# ANALISIS SENTIMEN TWITTER MENGGUNAKAN *ENSEMBLE FEATURE* DAN METODE *EXTREME LEARNING MACHINE (ELM)*
# (STUDI KASUS: SAMSUNG INDONESIA)

**SKRIPSI**

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Alqis Rausanfita
NIM: 145150207111066

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

# PENGESAHAN

ANALISIS SENTIMEN TWITTER MENGGUNAKAN *ENSEMBLE FEATURE* DAN METODE *EXTREME LEARNING MACHINE (ELM)*
(STUDI KASUS: SAMSUNG INDONESIA)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Alqis Rausanfita
NIM: 145150207111066

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

| Dosen Pembimbing I | Dosen Pembimbing II |
|---|---|
| Putra Pandu Adikara, S.Kom., M.Kom. | Sigit Adinugroho, S.Kom., M.Sc. |
| NIP.19850725 2008121002 | NIK: 2016078807011001 |

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 didalam 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, 16 Juli 2018


Algis Rausanfita

NIM: 145150207111066

# KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul "Analisis sentimen Twitter Menggunakan *Ensemble Feature* dan Metode *Extreme Learning Machine (ELM)* (Studi Kasus: Samsung Indonesia)". Penulis menyampaikan terima kasih kepada pihak-pihak yang telah membantu dan membimbing penulis selama pengerjaan skripsi ini sedari awal hingga penulis dapat menyelesaikannya dengan baik, yaitu diantaranya:

1. Bapak Putra Pandu Adikara, S.Kom., M.Kom., selaku dosen pembimbing I yang telah memberikan bimbingan, arahan, ilmu, dan saran dalam penyelesaian skripsi ini.
2. Bapak Sigit Adinugroho, S.Kom., M.Sc., selaku dosen pembimbing 2 yang telah membimbing, memberikan ilmu, saran dan juga arahan selama penyusunan skripsi ini.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.Kom., Bapak Heru Nurwasito, Ir., M.Kom., Bapak Suprapto, S.T., M.T., Drs., M.T., Bapak Edy Santoso, S.Si, M.Kom., selaku Dekan, Wakil Dekan I, Wakil Dekan II, Wakil Dekan III Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Dosen Fakultas Ilmu Komputer yang telah mendidik dan memberikan ilmu selama penulis menempuh pendidikan di Fakultas ini.
5. Orang tua dan saudara penulis yang selalu mendukung penulis baik secara moril dan materiil sedari awal menempuh pendidikan hingga penulis dapat menyelesaikan skripsi ini.
6. Seluruh keluarga besar kedua orang tua penulis yang selalu memberikan semangat dan memberikan bantuan selama ini
7. Teman-teman cincau penulis, Riski Puspa Dewi Diangga Putri, Indah Mutia Ayudita, Diajeng Ninda Armianti, dan Annisa Fitriani Nur yang telah membantu serta memberi semangat penulis dari awal menempuh pendidikan S1 sampai penyelesaian skripsi.
8. Teman-teman,kakak-kakak Advokesma BEM TIIK Bersatu III serta keluarga besar BEM TIIIK Bersatu III yang telah memberikan semangat dan membantu penulis dalam menyelesaikan skripsi ini.
9. Teman-teman,kakak-kakak, dan adik-adik Advokesma EMIF 2015/2016 serta keluarga besar EMIF 2015/2016 yang telah memberikan semangat dan membantu penulis dalam menyelesaikan skripsi ini.
10. Teman-teman BPH EMIF 2016/2017 serta keluarga besar EMIF 2016/2017 yang telah memberikan semangat dan membantu penulis dalam menyelesaikan skripsi ini.

11. Seluruh teman-teman kelas TIF-F yang telah memberikan semangat dan membantu penulis dalam menyelesaikan skripsi ini.
12. Seluruh teman-teman angkatan 2014 Informatika yang telah memberikan semangat dan membantu penulis dalam menyelesaikan skripsi ini.
13. Seluruh teman-teman angkatan 2014 Fakultas Ilmu Komputer yang telah membantu dan memberikan semangat penulis salam menyelesaikan skripsi ini.

Malang, 16 Juli 2018

Penulis
qisfit@gmail.com

# ABSTRAK

**Alqis Rausanfita, Analisis Sentimen Twitter Menggunakan *Ensemble Feature* dan Metode *Extreme Learning Machine (ELM*) (Studi Kasus:Samsung Indonesia)**

**Pembimbing: Putra Pandu Adikara, S.Kom., M.Kom. dan Sigit Adinugroho, S.Kom., M.Sc.**

Seiring berkembangnya teknologi perusahaan-perusahaan mengalami masa perallihan dengan mengembangkan usahanya menjadi berbasis digital. Aktivitas bisnis yang sangat krusial dan memiliki dampak nyata pada pertumbuhan organisasi dan *Return Of Investment (ROI)* yaitu memahami dan menanggapi secara tepat sentimen dari pelanggan dengan melakukan analisis sentimen. Dengan adanya analisis sentimen perusahaan dapat menjadi pedoman untuk mengevaluasi produk, layanan, reputasi merek, dan perusahaan dapat menjadi pemimpin pasar yang didukung dengan kondisi pelanggan yang sangat emosional sehingga produk/layanan yang membuat kecewa akan kehilangan komitmen pelanggan bahkan pelanggan akan mengalami kesusahan untuk menemukan kembali pengalaman pelanggan jika suatu perusahaan tidak mempedulikan ungkapan sentimen pelanggan. Berdasarkan penjelasan tersebut, penelitian ini di lakukan menggunakan *ensemble feature dan Extreme Learning Machine* untuk analisis sentimen Twitter. Data yang digunakan dalam penelitian ini sebanyak 72 *tweet* dengan perbandingan jumlah data *training* dan *testing* 70:30 di mana jumlah data tiap kelas seimbang. Sebelum dilakukan klasifikasi data tersebut dilakukan *preprocessing,* pembobotan *ensemble feature, serta* pembobotan kata. Hasil Pengujian pada penelitian ini didapatkan jumlah *hidden neuron* terbaik sebanyak 5000, fungsi aktivasi terbaik adalah *sigmoid bipolar,* dan *ensemble feature* berpengaruh terhadap hasil akurasi. Analisis sentimen Twitter menggunakan *ensemble feature* dan metode *Extreme Learning Machine* pada studi kasus Samsung Indonesia tidak mendapatkan akurasi yang tinggi. Akurasi yang didapatkan hanya sebesar 42,857 %. Rendahnya akurasi disebabkan munculnya *sparse data matrix* sehingga terjadi *overfitting* yang kemudian berakibat rendahnya hasil akurasi.

Kata kunci: *Extreme Learning Machine,* Twitter, *ensemble feature*, analisis sentimen, Samsung Indonesia

# ABSTRACT

**Alqis Rausanfita, Twitter Sentiment Analysis Using Ensemble Features and Extreme Machine Learning Methods (ELM) (Case Study: Samsung Indonesia)**

**Supervisors: Putra Pandu Adikara, S.Kom., M.Kom. dan Sigit Adinugroho, S.Kom., M.Sc.**

*Along with the development of technology companies have a lifetime by expanding its business to be digital-based. Business activity is very important and has a real impact on organizational growth and Return Of Investment (ROI) is to understand and respond to customer's proper sentiment by conducting sentiment analysis. Given the company's sentiment analysis can be a guide to evaluate product, service, brand reputation, and company can be a market leader powered by very emotional customer conditions so that disappointing products / services will lose the customer's commitment even customers will find it difficult to restore the customer experience if companies are not concerned with customer sentiment. Based on the explanation, this research is done using ensemble feature and extreme machine learning for Twitter sentiment analysis. The data used in this research is 72 tweets with the ratio of training amount and 70:30 testing data where the amount of data per class is balanced. Before the data classification was done preprocessing, weighting ensemble features, and weighting of words. The result of this research is to get the best hidden neuron number as much as 5000, the best activation function is bipolar sigmoid, and ensemble feature affect the accuracy result. Twitter sentiment analysis using ensemble features and extreme machine learning methods in the case study Samsung Indonesia did not get high accuracy. Accuracy in getting only amounted to 42.857%. The low accuracy caused by the data matrix rarely results in overfitting which then results in low accuracy results*

*Keywords: Extreme Learning Machine,* Twitter, *ensemble feature*, sentiment analysis , Samsung Indonesia

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# DAFTAR LAMPIRAN

# BAB 1 PENDAHULUAN

## 1.1 Latar belakang

Seiring berkembangnya teknologi perusahaan-perusahaan mengalami masa perallihan dengan mengembangkan usahanya menjadi berbasis digital. Aktivitas bisnis yang sangat krusial dan memiliki dampak nyata pada pertumbuhan organisasi dan *Return Of Investment (ROI)* yaitu memahami dan menanggapi secara tepat sentimen dari pelanggan dengan melakukan analisis sentimen. Dengan adanya analisis sentimen perusahaan dapat menjadi pedoman untuk mengevaluasi produk, layanan, reputasi merek, dan perusahaan dapat menjadi pemimpin pasar. Hal tersebut didukung dengan kondisi yang terjadi pada zaman sekarang, dimana pelanggan sangat emosional sehingga menurut laporan Mckinsey & Company (Govindasamy, 2017) lebih dari 70% konsumen mengurangi komitmen ketika produk/layanan membuat kecewa, bahkan ketika suatu perusahaan tidak memedulikan ungkapan sentimen pelanggan maka akan mengalami kesusahan untuk menemukan kembali pengalaman pelanggan walaupun telah melakukan pembelaan. Selain itu pelanggan juga suka berbagi pengalaman tentang produk atau layanan yang telah digunakan melalui aplikasi selular di berbagai platform yang dapat menjadi sebuah rekomendasi bagi teman maupun keluarga. Menurut Nielsen (Govindasamy, 2017) 83% responden global mengatakan bahwa mereka mempercayai rekomendasi keluarga dan teman.

Salah satu platform yang dapat dijadikan sumber data dalam melakukan analisis sentimen yaitu Twitter (Amalia, 2017). Semakin meningkatnya jumlah pengguna maka ada kemungkinan jumlah pelanggan yang akan mengungkapkan sentimen positif atau negatif terhadap suatu produk atau layanan juga akan meningkat. Semakin banyaknya jumlah pelanggan yang mengungkapkan sentimen positif atau negatif terhadap suatu produk atau layanan maka semakin melimpah juga data yang dapat digunakan pada analisis sentimen sehingga perusahaan yang menyediakan tempat untuk pelanggan berbagi pengalaman dengan membuat akun Twitter resmi perusahaan dapat melakukan evaluasi diri dari hasil yang telah dilakukan analisis sentimen.

Penelitian tentang analisis sentimen telah dilakukan dengan menggunakan beberapa metode seperti *Support Vector Machine (SVM), Naive Bayes, Decision Tree* dan *Back-Propagation Artificial Neural Network (BPANN).* Pada penelitian Azizah, Ivan dan Budi (2015) menggunakan 3 metode klasifikasi yaitu *Naive Bayes*, *Support Vector Machine*, dan *Decision Tree* pada penelitian tentang sentimen Twitter untuk menganalisis reputasi merek penyedia layanan internet dan didapatkan hasil bahwa metode *Support Vector Machine* memberikan kinerja yang lebih baik dibandingkan 2 metode lainnya. Sebelumnya Sharma dan Dey (2012) telah melakukan penelitian tentang analisis sentimen dengan menggunakan metode *BPANN* dengan jumlah iterasi sebanyak 500. Penelitian tersebut menggunakan 2 dataset dengan hasil akurasi 89-90% untuk dataset ulasan hotel dan mendapatkan akurasi 95% menggunakan dataset ulasan film.

Namun pada penelitian Sharma dan Dey (2012) didapatkan hasil bahwa metode *Extreme Learning Machine* mendapatkan hasil akurasi tertinggi dibandingkan *dengan* metode *Support Vector Machine.*

Metode *Extreme Learning Machine* pertama kali diusulkan untuk jaringan saraf *single hidden layer* dan kemudian di kembangkan ke *SLFN* secara umum. *ELM* memiliki waktu komputasi yang lebih sedikit dibandingkan dengan metode *neural network* yang lainnya*,* di karenakan pada metode sebelumnya nilai bias dan bobot *input* perlu ditentukan pada setiap pelatihan hingga kondisi berhenti dicapai sehingga dilakukan beberapa kali iterasi untuk mencapai hasil terbaik yang membutuhkan waktu yang cukup lama. *ELM* dapat mendekati fungsi target berkelanjutan dan mengklasifikasikan setiap daerah yang terpisah. Dalam kasus multi kelas *ELM* memiliki kinerja generalisasi yang lebih baik dan membutuhkan waktu yang lebih sedikit dibandingkan dengan *SVM* yang tradisional dan *LS-SVM* (Huang et al., 2012).

Pada penelitian Siddiqua, Ahsan dan Chy (2017) tentang kombinasi R*ule-Based classifier* dengan *ensemble features* dan *machine learning* untuk analisis sentimen pada microblog didapatkan hasil bahwa *lexicon based features* paling signifikan pentingnya dalam sentimen analisis. Sebelumnya Davidov, Tsur dan Rappoport (2010) melakukan penelitian tentang kalimat sarkastik pada ulasan produk *online* yang menerapkan *punctuation* yang digunakan sebagai basis fitur yang jika digabungkan dengan fitur lain dapat meningkatkan nilai akurasi, presisi, *recall* dan *F-measure.*

Berdasarkan penjelasan yang telah dijelaskan sebelumnya, peneliti ingin melakukan penelitian yang berjudul "Analisis Sentimen Twitter Menggunakan *Ensemble Feature,* seleksi fitur, dan Metode *Extreme Learning Machine* (Studi Kasus: Samsung Indonesia)". Harapannya perusahaan dapat menjadikannya sebagai bahan untuk evaluasi kualitas produk atau layanan perusahaan.

## 1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijabarkan sebelumnya, terbentuk beberapa rumusan masalah pada penelitian ini, sebagai berikut:

1. Bagaimana tingkat akurasi analisis sentimen Twitter Samsung Indonesia menggunakan metode *ELM* dan *ensemble feature*?
2. Bagaimana pengaruh *ensemble feature* terhadap hasil klasifikasi analisis sentimen Twitter Samsung?

## 1.3 Tujuan

Tujuan dari diadakannya penelitian ini sebagai berikut:

1. Mendapatkan hasil akurasi analisis sentimen Twitter Samsung Indonesia menggunakan metode *ELM* dan *ensemble feature*.
2. Mengetahui pengaruh penggunaan *ensemble feature* terhadap hasil klasifikasi analisis sentimen.

## 1.4 Manfaat

Manfaat dari penelitian ini antara lain:

1. Pengguna dapat mengetahui respon masyarakat terhadap *brand* Samsung dengan mengelompokkan opini masyarakat ke dalam sentimen positif, negatif, dan netral.
2. Diharapkan hasil dari penelitian ini penulis dapat memberikan wawasan tentang klasifikasi teks kepada pembaca.
3. Diharapkan dapat membantu pihak Samsung Indonesia dalam mengetahui respon masyarakat sehingga memudahkan dalam upaya perbaikan kualitas.

## 1.5 Batasan masalah

Berikut ini batasan masalah dalam penelitian analisis sentimen Twitter Samsung Indonesia yang digunakan untuk memfokuskan pada pokok permasalahan yang dihadapi, diantaranya:

1. Kategori yang digunakan adalah positif, negatif, dan netral untuk analisis sentimen.
2. Obyek yang digunakan adalah Twitter dengan kata kunci "@Samsung_id" dan "Samsung_id" .
3. Data yang digunakan pada penelitian ini adalah *tweet* berbahasa Indonesia.
4. Penelitian ini hanya menggunakan metode *Extreme Learning Machine,* tidak membandingkan dengan metode *machine learning* yang lain.
5. Penelitian ini tidak melakukan normalisasi kata tidak baku.

## 1.6 Sistematika pembahasan

Penyusunan skripsi ini menggunakan sistematika penulisan dengan kerangka sebagai berikut :

**BAB 1 Pendahuluan**

Bab ini merupakan bab yang berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dari analisis sentimen Twitter Samsung serta sistematika penulisan skripsi.

**BAB 2 Landasan Kepustakaan**

Bab ini berisi uraian dan pembahasan tentang teori, konsep, model, metode sistem, atau analisis dari kepustakaan ilmiah yang berkaitan dengan analisis sentimen Twitter menggunakan metode *Extreme Learning Machine* dan *ensemble feature*.

**BAB 3 Metodologi Penelitian**

Bab ini membahas tentang metode serta langkah kerja yang digunakan dalam membangun sistem.

**BAB 4 Perancangan dan Implementasi**

Bab ini menampilkan rancangan sistem dan laporan hasil pelaksanaan metode penelitian serta menyajikan data yang mendukung hasil penelitian.

**BAB 5 Hasil dan Pembahasan**

Bab ini berfungsi untuk menerjemahkan makna dari hasil yang diperoleh dalam penelitian.

**BAB 6 Penutup**

Bab ini memuat kesimpulan yang merupakan penegasan dari yang telah dijelaskan sebelumnya serta saran-saran untuk pengembangan selanjutnya.

# BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan menguraikan pembahasan tentang *review* penelitian-penelitian terdahulu yang berhubungan dengan permasalahan pada penelitian ini serta teori-teori pendukung yang digunakan pada penelitian ini.

## 2.1 Tinjauan Penelitian yang Terdahulu

Ada beberapa penelitian yang membahas tentang analisis sentimen. Penelitian tentang analisis sentimen telah dilakukan oleh Faradhillah, Kusumawardani dan Hafidz (2016) tentang klasifikasi analisis sentimen Twitter pada akun resmi Pemerintah kota Surabaya membuat sebuah aplikasi yang dapat mengklasfikasikan sentimen masyarakat yang terdapat pada akun Twitter SapawargaSby dengan menggunakan metode *Naive Bayes* dan *Support Vector Machine (SVM)*. Penelitian tersebut bertujuan untuk melihat kinerja pelayanan publik di Kota Surabaya. Penelitian tersebut melakukan eksperimen dengan 2 skenario dengan beberapa model dan mendapatkan hasil bahwa metode *SVM* memiliki nilai akurasi yang lebih tinggi dibandingkan *Naive bayes* dengan nilai akurasi sebesar 78,66%, *precision* sebesar 99,85%, *recall* sebesar 79,67% dan *f-measure* sebesar 88,6%.

Azizah, Ivan dan Budi (2015) melakukan penelitian yang membahas tentang analisis sentimen untuk mengukur tingkat kepuasan pengguna layanan telekomunikasi seluler Indonesia pada lima produk yang dikeluarkan yaitu 3G, 4G, sms, telepon, dan layanan internet, yang mana terdapat tiga buah penyedia layanan telekomunikasi seluler di Indonesia yang digunakan pada penelitian yaitu Indosat, Telkomsel, dan XL Axiata. Data yang digunakan sebanyak 10.000 pesan, dimana setelah data didapatkan akan dilakukan proses preprocessing, setelah itu dilakukan seleksi fitur berupa *POS Tagger*, dan melakukan klasifikasi dengan membandingkan 3 metode. Ketiga metode tersebut yaitu *Naive Bayes, Decision Tree,* dan *Support Vector Machine*. Akurasi tertinggi didapatkan ketika menggunakan metode *SVM* dibandingkan dengan kedua metode lainnya.

Penelitian selanjutnya dilakukan oleh Sharma dan Dey (2012) yang meneliti tentang analisis sentimen dengan menggunakan *dataset* yaitu ulasan film dan ulasan hotel. Penelitian tersebut melakukan seleksi fitur *Information Gain, lexicon,* dan metode *Back-Propagation Artificial Neural Network (BPANN).* Pada proses *Back-Propagation Artificial Neural Network (BPANN)* dilakukan iterasi sebanyak 500 kali.

Pada penelitian Roul et al. (2015) meneliti klasifikasi teks dengan membandingkan metode *Support Vector Machine, Extreme Learning Machine, KNN, Naive Bayes,* dan *Decision Tree.* Selain menggunakan metode klasifikasi, penelitian tersebut juga menggunakan seleksi fitur seperti *Chi-Square, Bi Normal Separation,* dan *Information Gain.* pada penelitian tersebut didapatkan hasil akurasi tertinggi sebesar 82,55% saat menggunakan metode *Chi-Square* dan *Extreme Learning Machine.* Penelitian tersebut juga menyatakan bahwa *Extreme*

*Learning Machine* merupakan teknik klasifikasi teks yang populer pada zaman sekarang.

Pada penelitian Siddiqua, Ahsan, dan Chy (2016) yang mengkombinasikan *Rule-based Classifier* dengan *Ensemble feature Set* dan *Machine Learning* pada microblog. Pada *supervised learning* penelitian ini mengekstrak *emoticon lexicon,* daftar fitur analisis sentimen, dan *Bag-of-Words (BoW)* yang terdapat pada *tweet*. Setelah dilakukan ekstrak fitur, dilakukan seleksi fitur mengaplikasikan *the Chi-Square* ( $\chi 2$ ) dan *Information Gain (IG)*. Dari hasil perhitungan pada ekstrak fitur didapatkan hasil bahwa *lexicon based features* memiliki hasil yang signifikan dibandingkan fitur lainnya. Selain itu, penelitian ini menerapkan *Rule-based classifier* berdasar pada kemunculan kata-kata dan *emoticon*. Pada *rule-based* didapatkan hasil terbaik pada *run9* dengan mengklasifikasikan positif atau negatif atau *unkown* yang kemudian dilakukan *majority voting* pada *run3, run4, run6, run7,dan run8.*

Untuk objek penelitian, metode serta perbandingan antara penelitian yang akan dilakukan dengan penelitian sebelumnya dijelaskan pada Tabel 2.1

**Tabel 2.1 Perbandingan penelitian sebelumnya dan penelitian skripsi**

| No | Judul | Objek | Metode – Akurasi | Perbandingan | |
|---|---|---|---|---|---|
| | | | | Studi Kepustakaan | Skripsi Penulis |
| 1 | Extreme Machine Learning in The Field Of The Text (Roul et al., 2015) | Text | *ELM - 82,55%, SVM - 81,87%, KNN - 52,44%, Naive Bayes - 79,50%, Decision Tree / 59,57%* | Text, seleksi fitur, ELM, SVM, KNN, Naive Bayes, Decision Tree | Twitter Samsung Indonesia, Sentimen positif, sentimen negatif, sentimen netral, *ELM, ensemble features* |
| 2 | Combining a Rule-based Classifier With Ensemble Of Feature Sets dan Machine Learning Techniques for Sentiment Analysis on Microblog (Siddiqua, U., A., Ahsan, T., Chy, A., N., 2016.) | Microblog | *Rule-based Classifier, supervised classifier – 87.74%* | Microblog, ensemble fitur, rule based classifier, supervised classifier | Twitter Samsung Indonesia, Sentimen positif, sentimen negatif, sentimen netral, *ELM, ensemble features* |

**Tabel 2.1 Perbandingan penelitian sebelumnya dan penelitian skripsi (lanjutan)**

| No | Judul | Objek | Metode – Akurasi | Perbandingan | |
|---|---|---|---|---|---|
| | | | | Studi Kepustakaan | Skripsi Penulis |
| 3 | Eksperimen Sistem Klasifikasi Analisa Sentimen Twitter Pada Akun Resmi Pemerintah Kota Surabaya Berbasis Pembelajaran Mesin (Faradhillah, Kusumawardani and Hafidz, 2016) | Akun resmi pemerintah kota Surabaya | *Naive Bayes , SVM – 78,66%* | Akun resmi pemerintah kota Surabaya, sentimen positif, sentimen negatif, sentimen netral, *Naive Bayes*, SVM | Twitter Samsung Indonesia, Sentimen positif, sentimen negatif, sentimen netral, *ELM, ensemble features* |
| 4 | Twitter Sentiment to Analyze Net Brdan Reputation of Mobile Phone Providers (Azizah, Ivan and Budi, 2015) | Pengguna penyedia layanan telekomunikasi selular Indonesia | *SVM – 82,40%, Naive bayes – 78,90%,* dan *Decision tree-72,90%* | Reputasi merek penyedia layanan telekomunikasi selular Indonesia, SVM, Naive Bayes, Decision Tree | Twitter Samsung Indonesia, Sentimen positif, sentimen negatif, sentimen netral, *ELM, ensemble features* |
| 5 | *A document-level sentiment analysis approach using artificial neural network dan sentiment lexicons* (Sharma dan Dey, 2012) | Review hotel dan film | *Back-Propagation Artificial Neural Network (BPANN), Information Gain – 95%* | *Back-Propagation Artificial Neural Network (BPANN),* seleksi fitur, ekstraksi fitur, sentimen positif, sentimen negatif | Twitter Samsung Indonesia, Sentimen positif, sentimen negatif, sentimen netral, *ELM, ensemble features* |

## *2.2 Text Mining*

*Text mining* adalah sebuah penggalian data teks yang didapatkan dari kumpulan kalimat atau dokumen dengan tujuan mencari maksud dari konten dan kemudian dilakukan analisa untuk mendapatkan sebuah informasi. Dengan menggabungkan teknik *data mining*, *machine learning, natural language processing, information retrieval*, dan *knowledge management* maka teks mining dapat menjadi solusi pada krisis informasi yang berlebihan (Susilowati et al., 2015). Hasil dari *text mining* dapat di gunakan untuk melakukan analisis sentimen pada Twitter Samsung Indonesia sehingga kecenderungan opini pelanggan Samsung terhadap loyalitas pelanggan akan didapatkan.

## 2.3 Analisis sentimen

Analisis sentimen (SA) atau *Opinion Mining* (OM) adalah pembelajaran komputasi dari sebuah opini, sikap, dan emosi seseorang terhadap suatu entitas. Entitas dapat mewakili individu, peristiwa, atau topik (Medhat, Hassan and Korashy, 2014). Menurut Liu (2012), Analisis sentimen adalah bidang studi yang menganalisis opini, sentimen, evaluasi, penilaian, sikap, dan emosi orang-orang terhadap entitas seperti produk, layanan, organisasi, individu, isu, peristiwa, topik, dan atribut. Analisis sentimen terdiri dari 3 level, yaitu:

a. Level dokumen. Pada level ini diasumsikan bahwa setiap dokumen mengungkapkan opini yang bersifat obyektif. Pada level dokumen, biasanya sentimen ditunjukkan dengan sentimen positif atau sentimen negatif. Contohnya: ulasan dari suatu produk, sistem menentukan ulasan tersebut menunjukkan opini positif atau negatif mengenai produk tersebut. Pada level ini diasumsikan bahwa sebuah dokumen menunjukkan opini pada satu entitas.
b. Level kalimat. Analisis pada level kalimat memiki keterkaitan dengan klasifikasi subjektifitas yang membedakan kalimat yang menunjukkan informasi yang faktual (kalimat objektif) dengan kalimat yang menunjukkan pandangan yang subjektif (kalimat subjektif).
c. Level Entitas dan aspek. Tujuan pada level ini untuk menemukan sentimen pada entitas dan/atau aspek kalimat tersebut.

Pada penelitian ini kita akan melakukan analisis pada level dokumen tentang *brand* Samsung pada Twitter.

Sebelum dilakukannya pengklasifikasian dokumen. Maka dokumen perlu dilakukan *preprocessing text* untuk menghilangkan *noise* yang dapat memengaruhi hasil klasifikasi .

## *2.4 Preprocessing text*

Data yang didapatkan perlu diolah agar menjadi lebih terstruktur. Salah satu tahapan teks mining untuk mengolah data menjadi terstruktur adalah *preprocessing*. *Preprocessing* dilakukan untuk menghilangkan *noise* yang terdapat pada data (Azizah, Ivan dan Budi, 2015). Pada tahapan ini data disiapkan menjadi data yang siap dianalisis. *Preprocessing* pada penelitian ini terdiri dari beberapa tahapan yaitu: proses tokenisasi, *filtering*, dan *part-of-speech*.

### 2.4.1 Tokenisasi

Tokenisasi adalah proses memisahkan kata (Susilowati et al., 2015). Contoh tahap tokenisasi ditunjukkan pada Tabel 2.2.

**Tabel 2.2 Contoh tahap tokenisasi**

| Data Masukan | Hasil Tokenisasi |
|---|---|
| @Samsung_ID kak disebelah kanan hp saya ada warna ungu, padahal baru 2 minggu beli. Ga jatuh ga ketidurin. Itu kenapa ya? | [‘@Samsung_ID ’, ‘kak ’, ‘disebelah‘, ‘kanan’, ‘hp’, ‘saya’, ‘ada’, ‘warna’, ‘ungu,’, ‘padahal’, ‘baru’, ‘2’, ‘minggu’, ‘beli’, ‘Ga’, ‘jatuh’, ‘ga’, ‘ketiduran.’, ‘Itu’, ‘kenapa’, ‘ya?’] |

### *2.4.2 Filtering*

*Filtering* atau *stopword removal* adalah Proses memilih kata-kata yang dianggap penting di dalam dokumen dengan membuang kata-kata yang dianggap tidak penting (Azizah, Ivan and Budi, 2015). Contoh tahap filterisasi ditunjukkan pada Tabel 2.3.

