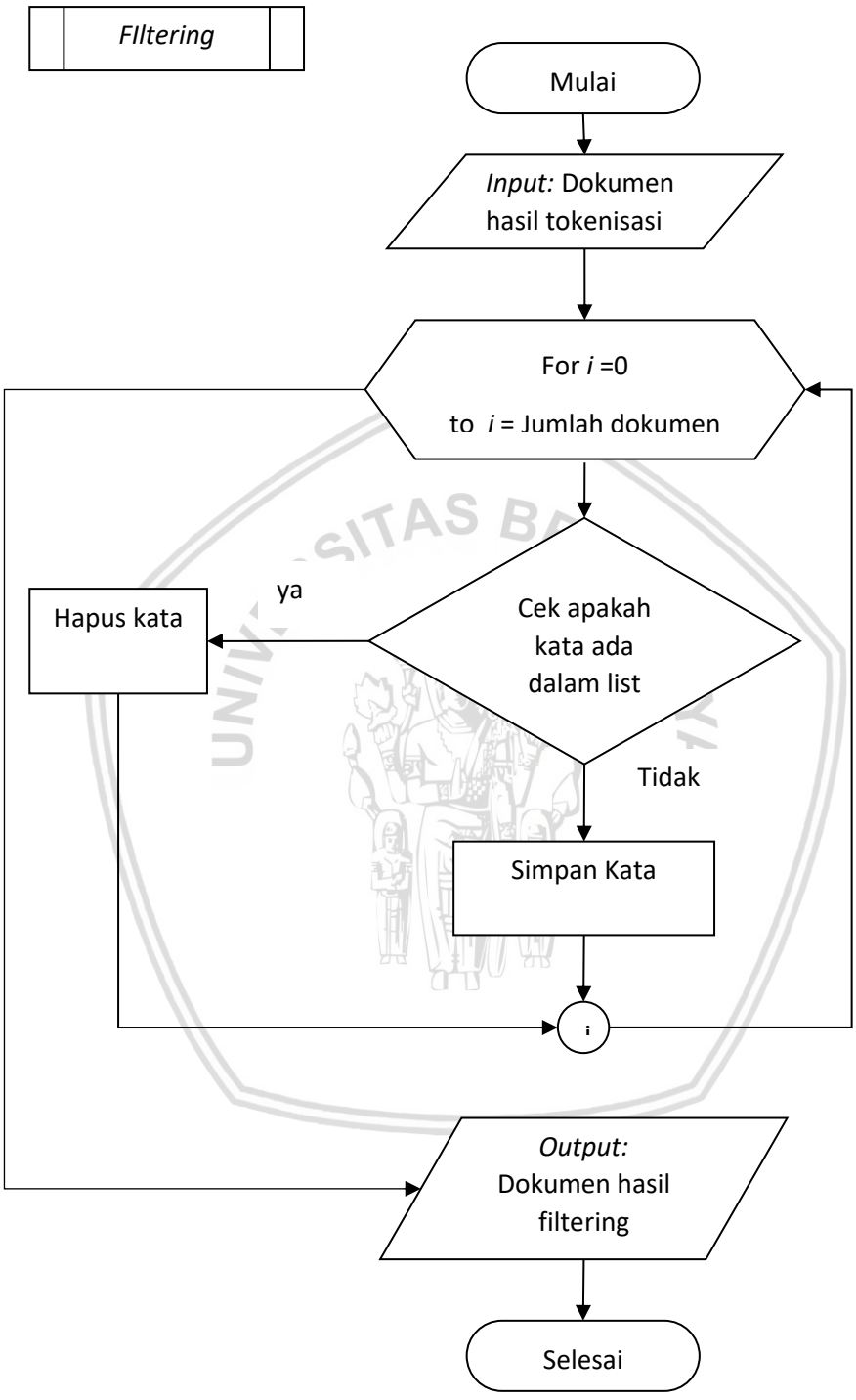
Pada proses *preprocessing* tahapan yang dilakukan adalah tokenisasi, *filtering* dan *stemming.* Tahap *preprocessing* bertujuan untuk mengolah data teks yang sebelumnya tidak terstruktur menjadi lebih terstruktur, masing-masing proses *preprocessing* digambarkan pada diagram-diagram di bawah ini. Untuk diagram tokenisasi digambarkan pada Gambar 4.3.



Gambar 0.3 Diagram Alir Proses Tokenisasi

Pada Gambar 4.3 menjelaskan alur dalam proses tokenisasi, data awal berupa dokumen teks *tweet*. Dokumen berisi kalimat yang kemudian dipecah menjadi potongan per kata. Selain memotong kata juga dihilangkan tanda baca. Pada proses ini semua kata diubah menjadi *lower case* atau awalan tiap kata diubah

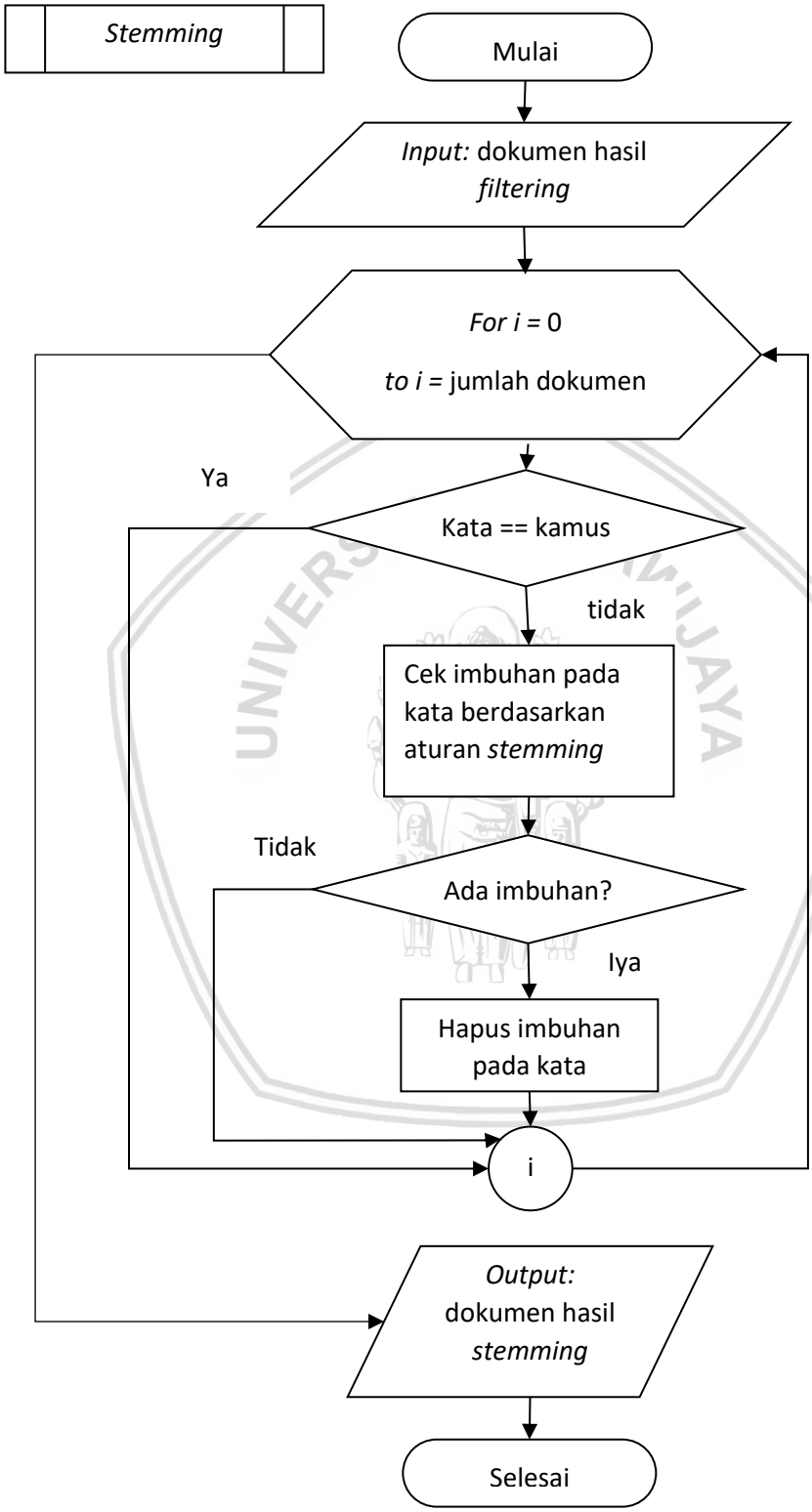
menjadi huruf kecil. Hasil proses tokenisasi kemudian disimpan kembali menjadi dokumen teks untuk selanjutnya diproses lebih lanjut.



Gambar 0.4 Diagram Alir Proses *Filtering*

Pada Gambar 4.4 menunjukkan proses *filtering,* setiap kata yang ada pada dokumen secara *loop* dicek apakah kata tersebut masuk dalam *liststopword*. Jika

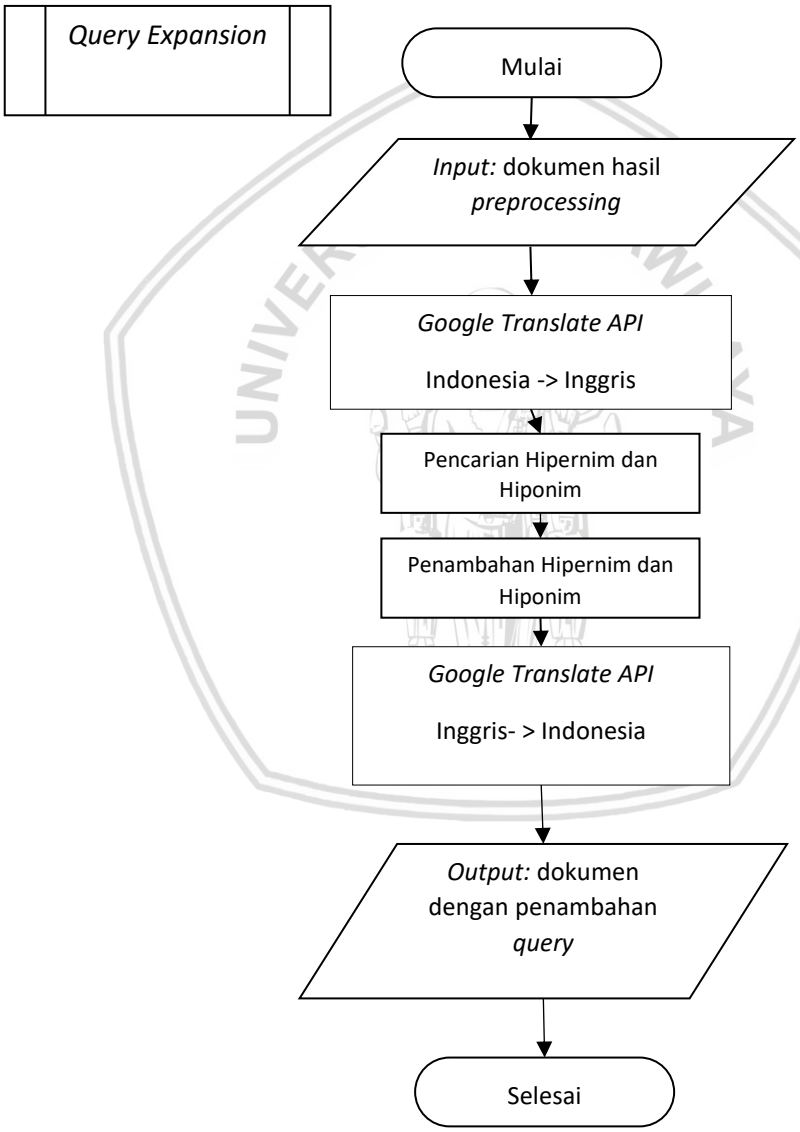
ada maka kata tersebut dihapus dari dokumen, jika tidak maka kata tersebut disimpan ke dalam dokumen baru untuk selanjutnya diproses lebih lanjut.



Gambar 0.5 Diagram Alir Proses *Stemming*

Pada Gambar 4.5, digambarkan proses *stemming*. Proses *stemming* bertujuan untuk mengembalikan kata menjadi kata dasar. Jika ada imbuham maka dihapuskan. Langkah pertama yang dilakukan adalah mengecek apakah kata ada dalam kamus.Jika ada maka langsung disimpan, jika tidak ada maka selanjutnya dicek apakah kata tersebut memiliki imbuhan. Jika memiliki imbuhan maka selanjutnya imbuhan tersebut dihapus dan kata disimpan.