**Tabel 2.3 Contoh tahap *filtering***

| Data inputan | Hasil Filtering |
|---|---|
| [ ‘@Samsung_ID ’, ‘kak ’, ‘disebelah‘, ‘kanan’, ‘hp’, ‘saya’, ‘ada’, ‘warna’, ‘ungu,’, ‘padahal’, ‘baru’, ‘2’, ‘minggu’, ‘beli’, ‘Ga’, ‘jatuh’, ‘ga’, ‘ketiduran.’, ‘Itu’, ‘kenapa’, ‘ya?’ ] | [‘@Samsung_ID’, 'kak', 'disebelah', 'kanan', 'hp', 'warna', 'ungu,', 'baru', ‘2’, 'minggu', 'beli', 'Ga', 'jatuh','ga', 'ketidurin', 'Itu', 'ya?’] |

### *2.4.3 Part of Speech (POS)*

*Part of speech (POS) tag* adalah suatu teknik dalam tata bahasa untuk mengkategorikan kelas kata, seperti kata benda, kata sifat, kata keterangan, kata kerja, dll. *POS Tagger* memiliki arti suatu aplikasi yang dapat melakukan proses anotasi *Part of Speech* pada suatu teks secara otomatis. *POS Tagger* Bahasa Indonesia telah dikembangkan oleh Rashel et al. (2014). *POS Tagger* Bahasa Indonesia tersebut dikembangkan menggunakan pendekatan *Rule-Based* berdasarkan aturan tata bahasa Indonesia. *POS Tagger* tersebut akan memberikan anotasi *POS tag* pada setiap kata di dalam dokumen, namun jika ada kata yang dianggap ambigu dan tidak didapatkan *rule-based* yang sesuai maka akan dibiarkan ambigu.

Contoh hasil *POS Tagger* yang dikembangkan oleh Rashel et al. (2014):

Kalimat : “Min @Samsung_ID s7 edge baterainya bisa diganti nggak ya? Baterai saya menggembung”

Hasil POS Tagger : Min/JJ @Samsung_ID/X s7/CD edge/X baterainya/NN bisa/PRP diganti/MD nggak/VB ya/NEG ?/X NEG/Z Baterai/NN saya/PRP menggembung/VB

## 2.5 Pembobotan Kata

Pembobotan kata bertujuan memberikan bobot pada kata berdasarkan jumlah frekuensi kemunculan kata. Menurut Manning, C & Raghavan (2009), pembobotan kata terdiri dari beberapa tahap yaitu:

### 2.5.1 Term frequency ($tf_{t,d}$)

Term frequency merupakan banyaknya kemunculan term/kata/token dalam sebuah dokumen. Bobot dari $tf_{t,d}$ atau biasa disebut *wtftd* dapat dihitung menggunakan rumus Persamaan 2.1.

$$tf_{t,d} = \begin{cases} 1+^{10}\log tf_{t,d} \text{ , jika } tf_{t,d} > 0 \\ 0, \text{ lainnya} \end{cases} \quad (2.1)$$

### 2.5.2 Document frequency ($df_t$)

*Document frequency* merupakan jumlah dokumen yang mengandung suatu kata/term/token.

### 2.5.3 Inverse document frequency ($idf_t$)

*Inverse document frequency* merupakan kebalikan dari *Document frequency*. Kata yang sering muncul dibanyak dokumen dianggap tidak penting, sedangkan kata yang jarang muncul di banyak dokumen memiliki nilai *inverse document frequency* yang tinggi. $Idf_t$ dapat dihitung menggunakan Persamaan 2.2.

$$idf_t = log_{10} \frac{N}{df_t} \quad (2.2)$$

dimana N menyatakan banyaknya dokumen yang ada

### 2.5.4 Term frequency – inverse document frequency (Wt,d)

Bobot *TF-IDF* dari sebuah term/token/kata merupakan hasil perkalian dari bobot *tf* dengan *Inverse document frequency*. $W_{t,d}$ dapat dihitung dengan menggunakan Persamaan 2.3.

$$W_{t,d} = tf_{t,d} X\ idf_t \quad (2.3)$$

## 2.6 Ekstraksi Fitur

Siddiqua, Ahsan, dan Chy (2017) mengekstrak beberapa fitur untuk digunakan pada proses analisis sentimen. Fitur-fitur tersebut dikategorikan ke dalam *Twitter specific features, textual features, Part-of-Speech features, lexicon based features,*dan *Bag-of-Words (BoW) feature* yang ditunjukkan pada Tabel 2.4

**Tabel 2.4 Fitur pada analisis sentimen**

| *Tipe* | ID | Keterangan Fitur |
|---|---|---|
| *Specific Features* | F 1 | Apakah tweet berisi #hashtag atau tidak. |
| | F 2 | Apakah tweet adalah tweet atau tidak. |
| | F 3 | Apakah tweet berisi nama pengguna atau tidak. |
| | F 4 | Apakah tweet berisi URL atau tidak. |
| *Textual Features* | F 5 | TweetLength: Jumlah kata dalam tweet. |
| | F 6 | AvgWordLength: Rata-rata panjang karakter kata-kata. |
| | F 7 | Jumlah tdana tanya tersedia di tweet. |
| | F 8 | Jumlah tdana seru tersedia di tweet. |
| | F 9 | Jumlah tdana kutip tersedia di tweet. |
| | F 10 | Jumlah kata yang dimulai dengan huruf besar dalam tweet |
| | F 11 | Apakah tweet berisi emoticon positif atau tidak. |
| | F 12 | Apakah tweet berisi emoticon negatif atau tidak. |

**Tabel 2.4 Fitur pada analisis sentimen (Lanjutan)**

| *Tipe* | ID | Keterangan Fitur |
|---|---|---|
| *Pos (Part-of-Speech)* | *F13* | Jumlah kata benda POS tersedia dalam tweet. |
| | F 14 | Jumlah POS kata sifat yang tersedia di tweet. |
| | F 15 | Jumlah kata kerja POS tersedia dalam tweet. |
| | F 16 | Jumlah POS keterangan tersedia di tweet. |
| | F 17 | Jumlah kata seru POS tersedia dalam tweet. |
| | F 18 | Persentase POS kata benda dalam tweet. |
| | F 19 | Persentase POS kata sifat dalam tweet. |
| | F 20 | Persentase kata kerja POS di tweet. |
| | F 21 | Persentase POS keterangan di tweet. |
| | F 22 | Persentase kata seru POS di tweet. |
| *Lexiconbased Features* | F 23 | Jumlah kata-kata positif yang tersedia di tweet. |
| | F 24 | Jumlah kata-kata negatif yang tersedia di tweet. |
| | F 25 | Jumlah kata-kata positif dengan POS kata sifat. |
| | F 26 | Jumlah kata-kata negatif dengan POS kata sifat. |
| | F 27 | Jumlah kata-kata positif dengan kata kerja POS. |
| | F 28 | Jumlah kata-kata negatif dengan kata kerja POS. |
| | F 29 | Jumlah kata-kata positif dengan POS kata keterangan |
| | F 30 | Jumlah kata-kata negatif dengan POS kata keterangan |
| | F 31 | Persentase kata-kata positif dengan POS kata sifat. |
| | F 32 | Persentase kata-kata negatif dengan POS kata sifat. |
| | F 33 | Persentase kata-kata positif dengan kata kerja POS. |
| | F 34 | Persentase kata-kata negatif dengan kata kerja POS. |
| | F 35 | Persentase kata-kata positif dengan POS kata keterangan |
| | F 36 | Persentase kata-kata negatif dengan POS kata keterangan |
| | F 37 | Jumlah kata intensifier tersedia di tweet. |

Sumber: Siddiqua, Ahsan, dan Chy (2017)

Pada penelitian ini kita akan menggunakan fitur *bag-of-words dan 37* fitur pada penelitian Siddiqua, Ahsan, dan Chy (2017)

## 2.7 Normalisasi *Min-Max*

Menurut Junaedi, Budianto, dan Maryati, (2011), Normalisasi merupakan proses transformasi yang mengubah atribut numerik menjadi lebih kecil dengan cara menskalakan atribut numerik tersebut dalam rentan yang lebih kecil seperti -1.0 sampai 1.0 atau 0.0 sampai 1.0. Ada beberapa metode yang dapat diterapkan untuk normalisasi data, yaitu *min-max, z-score*, dan normalisasi dengan penskalaan desimal. Pada penelitian ini normalisasi dilakukan dengan menggunakan metode *min-max*.

Normalisasi *min-max* memetakan sebuah nilai dari sebuah atribut ke dalam suatu batasan. Menurut Wirawan (2015, disitasi dalam Rofiqoh, 2017), metode normalisasi min-max memiliki kelebihan yaitu terdapat keseimbangan nilai perbandingan antara nilai data sebelum dinormalisasi dengan nilai data yang telah dinormalisasi. Rumus perhitungan normalisasi min-max ditunjukkan pada Persamaan 2.4.

$$v' = \frac{v - min}{\max - min}(maks_{baru} - min_{baru}) + min_{baru} \quad (2.4)$$

Keterangan:

v' = nilai dari data baru hasil normalisasi

v = nilai dari data yang akan dinormalisasi

min = nilai minimum data

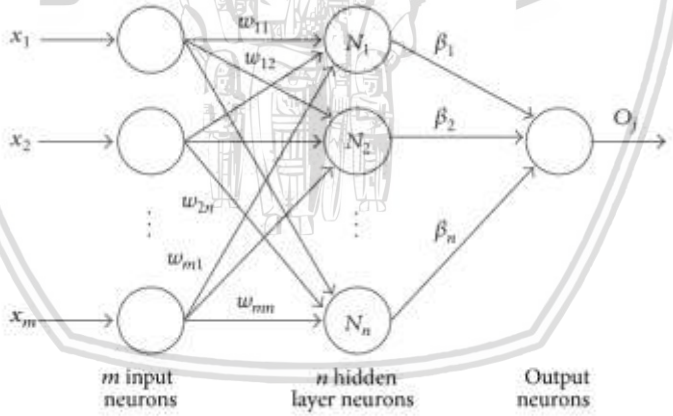
maks = nilai maksimum data

$maks_{baru}$ = nilai maksimum yang diharapkan didapatkan pada normalisasi

$min_{baru}$ = nilai minimum yang diharapkan didapatkan pada normalisasi

## *2.8 Extreme Learning Machine (ELM)*

*Extreme Learning Machines* di perkenalkan pertama kali oleh Huang pada tahun 2004 merupakan salah satu jenis jaringan saraf tiruan yang dapat di *training* dalam melakukan fungsi komplek di berbagai bidang dengan resolusi *linear* dan *non linear*, selain itu *ELM* dibuat untuk mengatasi kelemahan-kelemahan *feedforward* terutama dalam kecepatan belajar (Huang,et al., 2006 disitasi dalam Ariestyani, 2017). Berdasarkan teori Bartlett, bahwa *feedforward neural network* dapat menjangkau kesalahan pelatihan terkecil, semakin kecil nilai *norm* bobot, maka kinerja generalisasi jaringan akan semakin baik. (Huang, et al., 2012 disitasi dalam Ariestyani, 2017). Struktur *ELM* ditunjukkan pada Gambar 2.1.




**Gambar 2.1 Struktur Extreme Learning Machine**

Sumber: You et al. (2014)

Pada proses pelatihan *Extreme Learning Machine* membutuhkan target output (T) untuk klasifikasi dan input data (D) sebagai *training set*. *Extreme Learning Machine* memiliki beberapa tahap, sebagai berikut:

1. Melakukan perulangan di mulai dari i=1 dan berakhir pada N sejumlah data serta nilai bobot masukan ($W_i$) dengan *range* nilai -1 hingga 1 dan *hidden bias* ($b_i$) dengan *range* nilai 0 hingga 0.1 dipilih secara acak.
2. Menghitung matriks output *hidden neuron* ($H_{init}$)

$$H_{init} = \chi . \omega^T + b \quad (2.5)$$

3. Hitung matriks *output hidden neuron* dengan fungsi *aktivasi sigmoid biner (H)* dengan nilai eksponen = 2.718

$$H = \frac{1}{(1+exp-^H)} \quad (2.6)$$

4. Menghitung *pseudo-inverse* dengan *Moore-Penrose*

$$H^+ = (H^T . H)^{-1} . H^T \quad (2.7)$$

5. Menghitung bobot *output* dengan nilai $T = [t_1, \dots, t_N]^T$

$$\beta = H^+ . T \quad (2.8)$$

6. Menghitung Y prediksi (keluaran *output layer*)

$$\hat{Y} = H . \beta \quad (2.9)$$

## 2.9 Evaluasi

Desai dan Mehta (2016, disitasi dalam Rofiqoh, 2017) menyatakan bahwa pada umumnya ada empat indikator yang dihitung berdasarkan *confusion matrix* untuk dapat digunakan mengukur kemampuan teknik klasifikasi sentimen. Empat indikator tersebut yaitu: Akurasi, *Precision*, *Recall*, dan *F-measure. Confusion matrix* ditunjukkan pada Tabel 2.5.

**Tabel 2.5 *Confusion matrix***

| | | Nilai Prediksi | |
|---|---|---|---|
| | | *True* | *False* |
| Nilai Aktual | *True* | Nilai dari *True Positive (TP)* | Nilai dari *False Negative (FN)* |
| | *False* | Nilai dari *False Positive (FP)* | Nilai dari *True Negative (TN)* |

Akurasi merupakan nilai ketepatan record yang benar dari hasil prediksi pengujian dibandingkan dengan nilai aktual. Jika hasil akurasi berupa 100% maka menunjukkan bahwa kondisi yang diprediksi benar sesuai dengan aslinya (Rofiqoh, 2017). Akurasi dapat dihitung menggunakan rumus pada Persamaan 2.10.

$$\text{Akurasi} = \frac{TP+TN}{TP+FN+FP+TN} \quad (2.10)$$

*Precision* atau *confidence* adalah rasio antara *item* yang diprediksi positif yang juga positif pada sebenarnya dengan semua item yang diprediksi positif. *Precision* dapat dihitung menggunakan rumus pada Persamaan 2.11.

$$Precision = \frac{TP}{TP+FP} \quad (2.11)$$

*Recall* atau *Sensitivity* adalah rasio antara *item* yang diprediksi positif yang juga positif pada sebenarnya dengan semua item yang aktual positif. Rumus untuk mendapatkan nilai *recall* ditunjukkan pada Persamaan 2.12.

$$Recall = \frac{TP}{TP+FN} \tag{2.12}$$

*F-measure* atau *F-Score* adalah pengukuran yang menilai timbal balik antara *precision* dan *recall* (bobot *harmonic mean* antara presisi dengan *recall*). Nilai dari *F-measure* didapatkan dengan menghitung menggunakan rumus pada Persamaan 2.13.

$$F - measure = \frac{2 \times precision \times recall}{precision+recall} \tag{2.13}$$

# BAB 3 METODOLOGI

Pada bab ini berisi penjelasan tahapan-tahapan yang dilakukan dalam penelitian secara sistematik dan rancangan sistem yang berupa analisis kebutuhan sistem.

## 3.1 Tipe penelitian

Penelitan ini merupakan penelitian bertipe nonimplementatif analitik. Pada penelitian ini dilakukan penggalian informasi untuk menunjang aktivitas pada bidang ekonomi berbasis digital. Aktivitas yang dilakukan yaitu penggalian sentimen pelanggan yang terdapat pada Twitter yang kemudian dilakukan analisis, yang mana dengan dilakukannya analisis sentimen tersebut diharapkan dapat menjadi bahan evaluasi perusaahaan sehingga perusahaan dapat mempertahankan atau menambah pelanggan baru. Dengan bertambahnya pelanggan maka pendapatan perusahaan pun akan bertambah juga.

## 3.2 Strategi metode

Studi kasus pada peneltian ini yaitu sentimen yang terdapat Twitter Samsung Indonesia. Samsung meyediakan akun resmi pada beberapa platform yang dapat digunakan pelanggan untuk menyampaikan sentimennya terhadap produk yang telah digunakan. Dengan di sediakannya wadah untuk menampung sentimen, pelanggan tidak malu untuk mengungkapkan sentimen, baik sentimen positif ataupun negatif, dari pengalamannya setelah menggunakan produk atau layanan sehingga terciptalah data yang cukup untuk dilakukan analisis sentimen. Agar hasil sistem analisis sentimen dapat akurat maka data pada penelitian telah dikelompokkan sentimennya oleh pakar.

## 3.3 Subjek penelitian

Data pada penelitian ini didapatkan dari sentimen yang diungkapkan pelanggan Samsung setelah menggunakan produk atau layanan pada s*martphone* Samsung. Kemudian data yang didapatkan dikelompokkan ke dalam tiga kelas yaitu: positif, negatif, dan netral oleh pakar yang dianggap mengerti dan memahami maksud yang tersirat maupun tersurat pada sebuah kalimat. Pakar tersebut merupakan guru Bahasa Indonesia pada sekolah menengah atas yang bernama Dra. Dian Soegiharti.

## 3.4 Lokasi penelitian

Penelitian ini dilakukan di laboratorium riset Fakultas Ilmu Komputer Universitas Brawijaya Malang (FILKOM UB).

## 3.5 Pengumpulan data

Pada penelitian ini data Twitter didapatkan dengan cara *crawling* menggunakan R dengan memanfaatkan API Twitter. Ada beberapa tahap dalam melakukan *crawling* data menggunakan R, yaitu:

1. Menginstall *package* Twitter
2. Melakukan *authentication* Twitter
3. Melakukan pencarian *tweet* dengan kata kunci "@Samsung_id" dan "Samsung_id"
4. Melakukan penyimpanan data hasil pencarian *tweet* ke dalam excel

Pada penelitian ini pengambilan data Twitter dilakukan pada bulan Desember 2017 sampai Februari 2018. Dari hasil *crawling* Twitter tersebut peneliti hanya mengambil data yang berbahasa Indonesia dengan menggunakan R dan menyeleksi secara manual *tweet* yang membahas tentang ponsel cerdas. Berdasarkan penelitian Ariadi dan Fithriasari (2015, disitasi dalam Rofiqoh, 2017) dalam pembagian jumlah data, proses ini menghasilkan 862 data yang akan dibagi menjadi dua jenis data dengan pembagian 70% digunakan sebagai data *training* dan 30% digunakan sebagai data *testing*.

## 3.6 Peralatan Pendukung yang digunakan

Penelitian ini membutuhkan peralatan pendukung yaitu hardware dan kebutuhan *software*. Kebutuhan tersebut akan dijabarkan sebagai berikut:

### 3.6.1 Kebutuhan *hardware*

- Laptop dengan *processor* AMD A8-7410 APU

### 3.6.2 Kebutuhan *software*

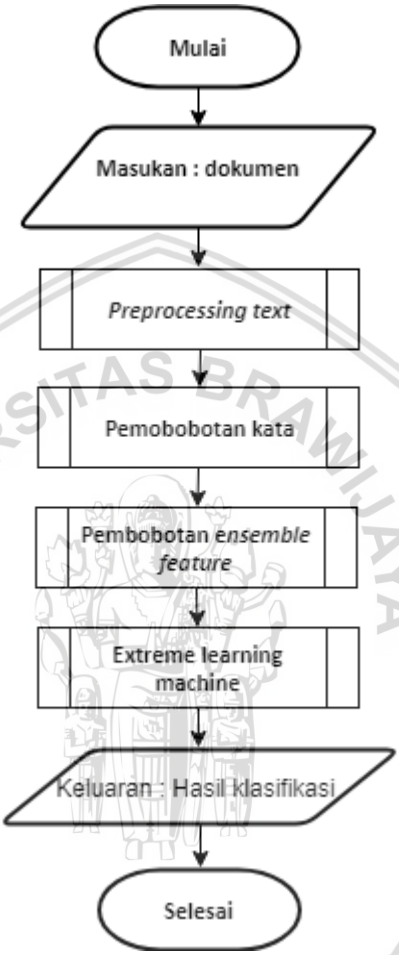
- Sistem operasi: Windows 10
- Bahasa pemrograman : R, Python
- Editor : Rstudio, IDLE Python
- Library : Sastrawi, Twitter

## 3.7 Perancangan sistem

Sistem pada penelitian ini dirancang dengan pendekatan berorientasi objek menggunakan bahasa pemrograman Python. Data yang dibutuhkan sistem berupa data *training*, data *testing*, data kamus sentimen (Wahid dan Azhari, 2016), dan data *stopword* (Tala, 2003). Cara kerja dari sistem ini yaitu data akan dilakukan *preprocessing text*dengan melakukan tokenisasi, *filtering*, serta pemberian anotasi *POS tag* menggunakan API yang merupakan pengembangan penelitian yang dilakukan oleh Rashel et al. (2014), kemudian dilakukan pembobotan kata serta *ensemble based features*, setelah itu data akan dilakukan

proses normalisasi *min-max* untuk menyeimbangkan nilai agar tidak memiliki jarak yang cukup jauh. Setelah itu data akan diolah menggunakan metode *Extreme Learning Machine (ELM)* agar didapatkannya kelas sentimen sehingga perusahaan dapat menggunakannya sebagai bahan untuk meningkatkan kualitas perusahaan.

Gambaran umum sistem ditunjukkan pada gambar 3.1




**Gambar 3.1 Gambaran umum sistem**

## 3.8 Implementasi

Implementasi merupakan tahapan yang dilakukan untuk membuat sistem berdasarkan hasil dari perancangan sistem yang telah dibuat sebelumnya. Implementasi sistem pada penelitian ini terdiri dari dua tahap, yaitu:

1. Mengimplementasikan kode logika dan metode ke dalam bahasa Python dengan menggunakan editor IDLE.
2. Menguji sistem dengan perhitungan evaluasi.

## 3.9 Pengujian dan evaluasi

Setelah program telah selesai dibuat, maka untuk mengevaluasi sistem pada penelitian ini dilakukan sebanyak tiga kali. Evaluasi pertama bertujuan untuk mengetahui pengaruh penggunaan *bag-of-words* disertai *ensemble feature* dengan tanpa *ensemble feature*, evaluasi kedua bertujuan untuk mengetahui perbandingan beberapa fungsi aktivasi, dan evaluasi ketiga bertujuan untuk mengetahui akurasi sistem analisis sentimen Twitter pada objek Samsung Indonesia.

## 3.10 Penyelesaian laporan

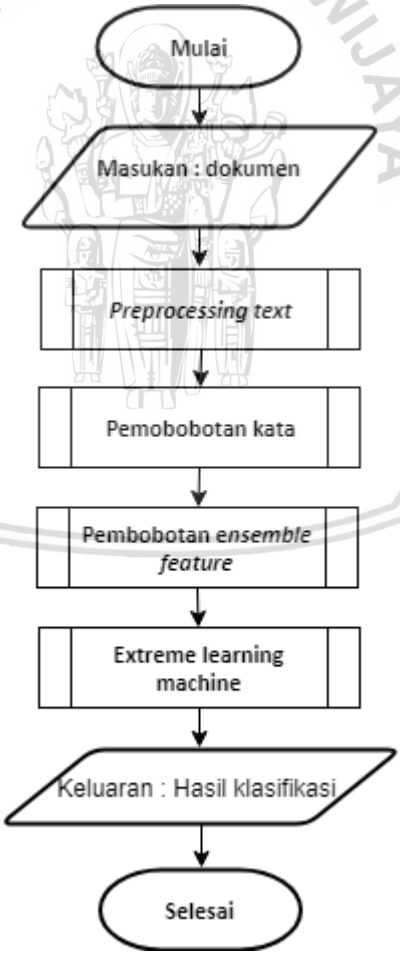
Laporan penelitian dibuat untuk melakukan dokumentasi kegiatan yang dilakukan oleh peneliti yang berisi latar belakang, perancangan, implementasi, Pengujian serta kesimpulan. Dengan adanya laporan yang terperinci diharapkan dapat menjadi referensi penelitian-penelitian selanjutnya.

# BAB 4 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini berisi tentang deskripsi umum sistem, algoritme yang digunakan, manualisasi perancangan sistem, perancangan antarmuka, dan implementasi sistem berupa kode program.

## 4.1 Deskripsi umum

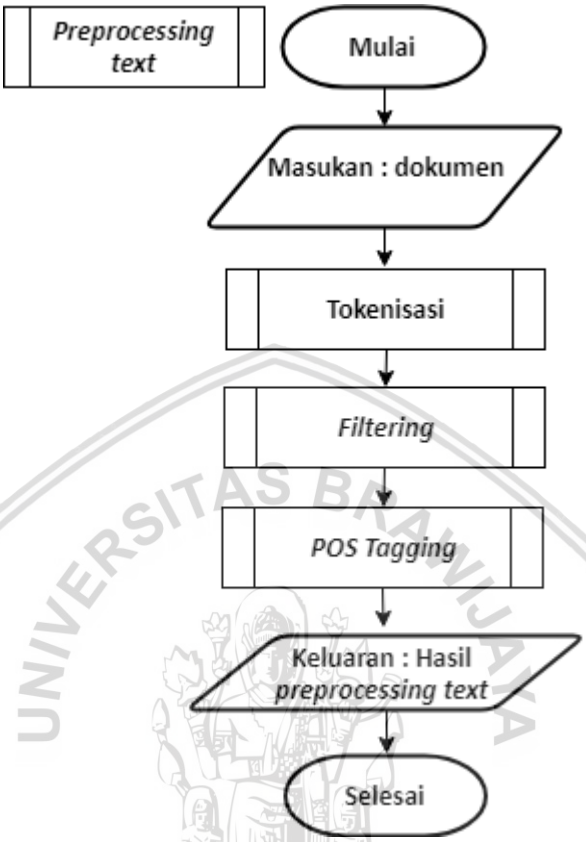
Analisis sentimen Twitter menggunakan *ensemble feature* dan *Extreme Learning Machine* merupakan sistem yang dipergunakan untuk mengevaluasi produk atau perusahaan agar perusahaan dapat menjadi pemimpin pasar. Data yang diambil menggunakan Twitter *search* API dibagi menjadi dua jenis, yaitu dokumen *training* dan dokumen *testing*. Dokumen *training* dan dokumen *testing* dimasukkan ke dalam proses *preprocessing* data yang kemudian dilakukan pembobotan 37 fitur serta pembobotan TF-IDF pada *bag-of-words.* Kemudian dilakukan proses klasifikasi *Extreme Learning Machine* untuk menentukan data baru termasuk ke dalam kelas positif, negatif atau netral. Tahap perancangan umum sistem ditunjukkan pada Gambar 4.1.




**Gambar 4.1 Diagram alir umum sistem**

## *4.2 Preprocessing text*

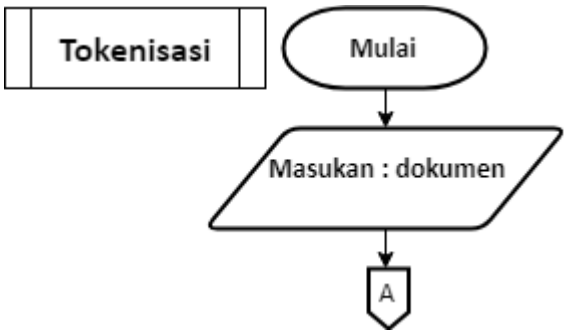
*Preprocessing* terdiri dari tiga tahapan yaitu: tokenisasi, *filtering,* dan *POS tagging.* TigaTahapan tersebut ditunjukkan pada Gambar 4.2.