Setelah melakukan proses *preprocessing* selanjutnya adalah proses *query expansion*. Proses ini memerlukan WordNet sebagai kamus yang berisi *synset* dari WordNet diambil hiponim dan hipernim dari kata. Selanjutnya adalah diagram alir dari proses *query expansion* yang digambarkan pada Gambar 4.6.

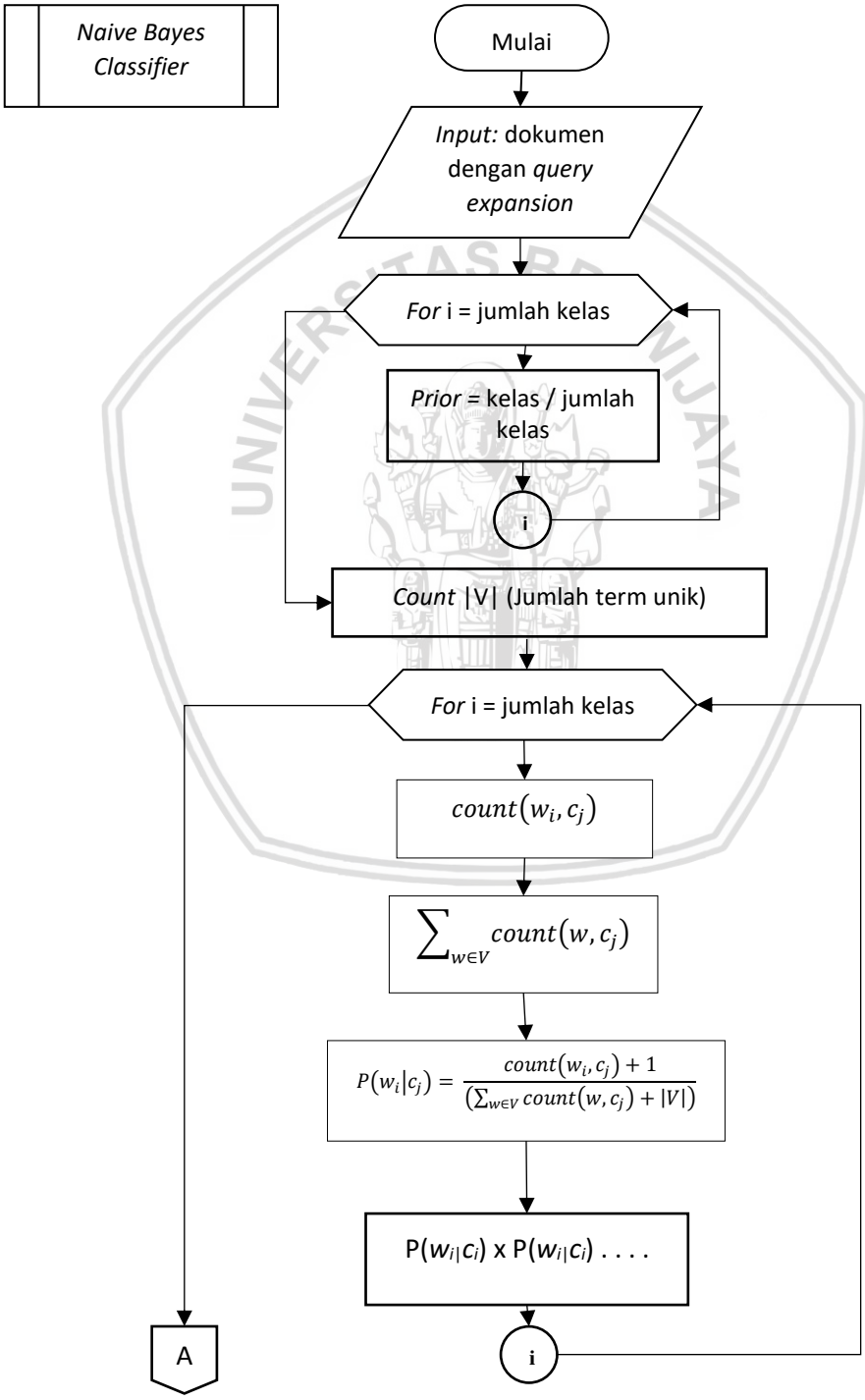


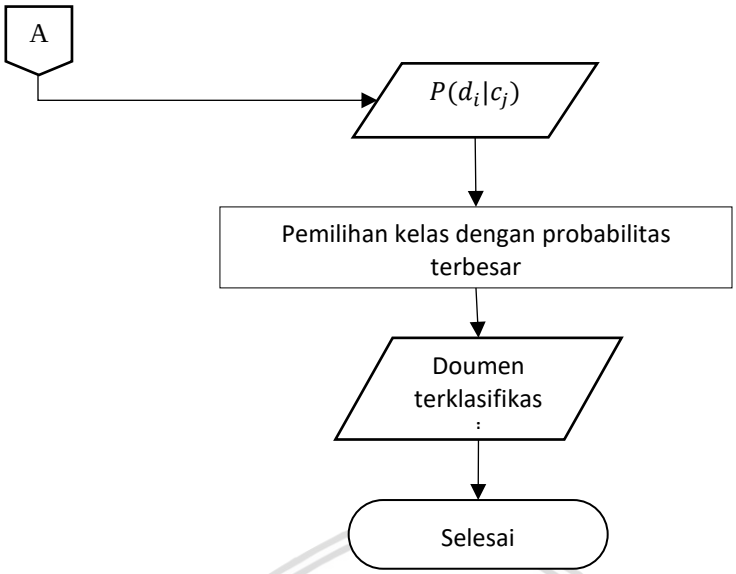
Gambar 0.6 Diagram Alir *query expansion*

Pada Gambar 4.6 proses dimulai dengan dokumen yang telah melalui tahap *prepocessing.* Dokumen tersebut kemudian diterjemahkan ke dalam Bahasa

Inggris terlebih dahulu, karena WordNet menggunakan Bahasa Inggris. Setelah diterjemahkan selanjutnya adalah mencari hipernim dan hiponim dari tiap-tiap *term* yang ada pada dokumen. Selanjutnya dilakukan penambahan *query* dengan menambahkan hipernim dan hiponim dari kata yang telah dicari sebelumnya. Terakhir dokumen dikembalikan ke Bahasa Indonesia untuk diolah dan dilakukan proses klasifikasi.

Kemudian klasifikasi menggunakan metode Naïve Bayes, diagram alir dari proses klasifikasi Naïve Bayesdigambarkan pada Gambar 4.7.





Gambar 0.7 Diagram alir Naive Bayes

Pada Gambar 4.7 digambarkan proses klasifikasi Naïve Bayes *multinomial*. Proses dimulai dengan melakukan perhitungan nilai *prior* yang merupakan probabilitas kemunculan kelas dibandingkan dengan jumlah seluruh kelas, perhitungan *prior* melakukan perulangan sebanyak kelas yang ada. Selanjutnya menghitung jumlah *term* unik yang ada pada data latih *(V)* kata unik merupakan seluruh *term* dari seluruh kelas. Langkah berikutnya, perulangan untuk setiap *term* (w_i) yang ada pada dokumen, dihitung jumlah kemunculan kata tersebut dalam tiap-tiap kelas (c_j). Nilai yang dihitung selanjutnya adalah jumlah kata yang ada pada setiap kelas, kata yang dihitung bukan kata unik melainkan keseluruhan kata yang ada pada kelas. Setelah didapatkan nilai-nilai tersebut maka selanjutnya adalah menghitung *probabilitas* setiap kata terhadap setiap kelas yang ada (*likelihood*). Nilai *likelihood* yang didapatkan kemudian dikalikan antara satu kata dengan kata lainnya dalam satu dokumen dan dikalikan *poste*rior dari kelas, sehingga dihasilkan probabilitas dokumen terhadap tiap-tiap kelas yang ada (*posterior*). Dari *posterior* yang ada kemudian diambil yang memiliki nilai paling besar dan menjadi kelas dari dokumen tersebut.

4.2 Perhitungan Manual

Manualisasi dilakukan dengan menghitung dan mendapatkan nilai yang nantinya menjadi acuan dan data untuk sistem yang dibangun. Manualisasi dilakukan dengan dokumen latih yang digunakan sebagai acuan perhitungan klasifikasi dan data uji.

4.2.1 Manualisasi tanpa*query expansion*

Manualisasi yang dilakukan berikut ini adalah proses klasifikasi *tweet* berita pada Twitter tanpa menerapkan *query expansion*. Pada proses manualisasi ini digunakan *dataset* yang ada pada Tabel 2.2

Tabel 0.1 *Dataset*

No	Tweet	Kategori
1	Awali Tahun 2016, Rupiah Berpeluang Menguat	Ekonomi
2	Tunda Pungutan Dana Ketahanan Energi	Ekonomi
3	Ini 5 Mata Uang Paling Jeblok Tahun 2015	Ekonomi
4	Getarkan DWP 2015, Dipha Barus Gandeng Rinni Wulandari	Entertaintment
5	Ayahanda Indra Bekti Tutup Usia	Entertaintment
6	Nikita Mirzani Dan Sejumlah Sessainya	Entertaintment
7	Kenalkan Sayur dan Buah Sejak Usia 6 Bulan	Kesehatan
8	5 Tanda Anda Berolahraga Lari Terlalu Banyak	Kesehatan
9	Jangan Lup 7 Area Tubuh Ini Saat Mandi	Kesehatan
10	10 Kafe di Dunia yang Berdesain Unik dan Artistik	Kuliner
11	Takeru Kobayashi, Jagoan Lomba M yang Bertubuh Atletis	Kuliner
12	Mau Burger Enak?	Kuliner
13	Evan cetak gol penyeimbang bagi MU. Skor sementara, MU 2-2 Braga. #UCL	Olahraga
14	Oscar cetak gol pada menit ke-88. Shakhtar 2-1 Chlesea. #UCL	Olahraga
15	Salah Kaprah Soal Jamur pada Bodi Mobil	Otomotif
16	Indonesia Bukan Basis Global Honda BR-V	Otomotif
17	Ponsel Android 7 Inci Lenovo Masuk Indonesia	Teknologi
18	Xperia "Malas" Jualan Android Murah	Teknologi
19	inilah Pariwisata jika berlibur raja ampat	Travel
20	4 keperluan yang harus Dilakukan Saat "berlibur"	Travel

Pada perhitungan manualisasi tidak secara jelas dipisahkan data uji dan data latih yang digunakan. Sebelumnya *dataset* pada Tabel 4.1 dibagi menjadi beberapa *k* bagian sesuai dengan metode pengujian *K-fold*, pada manualisasi ini *dataset* dibagi menjadi lima *K* dengan isi dokumen dari setiap kelas. Pembagian dokumen yang dilakukan pada manulaisasi ini dituliskan dalam Tabel 4.2.

Tabel 0.2 Pembagian Dokumen Tweet sejumlah *K*

K	Anggota *k*
1	Dok.1, Dok.4, Dok.7, Dok.10
2	Dok.2,Dok.5,Dok.8,Dok.11
3	Dok.3,Dok.6, Dok.9,Dok.12
4	Dok.13,Dok.15, Dok17,Dok.19
5	Dok.14,Dok.16,Dok.18,Dok.20

Setiap *k* berisi empat dokumen dengan kategori yang berbeda-beda. Setelah pembagian selanjutnya dilakukan *preprocessing* untuk seluruh *dataset*. Proses klasifikasi langsung dilakukan karena tidak menerapkan metode *query expansion*. Berikut merupakan proses *preprocessing* dan menghasilkan kata unik dan *term frequency*. Contoh proses *preproecssing* dapat dilihat pada Tabel 4.3.

Tabel 0.3Contoh *Preprocessing* pada dokumen 1

Tokenisasi	*Filtering*	*Stemming*	*Type*	*Term*
Awali	-	-	-	-
Tahun	-	-	-	-
Rupiah	rupiah	Rupiah	rupiah	rupiah
Berpeluang	berpeluang	Peluang	peluang	peluang
Menguat	menguat	Kuat	kuat	kuat

Pada Tabel 4.3 ditunjukkan langkah-langkah dalam melakukan proses *preprocessing* dimulai dengan *Filtering* hingga didapatkan *term* yang digunakan sebagai fitur, *term-term* tersebut adalah kata “rupiah”,”peluang”, dan “kuat”. Selanjutnya adalah melakukan pembobotan untuk setiap kata yang ada pada dokumen 1 terhadap seluruh dokumen, bobot yang digunakan adalah *term frequency (tf).* Contoh perhitungan nilai *tf* dituliskan dalam Tabel 4.4.