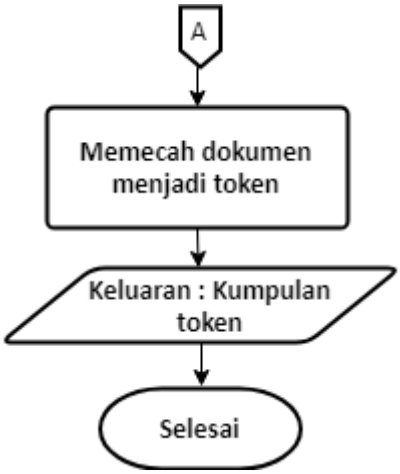



**Gambar 4.2 Diagram alir *preprocessing text***

### 4.2.1 Tokenisasi

Tokenisasi merupakan subproses dari *preprocessing text* yang dilakukan setelah sistem menerima dokumen. Dokumen tersebut berupa dokumen *training* atau dokumen *testing*. Pada tahap tokenisasi seluruh dokumen akan dipecah menjadi token/kata. Proses tokenisasi ditunjukkan pada Gambar 4.3

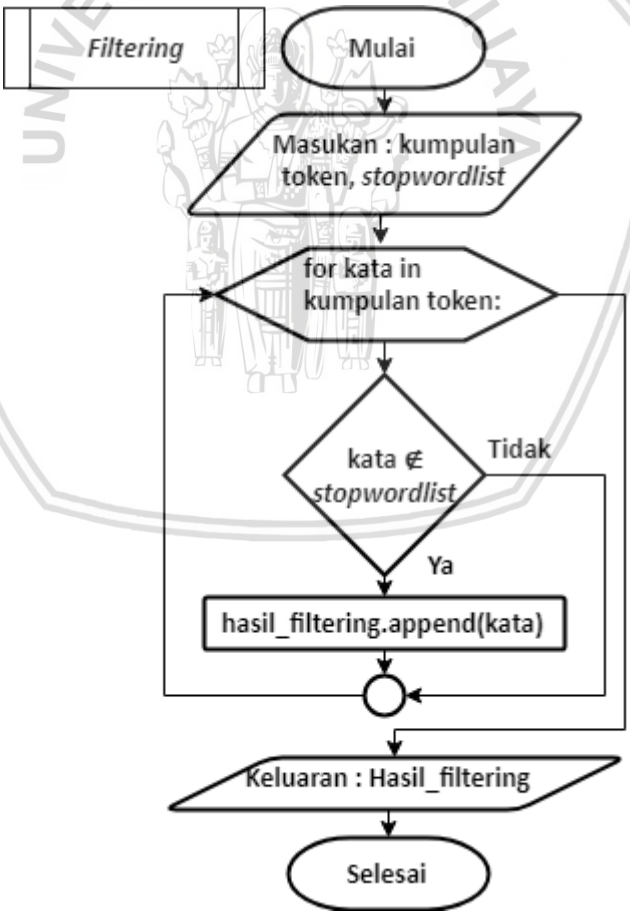





**Gambar 4.3 Diagram alir tokenisasi**

### 4.2.2 Filtering

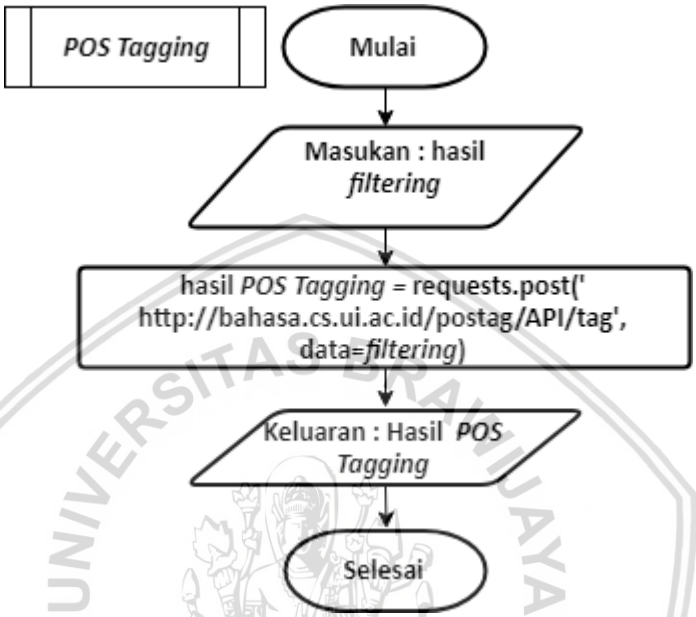
*Filtering* merupakan subproses dari *preprocessing text* yang mendapatkan masukan hasil dari tokenisasi. Pada tahap ini seluruh kata yang terdapat pada *stopwordlist* akan dihapus. Proses *filtering* ditunjukkan pada Gambar 4.4




**Gambar 4.4 Diagram alir *filtering***

### *4.2.3 POS Tagging*

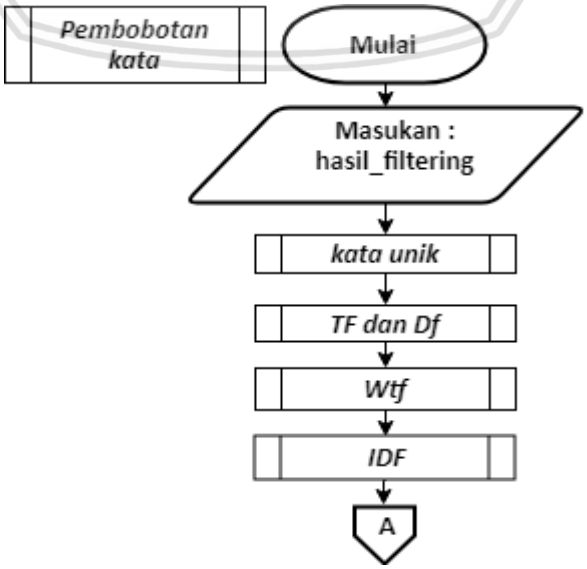
*POS Tagging* merupakan subproses dari *preprocessing text* yang mengolah data hasil dari *filtering.* Pada tahap ini seluruh kata hasil *filtering* akan diberi *tag* sesuai jenisnya dengan mengakses API hasil penelitian Rashel et al. (2014). Proses *POS Tagging* ditunjukkan pada Gambar 4.5.

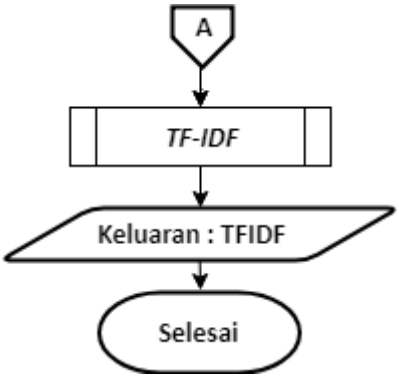



**Gambar 4.5 Diagram alir *POS Tagging***

### 4.2.4 Pembobotan kata

Ada 4 proses pada proses pembobotan kata yaitu perhitungan $T_f$ dan *DF* , perhitungan $w_{tf}$, perhitungan *IDF*, dan perhitungan *TF-IDF*. Pada pembobotan fitur mendapatkan masukan berupa token hasil dari filtering. Alur pembobotan kata ditunjukkan pada Gambar 4.6.

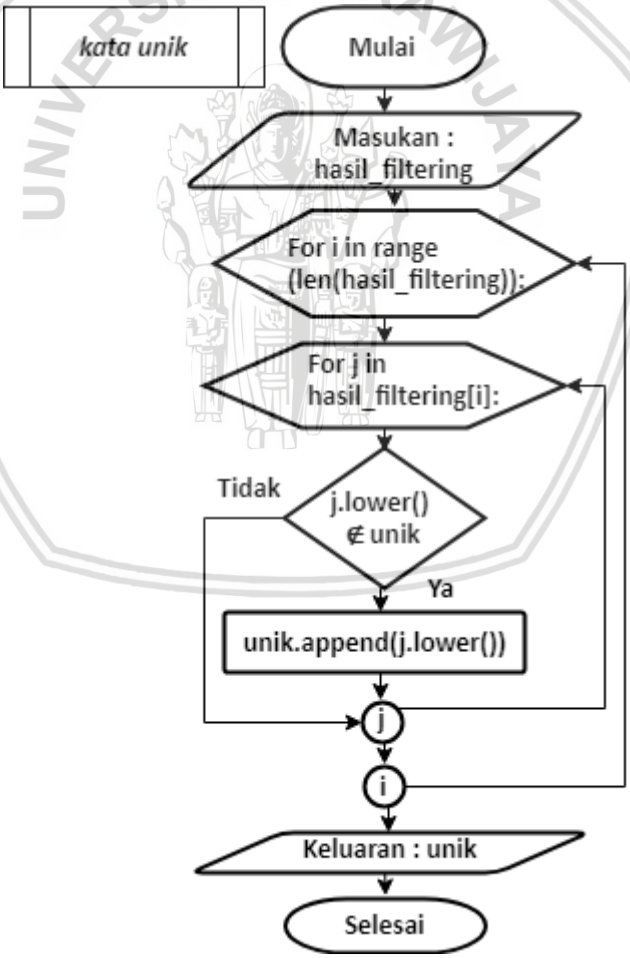





**Gambar 4.6 Diagram alir pembobotan kata**

### 4.2.5 Kata unik

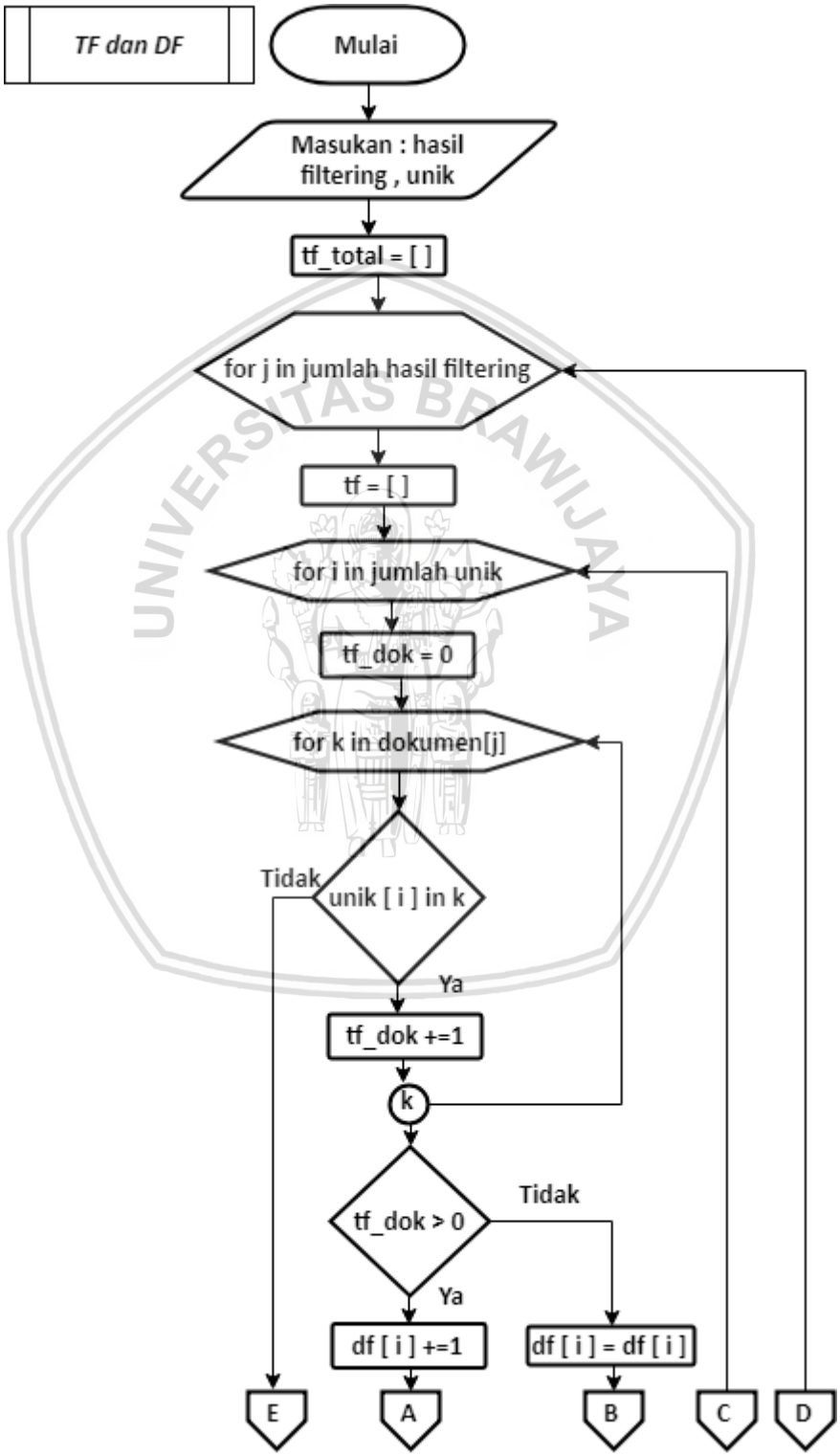
Proses pertama dalam pembobotan kata, membuat *list* kata unik, dengan cara mengumpulkan token-token yang belum terdapat pada *list* kata unik. Token yang terdapat di dalam *list* kata unik tidak boleh sama antara satu token dengan token lainnya. Proses pembuatan *list* kata unik ditunjukkan pada Gambar 4.7.

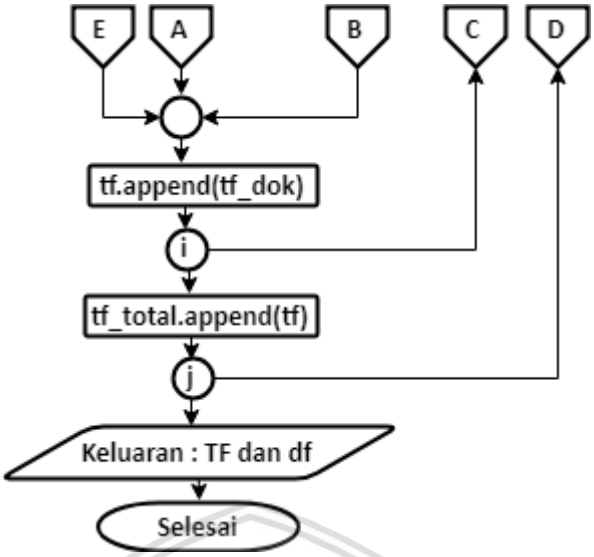



**Gambar 4.7 Diagram alir kata unik**

### 4.2.6 $T_f$ dan DF

Nilai $T_f$ merupakan frekuensi kata yang terdapat pada suatu dokumen, sedangkan *DF* merupakan *DF* merupakan frekuensi dokumen yang memiliki suatu kata. $W_{tf}$ merupakan nilai logaritma dari $T_f$. Proses penghitungan $T_f$ dan D*F* ditunjukkan pada Gambar 4.8.

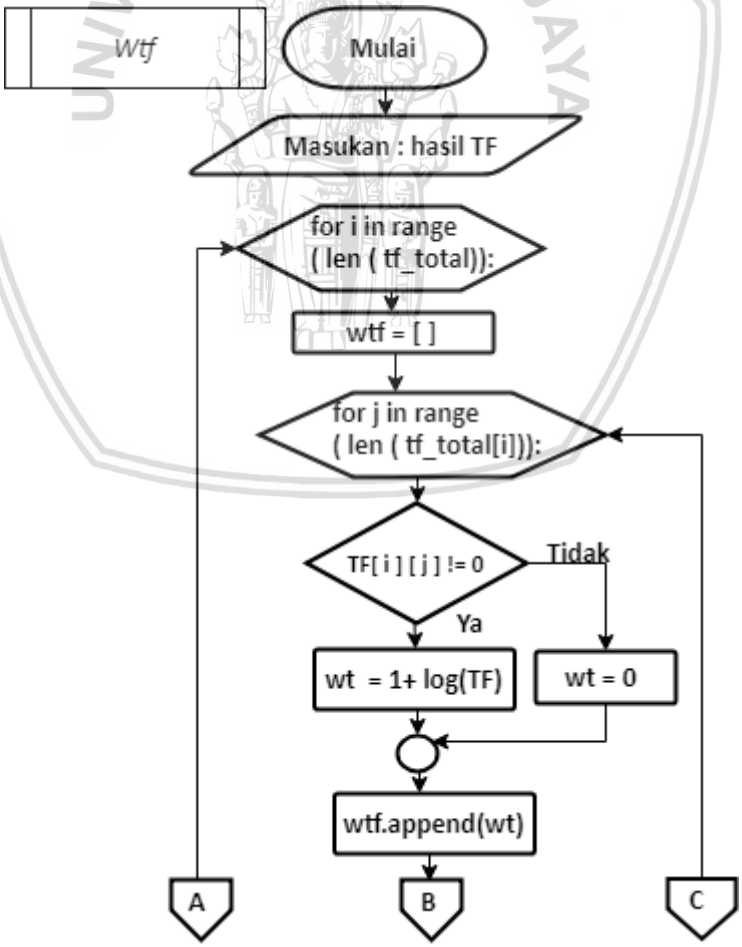


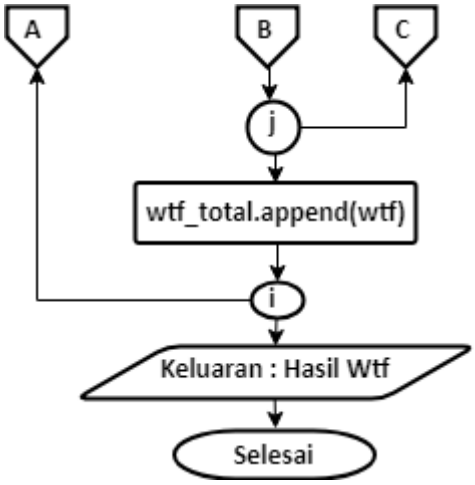



**Gambar 4.8 Diagram alir perhitungan nilai *Tf* dan *Df***

### *4.2.7 $W_{tf}$*

$W_{tf}$ merupakan nilai logaritma dari $T_f$. Proses penghitungan $W_{t\,f}$ ditunjukkan pada Gambar 4.9.

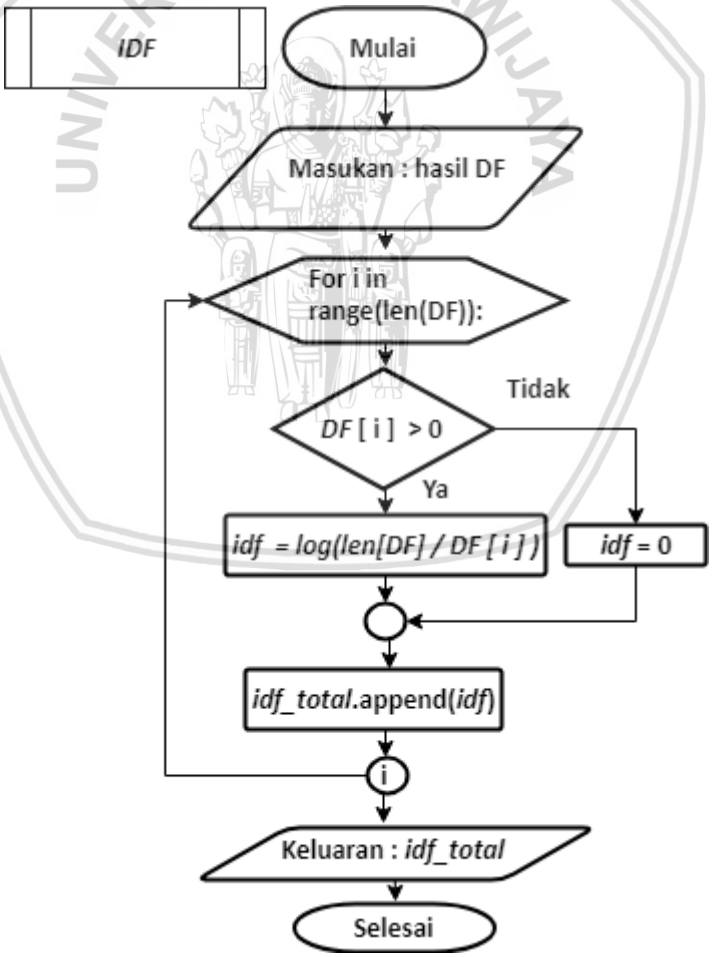





**Gambar 4.9 Diagram alir perhitungan nilai *wtf***

### 4.2.8 IDF

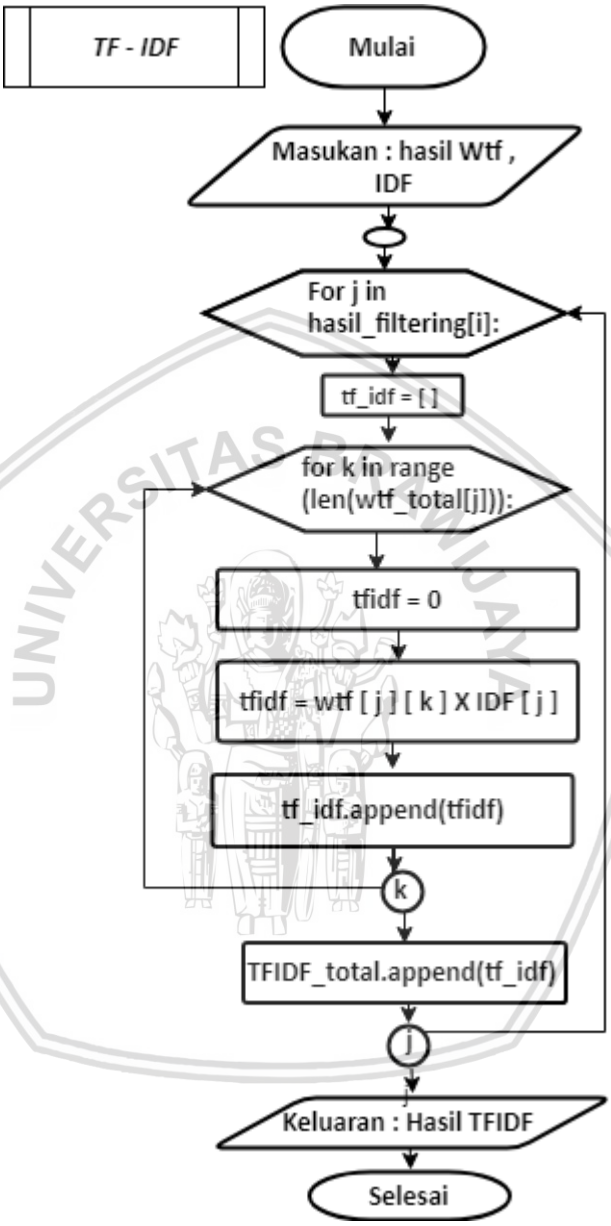
*Invers document frequency (IDF)* merupakan nilai kebalikan dari *DF.* Algoritme perhitungan nilai *IDF* ditunjukkan pada Gambar 4.10.




**Gambar 4.10 Diagram alir perhitungan *idf***

### 4.2.9 TF-IDF

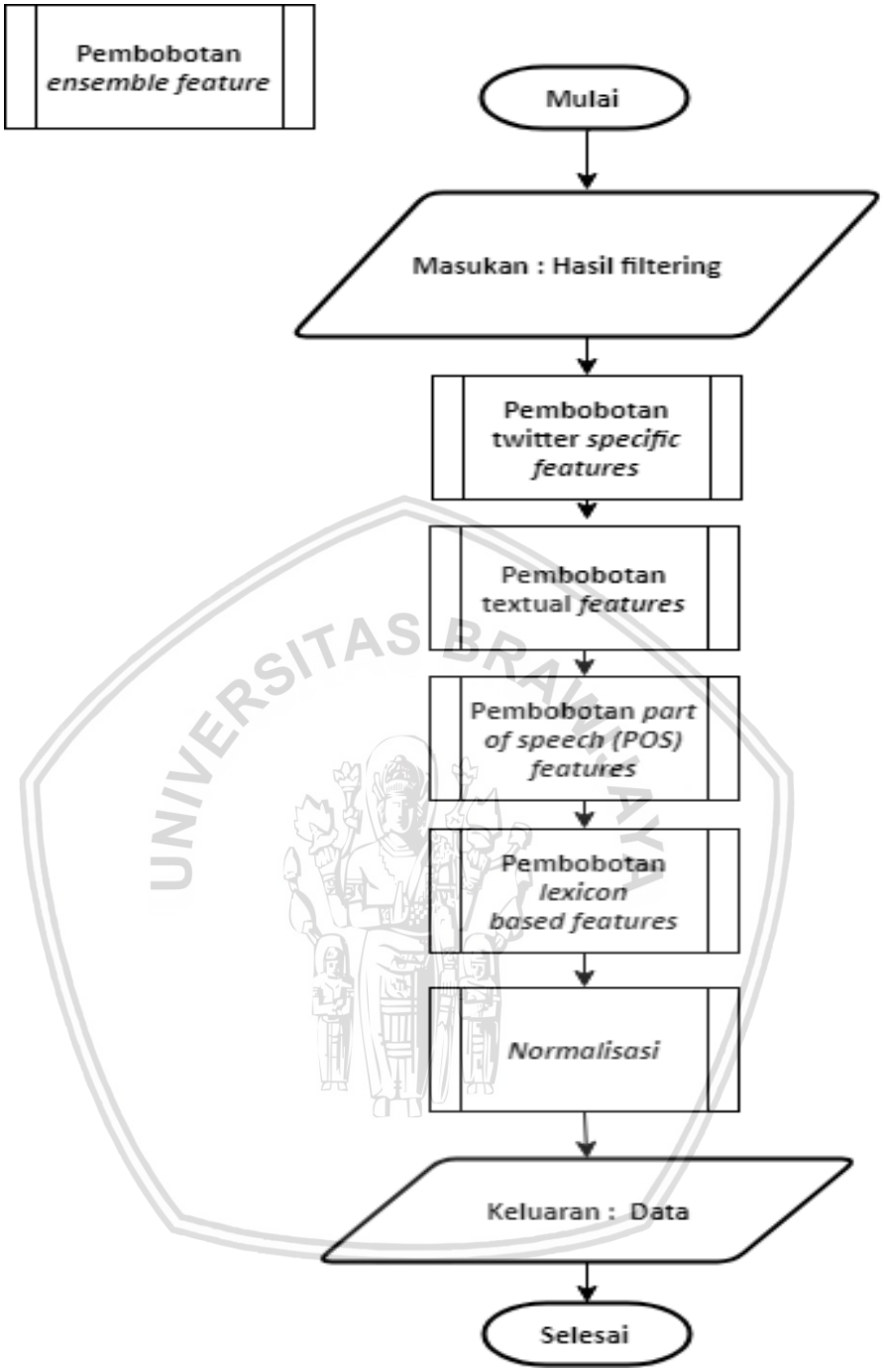
Setelah mendapatkan nilai $W_{td}$ dan nilai IDF maka dapat menghitung *nilai TF-IDF. Tf-IDF* merupakan hasil perkalian antara $W_{td}$ dengan *invers document frequency.*Proses perhitungan *Tf-IDF* ditunjukkan pada Gambar 4.11




**Gambar 4.11 Diagram alir perhitungan *TF-IDF***

## 4.3 Pembobotan ensemble feature

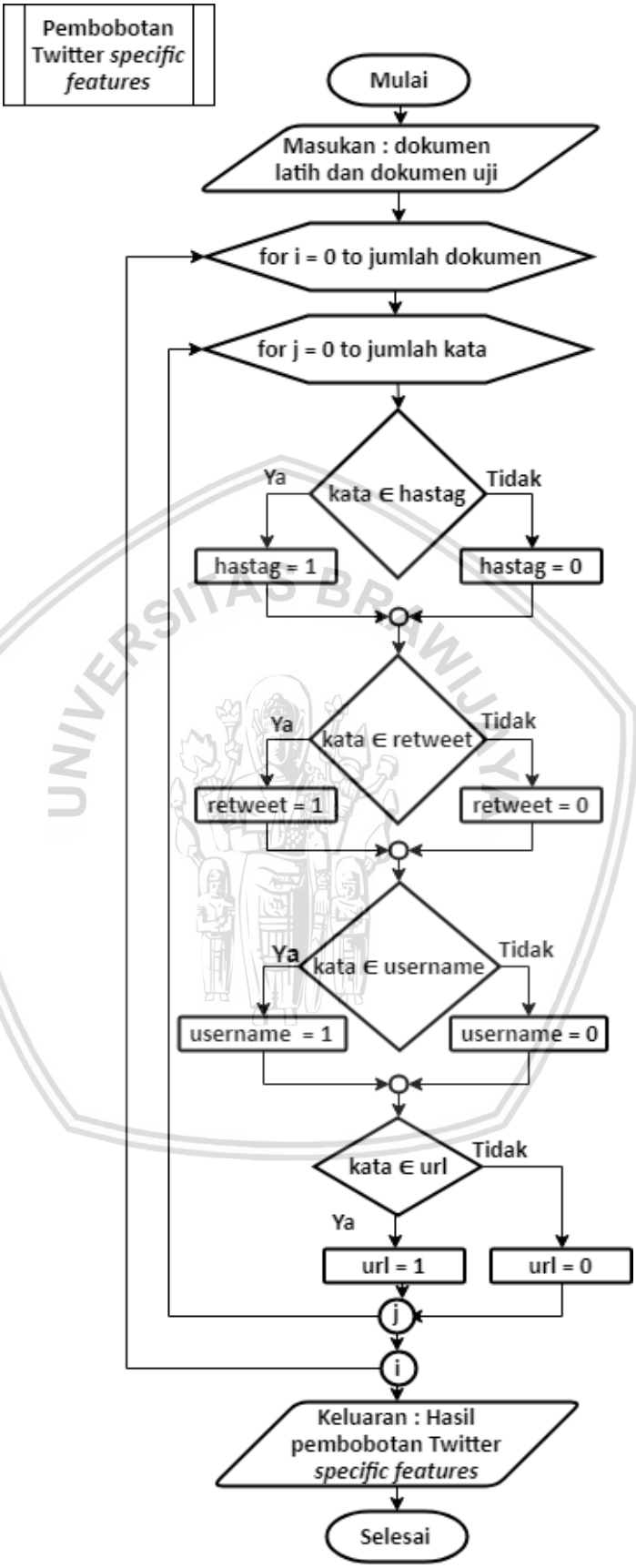
Pada penelitian ini melakukan pembobotan tiga puluh tujuh fitur yang telah dikelompokkan menjadi empat yaitu *Twitter specific features, textual features, part of speech features,* dan *lexicon based features* oleh (Siddiqua, Ahsan and Chy, 2017). Proses *ensemble features* di tunjukkan pada Gambar 4.12




**Gambar 4.12 Diagram alir *ensemble feature***

### 4.3.1 Twitter specific features

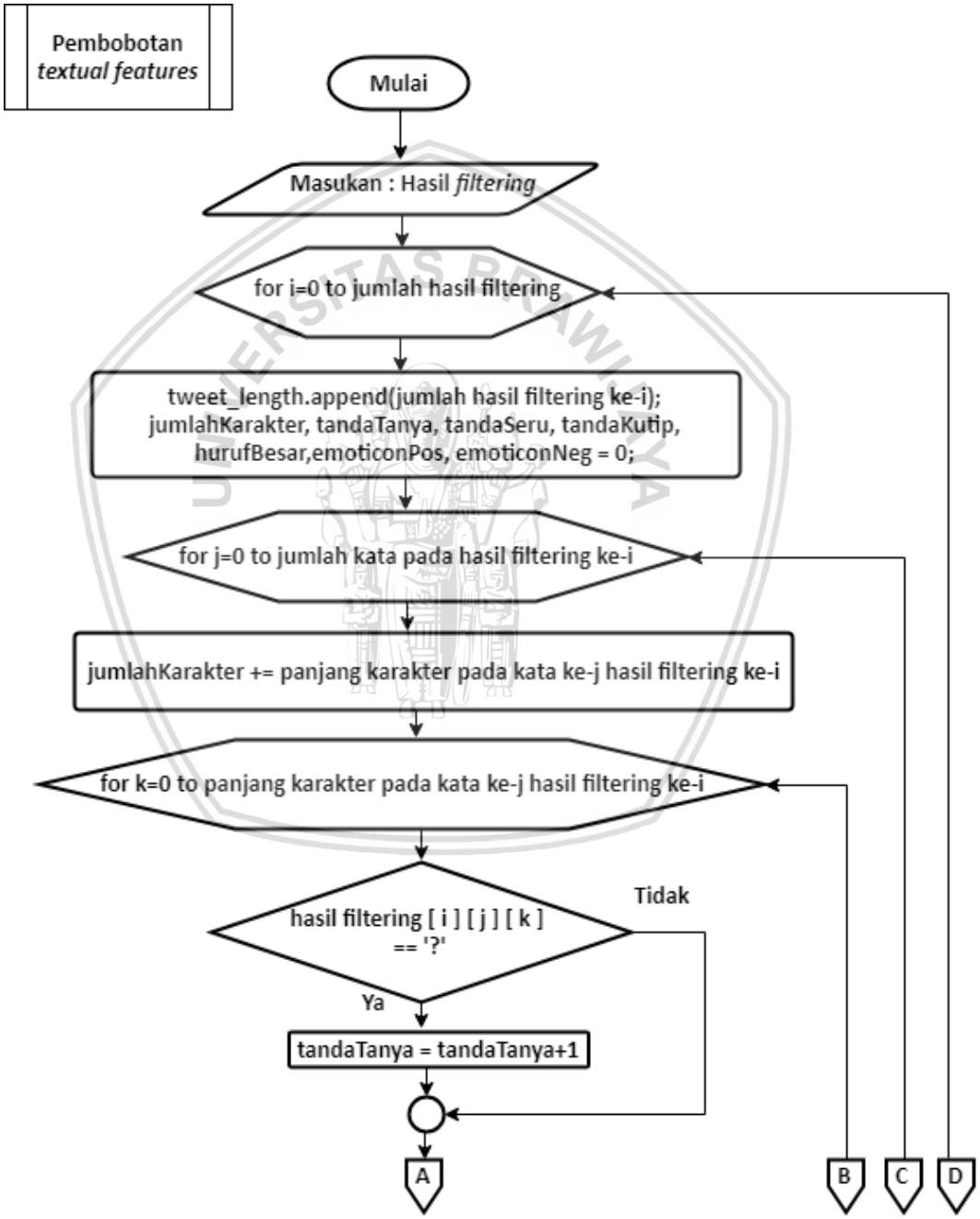
Pembobotan Twitter *specific features* dilakukan setelah proses *filtering*. Semua kata pada setiap dokumen akan diseleksi dengan syarat yang mengandung kata yang merupakan sebuah *url* atau *hashtag* atau *retweet* ataupun *username* akan diberi nilai 1, namun jika di dalam dokumen tersebut tidak terdapat kata yang menjadi syarat maka akan diberi nilai 0. Diagram alir pembobotan Twitter *specific features* ditunjukkan pada Gambar 4.13.

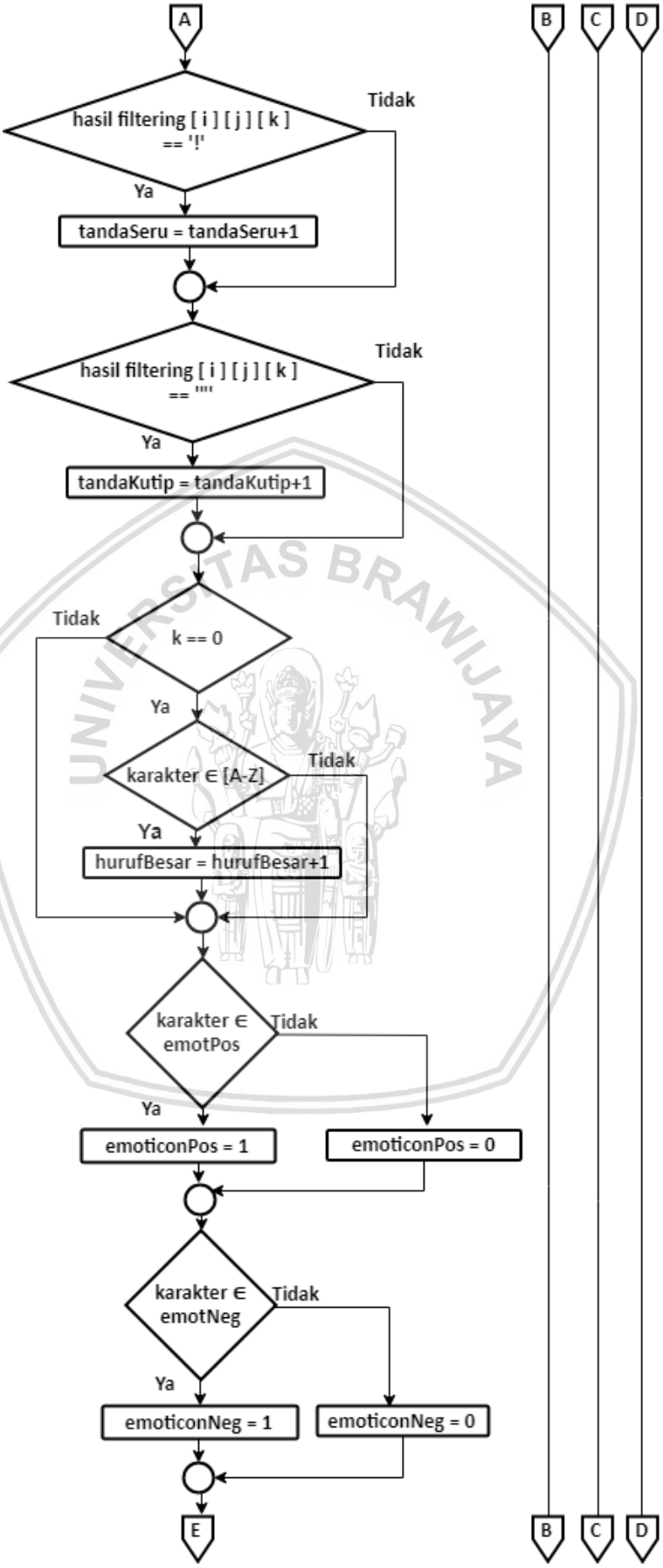


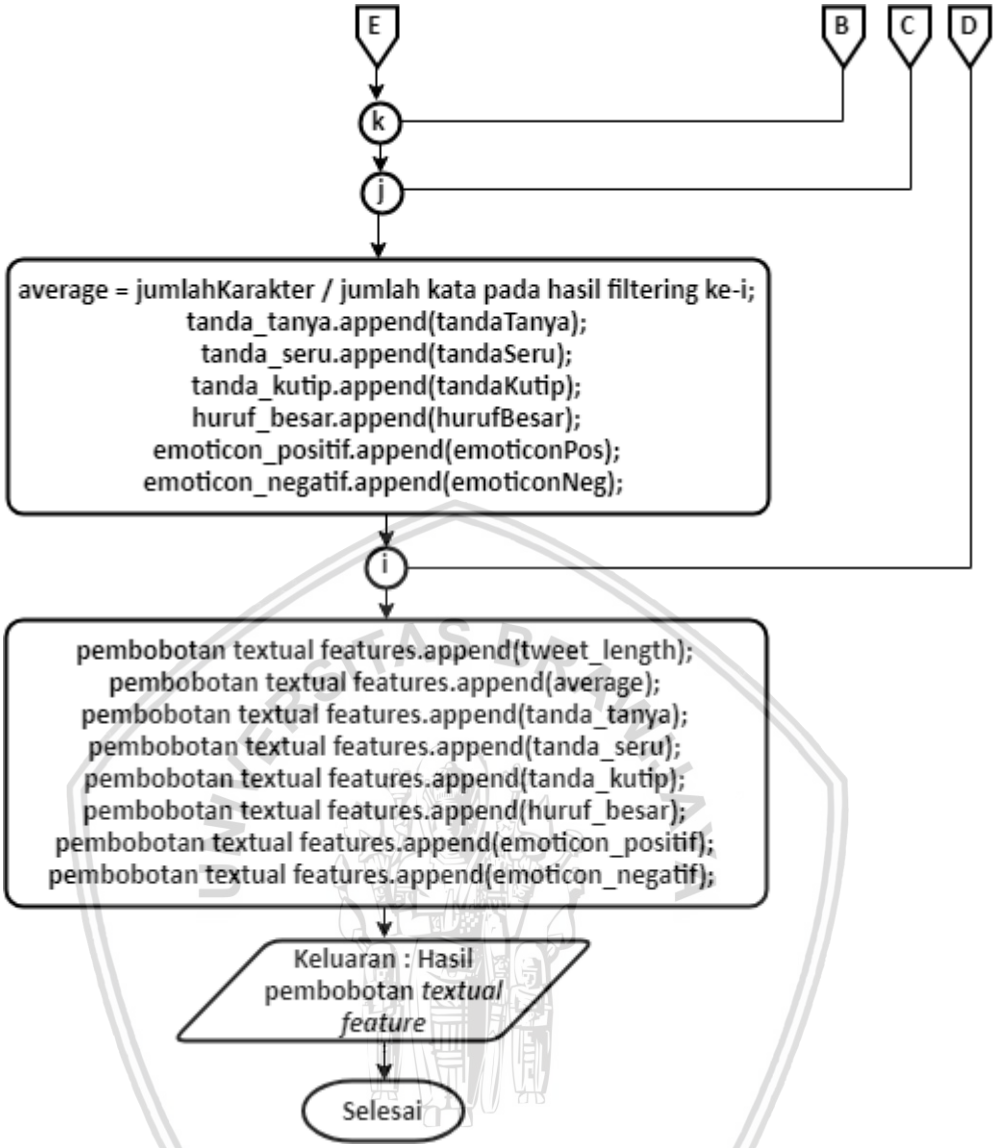

**Gambar 4.13 Diagram alir pembobotan Twitter *specific features***

### 4.3.2 Pembobotan *textual features*

Pembobotan *textual features* dilakukan untuk mengetahui jumlah karakter tanda tanya, tanda seru, tanda kutip, huruf besar, panjang *tweet*, rata-rata panjang kata, *emoticon* positif, dan *emoticon* negatif dengan menyeleksi kandungan karakter yang terdapat pada suatu dokumen. Bila suatu kata pada suatu dokumen memenuhi syarat maka akan dijumlahkan dengan syarat yang sama pada tiap dokumen. Diagram alir pembobotan *textual features* ditunjukkan pada Gambar 4.14.



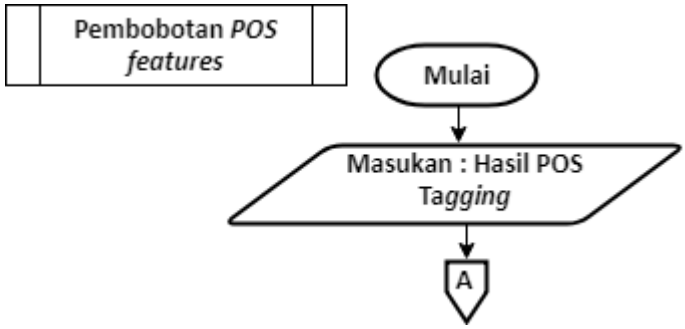

A
hasil filtering [ i ] [ j ] [ k ] == '!'
Tidak
Ya
tandaSeru = tandaSeru+1
hasil filtering [ i ] [ j ] [ k ] == '""'
Tidak
Ya
tandaKutip = tandaKutip+1
k == 0
Tidak
Ya
karakter ∈ [A-Z]
Tidak
Ya
hurufBesar = hurufBesar+1
karakter ∈ emotPos
Tidak
Ya
emoticonPos = 1
emoticonPos = 0
karakter ∈ emotNeg
Tidak
Ya
emoticonNeg = 1
emoticonNeg = 0
E
B
C
D
B
C
D

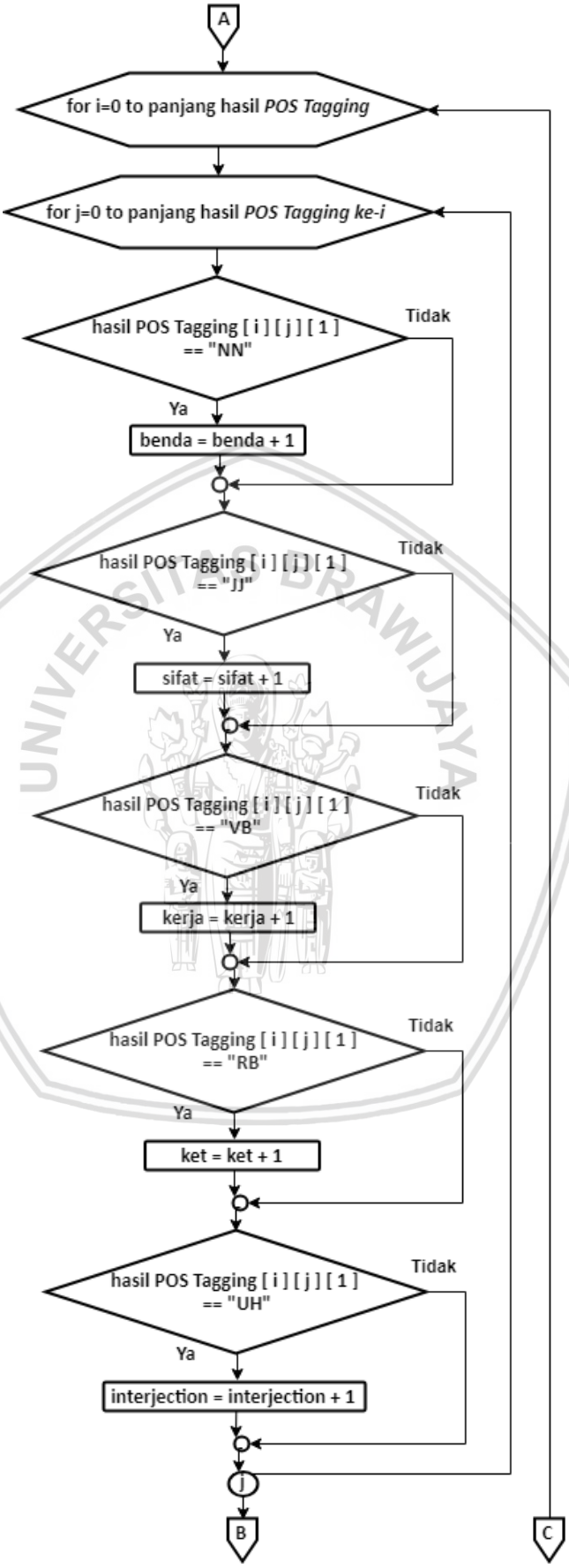


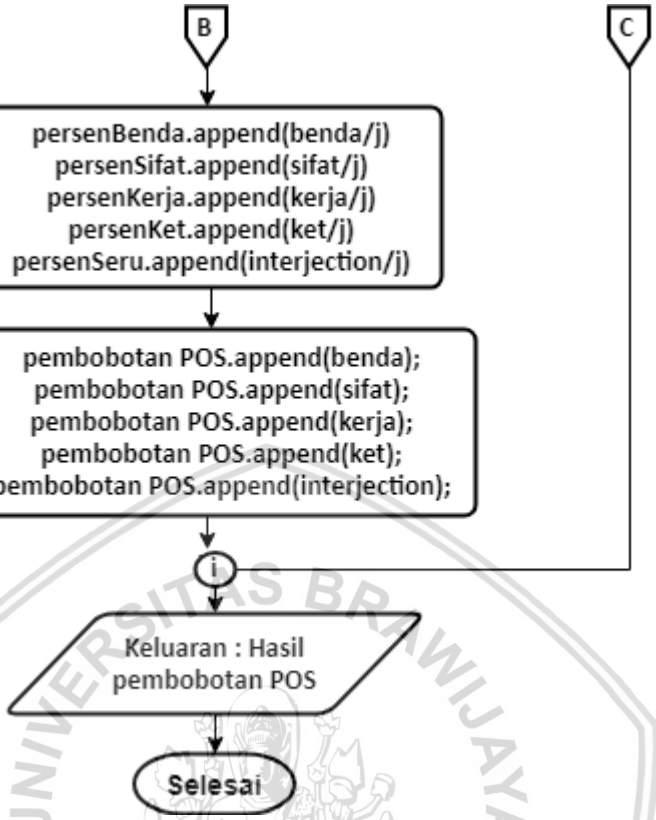

**Gambar 4.14 Diagram alir pembobotan *textual features***

### 4.3.3 Pembobotan *POS Tagging*

*POS Tagging* merupakan proses pemberian *tag* pada suatu kata. Setelah dilakukan pemberian *tag* dilakukan pembobotan berdasarkan kata benda, kata sifat, kata kerja, kata keterangan, *intensifier*. Proses *POS Tagging* ditunjukkan pada Gambar 4.15.



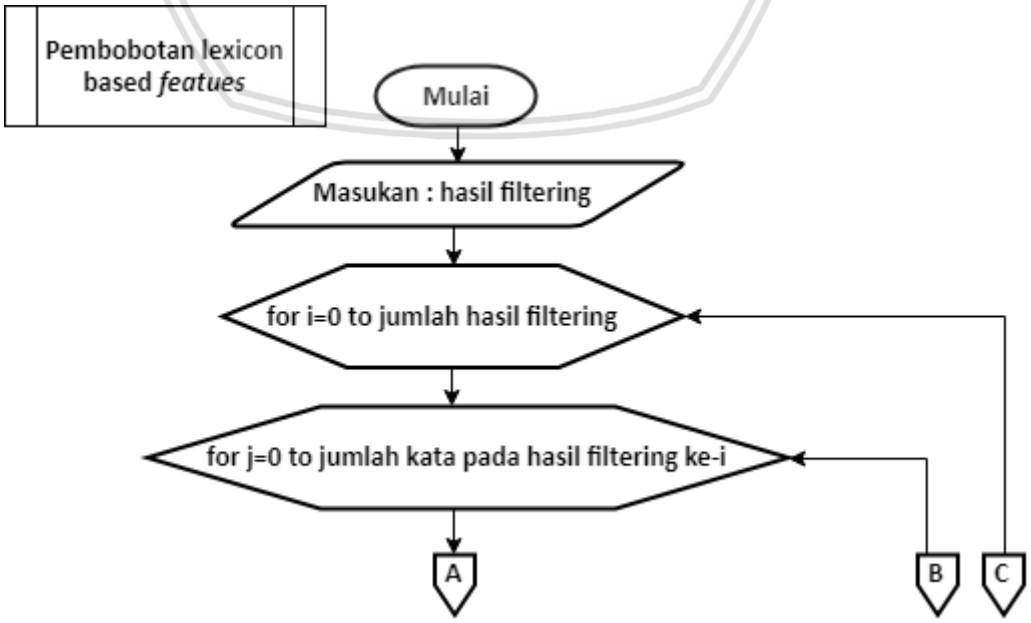

A
for i=0 to panjang hasil POS Tagging
for j=0 to panjang hasil POS Tagging ke-i
hasil POS Tagging [ i ] [ j ] [ 1 ] == "NN"
Tidak
Ya
benda = benda + 1
hasil POS Tagging [ i ] [ j ] [ 1 ] == "JJ"
Tidak
Ya
sifat = sifat + 1
hasil POS Tagging [ i ] [ j ] [ 1 ] == "VB"
Tidak
Ya
kerja = kerja + 1
hasil POS Tagging [ i ] [ j ] [ 1 ] == "RB"
Tidak
Ya
ket = ket + 1
hasil POS Tagging [ i ] [ j ] [ 1 ] == "UH"
Tidak
Ya
interjection = interjection + 1
j
B
C

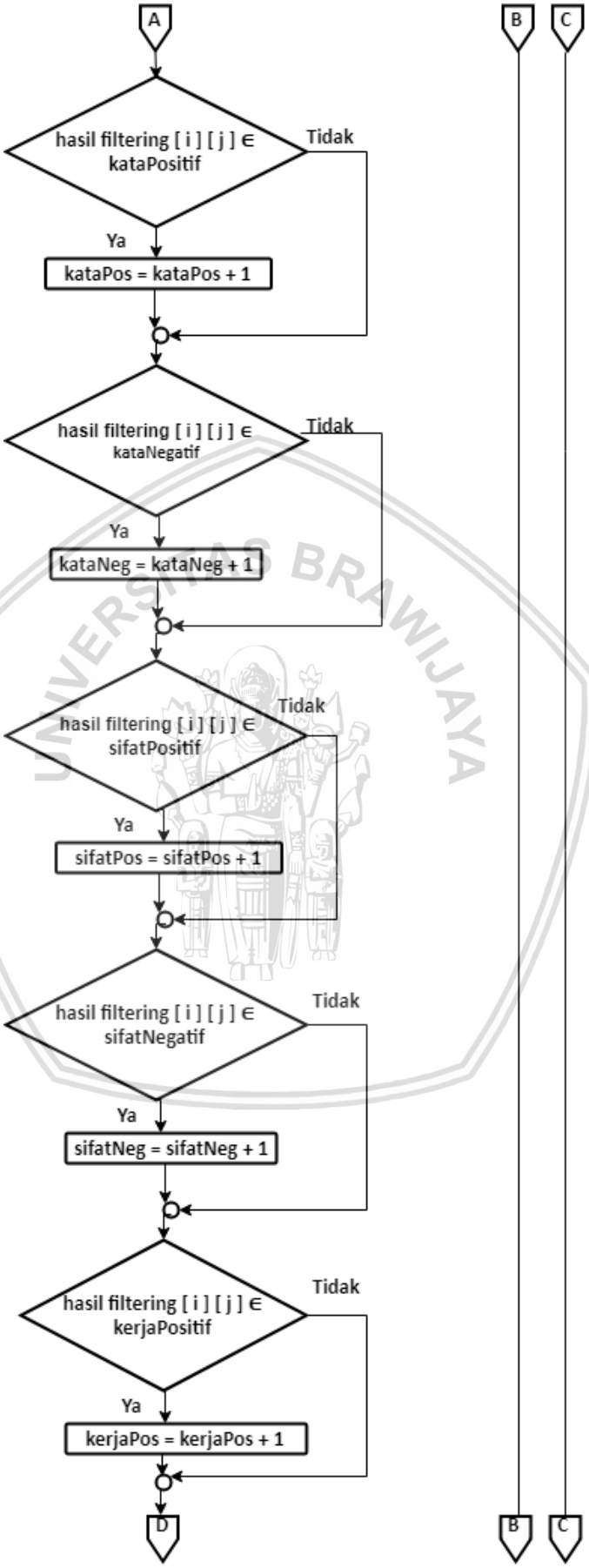


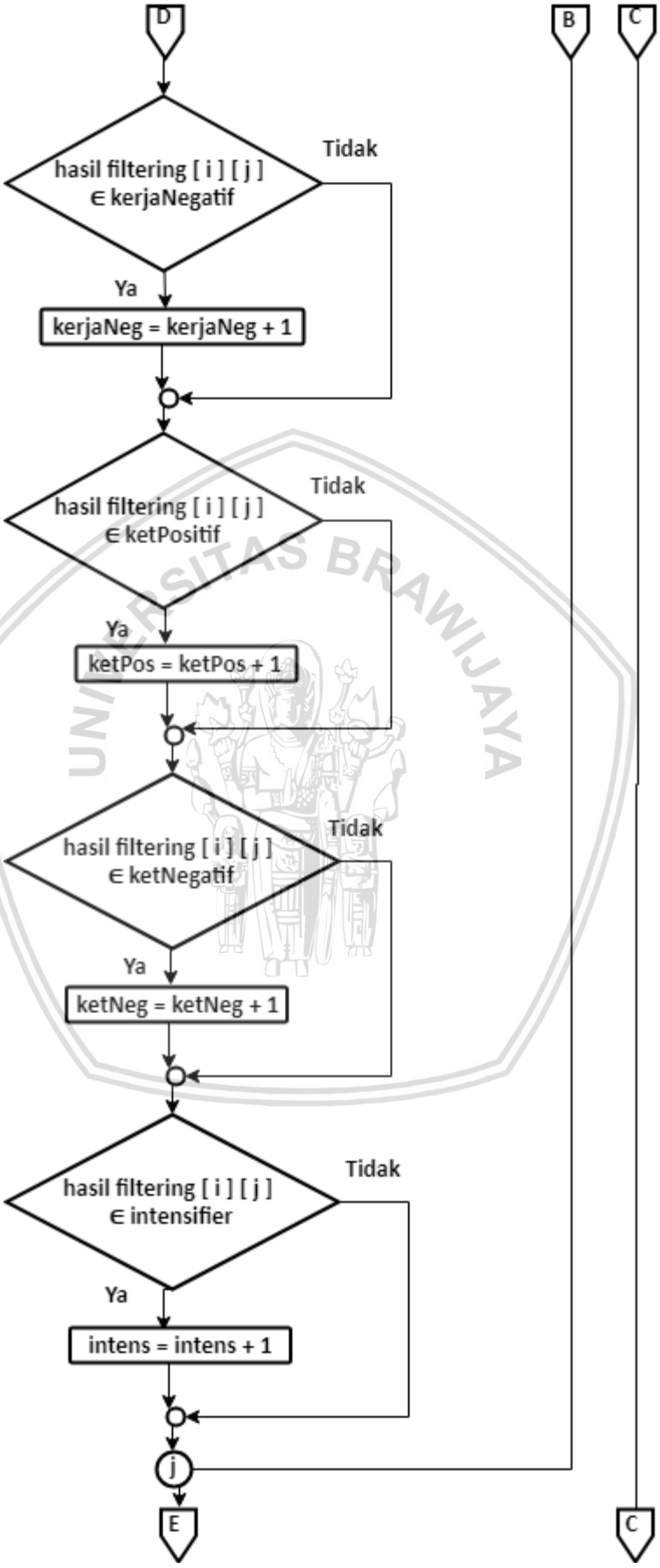

**Gambar 4.15 Diagram alir pembobotan *POS Tagging***

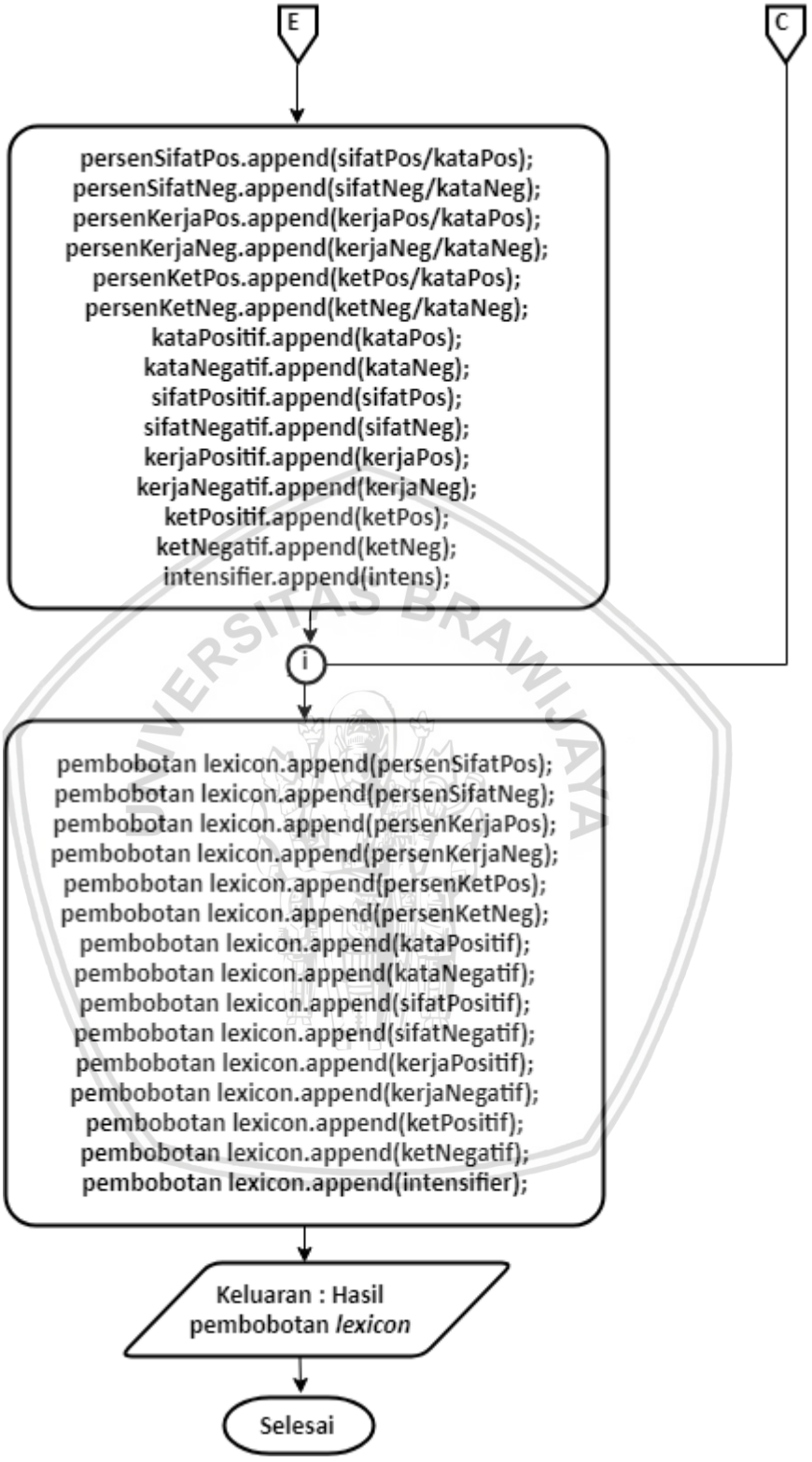
### 4.3.4 Pembobotan *lexicon based features*

Pada proses ini dilakukan pembobotan berdasarkan *lexicon* positif, negatif, dan positif/negatif kata sifat, kata kerja, ataupun kata keterangan. Proses pembobotan *lexicon based features* ditunjukkan pada Gambar 4.16.