Tabel 0.4 Contoh Perhitungan Nilai *tf*

Term	dok1	dok2	dok3	dok4	dok5	. . .	dok20
Rupiah	1	0	0	0	0		0
Peluang	1	0	0	0	0	⋮	0
Kuat	1	0	0	0	0		0

Nilai *tf* didapatkan dengan cara mengecek apakah sebuah *term* ada pada sebuah dokumen, jika ada maka *tf* nya bernilai 1 dan jika tidak ada m *tf* nya bernilai 0. Pada perhitungan *raw tf* jumlah kemunculan kata diperhatikan. Jadi jika

sebuah kata muncul lebih dari sekali maka nilai *tf* nya sesuai dengan jumlah kemunculan *term* tersebut .

Proses *preprocessing* dan perhitungan nilai *tf* dilakukan terhadap setiap dokumen untuk selanjutnya diperoleh *term* yang dihitung nilai *tf* nya. Perhitungan nilai *tf* untuk setiap *term* selanjutnya dituliskan ke dalam Tabel 4.5.

Tabel 0.5 Term Unik dan nilai tf

Term	Dok.1	Dok.2	Dok.3	Dok.4	...	Dok. 18	Dok. 19	Dok 20
Rupiah	1	0	0	0		0	0	0
Peluang	1	0	0	0		0	0	0
Kuat	1	0	0	0		0	0	0
Tunda	0	1	0	0		0	0	0
Pungut	0	1	0	0		0	0	0
Dana	0	1	0	0		0	0	0
Tahan	0	1	0	0		0	0	0
Energy	0	1	0	0		0	0	0
Uang	0	0	1	0		0	0	0
Jeblok	0	0	1	0	...	0	0	0
Getar	0	0	0	1		0	0	0
Dwp	0	0	0	1		0	0	0
Dipha	0	0	0	1		0	0	0
Barus	0	0	0	1		0	0	0
⋮	⋮					⋮		
Ampat	0	0	0	0		0	1	0
Laku	0	0	0	0		0	0	1

Dikarenakan banyaknya *term* dan dokumen maka pada Tabel 4.5 Penulisan *term* dan nilai *tf* tidak dituliskan secara keseluruhan. Untuk detail perhitungan nilai *tf* dapat dilihat pada tautan:

https://drive.google.com/open?id=1ylF4S3ap9puegBXjWHpwuQMNZLP0UNFh

Nilai *tf* yang didapatkan kemudian digunakan untuk menghitung nilai probabilitas setiap kata terhadap kategori yang ada. Pada manualisasi ini dilakukan perhitungan probabilitas untuk dokumen 1 dengan kategori ekonomi. Saat penghitungan probabilitas dari K1 maka *tweet* yang lain digunakan sebagai data latih. Selanjutnya adalah menghitung jumlah *term* unik pada data latih dan

menghitung jumlah *term* dari setiap kategori. Jumlah *term* unik dan jumlah *term* tiap kategori ditampilkan dalam Tabel 4.6.

Tabel 0.6 Jumlah *Term* Unik (|*V*|) dan Jumlah Kata Pada Kategori

|*V*|	Ekonomi	Entertaintment	Kesehatan	Kuliner	Olahraga	Otomotif	Teknologi	Travel
61	7	8	11	9	16	9	11	6

Jumlah kata unik dapat saja berubah jika diakukan perhitungan pada *K* yang lain dikarenakan data yang sebelumnya merupakan data latih digunakan sebagai data uji dan sebaliknya. Diperlukan perulangan untuk mencari jumlah dari *term* unik. Nilai yang didapatkan selanjutnya dihitung menggunakan Persamaan 2.3

Untuk setiap kata yang terdapat pada dokumen yang menjadi data uji dihitung probabilitasnya. Pada manualisasi kali ini dokumen *tweet* yaitu pada dokumen 1. Contoh perhitungan *probabilitas* kata "rupiah" terhadap kategori Ekonomi adalah sebagai berikut:

$|V| = 61$ (Jumlah kata unik)

$count\ (wi,cj) = 0$ (Kemunculan kata "rupiah" pada kategori ekonomi)

$\sum_{we V} count(w_i,c_j) = 7$ (Jumlah kata yang terdapat pada kategori ekonomi)

P(rupiah|Ekonomi) = (0) + 1 / (7) + 61

P(rupiah|Ekonomi) = 0,0147

Nilai *likelihood* dihitung terhadap seluruh kelas yang ada, perhitungan *likelihood* kata "rupiah" terhadap seluruh kelas dituliskan pada Tabel 4.7

Tabel 0.7 Perhitungan *Likelihood* Kata "Rupiah"

Term	Ekonomi	Entertainment	Kesehatan	Kuliner	Olahraga	Otomotif	Teknologi	Travel
rupiah	$\frac{(0)+1}{(7)+61} = 0{,}0147$	$\frac{(0)+1}{(8)+61} = 0{,}0145$	$\frac{(0)+1}{(11)+61} = 0{,}0139$	$\frac{(0)+1}{(9)+61} = 0{,}0143$	$\frac{(0)+1}{(16)+61} = 0{,}0130$	$\frac{(0)+1}{(9)+61} = 0{,}0143$	$\frac{(0)+1}{(11)+6} = 0{,}0139$	$\frac{(0)+1}{(6)+61} = 0{,}0149$

Perhitungan di atas dilakukan pada setiap kata yang ada di dalam dokumen sebanyak jumlah kelas yang ada. Hasil dari perhitungan *probabilitas* dapat dilihat pada Tabel 4.8.

Tabel 0.8 Likelihood Term Dokumen Tweet 1

P(W|C) dok1	Ekonomi	Entertaintment	Kesehatan	Kuliner	Olahraga	Otomotif	Teknologi	Travel
rupiah	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
peluang	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
kuat	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149

Nilai-nilai peluang dari kata yang ada pada dokumen kemudian dikalikan dengan *term* dalam satu kelas untuk menghitung *probabilitas* dikumen terhadap kategori. Rumus perhitungan nilai perkalian setiap kata pada satu dokumen

$P(dokumen_1|Ekonomi) = P(rupiah|Ekonomi) \times P(peluang|Ekonomi) \times P(kuat|Ekonomi)$

$P(dokumen_1|Ekonomi) = (0{,}0147) \times (0{,}0147) \times (0{,}0147)$

$P(dokumen_1|Ekonomi) = 3{,}1 \times 10^{-6}$

Probabilitas dokumen *tweet* 1 terhadap seluruh kategori dapat dituliskan dalam Tabel 4.9.

Tabel 0.9 *Poste*rior Dokumen Tweet 1

Dokumen	Ekonomi	Entertaintment	Kesehatan	Kuliner	Olahraga	Otomotif	Teknologi	Travel
Dok1	$3{,}1 \times 10^{-6}$	3×10^{-6}	$2{,}6 \times 10^{-6}$	$2{,}9 \times 10^{-6}$	$2{,}1 \times 10^{-6}$	$2{,}9 \times 10^{-6}$	$2{,}6 \times 10^{-6}$	$3{,}3 \times 10^{-6}$

Hasil yang didapatkan pada klasifikasi dokumen *tweet* 1 tanpa *query expansion* belum didapatkan nilai yang dapat digunakan untuk melakukan proses klasifikasi. Hal ini membuktikan bahwa dokumen *tweet* 1 yang dijadikan data uji belum mampu diklasifikasi. Setiap kategori memilki peluang yang sama, artinya dokumen *tweet* 1 bisa dikategorikan ke dalam seluruh kelas. Permasalahan diatas terjadi karena *term* yang terdapat pada dokumen *tweet* 1 tidak terdapat pada data latih, sehingga dalam perhitungan probabilitas didapatkan nilai yang sama untuk setiap kelas.

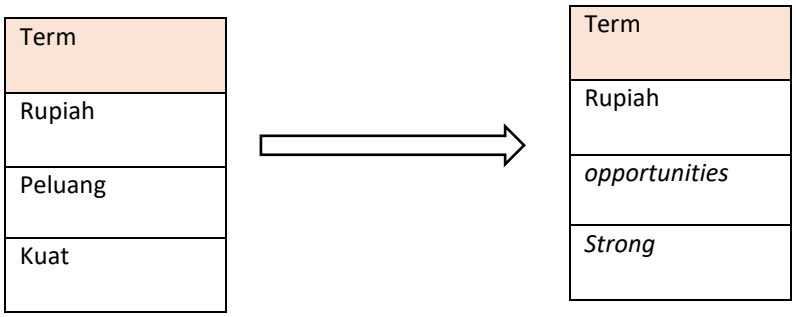
Permasalahan tersebut dapat diselesaikan dengan metode *query expansion* untuk meningkatkan jumlah *term* dan meningkatkan akurasi sistem dalam melakukan klasifikasi

4.3 Perhitungan Manual Menggunakan *query expansion*

Manualisasi menggunakan *query expansion* memilki tahapan yang sama dengan manualisasi tanpa *query expansion*, namun sebelum proses perhitungan dilakukan terlebih dahulu dokumen *tweet* 1 ditambahkan *query*. *Query* yang ditambahkan merupakan hipernim dan hiponim dari *query* awal, jumlah penambahan *query* pada manualisasi ini adalah ≤ 5 . Hal tersebut dilakukan untuk mencegah penambahan kata yang terlalu banyak sehingga mengurangi akurasi dan meningkatkan waktu komputasi. Pada manualisasi ini juga, kata yang diekspansi berjumlah satu kata, hal ini betujuan untuk mengurangi waktu komputasi dan meningkatkan akurasi.

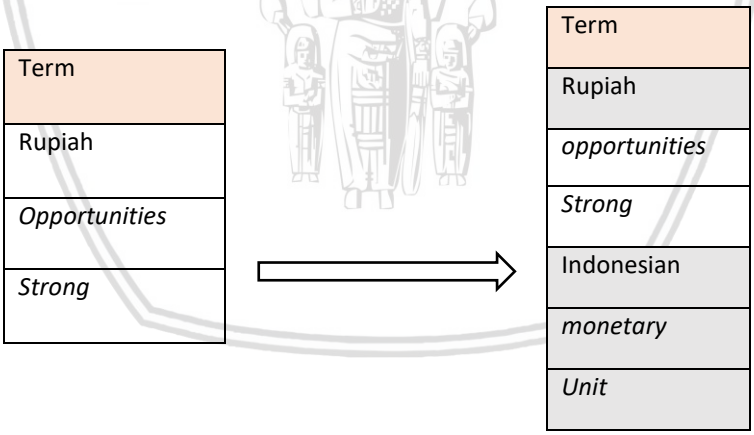
Tahapan *query expansion* yang pertama adalah mengubah dokumen *tweet* 1 yang sebelumnya berbahasa Indonesia menjadi Bahasa Inggris. Dokumen harus diterjemahkan terlebih dahulu karena WordNet yang digunakan berbahasa

Inggris. Berikut adalah proses translasi yang dilakukan menggunakan Google Translate API, yang dituliskan ke dalam Gambar 4.8.



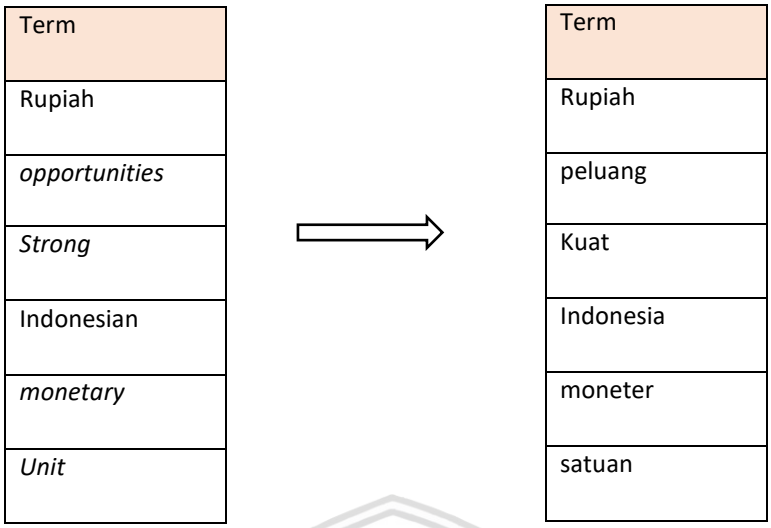
Gambar 0.8 Translate dari Bahasa Indonesia ke Bahasa Inggris

Setelah proses penerjemahan dilakukan selanjutnya adalah melakukan proses *query expansion. Query* yang baru melalui proses *preprocessing* terlebih dahulu untuk mendapatkan kata dasar untuk selanjutnya ditambahkan sebagai *term* dari dokumen *tweet* 1. Penambahan dilakukan tanpa memilih *term* mana yang diekspansi melainkan langsung mengambil *term* pertama pada dokumen. Penambahan *query* dituliskan ke dalam Gambar 4.9



Gambar 0.9 Query Expansion

Kata yang diekspansi merupakan kata "rupiah", kata rupian sendiri tidak memiliki hiponim sedangkan yang ditambahkan pada Gambar 4.9 adalah hipernim dari kata "rupiah". Kata yang ditambahkan dari kata "rupiah" hanya berjumlah tiga kata. Setelah ditambahkan selanjutnya kembali diterjemahkan ke dalam Bahasa Indonesia.



Gambar 0.10 *Translate* ke Bahasa Indonesia

Untuk setiap kata pada dokumen yang telah ditambahkan *query* menjadi data uji dan dihitung probabilitasnya. Perhitungan probabilitas dokumen *tweet* 1 dituliskan ke dalam Tabel 4.10

Tabel 0.10 *Likelihood Term* Dokumen Tweet 1

P(W|C) dok1	Ekonomi	Entertaintment	Kesehatan	Kuliner	Olahraga	Otomotif	Teknologi	Travel
rupiah	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
peluang	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
kuat	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
Indonesia	0,0147	0,0145	0,0139	0,0143	0,0130	0,0286	0,0278	0,0149
moneter	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
Satu	0,0147	0,0145	0,0139	0,0143	0,0130	0,0143	0,0139	0,0149
Posterior	$1{,}009 \times 10^{-11}$	$0{,}929 \times 10^{-11}$	$0{,}721 \times 10^{-11}$	$0{,}855 \times 10^{-11}$	$0{,}482 \times 10^{-11}$	$1{,}171 \times 10^{-11}$	$1{,}44 \times 10^{-11}$	$1{,}094 \times 10^{-11}$

Hasil yang didapatkan setelah melakukan *query expansion* dapat dilihat pada Tabel 4.10. Tidak terjadi perubahan *posterior* yang signifikan dikarenakan kata yang ditambahkan juga tidak terdapat pada data latih. Namun jika dilihat probabilitas terbesar, dokumen *tweet* 1 terklasifikasi ke dalam kategori Otomotif. Seharusnya dokumen *tweet* 1 terklasifikasi ke dalam kategori Ekonomi.

4.4 Perancangan Pengujian

Pada pengujian yang dilakukan, ada tiga jenis pengujian, yaitu pengujian akurasi sitem menggunakan metode *k-fold,* pengujian perubahan atau variasi *threshold* pada jumlah kata yang diekspansi dengan penambahan hiponim dan hipernim dan jenis kata yang ditambahkan, yaitu hipernim atau hiponim

4.4.1 Pengujian *k-fold*

Pada pengujian akurasi *k-fold* menggunakan pembagian dokumen menjadi 5 *fold*. Akurasi tiap-tiap *k* dihitung dengan perhitungan akurasi, membadingkan hasil klasifikasi sitem dengan data yang telah dilabeli sebelumnya. Untuk lebih jelasnya dapat dilihat pada Tabel 4.11.

Tabel 0.11 Perancangan Pengujian *k-fold*

K	K1	K2	K3	K4	K5
Akurasi k	**%	**%	**%	**%	**%

Setelah didapatkan akurasi dari masing-masing *k* maka selanjutnya adalah menghitung akurasi akhir sistem dengan cara mencari rata-rata dari akurasi seluruh *k.*

4.4.2 Pengujian variasi *Threshold*

Pengujian ini dilakukan dengan melakukan variasi terhadap jumlah kata yang diekspansi dan jumlah kata yang ditambahkan sebagai kata ekspansi. Pengujian ini dilakukan untuk mengetahui batas *threshold* yang memiliki akurasi tertinggi, dengan *range threshold* 0,0 – 1,00. Pada Tabel 4.12 ditampilkan perancangan pengujian *threshold.*

Tabel 0.12 Perancangan Pengujian *Threshold*

Data Set	*Threshold* kata yang diekspansi	*Threshold* kata yang ditambahkan	Akurasi hasil klasifikasi
500	0,1	0,1	**%
		. . .	**%
		0,**	**%
	. . .		
	0,**	0,1	**%
		. . .	**%
		0,**	**%

4.4.3 Pengujian *Query* yang Ditambahkan

Pengujian ini dilakukan dengan hanya menambahkan salah satu dari *query,* yaitudengan menambahkan hipernim atau hiponim saja dengan *threshold* yang diambil dari *threshold* terpilih yang menghasilkan akurasi tertinggi saat ditambahkan hipernim dan hiponim sekaligus. Pada Tabel 4.13 ditampilkan perancangan pengujian *query* yang ditambahkan. Pada pengujian ini akurasi juga dihitung menggunakan metode *k-fold*.

Tabel 0.13Pengujian Jenis *Query* yang Ditambahkan

Threshold	Akurasi Hipernim	Akurasi Hiponim
,	**%	**%

BAB 5 HASIL

Pada bab ini berisi implementasi berdasarkan pada metodologi dan perancangan yang telah dilakukan sebelumnya.

5.1 Batasan Implementasi

Dalam melakukan proses implementasi diterapkan batasan-batasan proses yang dapat dilakukan oleh sistem berdasarkan perancangan yang dibahas sebelumnya. Batasan dalam implementasi antara lain:

1. Penerapan klasifikasi berita pada Twitter menggunakan metode Naive Bayes dan *query expansion* hipernim-hiponim dilakukan dengan aplikasi IDE Spyder dengan bahsa Python2.7.
2. Metode yang diimplemetasikan adalah metode klasifikasi Naïve Bayes dan *query expansion*.
3. Data yan digunakan merupakan data yang telah ada sebelumnya dalam skripsi Bagaskoro,Fauzi, & Adikara 2018.
4. Hasil keluaran aplikasi berupa kategori dari dokumen.

5.2 Implementasi Algoritme *Preprocessing*

Preprocessing merupakan tahap awal dari rangkaian proses yang dilakukan pada sistem. *Preprocessing* dilakukan sebelum melakukan proses *query expansion* maupun klasifikasi. Ada beberapa tahapan dalam proses *preprocessing* yaitu:

1. Tokenisasi
2. *Filtering*
3. *Stemming*

5.2.1 Implementasi Kode Program Tokenisasi

Proses ini dilakukan untuk memecah dokumen enjadi potongan kata. Proses tokenosasi dilakukan menggunakan fungsi *regex*. Berikut implemetasi proses tokensisasi pada Kode Program 5.1.

Kode Program 0.1 Implementasi Proses Tokenisasi

```
1  def Tokenisasi(dok):
2      result=[]
3      for word in dok:
4          result.append( re.findall(r'\b[a-
   z]{3,15}\b',str(word).lower()))
5      return result
```

Penjelasan dari Kode Prgram 5.1 adalah sebagai berikut:

1. Baris 1 merupakan inisialisasi dari method dengan nama “Tokenisasi”
2. Baris 2 membuat *list* dengan nama *result*

3. Baris 3-4 merupakan perulangan untuk setiap *word*yang ada pada dokumen, dilakukan pencarian dengan fungsi *re.findall* dengan parameter sesuai dengan kode program sehingga didapatkan potongan kata yang selanjutnya ditambahkan ke dalam *list result.*
4. Baris 5 mengembalikan nilai yang disimpan dalam *list result*

5.2.2 Implementasi Kode Program *Filtering*

Proses ini dilakukan untuk menghilangkan kata-kata yang dianggap tidak penting seperti kata sambung. Kata-kata yang dianggap tidak penting dismpan dengan nama *stopword* yang didapatkan dari sumber:https://github.com/masdevid/IDStopwords/blob/master/id.stopwords.02.01.2016.txt.

Berikut implementasi dari proses *filtering* dituliskan dalam Kode Program 5.2.

Kode Program 0.2 Implementasi Proses *Filtering*

```
1   def Filtering(dok):
2       result=[]
3       term=dok
4       for word in term:
5           if word == "ekonomi" :
6               term.remove(word)
7           elif word == "entertaiment":
8               term.remove(word)
9           elif word == "sehat ":
10              term.remove(word)
11          elif word == "kuliner":
12              term.remove(word)
13          elif word == "olahraga":
14              term.remove(word)
15          elif word == "teknologi":
16              term.remove(word)
17          elif word == "otomotif":
18              term.remove(word)
19          elif word == "travel":
20              term.remove(word)
21
22      for word in term:
23          if word not in stopword:
24              result.append(word)
25      return result
```

Penjelasan dari Kode Program 5.2 adalah sebagai berikut:

1. Baris 1 inisisalisasi method dengan nama "*Filtering*"
2. Baris 2 membuat *list* dengan nama *result*
3. Baris 3 inisialisasivariabel *term* dengan nilai dok
4. Baris 4-20 perulangan yang bertujuan untuk menghapus kata kategori, karena saat *open file* dokumen kategori termasuk dalam *term*.
5. Baris 22-24 perulangan untuk setiap *word* pada *term* jika kata tersebut tidak ada dalam *stopword* maka selanjutnya ditambahkan ke dalam *list result*.

6. Baris 25 Mengembalikan nilai yang disimpan dalam *list result.*

5.2.3 Implementasi Kode Program *Stemming*

Proses ini dilakukan untuk mengembalikan *term* yang ada pada dokumen menjadi kata dasar, dengan cara menghapuskan imbuhan yang ada pada kata tersebut. Berikut implemenntasi dari proses *stemming* dituliskan dalam Kode Program 5.3.

Kode Program 0.3 Implementasi Proses *Stemming*

```
1   def stemming(dok):
2       result=[]
3       for word in dok:
4           result.append(stemmer.stem(str(word)))
5
6       return result
```

Penjelasan dari Kode Program 5.3 adalah sebagai berikut:

1. Baris 1 merupakan inisialisasi *method* dengan nama "*stemming*"
2. Baris 2 inisialisasi *list* dengan nama *result*
3. Baris 3-4 Perulangan untuk setiap *word* yang ada pada variabel dok(parameter). Setiap *word* dijadikan bentuk kata dasar menggunakan fungsi *stemmer sastrawi*. Selanjutnya hasil *stemming* disimpan ke dalam *list result.*
4. Baris 6 Mengenbalikan nilai yang disimpan dalam *list result*

5.3 Implementasi Algoritme *Query Expansion*

Query expansion merupakan tahapan setelah proses *preprocessing* dilakukan. Tahapan ini bertujuan menambahkan *query* tambahan kepada *query* awal. *Query expansion* memiliki beberapa tahapan yaitu:

1. *Translate* dokumen dari Bahasa Indonesia menuju Bahasa Inggris dan sebaliknya
2. Mencari kata ekspansi (hipernim dan hiponim dari *term*)
3. Memproses hipernim dan hiponim yang telah didapatkan
4. Menambahkan kata ekspansi ke-*term* asli

5.3.1 Implementasi Kode Program *Translate* Dokumen

Proses ini dilakukan untuk mengubah Bahasa dari dokumen, pada penelitian ini dilakukan translasi dari Bahasa Indonesia menuju Bahasa Inggris dan sebaliknya setelah proses penambahan selesai. Berikut implementasi dari proses *translate* dokumen dituliskan pada Kode Program 5.4.