A
hasil filtering [ i ] [ j ] ∈ kataPositif
Tidak
Ya
kataPos = kataPos + 1
hasil filtering [ i ] [ j ] ∈ kataNegatif
Tidak
Ya
kataNeg = kataNeg + 1
hasil filtering [ i ] [ j ] ∈ sifatPositif
Tidak
Ya
sifatPos = sifatPos + 1
hasil filtering [ i ] [ j ] ∈ sifatNegatif
Tidak
Ya
sifatNeg = sifatNeg + 1
hasil filtering [ i ] [ j ] ∈ kerjaPositif
Tidak
Ya
kerjaPos = kerjaPos + 1
D
B
C
B
C

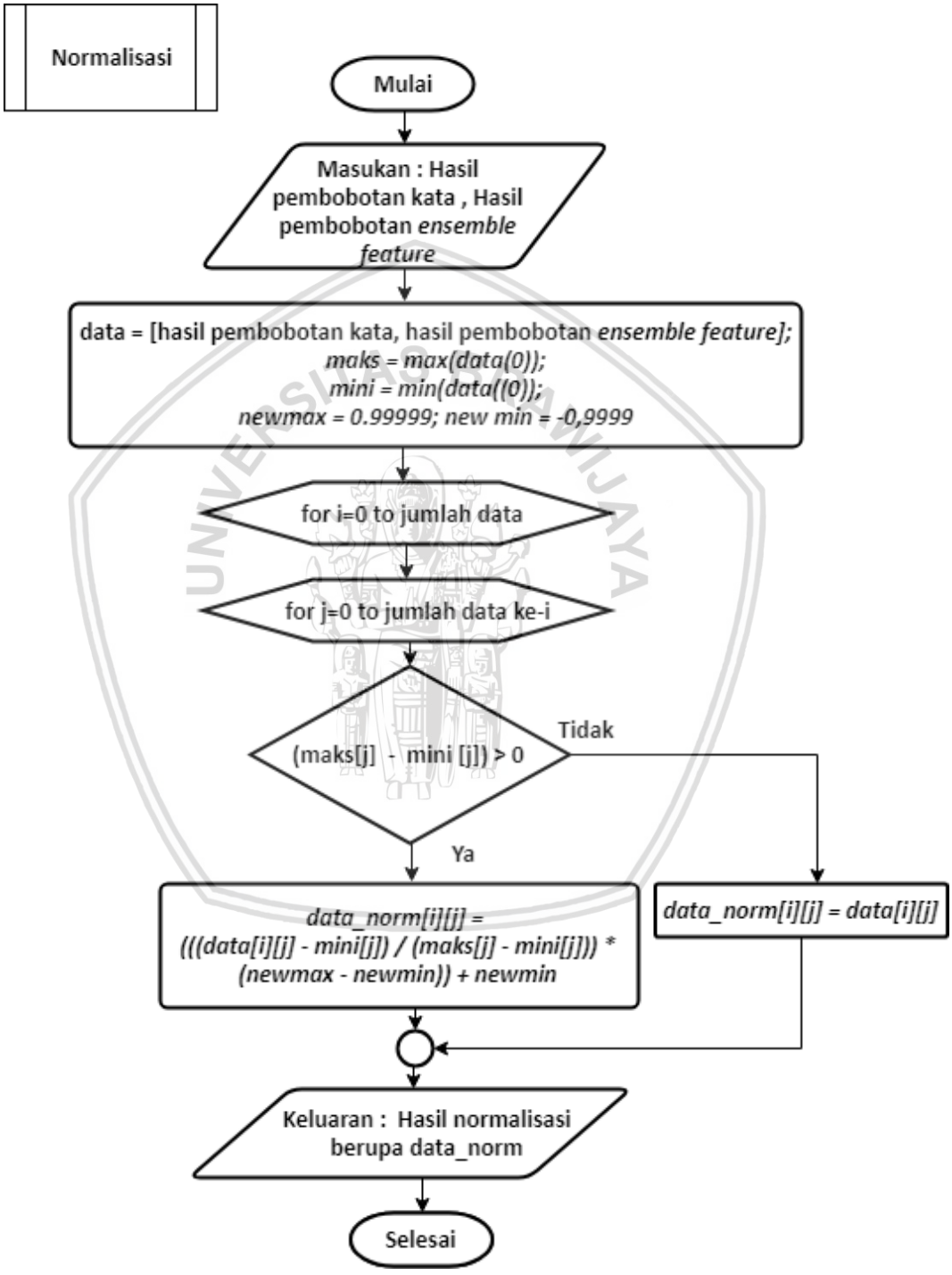

D
B
C
hasil filtering [ i ] [ j ]
∈ kerjaNegatif
Tidak
Ya
kerjaNeg = kerjaNeg + 1
hasil filtering [ i ] [ j ]
∈ ketPositif
Tidak
Ya
ketPos = ketPos + 1
hasil filtering [ i ] [ j ]
∈ ketNegatif
Tidak
Ya
ketNeg = ketNeg + 1
hasil filtering [ i ] [ j ]
∈ intensifier
Tidak
Ya
intens = intens + 1
j
E
C




**Gambar 4.16 Diagram alir pembobotan *lexicon based features***

### 4.3.5 Normalisasi

Penelitian ini melakukan normalisasi *min max* agar data hasil pembobotan kata dan pembobotan *ensemble features* tidak memiliki selisih nilai yang jauh dengan nilai maksimal 0,9999 serta nilai minimal -0,9999. Proses normalisasi ditunjukkan pada Gambar 4.17




**Gambar 4.17 Diagram alir normalisasi**

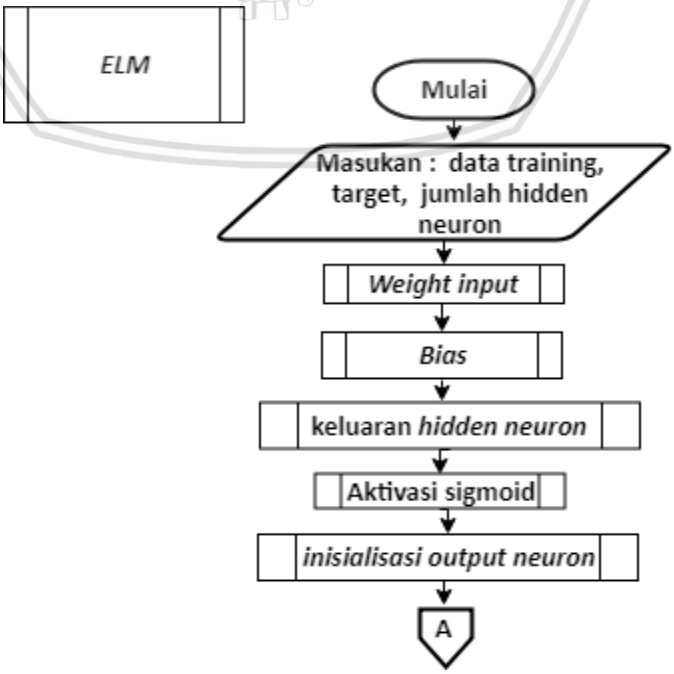
## *4.4 Extreme Learning Machine*

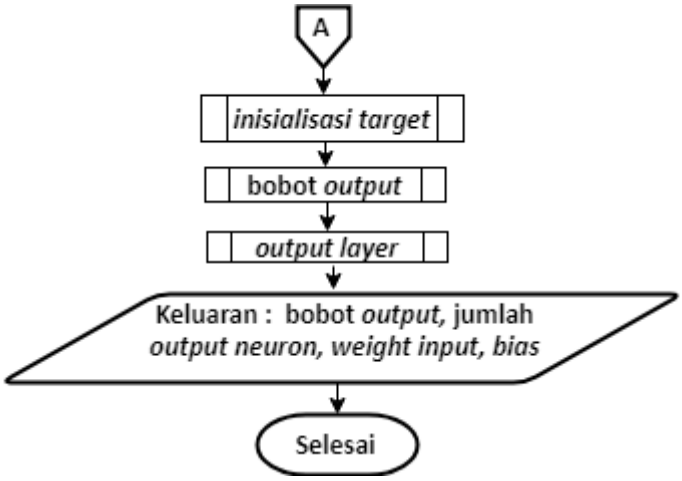
Pada tahap ini data berupa data *training* atau data *testing* akan dilakukan proses klasifikasi *Extreme Learning Machine* dengan jumlah kelas sebanyak tiga, yaitu: positif, negatif, dan netral. Proses klasifikasi *Extreme Learning Machine* terdiri dari 2 proses yaitu proses *training* dan proses *testing.*

### *4.4.1* Proses *training*

Proses training dilakukan untuk memperoleh nilai bobot keluaran yang kemudian akan digunakan pada proses testing. Proses *training* terdiri dari beberapa tahap yang ditunjukkan pada Gambar 4.18, yaitu:

1. Mendapatkan masukan berupa data *training,* dan target
2. Inisialisasi *weight input* dan *bias* secara *random*
3. Menghitung keluaran *hidden neuron* menggunakan Persamaan 2.5. diagram alir proses keluaran hidden neuron ditunjukkan Gambar 4.21
4. Menghitung keluaran *hidden neuron* dengan fungsi aktivasi menggunakan Persamaan 2.6. Diagram alir proses keluaran *hidden neuron* dengan fungsi aktivasi ditunjukkan pada Gambar 4.22
5. Inisialisasi *output neuron* dengan proses yang ditunjukkan pada Gambar 4.23
6. Inisialisasi matriks kelas target dengan proses yang ditunjukkan pada Gambar 4.24
7. Menghitung bobot *output* menggunakan Persamaan 2.8 dengan proses yang ditunjukkan pada Gambar 4.25
8. Keluaran sistem yaitu matriks *bobot output*.

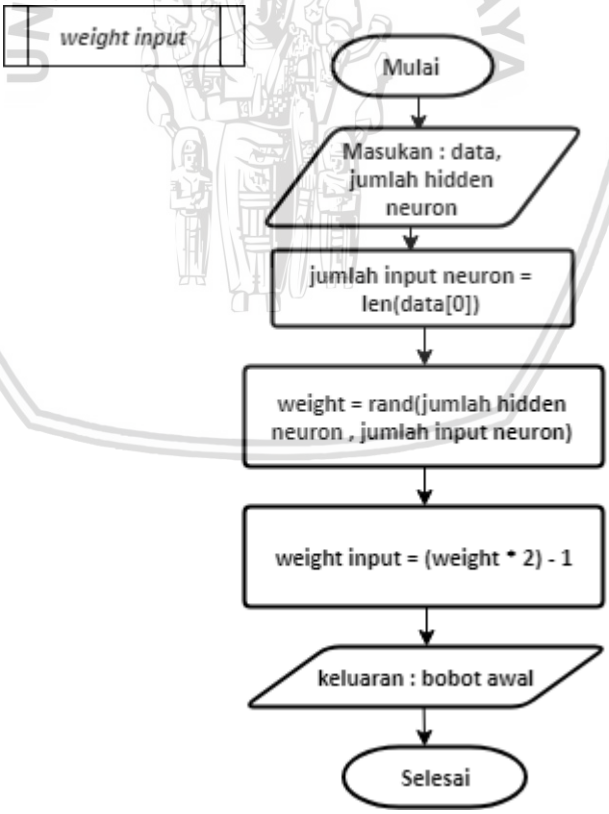





**Gambar 4.18 Diagram alir proses *training elm***

#### *4.4.1.1 Weight input*

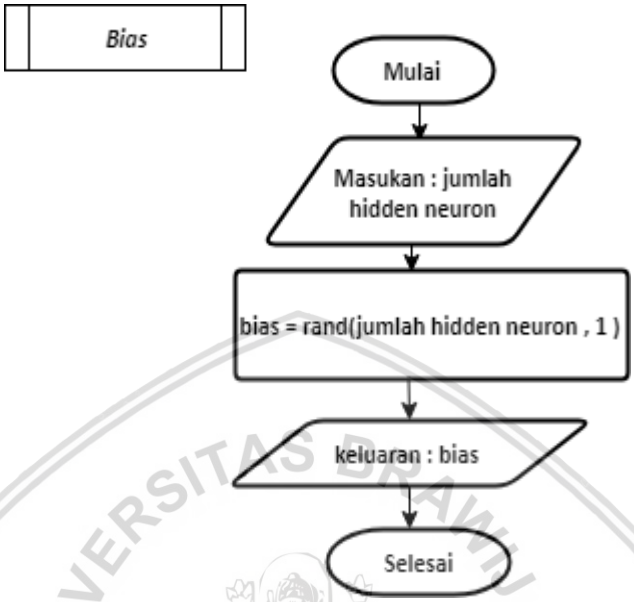
Pada proses klasifikasi *ELM,* yang pertama kali dilakukan yaitu inisialisasi bobot masukan secara *random* antara -1 sampai 1. Pada proses ini mendapatkan masukan berupa nilai *neuron* yaitu sejumlah fitur yang terdapat pada sistem dan nilai *hidden neuron.* Proses inisialisasi bobot masukan ditunjukkan pada Gambar 4.19.




**Gambar 4.19 Diagram alir *weight input***

#### *4.4.1.2 Inisialisasi nilai bias*

Proses setelah inisialisasi bobot masukan yaitu inisialisasi bias secara *random* antara 0 hingga 1. Pada proses ini mendapatkan masukan berupa jumlah *hidden layer*. Proses inisialisasi nilai bias ditunjukkan pada Gambar 4.20.

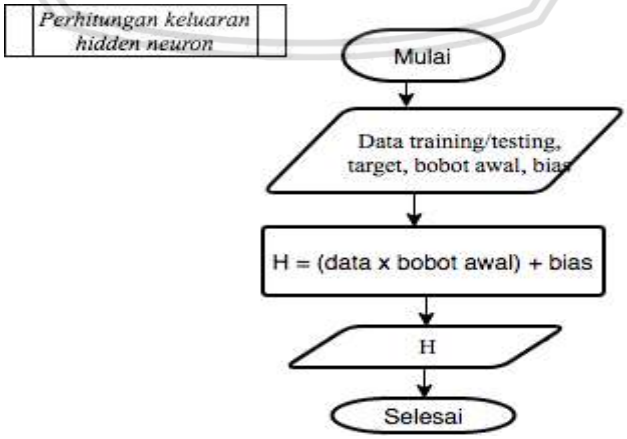



**Gambar 4.20 Diagram alir nilai *bias***

#### *4.4.1.3* Perhitungan keluaran *hidden neuron*

Proses perhitungan keluaran *hidden neuron* terdiri dari beberapa langkah yang ditunjukkan pada Gambar 4.21, yaitu:

1. Sistem menerima inputan berupa data *training/testing,* target, bobot awal, dan bias.
2. Menghitung keluaran *hidden neuron* menggunakan Persamaan 2.5.
3. Keluaran sistem yaitu matriks keluaran *hidden neuron.*

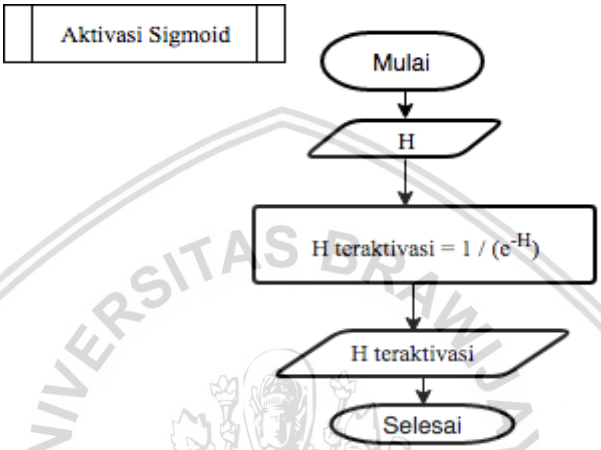



**Gambar 4.21 Diagram alir keluaran *hidden neuron***

#### 4.4.1.4 Aktivasi sigmoid

Aktivai sigmoid biner dilakukan pada proses *training* dan *testing.* Proses aktivasi *sigmoid biner* terdiri beberapa tahap yang ditunjukkan pada Gambar 4.22, yaitu:

1. Sistem menerima *input* berupa keluaran *hidden neuron*
2. Menghitung nilai *hidden neuron* dengan fungsi aktivasi *sigmoid biner* menggunakan Persamaan 2.6
3. Keluaran berupa matriks keluaran *hidden neuron* yang telah teraktivasi.

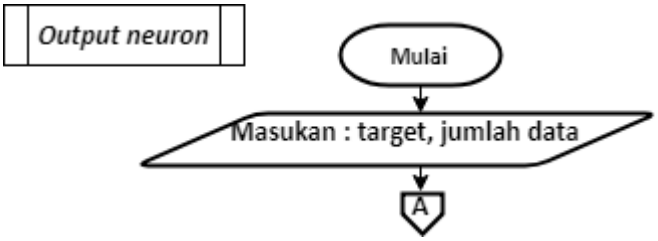


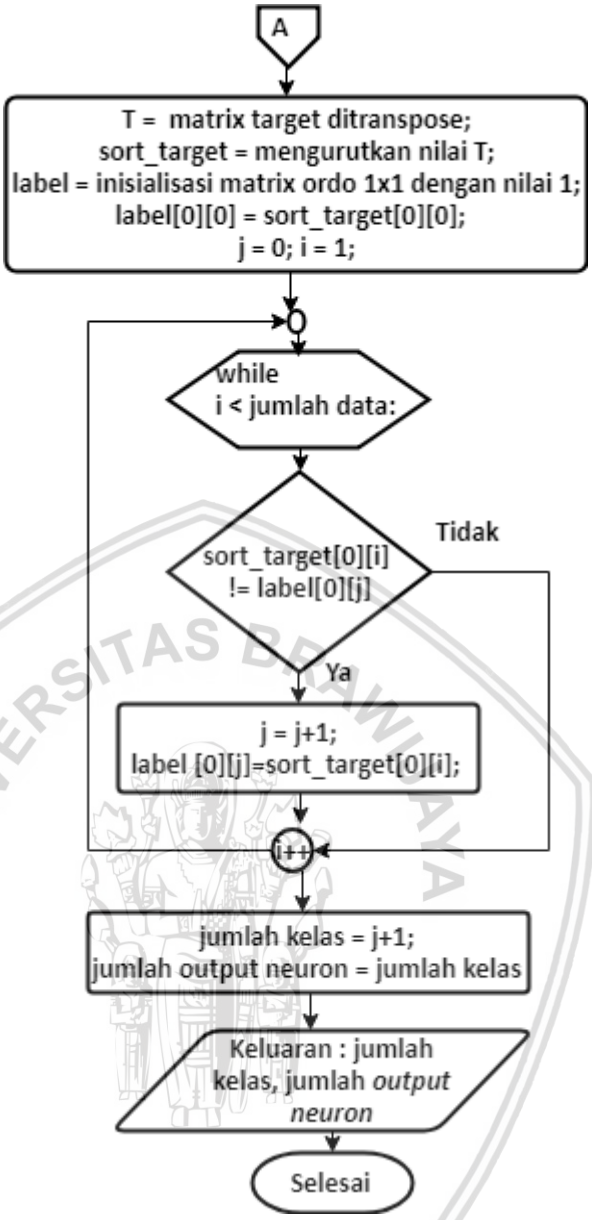

**Gambar 4.22 Diagram alir aktivasi *keluaran hidden neuron***

#### *4.4.1.5 Inisialisasi output neuron*

Proses perhitungan jumlah *outpun neuron* terdiri dari beberapa langkah yang ditunjukkan pada Gambar 4.23,yaitu:

1. Sistem menerima masukan berupa target dan jumlah data
2. Mengurutkan dan mentranspose target
3. Inisialisasi matriks dengan nilai nol
4. Melakukan perulangan sejumlah data
5. Melakukan percabangan if dan menyimpan nilai *sort_target* ke dalam variabel label jika memenuhi syarat if
6. Menyimpan jumlah *j* ke dalam variabel jumlah kelas
7. Inisialisasi variabel jumlah *output neuron* dengan nilai jumlah kelas
8. Keluaran berupa jumlah *output neuron* dan jumlah kelas





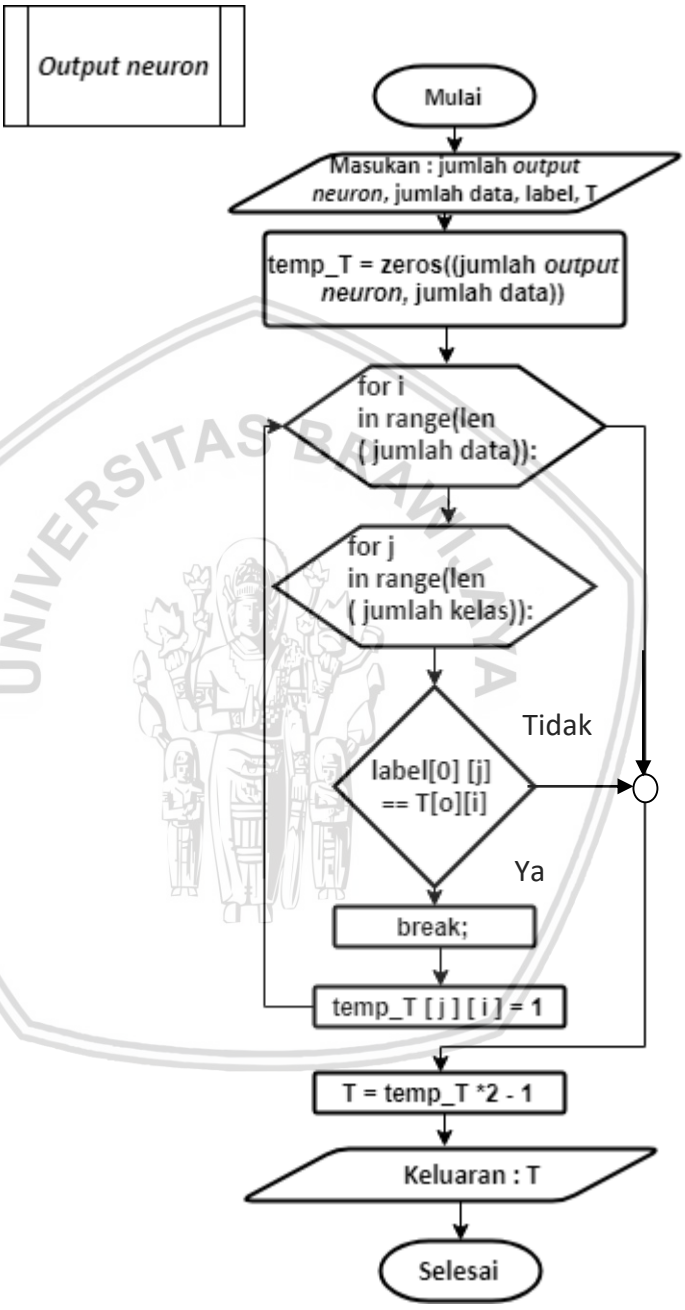

**Gambar 4.23 Diagram alir *output neuron***

#### 4.4.1.6 Matriks target

Proses perhitungan matrix target terdiri dari beberapa langkah yang ditunjukkan pada Gambar 4.24 ,yaitu:

1. Sistem menerima masukan berupa target data training, jumlah data, jumlah kelas, label, dan *T*
2. Inisialisasi matriks nol dengan ordo jumlah *output neuron* x *jumlah data* yang disimpan pada variable *temp_h*
3. Perulangan for *i* sampai sejumlah data
4. Perluangan for *j* sampai sejumlah kelas
5. Percabangan if

6. Menyimpan variabel *temp_h* dengan *indeks [j][i]* dengan nilai 1
7. Melakukan perkalian *temp_h* dengan angka 2, dan kemudian hasilnya dikurangi angka 1 yang disimpan ke dalam variabel *T*
8. Keluaran sistem berupa matriks target



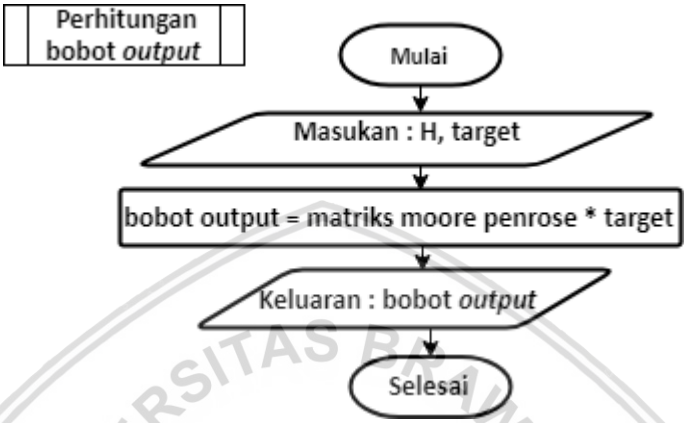

**Gambar 4.24 Diagram alir target**

### *4.4.1.7* Perhitungan bobot *output*

Proses perhitungan bobot *output* dilakukan hanya pada proses *training,* sedangkan pada proses *testing* memakai bobot *output* yang telah dihasilkan pada

proses *training.* Proses perhitungan bobot *output* terdiri dari beberapa tahap yang ditunjukkan pada Gambar 4.25, yaitu:

1. Menerima masukan berupa hasil matriks keluaran *hidden neuron* teraktivasi dan target
2. Menghitung bobot *output* menggunakan Persamaan 2.8
3. Keluaran berupa matriks bobot *output.*

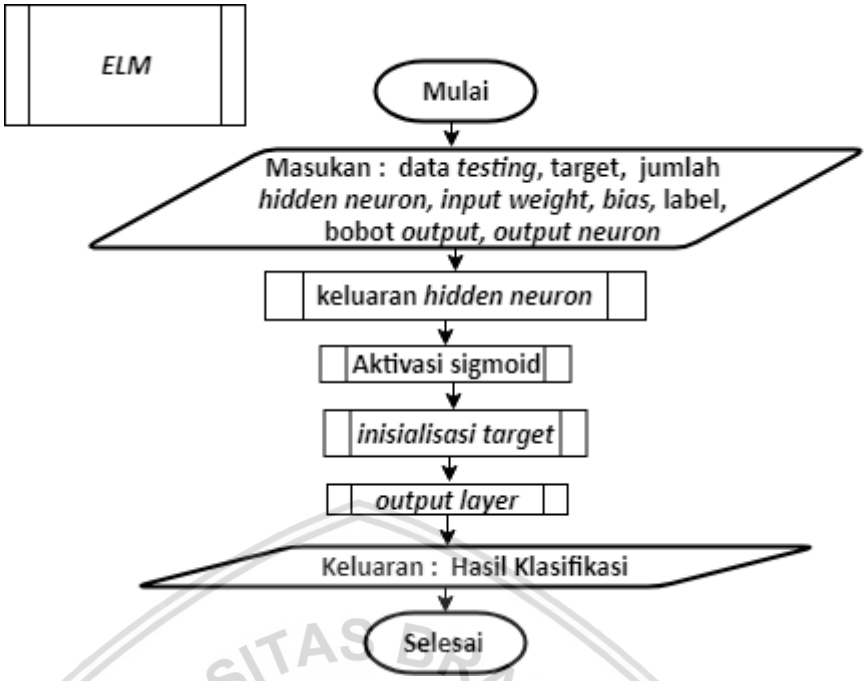



**Gambar 4.25 Diagram alir bobot *output***

### 4.4.2 Testing

Proses *training* dilakukan untuk memperoleh nilai bobot keluaran yang kemudian akan digunakan pada proses *testing*. Proses *testing* terdiri dari beberapa tahap yang ditunjukkan pada Gambar 4.26, yaitu:

1. Mendapatkan masukan berupa data *testing, target, weight input,* jumlah *hidden neuron, output neuron*, nilai *bias*, dan bobot *output*
2. Menghitung keluaran *hidden neuron* menggunakan Persamaan 2.5. diagram alir proses keluaran *hidden neuron* ditunjukkan pada Gambar 4.21
3. Menghitung keluaran *hidden neuron* dengan fungsi aktivasi menggunakan Persamaan 2.6. Diagram alir proses keluaran *hidden neuron* dengan fungsi aktivasi ditunjukkan pada Gambar 4.22
4. Inisialisasi matriks target
5. Menghitung *output layer* menggunakan Persamaan 2.9. diagram alir proses *perhitungan output layer* ditunjukkan pada Gambar 4.27
6. Keluaran sistem yaitu hasil klasifikasi.