Kode Program 0.4Implementasi Proses *Translate* Dokumen

```
1   def Translate(doc,destination):
2       result=[]
3       for dokumen in doc:
4           List=[]
5           for term in dokumen:
6               List.append(str((trans.translate(term,
    dest=destination)).text))
7           result.append(List)
8       return result
```

Penjelasan dari Kode Program 5.4 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama "Translate"
2. Baris 2 inisialisasi *list* dengan nama *result*
3. Baris 3-4 perulangan untuk setiap dokumen pada variabeldoc (parameter), setiap perulangan menginisialisasi *list* dengan nama *List*.
4. Baris 5-6 *nested loop* dari perulangan baris 3-4, perulangan untuk setiap *term* yang ada pada dokumen. Ditranslasikan sesuai dengan parameter *destination* dan diambil teksnya saja lalu ditambahkan ke dalam *list List*
5. Baris 7 menambahkan *list List* ke dalam *listresult* sesuai dengan perulangan baris 3-4
6. Baris 8 mengembalikan nilai yang disimpan dalam *list result*

5.3.2 Implementasi Kode Program Search*QueryExpansion*

Proses ini dilakukan setelah dokumen ditranslasi ke dalam Bahasa Inggris. Dari *synset* setiap *term* dicari hiperim dan hiponimnya. Berikut implementasi dari proses *query expansion* dituliskan ke dalam Kode Program 5.5

Kode Program 0.5 Implementasi Proses *SearchQueryExpansion*

```
1   def SearchQueryExpansion(dok):
2       result=[]
3       for word in dok:
4           for syn in wn.synsets(word):
5               if syn.hypernyms():
6                   result.append(syn.hypernyms()[0].lemma_names())
7               if syn.hyponyms():
8                   result.append(syn.hyponyms()[0].lemma_names())
9       return result
```

Penjelasan dari Kode Program 5.5 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama "SearchQueryExpansion"
2. Baris 2 inisialisasi *list* dengan nama *result*
3. Baris 3-5 perulangan untuk setiap word pada dok (parameter)
4. Baris 4 *nested loop* dari perlungan baris 3, untuk setiap *syn* (objek) yang ada pada *synset* word.

5. Baris 5-6 percabangan *if*. Apabila objek *syn*merupakan hipernim dari word maka ditambahkanke dalam *list result*
6. Baris 7-8 percabangan *if*. Apabila objek *syn* merupakan hiponim dari word maka ditambahkan ke dalam *list result*
7. Baris 9 Mengembalikan nilai yang disimpan dalam *list result*

5.3.3 Implementasi Kode Program Pengolahan Kata Ekspansi

Proses ini bertujuan untuk mengolah kata ekspansi yang didapatkan karena di dalamnya masih ada karakter-karakter yang tidak diperlukan seperti garis bawah (_). Berikut implementasi dari proses pengolahan kata ekspansi dituliskan ke dalam Kode Program 5.6.

Kode Program 0.6 Implementasi Proses Pengolahan Kata Ekspansi

```
1  def PengoLahanEkspansi(dok):
2      PengolahanEkspansi=[]
3      for x in range(len(dok)):
4          List=[]
5          for dokumen in dok[x]:
6              for kata in dokumen:
7                  List.append(str(re.sub("_"," ",kata)))
8          PengolahanEkspansi.append(List)
9      return PengolahanEkspansi
```

Penjelasan dari Kode Program 5.6 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama “PengoLahanEkspansi”
2. Baris 2 inisialisasi *list* dengan nama PengolahanEkspansi
3. Baris 3-8 perulangan sebanyak *x* kali (panjang dari dok (parameter))
4. Baris 4 inisialisasi *list* dengan nama *List*saat perulangan baris 3 dilakukan
5. Baris 5 *nested loop* dari baris 3, untuk setiap dokumen(objek) yang ada pada dok dengan indeks *x*. Nilai *x* bertambah setiap perulangan baris 3
6. Baris 6-7 *nested loop* dari baris 3 dan 5, untuk setiap kata(objek) yang ada pada dokumen. Mengganti nilai karakter “_” dengan karakter spasi “ ”.Kemudian ditambahkan ke dalam *listList*
7. Baris 8 menambahkan nilai *list List* ke dalam *list* PengolahanEkspansi
8. Baris 9 mengembalikan nilai yang disimpan dalam *list* PengolahanEkspansi

5.3.4 Implementasi Kode Program Penambahan

Proses ini dilakukan setelah didapatkan kata ekspansi yang telah diolah. Dari kata ekspansi yang ada kemudian ditambahkan ke dalam dokumen asli atau *term*asli. Berikut adalah implementasi dari proses penambahan kata ekspansi dituliskan ke dalam Kode Program 5.7

Kode Program 0.7 Implementasi Porses Penambahan Kata Ekspansi

```
1   def Penambahan(dokumen,ekspansi,TH):
2       result=[]
3       for iteration in range(0,len(dokumen)):
4           a=dokumen[iteration]
5           b=ekspansi[iteration]
6           threshold=math.trunc(len(b)*TH)
7
8           for x in b[0:threshold]:
9               a.append(x)
10          result.append(a)
11      return result
```

Penjelasan dari Kode Program 5.7 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama "Penambahan"
2. Baris 2 inisialisasi *list* dengan nama *result*
3. Baris 3-10 perulangan, untuk setiap kali perulangan sejumlah panjang dokumen dimulai dari 0
4. Baris4-5 inisialisasi variabel a dengan isi dokumen indeks ke-*iteration* dan variabel b dengan isi ekspansi indeks ke-*iteration*
5. Baris 6 inisialisasi variabel *threshold* dengan mengalikan panjang *b* dengan TH (parameter)
6. Baris 8-9 *nested loop* dari baris 3, untuk setiap *x* (ojbek) pada variabel b dengan indeks mulai dari 0 sampainilai *threshold* yang telah disimpan sebelumnya. Nilai yang didapatkan kemudian ditambahkan ke dalam variabel
7. Baris 10 menambahkan nilai pada *variabel a* ke dalam *list result* sesuai dengan perulangan baris 3
8. Baris 11 mengembalikan nilai yang disimpan dalam *list result*

5.4 Implementasi Algoritme Naïve Bayes

Proses ini dilakukan untuk mengklasifikasikan dokumen ke dalam kelas-kelas yang telah ditentukan sebelumnya. Naïve Bayes memiliki beberapa tahapan yang dilakukan yaitu:

1. Mencari *term* latih yang berisi *term* unik dan menghitung panjang atau jumlah dari *term* unik.
2. Mencari dan menghitung jumlah dari *term* tiap kelas
3. Melakukan perhitungan *likelihood*
4. Melakukan perhitungan *posterior*
5. Melakukan pemilihan kelas dari nilai *posterior*

5.4.1 Implementasi Kode Program Pencarian *Term* Latih

Proses ini dilakukan pertama kali saat melakukan klkasifikasi Naïve Bayes, bertujuan untuk mengambil *term* unik dari dokumen latih dan menghitung panjang atau banyaknya *term* unik. Berikut adalah implementasi dari proses pencarian *term* unik dituliskan dalam Kode Program 5.8.

Kode Program 0.8 Implementasi Proses Pencarian *Term* Unik

```
1  Data_latih=[isiK2,isiK3,isiK4,isiK5]
2  Term_latih=[]
3  for fold in Data_latih:
4      for doc in fold:
5          for word in doc:
6              if word not in Term_latih:
7                  Term_latih.append(word)
8
9  V_jumlah_term_unik=len(Term_latih)
```

Penjelasan dari Kode Program 5.8 adalah sebagai berikut:

1. Baris 1 inisilaisasi variabel *Data_latih* berupa *list*
2. Baris 2 inisialisasi *list* dengan nama *Term_latih*
3. Baris 3-7 perulangan untuk mencari *term* unik. Di dalamnya ada percabangan yang melakukan pengecekan, jika *term* yang sedang diproses tidak ada pada *list Term_latih* maka ditambahkan. Sehingga didapatkan *term*unik saja.
4. Baris 9 inisialisasi variabel*V_jumlah_term_unik*dengan isimenghitung panjang atau banyaknya *term*latih pada *list Term_latih.*

5.4.2 Implementasi Kode Program Pencarian *Term* Setiap Kelas

Proses ini dilakukan dengan menghitung banyaknya *term* yang ada pada tiap-tiap kelas. Berikut adalah implementasi dari proses pencarian *term* unik dituliskan dalam Kode Program 5.9. Contoh yang diberikan adalah penghitungan untuk kelas ekonomi.

Kode Program 0.9 Implementasi Proses Pencarian *Term*

```
1  Term_latih_Ekonomi=[isiK2[0:9],isiK3[0:9],isiK4[0:9],isiK5[0:9]]
2  sigma_term_ekonomi=len(Term_latih_Ekonomi)
```

Penjelasan dari Kode Program 5.9 adalah sebgai berikut:

1. Baris 1 inisialisasi *list* denga nama *Term_latih_Ekonomi*.
2. Baris 2 inisialisasi *list* dengan nama *sigma_term_*ekonomi yang memiliki nilai panjang atau banyaknya *term* yang ada pada *list Term_latih_Ekonomi*

5.4.3 Implementasi Kode Program Menghitung *Likelihood*

Proses ini adalah menghitung *likelihood* dari setiap *term* pada data uji, yaitu menghitung probabilitas setiap *term* terhadap setiap kelas yang ada. Berikut

adalah implementasi dari proses menghitung *likelihood* dituliskan dalam Kode Program 5.10.

Kode Program 0.10 Implementasi Proses Menghitung *Likelihood*

```
1   def likelihood(doc,V,Sigma,Term_Kelas):
2       result=[]
3       countWord=[]
4       term=doc
5       for word in term:
6           if word not in Term_latih:
7               term.remove(word)
8       for word in term:
9           countWord.append(Term_Kelas.count(word))
10
11      for count in countWord:
12          result.append(((count+1)/float(Sigma+V)))
13
14      return result
```

Penjelasan untuk Kode Program 5.10 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama *likelihood*
2. Baris 2-4 inisialisasi variabel
3. Baris 5-7 perulangan untuk setiap objek pada variabel, jika objek tersebut tidak ada pada *Term_Kelas* (paramenter)maka objek tersebut dihapus
4. Baris 8-9 perulangan untuk setiap objek yang ada pada variabel, menhitung banyaknya kemunculan objek tersebut dalam variabel
5. Baris 11-12 perulangan untuk setiap objek pada variabel, menambahkan nilai hasil perhitungan ke dalam *list*
6. Baris 14 mengembalikan nilai yang disimpan di dalam *listresult*

5.4.4 Implementasi Kode Program Menghitung *Posterior*

Proses ini dilakukan dengan mengkalikan setiap nilai likehood *term* dalam satu kelas pada satu dokumen untuk mendapatkan nilai probabilitas dokumen terhadap seluruh kelas yang telah ditentukan sebelumnya.Berikut adalah implementasi dari proses menghitung *posterior* dituliskan dalam Kode Program 5.11.

Kode Program 0.11 Implementasi Kode Program Menghitung *Posterior*

```
1   def Possterior(doc):
2       result=[]
3       hasil=1
4       for nilai in doc:
5           hasil=hasil*nilai
6       result.append(priorKelas*hasil)
7       return result
```

Penjelasan dari Kode Program 5.11 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama *posterior*
2. Baris 2-3 inisialisasi variabel

3. Baris 4-5 perulangan untuk setiap objek pada variabel. Menghitung nilai variabel *hasil*

4. Baris 6 menambahkan nilai pada *list result*

5. Baris 7 mengembalikan nilai yang disimpan di dalam *list result*

5.4.5 Implementasi Kode Program Memilih kelas

Proses ini dilakukan setelah didapatkan nilai *post*erior dari setiap dokumen. Proses pemilihan kelas dilakukan dengan mencari nilai *posterior* terbesar dokumen terhadap kelas. *Posterior* terbesar menjadi kelas dari dokumen tersebut. Berikut adalah implementasi dari proses memilih kelas dituliskan dalam Kode Program 5.12.

Kode Program 0.12 Implementasi Proses Pemilihan Kelas

```
1   def PilihKelas(doc,doc1,doc2,doc3,doc4,doc5,doc6,doc7):
2       result=[]
3       probTerpilih=[]
4       nilai=0
5       for x in range(len(doc)):
6           nilai =
    max(doc[x],doc1[x],doc2[x],doc3[x],doc4[x],doc5[x],doc6[x],doc7[x])
7           probTerpilih.append(nilai)
8
9       for x in range(len(doc)):
10          for prob in probTerpilih:
11              if prob == doc[x]:
12                  result.append("ekonomi")
13                  break
14              elif prob == doc1[x]:
15                  result.append("entertaintment")
16                  break
17              elif prob == doc2[x]:
18                  result.append("kesehatan")
19                  break
20              elif prob == doc3[x]:
21                  result.append("kuliner")
22                  break
23              elif prob == doc4[x]:
24                  result.append("olahraga")
25                  break
26              elif prob == doc5[x]:
27                  result.append("otomotif")
28                  break
29              elif prob == doc6[x]:
30                  result.append("teknologi")
31                  break
32              elif prob == doc7[x]:
33                  result.append("travel")
34                  break
35
36      return result
```

Penjelasan dari Kode Program 5.12 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama *PilihKelas*

2. Baris 2-4 inisialisasi *variabel result* dengan tipe *list, probTerpilih* dengan tipe *list,nilai* dengan tipe integer

3. Baris 5-7 perulangan sebanyak *x* kali yaitu sebanyak panjang dokumen *doc* (parameter), nilai *x* berubah +1 setiap perulangan dilakukan. Baris 6 mengubah nilai variabel *nilai* dengan mencari nilai maksimum menggunakan fungsi *max* dari setiap dokumen dengan indeks sesuaidengan nilai *x*. Baris 7 menambahkan nilai variabel *nilai* yang baru ke dalam *list probTerpilih.*

4. Baris 9-34 perulangan sebanyak *x* kali yaitu sebanyak panjang dokumen *doc* (parameter), nilai *x* berubah +1 setiap perulangan dilakukan. Baris 10 perulangan untuk setiap objek pada *list probTerpilih*

5. Baris 11-34 percabangan jika objek pada *probTerpilih* indeks ke *x* sesuai dengan nilai variabel maka *list result* ditambahkan kategori dari objek. Jika sudah ditambahkan maka proses perulangan baris 9-34 dilanjutkandengan mengabaikan percabangan pada baris bawahnya.