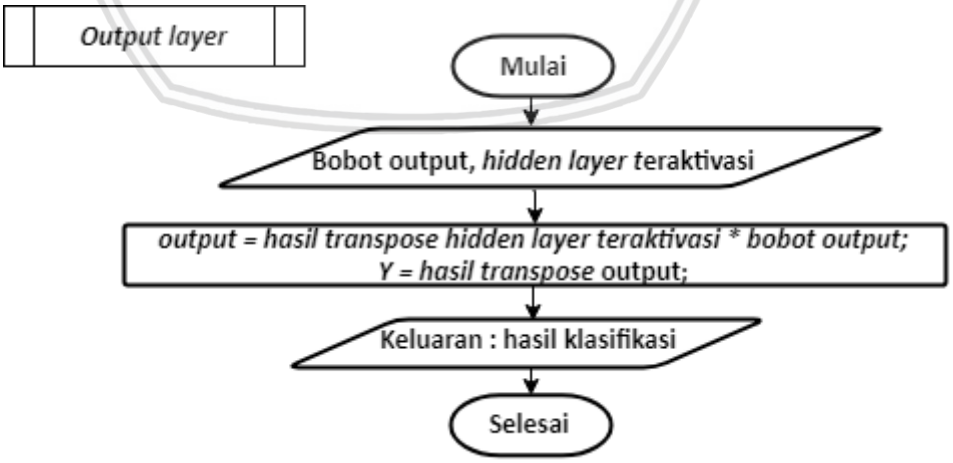



**Gambar 4.26 Diagram alir *testing***

### *4.4.2.1* Perhitungan prediksi kelas *output layer*

Untuk mendapatkan hasil klasfikasi kelas pada data *testing* maka dilakukan perhitungan pada *output layer*. Proses perhitungan kelas terdiri dari beberapa langkah yang ditunjukkan pada Gambar 4.27, yaitu:

1. Menerima masukan berupa bobot *output* dan *hidden layer* teraktivasi
2. Sistem melakukan perhitungan *output layer* dengan menggunakan Persamaan 2.9
3. Keluaran berupa hasil klasifikasi.




**Gambar 4.27 Diagram alir perhitungan *output layer***

## *4.5* Manualisasi Metode *Extreme Learning Machine*

Pada penelitian ini dokumen teks berupa *tweet* yang ditujukan kepada Samsung Indonesia tentang *smartphone* diambil dengan menggunakan Twiter API. Sebelum data diproses menggunakan perhitungan *Extreme Learning Machine,* terlebih dahulu dilakukan *preprocessing* pada data serta dilakukan pembobotan pada 37 fitur. Data yang telah diambil menggunakan Twitter API dibagi menjadi data *training* dan data *testing*. Data *training* merupakan data yang telah diketahui kelasnya yaitu positif, negatif, dan netral. Sedangkan data *testing* merupakan data yang belum diketahui kelasnya sehingga memerlukan proses klasifikasi menggunakan metode *Extreme Learning Machine* sampel data *training* dan data *testing* ditunjukkan pada Tabel 4.1.

**Tabel 4.1 Sample data *training***

| Tweet | Kelas |
|---|---|
| mas, kok sm galaxy a5'17 gue kalo buka play store sering freeze tb2, bahkan setiap buka malah, tolong dong pencerahannya @Samsung_ID | Negatif |
| Kenapa tidak bisa di charger atau tidak connect sama sekali ke hp? @Samsung_ID https://t.co/8N6xxXU0A6 | Netral |
| @tutisunaini @Samsung_ID oke mba makasih banyak yaa hehe :) | Positif |
| @Samsung_ID untuk partner e-commerce resmi Samsung apa ya min ? | Netral |
| @Samsung_ID Oh okey makasih | Netral |
| @Samsung_ID Baru kali ini saya kecewa sama smartphone. Dulu saya pake produk pesaing. Saya pindah pake Samsung karn… https://t.co/WuzAMiwS6R | Negatif |

**Tabel 4.2 Sample data *testing***

| Tweet | Kelas |
|---|---|
| @Samsung_ID my s8+ error. Lag. Hang. Atau apapun namanya. Selama 3 jam lebih?? Kirain cuma hang biasa selama 1 atau… https://t.co/0y0YIf3oeK | Negatif |
| @Samsung_ID Galaxy J5 2016 udah update OS Nougat, Galaxy J7 2016 kapan? Masa J5 doang? | Negatif |
| @Samsung_ID Terbaik @Samsung_ID terima kasih atas informasinya min...\xed\xa0\xbd\xed\xb1\x8d\xed\xa0\xbe\xed\xb4\x9d | Positif |
| @Samsung_ID cek garansi resmi atau distributor gimana ya? | Netral |

### 4.5.1 Tokenisasi

Langkah pertama penelitian ini, data *training* dan *testing* akan dipecah-pecah menjadi *term* atau token atau kata. Proses tersebut dinamakan tokenisasi. Hasil tokenisasi pada data *training* dan data *testing* ditunjukkan pada Tabel 4.3 dan Tabel 4.4

.

**Tabel 4.3 Hasil tokenisasi data *training***

| Hasil tokenisasi |
|---|
| [‘mas,’,’ kok’,‘sm’,‘galaxy’,‘a5'17’,‘gue’,‘kalo’,‘buka’,‘play’,‘store’,‘sering’,‘freeze’,‘tb2,’,‘bahkan’,‘setiap’,‘buka’,‘malah,’,‘tolong’,‘dong’,‘pencerahannya’,‘@Samsung_ID’] |
| [‘Kenapa’,‘tidak’,‘bisa’,‘di’,‘charger’,‘atau’ , ’tidak’,‘connect’,‘sama’,‘sekali’,‘ke’,‘hp?’,‘@Samsung_ID’,‘https://t.co/8N6xxXU0A6’] |
| [‘@tutisunaini’,‘@Samsung_ID’,‘oke’,‘mba’,‘makasih’,‘banyak’,‘yaa’,‘hehe’,‘:)’] |
| [‘@Samsung_ID’,‘untuk’,‘partner’,‘e-commerce’,‘resmi’,‘Samsung’,’apa’,‘ya’,‘min’,‘?’] |
| [‘@Samsung_ID’,‘Oh’,‘okey’,‘makasih’] |
| [‘@Samsung_ID’,‘Baru’,‘kali’,‘ini’,‘saya’,‘kecewa’,‘sama’,‘smartphone.’,‘Dulu’,‘saya’,‘pake’,‘produk’,‘pesaing.’,‘Saya’,‘pindah’,‘pake’,‘Samsung’,‘karn…’,‘https://t.co/WuzAMiwS6R’] |

**Tabel 4.4 Hasil tokenisasi data *testing***

| Hasil tokenisasi |
|---|
| [‘@Samsung_ID’, ‘my’, ‘s8+’, ‘error.’, ‘Lag.’ ,’Hang.’, ‘Atau’, ‘apapun’, ‘namanya.’, ‘Selama’, ‘3’, ‘jam’, ‘lebih??’, ‘Kirain’, ‘cuma’,’hang’, ‘biasa’,’selama’, ‘1’, ‘atau…‘, ‘https://t.co/0y0YIf3oeK’] |
| [‘@Samsung_ID’, ‘Galaxy’, ‘J5’, ‘2016’, ‘udah’, ‘update’, ‘OS’, ‘Nougat,’, ‘Galaxy’, ‘J7’, ‘2016’, ‘kapan?’, ‘Masa’, ‘J5’ ,’doang?’] |
| [‘@Samsung_ID’, ‘Terbaik’, ‘@Samsung_ID’, ‘terima’, ‘kasih’, ‘atas’, ‘informasinya’, ‘min...xed\xa0\xbd\xed\xb1\x8d\xed\xa0\xbe\xed\xb4\x9d’ |
| [‘@Samsung_ID’,‘cek’,‘garansi’,‘resmi’,‘atau’,‘distributor’,‘gimana’,‘ya?’ ] |

### 4.5.2 *Filtering*

Pada proses ini seluruh data *training* dan data *testing* hasil tokenisasi yang termasuk ke dalam daftar *stopword* akan dihapus. Hasil *filtering* pada data *training* dan data *testing* ditunjukkan pada Tabel 4.5 dan Tabel 4.6.

**Tabel 4.5 Hasil *filtering* data *training***

| Hasil Filtering |
|---|
| ['mas,', 'sm', 'galaxy', "a5'17", 'gue', 'kalo', 'buka', 'play', 'store', 'freeze', 'tb2,', 'buka', 'malah,', 'tolong', 'pencerahannya', '@Samsung_ID'] |
| ['charger', 'connect', 'hp?', '@Samsung_ID', 'https://t.co/8N6xxXU0A6'] |
| ['@tutisunaini', '@Samsung_ID', 'oke', 'mba', 'makasih', 'yaa', 'hehe', ':)'] |
| ['@Samsung_ID', 'partner', 'e-commerce', 'resmi', 'Samsung', 'ya', 'min', '?'] |
| ['@Samsung_ID', 'Oh', 'okey', 'makasih'] |
| ['@Samsung_ID', 'kali', 'kecewa', 'smartphone.', 'pake', 'produk', 'pesaing.', 'pindah', 'pake', 'Samsung', 'karn…', 'https://t.co/WuzAMiwS6R’] |

**Tabel 4.6 Hasil *filtering* data *testing***

| Hasil Filtering |
|---|
| ['@Samsung_ID', 'my', 's8+', 'error.', 'Lag.', 'Hang.', 'apapun', 'namanya.', '3', 'jam', 'lebih??', 'Kirain', 'hang', '1', 'atau…', 'https://t.co/0y0YIf3oeK'] |
| ['@Samsung_ID', 'Galaxy', 'J5', '2016', 'udah', 'update', 'OS', 'Nougat,', 'Galaxy', 'J7', '2016', 'kapan?', 'J5', 'doang?'] |
| ['@Samsung_ID', 'Terbaik', '@Samsung_ID', 'terima', 'kasih', 'informasinya', 'min...\\xed\\xa0\\xbd\\xed\\xb1\\x8d\\xed\\xa0\\xbe\\xed\\xb4\\x9d'] |
| ['@Samsung_ID', 'cek', 'garansi', 'resmi', 'distributor', 'gimana', 'ya?'] |

### *4.5.3 POS Tagging*

Pada langkah ini akan seluruh kata pada dokumen *training* dan dokumen *testing* akan diberi *tag*. Tabel 4.7 dan Tabel 4.8 menunjukkan hasil *POS Tagging* pada data *training* dan data *testing*.

**Tabel 4.7 Hasil *POS Tagging* data *training***

| Hasil *POS Tagging* |
|---|
| [['mas', 'X'], [',', 'Z'], ['sm', 'X'], ['galaxy', 'X'], ['a5', 'CD'], ['"\\\\\\"', 'Z'], ['17', 'CD'], ['gue', 'X'], ['kalo', 'X'], ['buka', 'VB'], ['play', 'X'], ['store', 'X'], ['freeze', 'X'], ['tb2', 'CD'], [',', 'Z'], ['buka', 'VB'], ['malah', 'CC'], [',', 'Z'], ['tolong', 'VB'], ['pencerahan', 'NN'], ['nya', 'PRP'], ['@Samsung_ID', 'X']] |
| [['charger', 'X'], ['connect', 'X'], ['hp', 'X'], ['?', 'Z'], ['@Samsung_ID', 'X'], ['https:\\\\\\\\\/\\\\\\\\\/t', 'X'], ['.', 'Z'], ['co\\\\\\\\\/8N6xxXU0A6', 'CD']] |
| [['@tutisunaini', 'X'], ['@Samsung_ID', 'X'], ['oke', 'X'], ['mba', 'X'], ['makasih', 'X'], ['yaa', 'NN'], ['hehe', 'X'], [':)', 'X']] |
| [['@Samsung_ID', 'X'], ['partner', 'X'], ['e-commerce', 'X'], ['resmi', 'JJ'], ['Samsung', 'X'], ['ya', 'X'], ['min', 'JJ'], ['?', 'Z']] |
| [['@Samsung_ID', 'X'], ['Oh', 'NNP'], ['okey', 'X'], ['makasih', 'X']] |
| [['@Samsung_ID', 'X'], ['kali', 'VB'], ['kecewa', 'JJ'], ['smartphone', 'X'], ['.', 'Z'], ['pake', 'X'], ['produk', 'NN'], ['pesaing', 'NN'], ['.', 'Z'], ['pindah', 'VB'], ['pake', 'X'], ['Samsung', 'X'], ['karn???', 'X'], ['https:\\\\\\\\\/\\\\\\\\\/t', 'X'], ['.', 'Z'], ['co\\\\\\\\\/WuzAMiwS6R', 'CD']] |

**Tabel 4.8 Hasil POS Tagging data testing**

| Hasil *POS Tagging* |
|---|
| [['@Samsung_ID', 'X'], ['my', 'X'], ['s8+', 'CD'], ['error', 'X'], ['.', 'Z'], ['Lag', 'NNP'], ['.', 'Z'], ['Hang', 'NNP'], ['.', 'Z'], ['apapun', 'X'], ['nama', 'NN'], ['nya', 'PRP'], ['.', 'Z'], ['3', 'CD'], ['jam', 'NND'], ['lebih', 'RB'], ['??', 'X'], ['Kirain', 'NNP'], ['hang', 'X'], ['1', 'CD'], ['atau???', 'X'], ['https:\\\\\\\\\/\\\\\\\\\/t', 'X'], ['.', 'Z'], ['co\\\\\\\\\/0y0YIf3oeK', 'CD']], |
| [['@Samsung_ID', 'X'], ['Galaxy', 'NNP'], ['J5', 'NNP'], ['2016', 'CD'], ['udah', 'X'], ['update', 'X'], ['OS', 'NNP'], ['Nougat', 'NNP'], [',', 'Z'], ['Galaxy', 'NNP'], ['J7', 'NNP'], ['2016', 'CD'], ['kapan', 'WH'], ['?', 'Z'], ['J5', 'CD'], ['doang', 'RB'], ['?', 'Z']] |
| [['@Samsung_ID', 'X'], ['Terbaik', 'NNP'], ['@Samsung_ID', 'X'], ['terima kasih', 'NN'], ['informasi', 'NN'], ['nya', 'PRP'], ['min', 'JJ'], ['\\\\\\\\\\\\\\\\\\xed\\\\\\\\\\\\\\\\\\xa0\\\\\\\\\\\\\\\\\\xbd', 'CD']] |
| [['@Samsung_ID', 'X'], ['cek', 'NN'], ['garansi', 'NN'], ['resmi', 'JJ'], ['distributor', 'NN'], ['gimana', 'X'], ['ya', 'X'], ['?', 'Z']] |

### 4.5.4 Hasil TF-IDF

Setelah dilakukan proses *preprocessing* data *training* dan data *testing* akan dilakukan proses pembobotan kata dengan menggunakan perhitungan nilai $T_f$ , $w_{tf}$, *DF*, *IDF*, dan *TF-IDF*. Hasil perhitungan $T_f$ ditunjukkan pada Tabel 4.9.

**Tabel 4.9 Hasil perhitungan $T_f$**

| | mas | sm | galaxy | a517 | ... | pesaing | Pindah | karn... | https://t.co/wuzamiws6r |
|---|---|---|---|---|---|---|---|---|---|
| **L1** | 1 | 1 | 1 | 1 | ... | 0 | 0 | 0 | 0 |
| **L2** | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| **L3** | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **U2** | 0 | 0 | 2 | 0 | ... | 0 | 0 | 0 | 0 |
| **U3** | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| **U4** | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |

Setelah didapatkan nilai $T_f$ maka hitung nilai $w_{tf}$ menggunakan Persamaan 2.1 dan nilai *DF*. Tabel 4.10 menunjukkan nilai hasil $w_{tf}$ dan *Df*.

**Tabel 4.10 Hasil perhitungan $w_{tf}$ dan *Df***

| | mas | sm | Galaxy | a517 | .. | pesaing | Pindah | karn... | https://t.co/wuzamiws6r |
|---|---|---|---|---|---|---|---|---|---|
| **L1** | 1 | 1 | 1 | 1 | .. | 0 | 0 | 0 | 0 |
| **L2** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **L3** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **...** | **...** | **...** | **...** | **...** | **..** | **...** | **...** | **...** | **...** |
| **U2** | 0 | 0 | 1,30103 | 0 | .. | 0 | 0 | 0 | 0 |
| **U3** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **U4** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **DF** | 1 | 1 | 1 | 1 | .. | 1 | 1 | 1 | 1 |

Setelah didapatkan nilai $w_{tf}$ dan *Df* maka hitung nilai *IDF* menggunakan Persamaan 2.2. Tabel 4.11 menunjukkan hasil *IDF*.

**Tabel 4.11 Hasil perhitungan *IDF***

| | mas | Sm | galaxy | a517 | .. | Pesaing | pindah | karn... | https://t.co/wuzamiws6r |
|---|---|---|---|---|---|---|---|---|---|
| **IDF** | 0,778151 | 0,778151 | 0,778151 | 0,778151 | .. | 0,778151 | 0,778151 | 0,778151 | 0,778151 |

Setelah didapatkan nilai *IDF* maka hitung nilai *TF-IDF* menggunakan Persamaan 2.3. Hasil perhitungan *TF-IDF* ditunjukkan pada Tabel 4.12.

**Tabel 4.12 Hasil perhitungan *TF-IDF***

| | mas | Sm | galaxy | a517 | .. | pesaing | pindah | karn... | https://t.co/wuzamiws6r |
|---|---|---|---|---|---|---|---|---|---|
| **L1** | 0,778151 | 0,778151 | 0,778151 | 0,778151 | .. | 0 | 0 | 0 | 0 |
| **L2** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **L3** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **U2** | 0 | 0 | 1,012398 | 0 | .. | 0 | 0 | 0 | 0 |
| **U3** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |
| **U4** | 0 | 0 | 0 | 0 | .. | 0 | 0 | 0 | 0 |

### *4.5.5 Lexicon based features*

Pada proses ini dilakukan pembobotan berdasarkan kamus dengan 15 fitur. Hasil pembobotan *lexicon based features* ditunjukkan pada Tabel 4.13.

**Tabel 4.13 Hasil pembobotan *lexicon based fetures***

| | **Positif** | **Negatif** | **kata sifat positif** | **...** | **persentase kata keterangan negatif** | **jumlah kata intesifier** |
|---|---|---|---|---|---|---|
| D1 | 1 | 0 | 1 | ... | 0 | 0 |
| D2 | 0 | 0 | 0 | ... | 0 | 0 |
| D3 | 0 | 0 | 0 | ... | 0 | 0 |
| D4 | 0 | 0 | 0 | ... | 0 | 0 |
| D5 | 0 | 0 | 0 | ... | 0 | 0 |
| D6 | 0 | 1 | 0 | ... | 0 | 1 |
| U1 | 0 | 1 | 0 | ... | 0 | 0 |
| U2 | 0 | 0 | 0 | ... | 0 | 0 |
| U3 | 0 | 0 | 0 | ... | 0 | 0 |
| U4 | 0 | 0 | 0 | ... | 0 | 0 |

### *4.5.6* Penggabungan *bag of words* dan *lexicon based features*

Pada proses ini hasil pembobotan kata dan pembobotan berdasarkan kamus digabungkan menjadi satu tabel. Tabel 4.14 menunjukkan hasil penggabungan *bag-of-word* dan *lexicon based features.*

**Tabel 4.14 Hasil penggabungan *bag-of-words* dan *lexicon based features***

| | mas | sm | galaxy | ... | *Persentase kata keterangan negatif* | *jumlah kata intesifier* |
|---|---|---|---|---|---|---|
| ***L1*** | 0,778151 | 0,778151 | 0,778151 | ... | *0* | *0* |
| ***L2*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***L3*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***L4*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***...*** | ***...*** | ***...*** | ***...*** | ***...*** | ***...*** | ***...*** |
| ***U2*** | 0 | 0 | 1,012398 | ... | *0* | *0* |
| ***U3*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***U4*** | 0 | 0 | 0 | ... | *0* | *0* |

### 4.5.7 Normalisasi

Pada proses ini hasil dari pembobotan kata dan pembobotan berdasarkan kamus yang telah digabungkan dilakukan normalisasi agar tidak terdapat selisih yang jauh. Tabel 4.15 menunjukkan hasil normalisasi.

**Tabel 4.15 Hasil normalisasi**

| | mas | sm | Galaxy | ... | *Persentase kata keterangan negatif* | *jumlah kata intesifier* |
|---|---|---|---|---|---|---|
| ***L1*** | 0,9999 | 0,9999 | 0,53719 | ... | *0* | *0* |
| ***L2*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***L3*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***L4*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***...*** | ***...*** | ***...*** | ***...*** | ***...*** | ***...*** | ***...*** |
| ***U2*** | 0 | 0 | 0,9999 | ... | *0* | *0* |
| ***U3*** | 0 | 0 | 0 | ... | *0* | *0* |
| ***U4*** | 0 | 0 | 0 | ... | *0* | *0* |

### *4.5.8 Weight input*

Inisialisasi matriks *weight input* dilakukan secara acak dengan aturan L x M yang kemudian dikalikan dengan 2 dan dikurangi 1, dimana M merupakan jumlah atribut, sedangkan L merupakan jumlah *hidden neuron.* Pada manualisasi penelitian ini menginisialisasi jumlah *hidden neuron* sebanyak 2. Tabel 4.16 menunjukkan nilai *weight input.*

**Tabel 4.16 Nilai *weight input***

| W | 1 | 2 |
|---|---|---|
| 1 | 0,032344 | 0,562458 |
| 2 | 0,643652 | -0,78421 |
| 3 | -0,06587 | 0,582826 |
| 4 | -0,94187 | 0,879402 |
| 5 | -0,81288 | -0,05699 |
| 6 | 0,842292 | 0,373506 |
| ... | ... | ... |
| 58 | 0,70433 | 0,417348 |
| 59 | 0,486537 | 0,326743 |

### 4.5.9 Bias

Inisialisasi matriks *bias* dilakukan secara acak dengan aturan *hidden neuron* x 1, sehingga setiap *neuron* memiliki nilai yang sama. Tabel 4.17 menunjukkan nilai *weight input.*

**Tabel 4.17 Nilai bias**

| B | 1 |
|---|---|
| 1 | 0,655304 |
| 2 | 0,164822 |

### 4.5.10 Proses Training

Tahapan yang terdapat pada proses *training*, sebagai berikut:

#### *4.5.10.1 Output hidden neuron*

*Output hidden neuron* dapat dihitung menggunakan Persamaan 2.5. berikut contoh perhitungan *output hidden*

$$H_{init} = \omega^T . \chi + bH_{init}$$
$$= ((0{,}03\ x\ \ 0{,}78) + (0{,}64\ x\ 0{,}78) + (-0{,}07\ x\ 0{,}78) + \cdots$$
$$+ \quad (0{,}70\ x\ 0) + (0{,}49x0)) + 0{,}65$$
$$H_{init} = 1{,}25$$

Hasil *output hidden neuron* ditunjukkan pada Tabel 4.18

**Tabel 4.18 Nilai *output hidden neuron***

| Hinit | 1 | 2 |
|---|---|---|
| 1 | 1,250951 | -1,4639 |
| 2 | 0,169783 | 0,770613 |
| 3 | -0,74445 | 0,139846 |
| 4 | 1,474661 | 0,051894 |
| 5 | 1,962681 | 1,823641 |
| 6 | -0,99455 | -1,90365 |

### *4.5.10.2* Menghitung fungsi aktivasi

*Setelah* didapatkan hasil o*utput hidden neuron* kemudian dipetakan menggunakan fungsi aktivasi *sigmoid biner* sehingga menghasilkan nilai pada rentang *0 sampai 1.* Berikut contoh perhitungan fungsi aktivasi dengan menggunakan Persamaan 2.6.

$$H = \frac{1}{\left(1 + exp^{-\chi.\omega^T + b}\right)} = \frac{1}{(1 + exp^{-1,25})} = 0,78$$

Hasil *output hidden neuron* setelah diaktivasi ditunjukkan pada Tabel 4.19

**Tabel 4.19 Nilai *output hidden neuron* dengan fungsi aktivasi**

| H | 1 | 2 |
|---|---|---|
| 1 | 0,777464 | 0,187872039 |
| 2 | 0,542344 | 0,683653473 |
| 3 | 0,322033 | 0,534904672 |
| 4 | 0,813765 | 0,512970496 |
| 5 | 0,876823 | 0,861002441 |
| 6 | 0,270014 | 0,12969599 |

### *4.5.10.3* Menghitung matriks *Moore-Penrose Pseudo Inverse* $(\boldsymbol{H}^{+})$

Untuk perhitungan matriks *Moore-Penrose Pseudo Inverse* dapat dilakukan menggunakan rumus pada Persamaan 2.7. Hasil perhitungan matriks *Moore-Penrose Pseudo Inverse* ditunjukkan pada Tabel 4.20*.*

**Tabel 4.20 Nilai matriks *Moore-Penrose Pseudo Inverse***

| $H^{+}$ | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 6 | 0.5625 | -3.5 | 1 | -0.5 | -2 |
| 2 | 1.625 | -1.140625 | 0.5 | -2.9375 | 0.5 | 3 |

### *4.5.10.4* Menghitung *output weight*

*Untuk mendapatkan output weight* maka dapat melakukan perkalian antara matriks *Moore-Penrose Pseudo Inverse* dengan target sesuai Persamaan 2.8. Hasil dari *output weight* akan digunakan untuk proses *testing.* Contoh perhitungan *output weight* sebagai berikut:

$$\beta = (6\,x - 1) + (0{,}5625\,x - 1) + (-3{,}5x1) + (1x - 1) + (-0{,}5x - 1) + (-2x - 1)$$

$$= -8{,}56$$

Hasil perhitungan *output weight* ditunjukkan pada Tabel 4.21

**Tabel 4.21 Nilai *output weight***

| $\beta$ | 1 | 2 | 3 |
|---|---|---|---|
| 1 | -8,5625 | 6,4375 | 0,5625 |
| 2 | -0,54688 | 7,703125 | -8,70313 |

### 4.5.11 Proses *testing*

Setelah tahap proses *training* selesai maka tahap berikutnya yaitu proses *testing.* Pada proses *testing* meggunakan *input weight, bias,* dan *output weight* yang didapatkan dari proses *training.* Tahapan yang terdapat pada proses *testing*, sebagai berikut:

#### *4.5.11.1 Output hidden neuron*

*Output hidden neuron* dapat dihitung menggunakan Persamaan 2.5. berikut contoh perhitungan *output hidden neuron*

$$H\_init = \omega\text{^}T.\,\chi + bH\_init$$

$$= ((0{,}03\,x\ \ 0) + (0{,}64\,x\,0) + (-0{,}07\,x\,0) + \cdots + (0{,}70\,x\,0) + (0{,}49x0)) + 0{,}65$$

$$H_{init} = 1{,}55$$

Hasil *output hidden neuron* ditunjukkan pada Tabel 4.22

**Tabel 4.22 Nilai *output hidden neuron***

| Hinit | 1 | 2 |
|---|---|---|
| 1 | 1,554682349 | -0,61099266 |
| 2 | 0,589438035 | 0,747589003 |
| 3 | 0,655303647 | 0,164821669 |
| 4 | 0,775490192 | 0,798519147 |
| 5 | 1,554682349 | -0,61099266 |
| 6 | 0,589438035 | 0.747589003 |

#### *4.5.11.2* Menghitung fungsi aktivasi

*Setelah* didapatkan hasil o*utput hidden neuron* kemudian dipetakan menggunakan fungsi aktivasi *sigmoid biner* sehingga menghasilkan nilai pada rentang *0 sampai 1.* Berikut contoh perhitungan fungsi aktivasi dengan menggunakan Persamaan 2.6.

$$H = \frac{1}{(1 + exp^{-\chi.\omega^T + b})} = \frac{1}{(1 + exp^{-1{,}55})} = 0{,}83$$

Hasil *output hidden neuron* setelah diaktivasi ditunjukkan pada Tabel 4.23.