6. Baris 36 mengembalikan nilai yang disimpan dalam *list result*

5.5 Implentasi Algoritme Menghitung Akurasi

Setelah didapatkan kelas dari setiap dokumen, maka selanjutnya adalah menghitung tingkat akurasi dari klasifikasi dengan membandingkan kelas yang didapatkan dengan kelas yang seharusnya dari dokumen tersebut. Berikut adalah implementasi dari proses menghitung akurasi dituliskan dalam Kode Program 5.13.

Kode Program 0.13 Implentasi Proses Menghitung Akurasi

```
1   def akurasi(doc):
2       benar=0
3       for kelas in doc[0:9]:
4           if kelas == "ekonomi":
5               benar+=1
6       for kelas in doc[10:19]:
7           if kelas == "entertaintment":
8               benar+=1
9       for kelas in doc[20:29]:
10          if kelas == "kesehatan":
11              benar+=1
12      for kelas in doc[30:39]:
13          if kelas == "kuliner":
14              benar+=1
15      for kelas in doc[40:49]:
16          if kelas == "olahraga":
17              benar+=1
18      for kelas in doc[50:59]:
19          if kelas == "otomotif":
20              benar+=1
21      for kelas in doc[60:69]:
22          if kelas == "teknologi":
23              benar+=1
24      for kelas in doc[70:79]:
25          if kelas == "travel":
26              benar+=1
27
28      akurasi = float(benar) / len(doc)
29      return akurasi
```

Penjelasan dari Kode Program 5.13 adalah sebagai berikut:

1. Baris 1 inisialisasi *method* dengan nama *akurasi*
2. Baris 2 inisialisasi variabel *benar*dengan nilai 0
3. Baris 3-26 Perlangan untuk setiap objek pada *variabel doc* (parameter) dengan indeks tertentu. Jika objek sama dengan kelas yang dimaksud maka nilai variabel *benar* ditambah 1.
4. Baris 28 menghitung nilai dari variabel akurasi
5. Baris 29 mengembalikan nilai yaitu nilai dari variabel *akurasi* yang telah didapatkan sebelumnya

BAB 6 PEMBAHASAN DAN ANALISIS

Pada bab ini menjelaskan mengenai pengujian yang dilakukan pada saat penelitian. Bab ini juga berisi analisis terhadap hasil dari pengujian dan implementasi yang telah dilakukan. Tujuan dari bab ini adalah untuk mengetahui efektifitas dari proses *query expansion* dan klasifikasi Naïve Bayes dalam melakukan klasifikasi *tweet* berita pada Twitter. Pada bab ini ada 3 jenis pengujian yang dilakukan.

1. Pengujian perhitungan akurasi tanpa melakukan *query expansion* (*k-fold*)
2. Pengujian variasi *threshold* jumlah kata yang ditambahkan
3. Pengujian jenis kata yang ditambahkan (hipernim dan hiponim)

6.1 Pengujian Perhitungan Akurasi dengan *K-Fold*

Pada pengujian ini dilakukan perhitungan akurasi menggunakan metode *k-fold* menggunakan 5 *fold*. Dokumen-dokumen yang terdapat di dalam setiap *fold* dipilih secara manual oleh Penulis. Namun pemilihan dokumen dilakukan secara acak agar menghindari adanya manipulasi data. Jumlah dokumen dari setiap *fold* adalah 80 dokumen dengan isi 10 dari tiap-tiap kelas yang berjumlah 8 kelas. Perhitungan akurasi klasifikasi dilakukan tanpa melakukan *query expansion*. Perhitungan akurasi sistem dilakukan dengan merata-rata jumlah dari seluruh akurasi *fold*. Tujuan penggunaan *k-fold* sebagai metode pengujian bertujuan untuk menguji *robustness* atau kehandalan dari sistem terhadap setiap data uji yang digunakanakan. Selain menguji kehandalan sistem, pengujian menggunakan *k-fold* menghindari terjadinya pemilihan data uji oleh peneliti agar mendapatkan akurasi yang tinggi. Berikut adalah hasil perhitungan akurasi dari masing-masing *fold* yang dituliskan ke dalam Tabel 6.1.

Tabel 6.1 Hasil Perhitungan Akurasi Dengan *K-Fold*

K	1	2	3	4	5	Sistem
Akurasi	77,5%	83,75%	75%	67,5%	56,25%	72%

Pada Tabel 6.1 berisikan hasil klasfikasi tanpa menggunakan *query expansion* memiliki akurasi sebesar 72%. Hasil akurasi dari tiap *fold* tidak memengaruhi*fold* yang lain. Akurasi *fold* tertinggi adalah 83,75% pada *fold* 2 dan akurasi *fold* terendah adalah 56,25% pada *fold* 5. Rendahnya tingkat akurasi pada proses klasifikasi *tweet* ini disebabkan oleh *dataset* yang digunakan . Persebaran dari dokumen dengan *term* tertentu sangat memengaruhi hasil dari proses klasifikasi, maksudnya jika sebuah *term* hanya muncul pada satu *fold* maka dipastikan proses klasifikasi menjadi kurang maksimal. Selain itu, dari*dataset* yang digunakan beberapa dokumen dengan kelas tertentu memiliki satu atau beberapa *term* yang sama dengan kelas lain sehingga juga menyebabkan proses klasifikasi menjadi

tidak optimal. Contoh dari dokumen yang memiliki kesamaan salah satu atau beberapa *term* dapat dilihat pada Tabel 6.2.

Tabel 6.2 Dokumen Dengan Kesamaan *Term*

Dokumen	Kelas	*Fold*
Pantai Karang Jahe, Wisata Bahari Andalan dari Rembang	Ekonomi	2
Yang beda di Thailand, Wisata ke Kebun Anggur	Travel	2
Pengusaha Pesawat Carter Bidik Sektor Pariwisata dan 'Taksi' Udara	Travel	3
Mau Wisata Kuliner di Eropa? Ikuti 10 Tips M Enak Ini (2)	Kuliner	5
Jokowi Ingin Raja Ampat Jadi Lokasi Wisata Eksklusif	Travel	5

Dari beberapa contoh dokumen Tabel 6.2masing-masing dokumen mengandung *term* "wisata".Hal ini memengaruhi saat proses perhitungan *likelihood*, apalagi dokumen dengan *term* yang sama berada pada *fold* yang berbeda karena ketika sebuah *fold* menjadi dokumen uji maka dokumen lainnya yang ada pada *fold* lain menjadi data latih. Permasalahan selanjutnya adalah adanya dokumen yang seharusnya *single*label namun kenyataanya dapat diklasifikasikan ke dalam kelas lain. Contoh dokumen yang dapat diklasifikasikan dalam beberapa kelas dapat dilihat pada Tabel 6.3.

Tabel 6.3 Dokumen Dengan Kemungkinan Multi Kelas

No	Isi Dokumen	Kelas Label	Kelas Sistem
1	Pemerintah Tambah 84 Negara Bebas VisaWisata ke RI	Ekonomi	Travel
2	Tiket Rp 15.000/Orang, Kapal ke Pulau Seribu Baru Terisi 50%	Ekonomi	Travel

Pada dokumen pertama yang ada pada Tabel 6.3, kelas seharusnya dari dokumen adalah kelas ekonomi karena menekankan pada *term* "visa" namun ketika diklasifikasi oleh sistem menjadi dokumen travel karena menekankan pada *term* "wisata". Pada dokumen kedua, dokumen seharusnya diklasifikasikan sebagai kelas ekonomi karena secara konteks membahas mengenai penjualan tiket namun oleh sistem diklasifikasikan sebagai kelas travel karena *term* yang

terdapat di dalamnya ada *term* yang berkaitan dengan tempat wisata yaitu *term* "Pulau Seribu".

Proses klasifikasi yang dilakukan merupakan proses klasfikasi dengan fitur *bag of words* yang berarti bahwa proses klasfikasi sepenuhnya bergantung pada *term* yang terdapat pada dokumen, jika terdapat kesamaan *term* antar dokumen maka tentu saja sangat berdampak pada hasil klasifikasi. Selain itu proses klafikasi tidak berdasarkan konteks dan maksud dari dokumen, sehingga proses klasifikasi sangat bergantung pada *term* yang ada dalam dokumen itu sendiri.

6.2 Pengujian Variasi *Threshold* Kata yang Ditambahkan

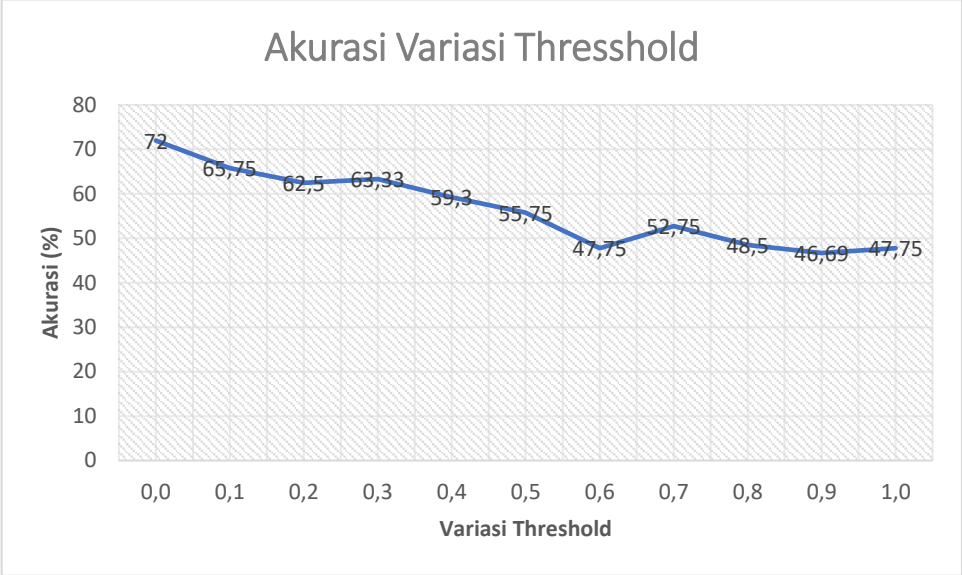
Pada pengujian ini dilakukan proses *query expansion* terhadap *term* dokumen asli. Jumlah dari kata yang ditambahkan kemudian divariasi dari rentang 0,1 - 1,0 untuk mendapatkan akurasi tertinggi berdasarkan jumlah kata ekspansi yang ditambahkan. Penambahan berdasarkan *threshold* dilakukan dengan mengalikan panjang dokumen (banyaknya kata ekspansi dalam satu dokumen) dengan *threshold*. Nilai yang didapatkan kemudian menjadi batas indeks kata ekspansi yang ditambahkan ke dalam dokumen asli. Misalkan ketika pada dokumen 1 ada 20 kata ekspansi dari seluruh *term* yang ada pada dokumen 1, saat *threshold* = 0,1 maka batas dari kata yang ditambahkan adalah 20 X 0,1 = 2, jadi kata yang ditambahkan merupakan kata ekspansi dengan indeks 0 sampai 2. Bertambahnya nilai *threshold* berbanding lurus dengan pertambahan kata ekspansi yang ditambahkan ke dalam dokumen. Pada pengujian ini, kata ekspansi yang ditambahkan merupakan gabungan antara hipernim dan hiponim dari *term*. Sebelumnya dokumen berbahasa Indonesia harus melalui proses *preprocessing* yang kemudian dilanjutkan dengan proses *translate* ke dalam Bahasa Inggris dikarenakan WordNet yang merupakan kamus acuan penambahan kata berbahasa merupakan WordNet berbahasa Inggris. Hasil dari variasi *threshold* dari kata ekspansi yang ditambahkan ditampilkan ke dalam Tabel 6.4.