**Tabel 4.23 Nilai *ouput hidden neuron* dengan fungsi aktivasi**

| H | 1 | 2 |
|---|---|---|
| 1 | 0,82558898 | 0,351832792 |
| 2 | 0,643236194 | 0,678653128 |
| 3 | 0,65820463 | 0,541112387 |
| 4 | 0,684707334 | 0,689657623 |
| 5 | 0,82558898 | 0,351832792 |
| 6 | 0,643236194 | 0,678653128 |

### *4.5.11.3* Menghitung $\hat{Y}$ prediksi

Proses terakhir yaitu menghitung $\hat{Y}$ prediksi. $\hat{Y}$ prediksi dapat dihitung menggunakan Persamaan 2.9. Hasil dari $\hat{Y}$ prediksi menentukan kelas prediksi dari data. Setelah didapatkan nilai $\hat{Y}$ prediksi maka dicari nilai maksimal. Nilai maksimal tersebut merepresentasikan kelas data baru. Contoh perhitungan $\hat{Y}$ prediksi sebagai berikut:

$$\hat{Y} = (0{,}83\,x - 8{,}56) + (0{,}35x - 0{,}55)$$

$$\hat{Y} = -7{,}26$$

Hasil $\hat{Y}$ prediksi beserta hasil klasifikasi ditunjukkan pada Tabel 4.24.

**Tabel 4.24 Hasil $\hat{Y}$ prediksi**

| $\hat{Y}$ | Positif | negatif | netral |
|---|---|---|---|
| 1 | -7,261514 | 8,024941035 | -2,59765 |
| 2 | -5,878848 | 9,3685829 | -5,54458 |
| 3 | -5,931798 | 8,405448665 | -4,33913 |
| 4 | -6,239963 | 9,720322343 | -5,61703 |

Setelah didapatkan nilai $\hat{Y}$ prediksi maka dicari nilai yang maksimum. Nilai maksimum tersebut merepresentasikan hasil kelas proses klasifikasi. Tabel 4.25 menunjukkan hasil klasifikasi.

**Tabel 4.25 Hasil klasifikasi**

| | Positif | negatif | netral | maks | prediksi | target |
|---|---|---|---|---|---|---|
| 1 | -7,261514 | 8,024941 | -2,59765 | 8.024941 | Negatif | Negatif |
| 2 | -5,878848 | 9,368582 | -5,54458 | 9.368583 | Negatif | Negatif |
| 3 | -5,931798 | 8,405449 | -4,33913 | 8.405449 | Negatif | Positif |
| 4 | -6,239963 | 9,720322 | -5,61703 | 9.720322 | Negatif | Netral |

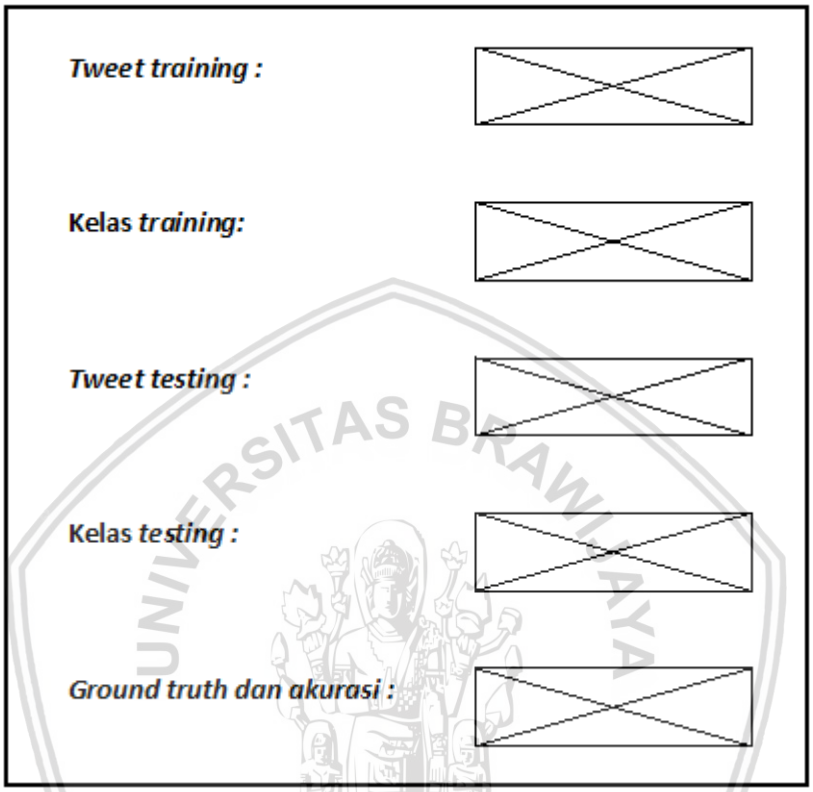
## 4.6 Hasil akurasi

Akurasi hasil klasifikasi dapat dihitung menggunakan Persamaan 2.15. Contoh perhitungan akurasi sebagai berikut.

$\text{Akurasi} = \frac{2}{4} = 0{,}5$

## 4.7 Perancangan antarmuka sistem

Perancangan antarmuka sistem analisis sentimen Twitter menggunakan metode *Extreme Learning Machine* dan *ensemble* fitur dapat dilihat pada Gambar 4.28.




**Gambar 4.28 Perancangan antarmuka**

## 4.8 Perancangan Pengujian

Pada sistem ini terdapat tiga macam Pengujian yaitu pengaruh Pengujian perbandingan fungsi aktivasi, pengaruh Pengujian *ensemble features* terhadap akurasi, dan *confusion matrix* dari parameter yang paling optimal.

### 4.8.1 Pengujian pengaruh jumlah *hidden neuron terhadap akurasi*

Pengujian pengaruh jumlah *hidden layer* terhadap akurasi dilakukan dengan lima kali percobaan serta menggunakan jumlah *hidden neuron* secara acak yaitu: 10, 50, 100, 500, 1000, 5000, dan 10000. Fungsi aktivasi yang digunakan adalah sigmoid bipolar, dimana data yang digunakan memiliki rentang antara 0 sampai 1. Pengujian ini dilakukan untuk mengetahui apakah jumlah *hidden neuron* memiliki pengaruh terhadap hasil akurasi sistem. Perancangan Pengujian pengaruh jumlah *hidden neuron* ditunjukkan pada Tabel 4.26.

**Tabel 4.26 Perancangan Pengujian jumlah *hidden neuron***

| Nilai akurasi sistem | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Percobaan ke** | **10** | **50** | **100** | **500** | **1000** | **5000** | **10000** |
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |
| 5 | | | | | | | |
| Rata-rata | | | | | | | |
| Standar deviasi | | | | | | | |

### 4.8.2 Pengujian perbandingan fungsi aktivasi

Pengujian fungsi aktivasi bertujuan untuk mengubah suatu *input* menjadi suatu *output* tertentu. (Ariestyani,2017). Pada penelitian Srimuang(2015) terdapat beberapa fungsi aktivasi, yaitu: *sigmoid, hard limit, triangular basis, radial basis, linear,* dan *sin*. Perancangan Pengujian pengaruh *ensemble features* terhadap akurasi sistem ditunjukkan pada Tabel 4.27.

**Tabel 4.27 Perancangan Pengujian perbandingan fungsi aktivasi**

| Nilai akurasi sistem | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Percobaan ke** | **Sigmoid biner** | **Hard limit** | **Triagular basis** | **Radial basis** | **Linear** | **Sin** | **Sigmoid bipolar** |
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |
| 5 | | | | | | | |
| Rata-rata | | | | | | | |
| Standar deviasi | | | | | | | |

### 4.8.3 Pengujian *K-Fold Crossvalidation*

Pengujian *K-Fold Crossvalidation* di lakukan dengan jumlah *K* sebanyak 4, di mana setiap *K* memiliki 18 data dengan jumlah setiap kelas yang seimbang sebanyak 6 data tiap kelas. Jumlah *hidden neuron* yang digunakan pada Pengujian ini sebanyak 5000 serta menggunakan fungsi aktivasi *sigmoid bipolar*. Nilai hasil akurasi Pengujian *K-Fold* ditunjukkan pada Tabel 4.28.

**Tabel 4.28 Pengujian *K-Fold crossvalidation***

| Fold 1 | | Fold 2 | | Fold 3 | | Fold 4 | | Rata-rata | |
|---|---|---|---|---|---|---|---|---|---|
| Training | Testing | Training | Testing | Training | Testing | Training | Testing | Training | Testing |
| | | | | | | | | | |

### 4.8.4 Pengujian pengaruh *ensemble features* terhadap akurasi sistem

Pengujian pengaruh *ensemble feature* digunakan untuk mengetahui apakah analisis sentimen menggunakan *bag-of-word* dan *ensemble features* memiliki pengaruh yang lebih besar dengan nilai akurasi yang lebih tinggi dibandingkan dengan hanya menggunakan *bag-of-word* dan *lexicon based features* yang pada penelitan Rofiqoh (2017) memiliki pengaruh terhadap analisis sentimen. Perancangan Pengujian pengaruh *ensemble features* terhadap akurasi sistem ditunjukkan pada Tabel 4.29.

**Tabel 4.29 Perancangan pengujian pengaruh *ensemble features***

| Nilai akurasi sistem | | | |
|---|---|---|---|
| **Percobaan ke** | **Ensemble features** | ***Bag-of-words*** | ***Lexicon based feature* dan bag of word** |
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| Rata-rata | | | |

### 4.8.5 *Confusion matrix* dari parameter yang paling optimal

Pengujian *confusion matrix* bertujuan untuk mengukur kemampuan teknik klasifikasi sentimen. Perancangan *confusion matrix* dari parameter yang paling optimal ditunjukkan pada Tabel 4.30.

**Tabel 4.30 Perancangan pengujian *confusion matrix***

| Sistem / Pakar | Kelas Positif | Kelas Negatif | Netral |
|---|---|---|---|
| Kelas Positif | | | |
| Kelas Negatif | | | |
| Kelas Netral | | | |
| Rata-rata | | | |

## 4.9 Implementasi sistem

Pada sub bab ini akan dijelaskan *source code* metode *ELM* yang merupakan implementasi dari algoritme *preprocessing*, pembobotan kata, pembobotan

*ensemble features,* dan metode *Elm* pada sub bab perancangan yang digunakan dalam proses klasifikasi.

### *4.9.1 Preprocessing text*

Preprocessing text terdiri dari beberapa tahap yaitu: tokenisasi, *filtering, dan POS Tagging.*

```
1   def Tokenisasi(dokumen):
2       kata = dokumen.split()
3       return kata
4   def Filtering(dokumen):
5       hasil=[]
6       list_stopword=[]
7       with open("D:/fita/leksikon/id.stopwords.02.01.2016.txt") as f:
8           content = f.readlines()
9       list_stopwords = [x.strip() for x in content]
10      for word in dokumen:
11          if word.lower() not in list_stopwords:
12                  hasil.append(word)
13
14      return hasil
15  def Postagger(dokumen):
16      payload = {
17      "Text[value]": dokumen
18      }
19      r    =    requests.post('http://bahasa.cs.ui.ac.id/postag/API/tag',
20  data=payload)
21      return (r.content)
```

**Source code 4.1 *preprocessing text***

Penjelasan Source code 4.1 tentang implementasi proses *preprocessing text* sebagai berikut:

Baris 1-3 : proses tokenisasi

Baris 4-14 : proses *filtering*

Baris 15-21: proses *POS Tagging*

### 4.9.2 Pembobotan Kata

```
1   def KataUnik(dokumen):
2
3       for i in range(len(dokumen)):
4           for j in dokumen[i]:
5               if j.lower() not in unik:
6                   unik.append(j.lower())
7                   df.append(0)
8       return unik
9
10  #jumlah Tf dan df pada seluruh dokumen
11  def TermFrequency(dokumen,train):
12      tf_total = []
13      for j in range(len(dokumen)):
14          tf=[]
15          for i in range(len(unik)):
16              tf_dok = 0
17              for k in dokumen[j]:
18                  if unik[i] in k:
19                      tf_dok+=1 #jumlah kata unik pada suatu dokumen
20              if(train==0):
21                  if(tf_dok>0):
```

```
                    df[i]= df[i]+1 #jumlah dokumen yg memiliki kata
unik
                else:
                    df[i]=df[i]
            tf.append(tf_dok)

        tf_total.append(tf)
    return tf_total

#menghitung wtft
def LogTermFrequency(tf_total):
    wtf_total = []
    for i in range(len(tf_total)):
        wtf=[]
        for j in range(len(tf_total[i])):
            if tf_total[i][j]!=0:
               wt=1+math.log10(tf_total[i][j])
            else:
                wt = 0
            wtf.append(wt)
        wtf_total.append(wtf)
    return wtf_total

#menghitung invers dokumen frekuensi
def InversDocumentFrequency():
    idf_total = []
    for i in range(len(df)):
        if(df[i])>0:
            idf= math.log10(len(df)/df[i])
        else:
            idf=0
        idf_total.append(idf)
    return idf_total

#menghitung Tf x Idf
def TFxIDF(wtf_total, idf_total):
    tfidf_total = []
    for j in range(len(wtf_total)):
        tf_idft=[]
        for k in range(len(wtf_total[j])):
            tfidf=0
            tfidf= wtf_total[j][k]*idf_total[j]
            tf_idft.append(tfidf)
        tfidf_total.append(tf_idft)
    return tfidf_total
```

**Source code 4.2 pembobotan kata**

Penjelasan *Source Code 4.1* tentang implementasi proses inisialisasi *input weight* dan bias adalah sebagai berikut:

Baris 1-8 : proses mengumpulkan kata yang terdapat pada seluruh dokumen yang kemudian akan dijadikan sebagai kata unik

Baris 10-29: proses perhitungan *term frequency* dan *document frequency*

Baris 31-43: proses perhitungan bobot dari *term frequency*

Baris 45-54: proses perhitungan *invers document frequency*

Baris 56-66: proses perhitungan *TF-IDF*

### *4.9.2.1 Pembobotan ensemble features*

```
def Hastag(tweet):
    bobot_hastag = []
    for i in range(len(tweet)):
        bobot=0
        for j in range(len(tweet[i])):
            matchObj = re.search( r'#[\w|\W].*', tweet[i][j])
            if matchObj:
                bobot = 1;
            else:
                bobot = 0;
        bobot_hastag.append(bobot)
    return bobot_hastag

def Retweet(tweet):
    bobot_rt = []
    for i in range(len(tweet)):
        bobot=0
        for j in range(len(tweet[i])):
            matchObj = re.match( r'RT', tweet[i][j])
            if matchObj:
                bobot = 1
        bobot_rt.append(bobot)
    return bobot_rt

def Username(tweet):
    bobot_username = []
    for i in range(len(tweet)):
        bobot=0
        for j in range(len(tweet[i])):
            matchObj = re.search( r'@[\w|\W].*', tweet[i][j])
            if matchObj:
                bobot = 1
        bobot_username.append(bobot)
    return bobot_username
def Url(tweet):
    bobot_url = []
    for i in range(len(tweet)):
        bobot=0
        for j in range(len(tweet[i])):
            matchObj = re.search( r'(https)[\w|\W].*', tweet[i][j])
            if matchObj:
                bobot = 1
        bobot_url.append(bobot)
    return bobot_url
def TweetLength(tweet):
    tweet_length = []
    average_length = []
    for i in range(len(tweet)):
        tweet_length.append(len(tweet[i]))
        jumlahKarakter = 0
        for j in range(len(tweet[i])):
           jumlahKarakter += len(tweet[i][j]) #jumlah panjang karakter
pada suatu dokumen/ kalimat/ tweet
        average = jumlahKarakter/len(tweet[i])
        average_length.append(average)
    return tweet_length, average_length

def TdanaTanya(tweet):
    tdana_tanya=[]
    for i in range(len(tweet)):
        tanya = 0
        for j in range(len(tweet[i])):
            for k in range(len(tweet[i][j])):
```

```
                if tweet[i][j][k] == '?':
                    tanya+=1
        tdana_tanya.append(tanya)
    return tdana_tanya

def TdanaSeru(tweet):
    tdana_seru = []
    for i in range(len(tweet)):
        seru = 0
        for j in range(len(tweet[i])):
            for k in range(len(tweet[i][j])):
                if tweet[i][j][k] == '!':
                    seru+=1
        tdana_seru.append(seru)
    return tdana_seru

def Quote(tweet):
    tanya = 0
    quote = []

    for i in range(len(tweet)):
        petik = 0
        for j in range(len(tweet[i])):
            for k in range(len(tweet[i][j])):
                if tweet[i][j][k] == '"':
                    petik +=1
        quote.append(petik)
    return quote

def Upper(tweet):
    upper = []
    for i in range(len(tweet)):
        up = 0
        for j in range(len(tweet[i])):
            matchObj = re.search( r'^[A-Z]', tweet[i][j])
            if matchObj:
                up+= 1;
        upper.append(up)
    return upper
def EmotPositif(tweet, lexicon_emot_positif):
    emoticon_positif = []
    for i in range(len(tweet)):
        emoticonP=0
        for j in range(len(tweet[i])):
            for k in range(len(lexicon_emot_positif)):
                if tweet[i][j] == lexicon_emot_positif[k]:
                    emoticonP = 1
        emoticon_positif.append(emoticonP)
    return emoticon_positif

def EmotNegatif(tweet, lexicon_emot_negatif):
    emoticon_negatif = []
    for i in range(len(tweet)):
        emoticon=0
        for j in range(len(tweet[i])):
            for k in range(len(lexicon_emot_negatif)):
                if tweet[i][j] == lexicon_emot_negatif[k]:
                    emoticon = 1
        emoticon_negatif.append(emoticon)
    return emoticon_negatif

def Noun(postager):
    number_noun = []
    for i in range(len(postager)):
        pos=0
```

```
129         for j in range(len(postager[i])):
130             if(postager[i][j][1]=="NN"):
131                 pos=pos+1
132         number_noun.append(pos)
133 
134     return number_noun
135 def Adjective(postager):
136     number_adjective = []
137     adjective_positif = []
138     adjective_negatif = []
139     for i in range(len(postager)):
140         positif=0
141         negatif=0
142         pos=0
143         for j in range(len(postager[i])):
144             if(postager[i][j][1]=="JJ"):
145                 pos=pos+1
146                 for k in range(len(leksikon_positif)):
147                     if        (postager[i][j][0]).lower()          ==
148 leksikon_positif[k]:
149                         positif+= 1
150                 for k in range(len(leksikon_negatif)):
151                     if        (postager[i][j][0]).lower()          ==
152 leksikon_negatif[k]:
153                         negatif+= 1
154         number_adjective.append(pos)
155         adjective_positif.append(positif)
156         adjective_negatif.append(negatif)
157     return number_adjective,adjective_positif, adjective_negatif
158 
159 def Verb(postager):
160     number_verb = []
161     verb_positif = []
162     verb_negatif = []
163     for i in range(len(postager)):
164         positif=0
165         negatif=0
166         pos=0
167         for j in range(len(postager[i])):
168             if(postager[i][j][1]=="VB"):
169                 pos=pos+1
170                 for k in range(len(leksikon_positif)):
171                     if          (postager[i][j][0]).lower()          ==
172 leksikon_positif[k]:
173                         positif+= 1
174                 for k in range(len(leksikon_negatif)):
175                     if          (postager[i][j][0]).lower()          ==
176 leksikon_negatif[k]:
177                         negatif+= 1
178         number_verb.append(pos)
179         verb_positif.append(positif)
180         verb_negatif.append(negatif)
181     return number_verb,verb_positif, verb_negatif
182 def Adverb(postager):
183     number_adverb =[]
184     adverb_positif =[]
185     adverb_negatif =[]
186     for i in range(len(postager)):
187         positif=0
188         negatif=0
189         pos=0
190         for j in range(len(postager[i])):
191             if(postager[i][j][1]=="RB"):
192                 pos=pos+1
193                 for k in range(len(leksikon_positif)):
```

```
                    if          (postager[i][j][0]).lower()          ==
leksikon_positif[k]:
                        positif+= 1
                for k in range(len(leksikon_negatif)):
                    if          (postager[i][j][0]).lower()          ==
leksikon_negatif[k]:
                        negatif+= 1
        number_adverb.append(pos)
        adverb_positif.append(positif)
        adverb_negatif.append(negatif)
    return number_adverb,adverb_positif, adverb_negatif

def Interjection(postager):
    number_interjection = []
    for i in range(len(postager)):
        pos=0
        for j in range(len(postager[i])):
            if(postager[i][j][1]=="UH"):
                pos=pos+1
        number_interjection.append(pos)
    return number_interjection
def Percentage(a,b):
   %_pos=[]
    for i in range(len(a)):
        if(a[i]!=0):
            if(b[i]!=0):
               % = a[i]/b[i]
            else:
               %=0
        else:
           %=0
       %_pos.append(persen)
    return%_pos

def PositifTweet(tweet):
    positif_tweet = []
    for i in range(len(tweet)):
        kata =0
        for j in range(len(tweet[i])):
            for k in range(len(leksikon_positif)):
                if tweet[i][j] == leksikon_positif[k]:
                    kata+= 1
        positif_tweet.append(kata)
    return positif_tweet

def NegatifTweet(tweet):
    negatif_tweet = []
    for i in range(len(tweet)):
        kata =0
        for j in range(len(tweet[i])):
            for k in range(len(leksikon_negatif)):
                if tweet[i][j] == leksikon_negatif[k]:
                    kata+= 1
        negatif_tweet.append(kata)
    return negatif_tweet

def Intensifier(tweet):
    intensifier=[]
    for i in range(len(tweet)):
        kata =0
        with open("D:/fita/leksikon/intensifier.txt") as f:
            content = f.readlines()
        leksikon_intensifier = [x.strip() for x in content]
        for j in range(len(tweet[i])):
            for k in range(len(leksikon_intensifier)):
```

```
259                 if                (tweet[i][j]).lower()                ==
260 leksikon_intensifier[k].lower():
261                     kata+= 1
262         intensifier.append(kata)
263     return intensifier
```

***Source code* 4.3 pembobotan *ensemble features***

Penjelasan *Source code* 4.3 tentang pembobotan *ensemble features Twitter specific* sebagai berikut:

Baris 1-44 : proses pembobotan *Twitter specific*

Baris 45-123 : proses pembobotan *textual features*

Baris 125- 226 : proses pemobobotan *POS Tagging* dan *lexicon based features*

Baris 228-263 : proses pembobotan *lexicon based features*

### 4.9.3 Proses inisialisasi *weight input* dan bias

```
1 def InputWeightBias(number_of_hidden_neurons,number_of_input_neurons):
2
3 input_weight=(rdanom.rdan(number_of_hidden_neurons,number_of_input_neurons))*2-
4 1
5     bias_of_hidden_neurons = rdanom.rdan(number_of_hidden_neurons,1)
      return input_weight, bias_of_hidden_neurons
```

***Source code 4.4* inisialisasi weight input dan bias**

Penjelasan *Source Code 4.4* tentang implementasi proses inisialisasi *input weight* dan bias adalah sebagai berikut:

Baris 2-3 : proses inisialisasi matriks *input weight* secara *random* dengan rentang 0 hingga 1 kemudian dikalikan dengan 2 dan setelahnya dikurangi satu.

Baris 4 : proses inisialisasi matriks bias secara *random* dengan rentan 0 hingga 1

### 4.9.4 Proses *training*

```
1  def TrainingElm(data_training, target):
2  print("#########################################################  PROSES
3  TRAINING ELM ############################################################")
4      number_of_hidden_neurons = 5
5      print("Data Training:")
6      print(data_training)
7      print("Kelas Target:")
8      print(target)
9      "mentranspose matrix data_training"
10     P = (data_training).transpose()
11     "mentranspose matrix target dari data training"
12     T = (target).transpose()
13     number_of_data = len(P[0])
14     number_of_input_neurons = len(P)
15     sorted_target = sort(T)
16     label = zeros((1,1))
17     j= 0
18     i=1
19     label[0][0]=sorted_target[0][0]
20     while(i <number_of_data):
```

```
21          if (sorted_target[0][i] != label[0][j]):
22              j= j + 1
23              label = insert(label, [j], [[sorted_target[0][i]]],axis=1)
24          i=i+1
25      number_class=j+1
26      number_of_output_neurons = number_class
27      T    =    TargetTraining(number_of_output_neurons,    number_class,
28  number_of_data, label , T)
29      input_weight,               bias_of_hidden_neurons               =
30  InputWeightBias(number_of_hidden_neurons,number_of_input_neurons)
31      temp_h, bias_matrix = NilaiH(input_weight, bias_of_hidden_neurons,
32  P, number_of_data)
33      print("Input Weight:")
34      print(input_weight)
35      print("Bias Matrix:")
36      print(bias_matrix)
37      print("H:")
38      print(temp_h)
39      print("Proses Aktivasi H menggunakan Fungsi Sigmoid Biner:")
40      output_weight_biner  =  OutputTraining("sigmoid  biner",  temp_h,  T,
41  label, number_of_data)
42      print("Proses Aktivasi H menggunakan Fungsi Sin:")
43      output_weight_sin   =   OutputTraining("sin",   temp_h,   T,   label,
44  number_of_data)
45      print("Proses Aktivasi H menggunakan Fungsi Hardlimit:")
46      output_weight_hardlimit  =  OutputTraining("hardlimit",  temp_h,  T,
47  label, number_of_data)
48      print("Proses Aktivasi H menggunakan Fungsi Triangular:")
49      output_weight_triangular = OutputTraining("triangular", temp_h, T,
50  label, number_of_data)
51      print("Proses Aktivasi H menggunakan Fungsi Radial Basis:")
52      output_weight_radial = OutputTraining("radial", temp_h, T, label,
53  number_of_data)
54      print("Proses Aktivasi H menggunakan Fungsi Linear:")
55      output_weight_linear = OutputTraining("linear", temp_h, T, label,
56  number_of_data)
57      print("Proses Aktivasi H menggunakan Fungsi Sigmoid Bipolar:")
58      output_weight_bipolar = OutputTraining("sigmoid bipolar", temp_h,
59  T, label, number_of_data)
60      return      number_of_output_neurons,      label,      input_weight,
61  bias_of_hidden_neurons,     output_weight_biner,     output_weight_sin,
62  output_weight_hardlimit,                        output_weight_triangular,
63  output_weight_radial ,output_weight_linear ,output_weight_bipolar
```

***Source code 4.5* proses *training***

Penjelasan *Source Code 4.5* tentang implementasi proses *training* adalah sebagai berikut:

Baris 2-8 : proses inisialisasi *hidden neuron*, cetak data data *training, serta kelas target*.