Tabel 6.4 akurasi Dari Variasi *Threshold*

Threshold	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1,0
Akurasi (%)	72,0	65,75	62,5	63,33	59,25	55,75	47,75	52,75	48,5	46,69	47,75

Pada Tabel 6.4, dimulai dari *threshold* 0,0 yang berarti bahwa proses klasifikasi tidak mengalami atau melalui proses *query expansion* dengan akurasi mencapai 72,0 %. Pada *threshold* selanjutnya akurasi mengalami penurunan akurasi sebesar 6,25%, akurasi sistem menjadi65,75% pada *threshold* 0,1. Nilai akurasi kemudian mengalami penurunansampai dengan *threshold*0,2 dan kembali meningkat pada *threshold* 0,3. Setelah mengalami peningkatan akurasi selanjutnya terjadi penurunan hingga *threshold* 1,0 dengan akurasi sebesar 47,75%. *Threshold* terbaik yang digunakanakan dalam penelitian ini adalah 0,1. Untuk lebih jelas dalam

melihat peningkatan dan penurunan akurasi dari variasi *threshold* maka akurasi ditampilkan pada Gambar 6.1.



Gambar 6.1 Akurasi Variasi *Threshold*

Perubahan akurasi baik penurunan maupun peningkatan terjadi karena adanya kata ekspansi yang ditambahkan. Pada Gambar 6.1 grafik menunjukkan bahwa dari *threshold* 0,1 ke *threshold* 0,2 mengalami penurunan tingkat akurasi sebesar 5,25%. Hal ini disebabkan saat penambahan kata ekspansi, kata yang ditambahkan pada *threshold* 0,2 memberikan kata ekspansi yang tidak cocok, sehingga akurasi klasifikasi menurun. Berikut ditampilkan hasil klasifikasi pada saat K2 menjadi data uji, *threshold* 0,1 dan 0,2 pada Tabel 6.5.

Tabel 6.5Hasil Klasifikasi *K2* saat *Threshold* = 0,1

Threshold	Klasifikasi Dokumen	Kelas yang benar
0,1	'ekonomi', 'ekonomi', 'olahraga', 'kesehatan', 'ekonomi', 'ekonomi', 'kesehatan', 'ekonomi', 'ekonomi', 'ekonomi',	Ekonomi
	'travel', 'ekonomi', 'entertaintment', 'entertaintment', 'entertaintment', 'otomotif', 'entertaintment', 'entertaintment', 'entertaintment', 'travel',	Entertaintmen
	'kesehatan', 'ekonomi', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan',	Kesehatan

Tabel 6.5 Hasil Klasifikasi K2 saat *Threshold* = 0,1(Lanjutan)

Threshold	Klasifikasi Dokumen	Kelas yang benar
0,1	'kuliner', 'kuliner', 'travel', 'kuliner', 'kuliner', 'kuliner', 'kuliner', 'kuliner', 'kuliner', 'kuliner',	Kuliner
	'kuliner', 'olahraga', 'teknologi', 'olahraga', 'olahraga', 'olahraga', 'olahraga', 'kuliner', 'olahraga', 'ekonomi',	Olahraga
	'ekonomi', 'otomotif', 'otomotif', 'entertaintment', 'otomotif', 'otomotif', 'otomotif', 'otomotif', 'travel', 'kuliner',	Otomotif
	'teknologi', 'teknologi', 'teknologi', 'teknologi', 'teknologi', 'otomotif', 'teknologi', 'teknologi', 'teknologi', 'teknologi'	Teknologi
	'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel'	Travel

Pada Tabel 6.5 hasil klasifikasi menggunakan metode *query expansion* dengan *threshold* 0,1 didapatkan hanya 62 dokumen dengan kategori yang benar dan menghasilkan 77,5% akurasi. Kemudian berikutnya adalah hasil dari klasifikasi saat *threshold* dinaikkan menjadi 0,2 ditampilkan ke dalam Tabel 6.6.

Tabel 0.6 Hasil Klasifikasi K2 saat *Threshold* = 0,2

Threshold	Klasifikasi Dokumen	Kelas yang benar
0,2	'ekonomi', 'ekonomi', 'olahraga', 'kesehatan', 'ekonomi', 'ekonomi', 'kesehatan', 'ekonomi', 'ekonomi', 'ekonomi',	Ekonomi
	'travel', 'ekonomi', 'entertaintment', 'entertaintment', 'entertaintment', 'otomotif', 'entertaintment', 'entertaintment', 'entertaintment', 'travel',	Entertaintment
	'ekonomi', 'ekonomi', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan', 'kesehatan',	Kesehatan
	'kuliner', 'travel', 'travel', 'kuliner', 'kuliner', 'kuliner', 'kesehatan', 'kuliner', 'kuliner', 'kuliner',	Kuliner
	'kuliner', 'olahraga', 'teknologi', 'olahraga', 'teknologi', 'olahraga', 'olahraga', 'kuliner', 'olahraga', 'teknologi',	Olahraga
	'ekonomi', 'otomotif', 'otomotif', 'entertaintment', 'otomotif', 'otomotif', 'otomotif', 'ekonomi', 'travel', 'kuliner',	Otomotif
	'teknologi', 'teknologi', 'teknologi', 'teknologi', 'teknologi', 'otomotif', 'teknologi', 'teknologi', 'teknologi', 'teknologi',	Teknologi
	'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel', 'travel'	Travel

Pada Tabel 6.6 hasil dari klasifikasi saat *threshold* dinaikkan menjadi 0,2 adalah menurunnya jumlah dokumen dengan klasifikasi yang benar dan akurasi menjadi 71,25%. Contoh dari kelas dokumen yang sebelumnya benar kemudian menjadi salah adalah dokumen kesehatan, yaitu pada dokumen ke-32 (yang diberi garis bawah). Pada saat *threshold* 0,1 dokumen tersebut diklasifikasikan ke dalam kelas kuliner sebagaimana kelas yang benar dan diklasifikasikan ke dalam kelas *travel* saat *threshold* 0,2. Hal ini terjadi karena kata ekspansi yang sebelumnya tidak ditambahkan saat *threshold* 0,1 kemudian ditambahkan saat *threshold* 0,2. Perbedaan dari kata ekspansi yang ditambahkan dituliskan ke dalam Tabel 6.7.

Tabel 6.7 Perbandingan kata ekspansi yang ditambahkan

Threshold	No. Dok	Dokumen + *query expansion*
0,1	32	warga,paris,tolak,isis,cafe,nasional,subyek
0,2	32	warga,paris,tolak,isis,cafe,nasional,subyek,warga,negara,aktif, genus,tanam,putus

Penambahan kata ekspansi saat *threshold* 0,2 memberikan tambahan kata "warga" yang pada *threshold* 0,1 tidak ditambahkan, kata "warga" tersebut terdapat pada data latih kelas *travel*. Sehingga saat perhitungan *likelihood*, nilai probabilitas *term* "warga"meningkat pada kelas *travel*. Penurunan akurasi terjadi ketika sebuah *term* ditambahkan dan *term* tersebut mmeningkatkan nilai *posterior* dari kelas lain. Sehingga dokumen dikelmpokkan ke dalam kelas yang salah.

6.2.1 Permasalahan *Translate* saat Penambahan *Query*

Faktor selanjutnya yang menjadi penentu hasil klasifikasi pada penelitian ini adalah proses *translate* dokumen. Pada penelitian ini digunakan data berupa *tweet* dengan Bahasa Indonesia. Sedangkan sumber yang digunakan untuk melakukan*query expansion* untuk mendapatkan kata ekspansi merupakan WordNet berbahasa Inggris. Tahapan yang dilakukan adalah men*translatetweet* ke dalam Bahasa Inggris, kemudian dilakukanlah proses *query expansion.* Setelah kata ekspansi ditambahkan selanjutnya dokumen dikembalikan ke Bahasa Indonesia. Saat proses *translate* kembali ke Bahasa Indonesia ini beberapa *term* di *translate* tidak sama dengan dokumen aslinya.Proses *translate* sepenuhnya menggunakan *google translate API*. Berikut adalah beberapa *term* yang berubah akibat *translate*, ditampilkan dalam Tabel 6.8.

Tabel 6.8 Perbedaan *Translate*

No	Dokumen Asli	Dokumen setelah *translate*
1	rapat,tutup,jam,dpr,bahas, rahasia,strategi,kelola, devisa	temu,tutup,jam,dpr,bahas,rahasia,strategi,kelola, tukar,temu,majelis,temu,dewan,rapat,komite, kumpul,sosial,urus,sosial,biarawati,konvergensi, konjungsi,jajar,majelis,majelis,temu,temu,

Tabel 6.9 Perbedaan *Translate*(Lanjutan)

No	Dokumen Asli	Dokumen Setelah *translate*
1		konvergensi,konvergen,konvergensi,concourse, temu,titik,geografis
2	city,intip,peluang,arsenal	kota,intip,peluang,arsenal,kotamadya
3	kejut,ponsel,selfie,oppo	syok,telepon,selfie,oppo,,lumpuh, tarung,kelah,tempur,batal,refleks,respon,refleks

Pada Tabel 6.8 dapat dilihat bahwa adanya perbedaan *term* saat sebelum dan sesudah mengalami proses *translate*. Perubahan *term* ini memengaruhi proses klasifikasi dikarenakan *term* yang semula ada pada data uji dan data latih digantikan *term lain* yang belum tentu ada pada data latih. Hal ini dapat menyebabkan proses klasifikasi dokumen menjadi salah, contohnya pada Tabel 6.8 No1, *term* "devisa" yang ada pada data uji juga terdapat pada data latih, sedangkan hasil terjemahan nya yaitu *term "*tukar*"* tidak terdapat pada data latih.

6.2.2 Permasalahan Penambahan *Query* Tidak Sesuai Konteks

Pada proses *query expansion*, kata ekspansi yang ditambahkan merupakan hipernim dan hiponim dari *term* awal. Kata ekspansi yang ditambahkan diambil dari *synset* yang ada pada WordNet. Hasil dari penambahan yang dilakukan memengaruhi proses klasifikasi yang dilakukan, salah satu dampak yang dihasilkan dari penambahan kata ekspansi adalah meningkatnya akurasi dikarenakan kata ekspansi yang ditambahkan sesuai dan ada pada data latih. Namun pada penelitian ini didapatkan beberapa kata ekspansi yang tidak sesuai dengan dokumen awal dalam hal konteks. Berikut contoh kata ekspansi yang ditambahkan, namun tidak sesuai konteks. Dokumen dan kata ekspansi dituliskan dalam Tabel 6.9.

Tabel 6.10Contoh Kata Ekspansi di Luar Konteks

Dokumen Asli	kata Ekspansi
hijau,daun,rilis,album, negara,asia	hijau,daun,lepas,album,negara,asia,warna,kromatik, warna,kromatik,warna,spektral,warna,spektral,hijau ,biru,biru,hijau,teal,sistem,bidang,tanah,potong,tan ah,bungkus,tanah,parcel,taman,hibur,pasar,dgn,atr aksi,tanah,senang

Pada Tabel 6.9 dokumen asli merupakan dokumen dengan kelas Entertaintment saat proses klasifikasi tanpa *query expansion* dilakukan, sistem mengklasifikasikan dokumen asli ke dalam kelas Entertaintment. Namun ketika proses *query* expansion dilakuan dengan *threshold* 0,1 didapatkan dokumen seperti pada Tabel 6.9 kolom Kata ekspansi yang di dalamnya terdapat beberapa *term* yang sama sekali tidak berkaitan dengan *term* maupun dokumen awal.

Contoh kata yang ditambahkan adalah " warna","taman","hibur", dan seterusnya. Kata-kata tersebut memang merupakan hipernim dan hiponim dari *term* awal, namun penambahannya sama sekali tidak memperhatikan konteks. Penambahan yang dilakukan justru menyebabkan hasil klasifikasi salah, dokumen yang telah ditambahkan kata ekspansi diklasifikasikan ke dalam kelas *travel*. Hal tersebut terjadi karena kata ekspansi yang walaupun sebenarnya hipernim dan hiponim dari *term,* namun pada data latih kata-kata tersebut merujuk pada kelas yang lain. Pada penelitian ini seluruh kata ekspansi yang ditambahkan tidak memperhatikan konteks.

6.3 Pengujian Jenis Kata yang Ditambahkan

Pada pengujian kali ini dilakukan proses *query expansion* dengan menambahkan hipernim atau hiponim saja. Hal ini dilakukan untuk mengetahui pengaruh dari jenis kata eskpansi yang ditambahkan. *Threshold* dari kata ekspansi yang ditambahkan menggunakan *threshold* pada penambahan kata ekspansi hipernim dan hiponim sekaligus yang menghasilkan akuasi tertinggi yaitu 0,1.

6.3.1 Penambahan Hiponim

Pada bagian ini dilakukan *query expansion* dengan menambahkan kata ekspansi yang merupakan hiponim dari *term* yang ada pada dokumen uji. *Threshold* yang digunakan dalam melakukan *query expansion* adalah 0,1. Tujuan dari pengujian ini adalah untuk mengetahui jenis *synset* yang efektif untuk ditambahkan sebagai kata ekspansi. Berikut adalah Tabel 6.10 yang berisi peritungan akurasi menggunakan *k-fold* pada dokumen setelah ditambahkan hiponim dari *term* .

Tabel 6.11 Akurasi Penambahan Hiponim

K	1	2	3	4	5	Sistem
Akurasi	76,25%	77,5%	51,81%	70%	56,25%	66,36%

Hasil yang didapatkan dari proses klasifikasi dengan penambahan hiponim sebagai kata ekspansi memberikan hasil akurasi sebesar 66,36%. Nilai akurasi yang didapatkan lebih besar jika dibandingkan dengan penambahan hipernim dan hiponim sekaligus pada *threshold* 0,1 yaitu sebesar 65,75%. Akurasi yang didapatkan pada proses klasifikasi dengan penambahan hiponim saja menunjukkan bahwa adanya penambahan kata ekspansi menurunkan akurasi, sehingga belum bisa didapatkan akurasi yang optimal dikarenakan beberapa masalah yang telah dibahas sebelumnya.

6.3.2 Penambahan Hipernim

Selanjutnya dilakukan *query expansion* dengan menambahkan kata ekspansi berupa hipernim dari *term* yang ada pada dokumen uji. *Threshold* yang digunakan

dalam melakukan *query expansion* adalah 0,1. Tujuan dari pengujian ini adalah untuk mengetahui jenis *synset* yang efektif untuk ditambahkan sebagai kata ekspansi. Berikut adalah Tabel 6.11 yang berisi peritungan akurasi menggunakan *k-fold* pada dokumen setelah ditambahkan hipernim dari *term* .

Tabel 6.12 Akurasi Penambahan Hipernim

K	1	2	3	4	5	Sistem
Akurasi	70%	81,25%	67,5%	61,25%	57,5%	67,5%

Hasil akurasi yang diperoleh saat penambahan hipernim sebagai kata ekspansi memiliki akurasi 67,5%. Hal ini lebih tinggi dibandingkan daripada akurasi penambahan hiponim saja dan penambahan keduanya sekaligus. Akurasi dari penambahan hipernim lebih tinggi dibandingkan denganhiponim dikarenakan kebany dari *term* hanya memiliki sedikit atau bahkan tidak memiliki hiponim(bentuk khusus kata). Sedangkan jika dibandingkan dengan penambahan keduanya, penambahan keduanya sekaligus dapat menyebabkan turunnya akurasi. Turunnya akurasi bisa terjadi ketika hiponim memberikan *term* yang cocok sedangkan hipernim tidak dan sebaliknya. Berikut adalah Tabel perbandingan ketika *query expansion* dilakukan dengan hiponim atau hipernim pada Tabel 6.12.

Tabel 6.13 Perbandingan Penambahan *Query*

Dokumen Asli	Dokumen + HIPONIM	Dokumen + HIPERNIM
perintah,negara, bebas,visa,wisata	perintah,negara,bebas,wisata, perintah,komando,tempur,udara, acc	perintah,negara,bebas, wisata,tindak,unit,militer, kuat,militer,kelompok, militer,paksa,wewenang, otorisasi,otorisasi
nyeri,punggung, depresi,sehat	sakit,depresi,sehat,sakit,sakit,sulit,saki t	sakit,depresi,sehat,gejala, somesthesia,somaesthesia,so matesthesia,sensasi, somatik,senang
ponsel,android, ngacir,hapus, facebook	telepon,android,ngacir,hapus, facebook,telepon,meja	telepon,android,ngacir, hapus,facebook,alat, elektronik

Pada Tabel 6.12 dapat dilihat bahwa jumlah kata yang ditambahkan saat penambahan hiponim tidak terlalu banyak, walaupun terjadi perbedaan jumlah kata ekspansi tidak menjamin bahwa hipernim lebih baik dari hiponim. Hanya saja seperti yang telah dibahas di atas bahwa kebany dari *term* hanya memiliki sedikit atau bahkan tidak memiliki hiponim (mengacu pada kamus yang digunakan). Terjadinya penurunan akurasi disebabkan banyak faktor dan salah

satu yang terjadi pada penambahan hipernim atau hiponim adalah penambahan kata ekspansi yang tidak memperhatikan konteks dokumen.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil, analisis serta pemBahasan dari penelitian implementasi Naïve Bayes dan *query expansion* berbasis WordNet untuk klasifikasi *tweet* berita pada Twitter maka dapat disimpulkan bahwa penambahan *query* pada dokumen sebelum dilakukan proses klasifikasi menurunkan tingkat akurasi. Pada proses klasfikasi tanpa menggunakan *query epansion* didapatkan akurasi sebesar 72%. Penambahan *query* berupa hipernim dan hiponim menghasilkan akurasi paling tinggi sebesar 65,75%, penambahan hipernim saja menghasilkan akurasi 67,5%, dan penambahan hiponim saja sebesar 66,3% semua pada *threshold* 0,1. Penurunan akurasi terjadi karena beberapa hal seperti hasil terjemahan dari Google Translate API mengubah *term* awal menjadi *term* lain yang tidak ada pada data latih, penambahan *query* sama sekali tidak memperhatikan konteks dokumen.

7.2 Saran

Berdasarkan pada hasil penelitian yang dilakukan maka dapat diberikan beberapa saran untuk penelitian selanjutnya. Saran-saran yang diberikan adalah sebagai berikut.

1. Penggunaan WordNet berbahasa Inggris kurang efektif saat *dataset* berbahasa Indonesia karena perlu melakukan proses penerjemahan, maka disarankan untuk menggunakan WordNet berbahsa Indonesia.
2. Jika menggunakan WordNet berbahasa Inggris maka perlu cara atau proses penerjemahan lain yang lebih efektif, selain menggunakan *google translate API*
3. Menambahkan metode untuk menyeleksi kata yang ditambahkan agar tetap sesuai dengan konteks dokumen awal.

Mencoba menggunakan relasi antar kata selain hiponim dan hipernim.

DAFTAR PUSTAKA

Anguita, D. et al., 2012. *The 'K' in K-fold Cross Validation.* Genova-Italy, Department of Biophysical and Electronic Engineering Via Opera Pia 11A, I-16145 Genova - Italy.

Asian, J., 2007. *Effective Techniques for Indonesian Text Retrieval,* Melbourne: School of Computer Science and Information Technology Science, Engineering, and Technology Portfolio RMIT University.

Bagaskoro, G. N., Fauzi, M. A. & Adikara, P. P., 2017. Penerapan Klasfikasi Tweets pada Berita Twitter Menggunakanakan Metode k-nearest neighbor dan Query Expansion Berbasis Distributional Semantic.. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* Issue 10, pp. 3849-3855.

Buzic, D. & Dobša, J., 2018. *Lyrics Classification Using Naive Bayes.* croatia, s.n., pp. Conference: MIPRO 2018, 41st International Convention on Information and Comunication Tachnoloy, Elecronics and Microelectornics.

Elberrichi, Z., Rahmoun, A. & Bentaalah, M. A., 2008. Using WordNet for Text Categorization. *The International Arab Journal of Information Technology,* pp. 16-24.

Fasilkom Universitas Indonesia, 2018. *WORDNET BAHASA INDONESIA.* [Online] Available at: http://bahasa.cs.ui.ac.id/wordnet/user/index.php [Diakses 16 Maret 2018].

Feldman, R. & Sanger, J., 2007. Introduction to Text Mining. Dalam: *The Text Mining HandBook : Advanced Approaches in Analyzing Unstructured Data.* New York , United States of America: Cambridge University Press, p. 1.

Feldman, R. & Sanger, J., 2007. Introduction to Text Mining. Dalam: *The Text Mining HandBook : Advanced Approaches in Analyzing Unstructured Data.* New York: United States of America by Cambridge University Press, p. 1.

Google Cloud Platform, 2018. *Cloud Translation API.* [Online] Available at: https://cloud.google.com/translate/ [Diakses 16 Maret 2018].

Korde, V., 2012. Text Classification and Classifiers:A Survey.. *International Journal of Artificial Intelligence & Applications (IJAIA),.*

kwak, H., Lee, c., Park, H. & moon, S., 2010. *What is Twitter , a social Network or a News Media ?.* Korea, Departmen of computer Science, KAIST.

Librian, A., 2016. *Stemming Bahasa Indonesia.* [Online] Available at: https://github.com/sastrawi/sastrawi/wiki/Stemming-Bahasa-

Indonesia
[Diakses 15 February 2018].

Li, W., Ganguly, D. & J.F.Jones, G., 016. Using WordNet for Query Expansion: ADAPT @ FIRE 2016 Microblog Track.

M.Ali Fuzi, M., Arifin, A. Z. & Yuniarty, A., 2014. Term Weighting Berbasis Indeks Buku dan Kelas untuk Perangkingan Dokumen Berbahasa Arab. *LONTAR KOMPUTER,* 5(2), pp. 435-442.

Mansuy, T. & Hilderman, R. J., 2006. *Evaluating WordNet Features in Text Classification Models.* Florida,USA, s.n., pp. 11-13.

McDonald, D. & Kelly, U., 2012. *The Value and Benefits of Text Mining.* United Kingdom: s.n.

Miller, G. A., Beckwith, R. & Fellbaum, C., 1993. Introduction to WordNet: An On-line Lexical Database,Derek Gross, and Katherine Miller. *International Journal of Lexicography.*

Nugroho, S. A., 2009. Query Expansion Dengan Menggabungkan Metode Ruang Vektor Dan Wordnet Pada Sistem Information Retrieval. *JURNAL INFORMATIKA,* 5(1).

O'Reilly, T. & Milstein, S., 2009. What is Twitter ?. Dalam: J. Wikert, penyunt. *The Twitter Book.* United States of America: O`Reilly Media , Inc, p. 7.

Perdana, R. S., Suprapto & Regasari, R., 2013. Pengkategorian Pesan Singkat Berbahasa Indonesia Pada Jejaring Sosial Twitter Dengan Metode Klasifikasi Naïve Bayes. *S1. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.*

Rahman, A., Wiranto & Doewes, A., 2017. Online News Classification Using Multinomial Naive Bayes. *Jurnal Ilmiah Teknologi dan Informasi,* pp. 32-38.

Sriram, B., 2010. *Short Text Classification In Twitter To Improve Information Filtering.* ohio: The Ohio State University.

Taheri, S. & Mammadov, M., 2013. Learning The Naive Bayes Classifier With Optimization Models. *International Journal of Applied Mathematics and Computer Science,* 23(4), pp. 787-795.

Tang, J., Wang, Y., Zheng, K. & Mei, Q., 2017. *End-to-end Learning for Short Text Expansion,* Michigan: Department of Informatics, University of California, Irvine.

Tanjung, J. P. R., Fauzi, M. A. & Indriati, 2017. Klasifikasi Tweets Pada Twitter Dengan Menggunakanakan Metode Fuzzy K-Nearest (Fuzzy K-NN) dan Query

Expansion Berbasis Apriori. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 1(5), pp. 405-414.

The Open University, 2018. *Being Digital Writing a Good Tweet,* Milten Keynes: The Open University.

Vidhya, K. & Aghila, G., 2010. A Survey of Naïve Bayes Machine Learning approach in Text Document Classification. *International Journal of Computer Science and Information Security,,* pp. 206-211.

Vijayaran, S., Ilamathi, J. & Nithya, 2015. Preprocessing Techniques for Text Mining - An Overview. *International Journal of Computer Science & Communication Networks,* 5(1), pp. 7-16.

Zhang, J., Deng, B. & Li, X., 2009. *Concept Based Query Expansion Using WordNet.* Beijing, Dept. Electronic Engineering, Tsinghua Univ. Beijing, 100084, China.