Baris 9-10 : proses transpose matriks data training

Baris 10-11: proses transpose matriks kelas target

Baris 13-19: proses inisialisasi nilai

Baris 20-28: proses membuat matriks target kelas

Baris 29-30: proses *input weight* serta bias

Baris 31-32: proses *output hidden neuron*

Baris 39-57: proses aktivasi fungsi *output hidden neuron*

Baris 58-63: proses *output weight* serta hasil akurasi

### 4.9.5 Proses target *training*

```
1   def       TargetTraining(number_of_output_neurons,       number_class,
2   number_of_data, label, T):
3
4       temp_T = zeros((number_of_output_neurons, number_of_data))
5
6       for i in range(number_of_data):
7           for j in range(number_class):
8               if (label[0][j] == T[0][i]):
9                   break;
10          temp_T[j][i]=1
11      T=temp_T*2-1
12      return T
```

***Source code 4.6* proses target *training***

Penjelasan *Source Code 4.6* tentang implementasi proses target *training* adalah sebagai berikut:

Baris 1-4 : proses inisialisasi matriks *zero* dengan ordo jumlah *output neuron x jumlah data*

Baris 6-12 : proses *update* nilai target kelas sesuai dengan data

### 4.9.6 Proses keluaran hidden neuron

```
1   def NilaiH(input_weight, bias_of_hidden_neurons, P, number_of_data):
2       temp_h= matmul(input_weight,P)
3       ind =ones((1,number_of_data))
4       bias_matrix = matmul(bias_of_hidden_neurons, ind)
5       temp_h = temp_h + bias_matrix
6       return temp_h, bias_matrix
```

***Source code 4.7* proses keluaran *hidden neuron***

Penjelasan *Source Code 4.7* tentang implementasi proses target *training* adalah sebagai berikut:

Baris 2 : proses perkalian matriks antara *input weight* dengan data

Baris 3 : proses inisialisasi matriks *ones* dengan ordo 1 x jumlah data

Baris 4 : proses perkalian matriks antara matriks bias *hidden neuron* dengan matriks *ones*

Baris 5-6 : proses perhitungan matriks keluaran *hidden neuron*

### 4.9.7 Proses fungsi aktivasi

```
1   def FungsiAktivasi(fungsi_aktivasi, temp_h):
2       H = zeros((len(temp_h),len(temp_h[0])))
3       for i in range(len(temp_h)):
4           for j in range(len(temp_h[0])):
5               if(fungsi_aktivasi.lower() == "sigmoid biner"):
6                   H[i][j] = 1/(1+ math.exp(-temp_h[i][j]))
7               elif(fungsi_aktivasi.lower() == "sin"):
8                   H[i][j] = math.sin(temp_h[i][j])
9               elif(fungsi_aktivasi.lower() == "hardlimit"):
10                  if(temp_h[i][j]>=0):
11                      H[i][j] = 1
12                  else:
```

```
13                          H[i][j]= 0
14                elif(fungsi_aktivasi.lower() == "triangular"):
15                     if(temp_h[i][j]<=1):
16                         if(temp_h[i][j]>=-1):
17                             H[i][j] = 1-abs(temp_h[i][j])
18                     else:
19                         H[i][j]= 0
20                elif(fungsi_aktivasi.lower() == "radial"):
21                     H[i][j] = math.exp(-(math.pow(temp_h[i][j],2)))
22                elif(fungsi_aktivasi.lower() == "linear"):
23                     H[i][j] = temp_h[i][j]
24                elif(fungsi_aktivasi.lower() == "sigmoid bipolar"):
25                     H[i][j]  =  (1-math.exp(-temp_h[i][j]))/(1+math.exp(-
26   temp_h[i][j]))
27       return H
```

***Source code 4.8* proses fungsi aktivasi**

Penjelasan *Source Code 4.8* tentang implementasi proses aktivasi adalah sebagai berikut:

Baris 2 : proses inisialisasi awal matrik *zeros*

Baris 3-27 : proses perhitungan keluaran *hidden neuron* teraktivasi

### 4.9.8 Proses bobot *output training*

```
1    def OutputTraining(fungsi_aktivasi, temp_h, T, label, number_of_data):
2        H = FungsiAktivasi(fungsi_aktivasi, temp_h)
3        print("H setelah aktivasi:")
4        print(H)
5        output_weight = matmul(linalg.pinv(H.transpose()), T.transpose())
6        print("Output Weight:")
7        print(output_weight)
8        Y = (matmul(H.transpose(),output_weight)).transpose()
9        print("Output ELM:")
10       print(Y)
11       kelas_hasil_sistem,  hasil_akurasi  =  Akurasi(T,  Y,  label,
12   number_of_data)
13       print("Kelas Hasil Klasifikasi:")
14       print(kelas_hasil_sistem)
15       print("Hasil Akurasi sistem:")
16       print(hasil_akurasi)
17       return output_weight
```

***Source code 4.9* proses bobot *output***

Penjelasan *Source Code 4.9* tentang implementasi proses bobot *output* adalah sebagai berikut:

Baris 2-4 : proses keluaran hidden neuron teraktivasi

Baris 5-7 : proses perhitungan *output weight*

Baris 8-10 : proses perkalian matrik untuk mendapatkan nilai *output* metode *ELM*

Baris 11-17: proses perhitungan akurasi metode pada proses *training*

### 4.9.9 Proses akurasi

```
1    def Akurasi(T, Y, label, number_of_data):
2        label_expected=[]
3        label_actual=[]
4        output=[]
```

```
5       miss_classification=0
6       t_max = T.max(0) #mencari nilai tertinggi tiap kolom "max(0)"
7       for i in range(len(T[0])):
8           for j in range(len(T)):
9               if(t_max[i] == T[j][i]):
10                  label_expected.append(j)
11                  break
12      y_max = Y.max(0)
13      for i in range(len(Y[0])):
14          for j in range(len(Y)):
15              if(y_max[i] == Y[j][i]):
16                  label_actual.append(j)
17                  output.append(label[0][j])
18                  break
19          if(label_actual[i] != label_expected[i]):
20              miss_classification = miss_classification + 1
21
22      training_akurasi = 1 - (miss_classification/number_of_data)
23      return label_actual, training_akurasi
```

***Source code 4.10* proses akurasi**

Penjelasan *Source Code 4.10* tentang implementasi proses akurasi adalah sebagai berikut:

Baris 2-4 : proses inisialisasi *matrik*

Baris 6 : proses pencarian nilai tertinggi pada target

Baris 7-11 : proses *update matrik untuk kelas target*

Baris 12 : proses pencarian nilai tertinggi pada setiap atribut

Baris 13-20: proses pengecekan ketepatan hasil klasifikasi dengan kelas target

Baris 21-23: proses perhitungan akurasi

### 4.9.10 Proses *testing*

```
1   def TestingElm(data_testing, target, number_of_output_neurons, label,
2   input_weight,       bias_of_hidden_neurons,       output_weight_biner,
3   output_weight_sin, output_weight_hardlimit, output_weight_triangular,
4   output_weight_radial ,output_weight_linear ,output_weight_bipolar):
5
6   print("############################################################
7   PROSES                              TESTING                              ELM
8   ############################################################")
9       print("Data Testing:")
10      print(data_testing)
11      print("Kelas Target:")
12      print(target)
13
14      "mentranspose matrix data_training"
15      P = (data_testing).transpose()
16      "mentranspose matrix target dari data training"
17      T = (target).transpose()
18
19      number_of_data = len(P[0])
20      print(number_of_data)
21
22      T = TargetTesting(number_of_output_neurons, number_of_data, label,
23  T)
24      temp_h, bias_matrix = NilaiH(input_weight, bias_of_hidden_neurons,
25  P, number_of_data)
26
```

```
27      print("H:")
28      print(temp_h)
29
30      print("Proses Aktivasi H menggunakan Fungsi Sigmoid Biner:")
31      OutputTesting("sigmoid biner", temp_h, T, label, number_of_data,
32  output_weight_biner)
33
34      print("Proses Aktivasi H menggunakan Fungsi Sin:")
35      OutputTesting("sin",    temp_h,    T,    label,    number_of_data,
36  output_weight_sin)
37
38      print("Proses Aktivasi H menggunakan Fungsi Hardlimit:")
39      OutputTesting("hardlimit",   temp_h,   T,   label,   number_of_data,
40  output_weight_hardlimit)
41
42      print("Proses Aktivasi H menggunakan Fungsi Triangular:")
43      OutputTesting("triangular",   temp_h,   T,   label,   number_of_data,
44  output_weight_triangular)
45
46      print("Proses Aktivasi H menggunakan Fungsi Radial Basis:")
47      OutputTesting("radial",   temp_h,   T,   label,   number_of_data,
48  output_weight_radial)
49
50      print("Proses Aktivasi H menggunakan Fungsi Linear:")
51      OutputTesting("linear",   temp_h,   T,   label,   number_of_data,
52  output_weight_linear)
53
54      print("Proses Aktivasi H menggunakan Fungsi Sigmoid Bipolar:")
55      OutputTesting("sigmoid bipolar", temp_h, T, label, number_of_data,
56  output_weight_bipolar)
```

***Source code 4.11* proses *testing***

Penjelasan *Source Code 4.11* tentang implementasi proses *testing* adalah sebagai berikut:

Baris 9-12 : cetak data data *testing, serta kelas target*.

Baris 14-15: proses transpose matriks data testing

Baris 16-17: proses transpose matriks kelas target

Baris 19-20: proses inisialisasi nilai

Baris 22-23: proses membuat matriks target kelas

Baris 24-29: proses *output hidden neuron*

Baris 30-57: proses aktivasi fungsi *output hidden neuron*

Baris 30-56: proses *output weight* serta hasil akurasi

### 4.9.11 Proses bobot *output testing*

```
1   def OutputTesting(fungsi_aktivasi, temp_h, T, label, number_of_data,
2   output_weight):
3       H = FungsiAktivasi(fungsi_aktivasi, temp_h)
4       print("H setelah aktivasi:")
5       Y = (matmul(H.transpose(),output_weight)).transpose()
6       print("Output ELM:")
7       print(Y)
8       kelas_hasil_sistem,kelas_target,  hasil_akurasi  =  Akurasi(T,  Y,
9   label, number_of_data)
10      print("Kelas Hasil Klasifikasi:")
11      print(kelas_hasil_sistem)
```

```
12      print("Kelas Hasil Target:")
13      print(kelas_target)
14      print("Hasil Akurasi sistem:")
15      print(hasil_akurasi)
```

***Source code 4.12* proses bobot *output***

Penjelasan *Source Code* 4.12 tentang implementasi proses bobot *output* adalah sebagai berikut:

Baris 2-4 : proses keluaran hidden neuron teraktivasi

Baris 5-7 : proses perhitungan *output weight*

Baris 8-15 : proses perhitungan akurasi metode pada proses *testing*

## 4.10 Implementasi antarmuka

### 4.10.1 Antarmuka *tweet training*

**Gambar 4.29 Antarmuka *tweet training***

Antramuka *tweet training* menampilkan isi dari dokumen *training* yang berisi kumpulan *tweet* yang digunakan sebagai dasar dari proses *training* yang kemudian akan dilakukan proses pembobotan dan klasifikasi menggunakan metode *Extreme Learning Machine.*

### 4.10.2 Antarmuka kelas data *training*


**Gambar 4.30 Antarmuka kelas data *training***

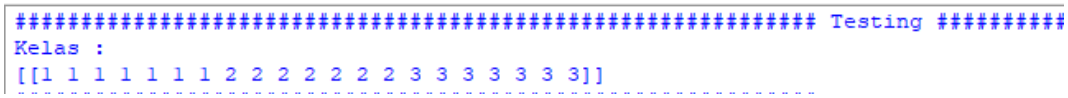
Antarmuka kelas data training pada Gambar 4.30 Antarmuka kelas data *training* Gambar 4.30 menampilkan daftar kelas dari data *training,* di mana angka 1 merupakan kelas positif, angka 2 merupakan kelas negatif, dan angka 3 merupakan kelas netral.

### 4.10.3 Antarmuka *tweet testing*

**Gambar 4.31 Antarmuka *tweet testing***

Antramuka *tweet testing* pada Gambar 4.31 menampilkan isi dari dokumen *testing* yang berisi kumpulan *tweet* yang digunakan sebagai dasar dari proses *testing* yang digunakan sebagai data baru untuk mengecek sistem yang sudah dilakukan *training.*

### 4.10.4 Antarmuka kelas data *testing*




**Gambar 4.32 Antarmuka kelas data *testing***

Antarmuka kelas data *training* pada Gambar 4.32 menampilkan daftar kelas dari data *testing,* dimana angka 1 merupakan kelas positif, angka 2 merupakan kelas negatif, dan angka 3 merupakan kelas netral.

### 4.10.5 Ground truth dan akurasi

**Gambar 4.33 Antarmuka *ground truth* dan akurasi**

Pada Gambar 4.33 merupakan antarmuka *ground truth* dan akurasi hasil dari sistem klasifikas menggunakan metode *ELM.*

# BAB 5 HASIL DAN PEMBAHASAN

Pada bab ini berisi tentang proses Pengujian terhadap sistem yang dibuat menggunakan metode E*xtreme Learning Machine (ELM).* Proses Pengujian pada bab ini berdasarkan perancangan yang telah dibuat pada bab sebelumnya dengan perbandingan data *training* dan data *testing* sebesar 70:30. Jumlah data yang digunakan sebagai data *training* sebanyak 51 *tweet* dan data *testing* sebanyak 21 *tweet,* dimana jumlah tiap kelas seimbang. Proses Pengujian tersebut yaitu: Pengujian pengaruh jumlah *hidden neuron* terhadap akurasi*,* Pengujian perbandingan fungsi aktivasi, Pengujian pengaruh *ensemble features* terhadap akurasi, dan *confusion matrix* dari parameter yang paling optimal.

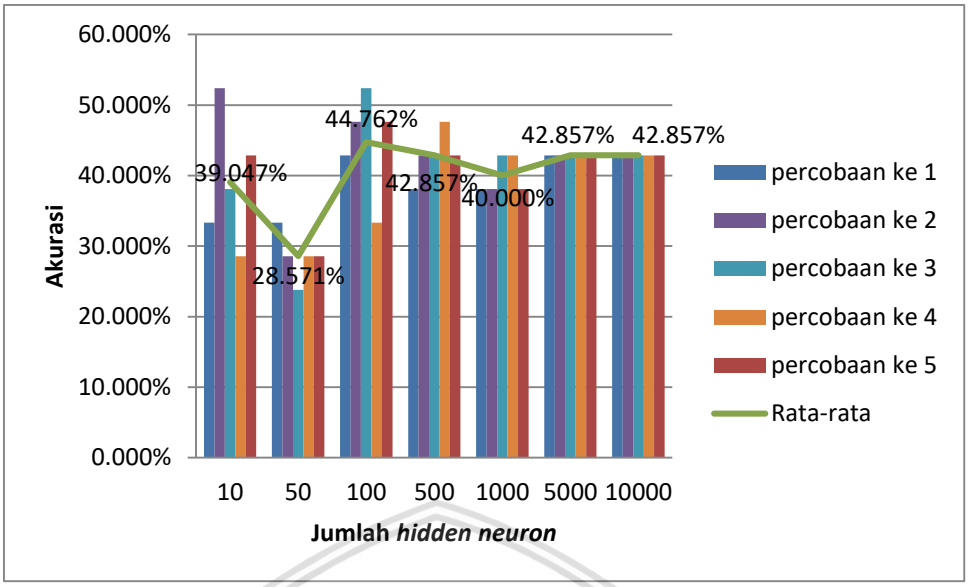
## *5.1* Pengujian pengaruh jumlah *hidden neuron* terhadap akurasi

Pengujian ini dilakukan untuk mengetahui apakah jumlah *hidden neuron* memiliki pengaruh terhadap hasil akurasi sistem. Pengujian pengaruh jumlah *hidden layer* dilakukan dengan lima kali percobaan dengan menggunakan jumlah *hidden neuron* secara acak yaitu: 10, 50, 100, 500, 1000, 5000, dan 10000. Fungsi aktivasi yang digunakan adalah sigmoid bipolar, di mana data yang digunakan memiliki *range* antara 0 sampai 1. Nilai hasil akurasi Pengujian pengaruh jumlah *hidden neuron* ditunjukkan pada Tabel 5.1

**Tabel 5.1 Pengujian pengaruh jumlah *hidden neuron* terhadap akurasi**

| | Jumlah *hidden neuron* | | | | | | |
|---|---|---|---|---|---|---|---|
| **Percobaan ke** | **10** | **50** | **100** | **500** | **1000** | **5000** | **10000** |
| 1 | 33,333% | 33,333% | 42,857% | 38,095% | 38,095% | 42,857% | 42,857% |
| 2 | 52,381% | 28,571% | 47,619% | 42,857% | 38,095% | 42,857% | 42,857% |
| 3 | 38,095% | 23,810% | 52,381% | 42,857% | 42,857% | 42,857% | 42,857% |
| 4 | 28,571% | 28,571% | 33,333% | 47,619% | 42,857% | 42,857% | 42,857% |
| 5 | 42,857% | 28,571% | 47,619% | 42,857% | 38,095% | 42,857% | 42,857% |
| **Rata-rata** | 39,047% | 28,571% | 44,762% | 42,857% | 40,000% | 42,857% | 42,857% |
| **Simpangan baku** | 9,16% | 3,37% | 7,22% | 3,37% | 2,61% | 0,00% | 0,00% |

Grafik hasil akurasi pengaruh jumlah *hidden neuron* terhadap akurasi ditunjukkan pada Gambar 5.1.




**Gambar 5.1 Grafik hasil akurasi Pengujian pengaruh jumlah *hidden neuron***

Dari Tabel 5.1 dan Gambar 5.1 didapatkan hasil bahwa nilai akurasi terendah didapatkan saat menggunakan jumlah hidden sebanyak 50 dengan nilai akurasi sebesar 23,810%, sedangkan nilai akurasi tertinggi didapatkan saat menggunakan jumlah *hidden neuron* sebanyak 10 dan 100 dengan nilai akurasi sebesar 52,381%, namun jumlah *hidden neuron* tersebut bukanlah jumlah *hidden neuron* terbaik. Hal tersebut dikarenakan pada lima kali percobaan dengan menggunakan *hidden neuron* sebanyak 10 didapatkan hasil akurasi yang tidak stabil dengan nilai simpangan baku sebesar 0,839% dan pada jumlah *hidden neuron* sebanyak 100 juga didapatkan hasil akurasi yang tidak stabil dengan nilai simpangan baku sebesar 0,522%. Hal tersebut dapat membuat spekulasi bahwa hasil akurasi tertinggi tersebut hanya sebuah kebetulan yang di mana saat dilakukan percobaan kembali akan sangat memungkinkan mendapatkan hasil akurasi yang sangat rendah. Hasil akurasi yang stabil dengan nilai simpangan baku 0% di dapatkan pada jumlah *hidden neuron* sebanyak 5000 dan 10000.

Pada jumlah *hidden neuron* sebanyak 5000 dan 10000 mendapatkan nilai rata-rata akurasi yang sama dengan nilai sebesar 42,857%. Nilai rata-rata akurasi tersebut merupakan rata-rata dari lima kali percobaan yang selalu menghasilkan nilai akurasi sebesar 42,857% sehingga didapatkan nilai simpangan baku sebesar 0%. Pada *hidden neuron* sebanyak 5000 dan 10000 mendapatkan nilai rata-rata akurasi yang sama, namun secara teoritis memerlukan waktu yang berbeda, di mana semakin banyak jumlah *hidden neuron* maka struktur jaringannya akan semakin lebih kompleks sehingga secara teoritis akan memerlukan waktu yang lebih lama. Dengan nilai rata-rata akurasi yang sama namun memerlukan waktu yang berbeda maka pada Pengujian pengaruh jumlah *hidden neuron* terbaik di dapatkan dengan menggunakan jumlah *hidden neuron* sebanyak *5000.*

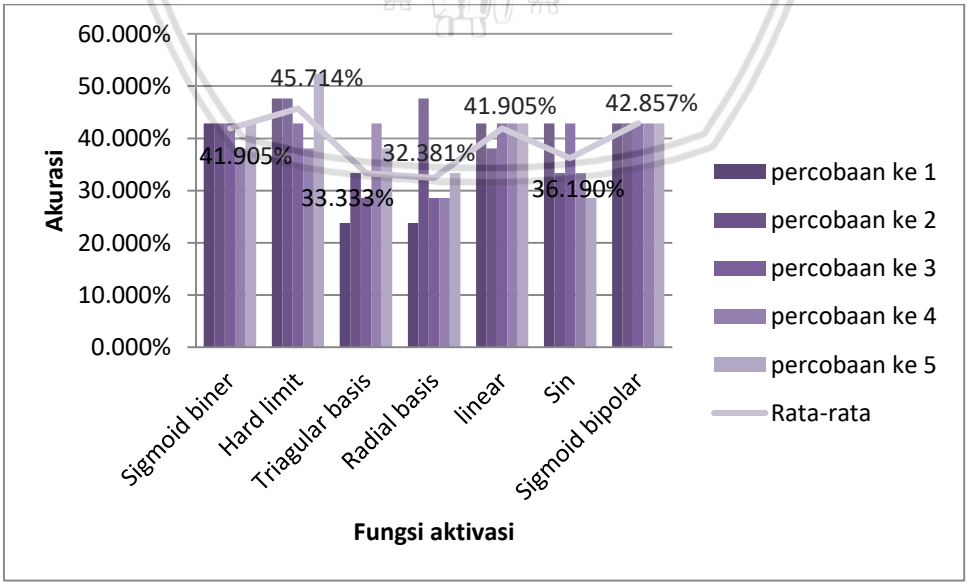
## 5.2 Pengujian perbandingan fungsi aktivasi

Pengujian perbandingan aktivasi di lakukan untuk mengetahui pengaruh atau tidaknya suatu fungsi tertentu terhadap hasil akurasi. Terdapat 7 macam fungsi aktivasi yang digunakan pada Pengujian penelitian ini, yatu *sigmoid biner, hard limit, triangular basis, radial basis, linear, sin, dan sigmoid bipolar.* Pengujian ini akan dilakukan sebanyak lima kali pada setiap fungsi aktivasi dengan perbandingan jumlah data *training* 70:30 data *training* dan *testing* serta jumlah *hidden neuron* sebesar 5000*.* Nilai hasil akurasi pada pengujian perbandingan fungsi aktivasi ditunjukkan pada Tabel 5.2

**Tabel 5.2 Pengujian perbandingan fungsi aktivasi**

| Nilai akurasi sistem | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Percobaan ke** | **Sigmoid biner** | **Hard limit** | **Triagular basis** | **Radial basis** | **Linear** | **Sin** | **Sigmoid bipolar** |
| 1 | 42,857% | 47,619% | 23,810% | 23,810% | 42,857% | 42,857% | 42,857% |
| 2 | 42,857% | 47,619% | 33,333% | 47,619% | 38,095% | 33,333% | 42,857% |
| 3 | 42,857% | 42,857% | 28,571% | 28,571% | 42,857% | 42,857% | 42,857% |
| 4 | 38,095% | 38,095% | 42,857% | 28,571% | 42,857% | 33,333% | 42,857% |
| 5 | 42,857% | 52,381% | 38,095% | 33,333% | 42,857% | 28,571% | 42,857% |
| **Rata-rata** | 41,905% | 45,714% | 33,333% | 32,381% | 41,905% | 36,190% | 42,857% |
| **Simpangan baku** | 2,13% | 5,43% | 7,53% | 9,16% | 2,13% | 6,39% | 0,00% |

Grafik hasil akurasi pengujian perbandingan fungsi aktivasi ditunjukkan pada Gambar 5.2.




**Gambar 5.2 Grafik hasil akurasi pada pengujian perbandingan fungsi aktivasi**

Dari Tabel 5.2 dan Gambar 5.2 didapatkan nilai akurasi terendah saat menggunakan fungsi aktivasi *triangular basis* dan *radial basis* dengan nilai akurasi sebesar 23,810%, sedangkan nilai akurasi tertinggi didapatkan saat menggunakan fungsi aktivasi *hard limit* dengan nilai akurasi sebesar 52,391% namun hasil akurasi dari lima kali percobaan didapatkan bahwa saat menggunakan fungsi aktivasi *hard limit* nilai akurasinya tidak stabil. Hasil akurasi yang cenderung tidak stabil juga didapatkan saat menggunakan fungsi aktivasi *sigmoid biner, triangular basis, radial basis, linear, dan sin* tergantung nilai dari *weight input* yang didapatkan secara acak. Hal tersebut dapat membuat spekulasi bahwa hasil akurasi tertinggi yang didapatkan saat menggunakan fungsi aktivas *hardlimit* hanya sebuah kebetulan, di mana saat dilakukan percobaan kembali akan sangat memungkinkan mendapatkan hasil akurasi yang sangat rendah tergantung pada nilai *input weight* dan *bias* yang didapatkan secara acak. Nilai akurasi yang stabil didapatkan saat menggunakan fungsi aktivasi *sigmoid bipolar* dengan nilai akurasi sebesar 42,857%.

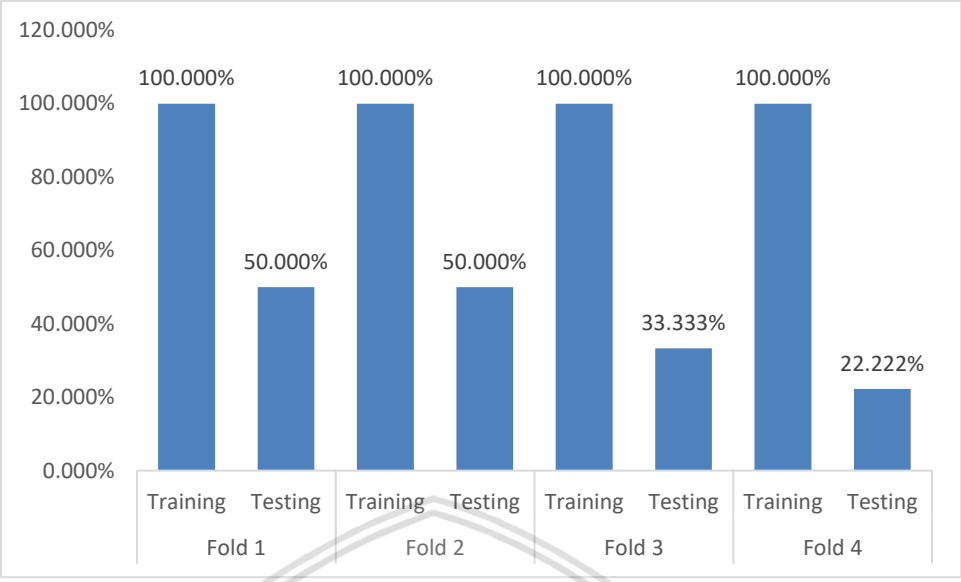
Pada Pengujian perbandingan fungsi aktivasi menggunakan fungsi *sigmoid bipolar* didapatkan nilai rata simpangan baku sebesar 0% yang berarti sebaran datanya 0. Jika sebarannya bernilai 0 maka nilai semua datanya sama, di mana semakin besar nilai sebarannya maka datanya semakin bervariasi sehingga dapat dikatakan bahwa hasil akurasi pengujian menggunakan fungsi aktivasi *sigmoid bipolar* bersifat stabil dengan nilai akurasi pada setiap percobaan sebesar 42,857%. Pada Pengujian perbandingan fungsi aktivasi didapatkan hasil terbaik pada saat menggunakan fungsi aktivasi *sigmoid bipolar* karena dengan menggunakan *sigmoid bipolar* didapatkan hasil akurasi yang stabil dibandingkan dengan menggunakan *sigmoid biner, sin, hardlimit, triangular basis, radial basis,* dan *linear.*

## *5.3 Pengujian K-Fold Crossvalidation*

Pengujian *K-Fold Crossvalidation* di lakukan dengan jumlah *K* sebanyak 4, di mana setiap *K* memiliki 18 data dengan jumlah setiap kelas yang seimbang sebanyak 6 data tiap kelas. Jumlah *hidden neuron* yang digunakan pada Pengujian ini sebanyak 5000 serta menggunakan fungsi aktivasi *sigmoid bipolar*. Nilai hasil akurasi pengujian *K-Fold* ditunjukkan pada Tabel 5.3.

**Tabel 5.3 Pengujian *K-Fold Crossvalidation***

| Fold 1 | | Fold 2 | | Fold 3 | | Fold 4 | | Rata-rata | |
|---|---|---|---|---|---|---|---|---|---|
| Training | Testing | Training | Testing | Training | Testing | Training | Testing | Training | Testing |
| 100% | 50% | 100% | 50% | 100% | 33,333% | 100% | 22,222% | 100% | 38,889% |




**Gambar 5.3 Pengujian *4-fold crossvalidation***

Dari Tabel 5.3 dan Gambar 5.3 didapatkan hasil akurasi pada data *training* sebesar 100% untuk 4 *Fold*, sedangkan untuk data *testing* hasil akurasi terendah pada saat *Fold* ke 3 dengan nilai akurasi sebesar 33,333%. Nilai akurasi tertinggi data *testing* didapatkan saat *Fold* ke 1 dan *Fold* ke 2 dengan nilai akurasi sebesar 50%. Dari 4 *Fold* data *testing* didapatkan hasil rata-rata akurasi sebesar 38,889%. Rendahnya nilai akurasi yang didapatkan pada saat *testing* dikarenakan terjadinya *overfitting,* di mana fitur yang digunakan tidak sesuai pada data *testing* namun sesuai dengan data *training* sehingga rata-rata nilai akurasi pada data *training* tinggi yaitu sebesar 100%, sedangkan pada data *testing* rata-rata nilai akurasinya hanya 38,889%. Ketidaksesuaian fitur yang digunakan pada data *testing* dilihat dari munculnya *sparse data* pada hasil *testing.*

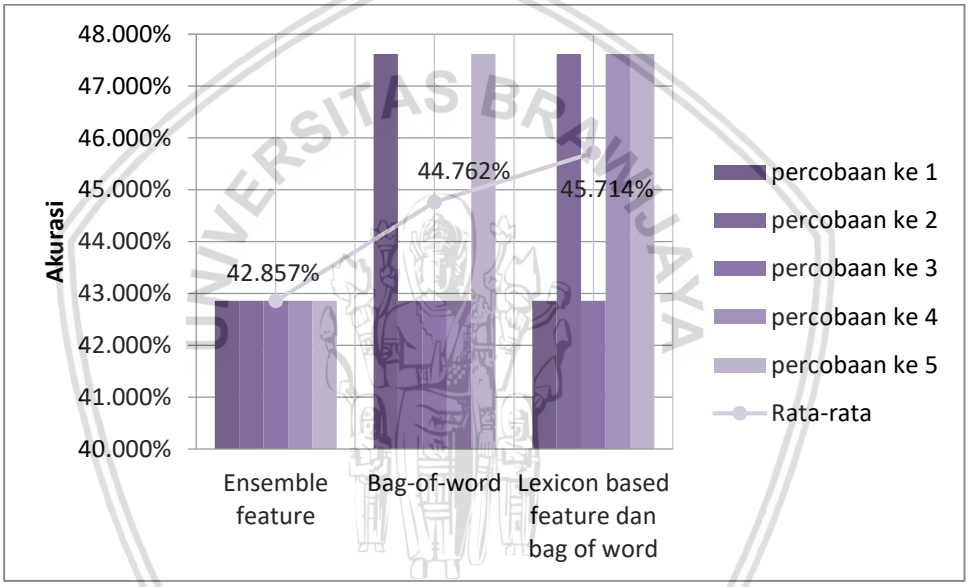
## 5.4 Peng*ujian* pengaruh *ensemble features* terhadap akurasi

Pengujian pengaruh *ensemble feature* digunakan untuk mengetahui apakah analisis sentimen menggunakan *bag-of-word* dan *ensemble features* memiliki pengaruh yang lebih besar dengan nilai akurasi yang lebih tinggi dibandingkan dengan menggunakan *bag-of-word* dan *lexicon based features* yang pada penelitan Rofiqoh (2017) memiliki pengaruh terhadap analisis sentimen serta dibandingkan dengan hanya menggunakan *bag-of-word* saja. Pada pengujian ini akan dilakukan lima kali percobaan dengan menggunakan perbandingan data *training* dan data *testing* sebesar 70:30 serta menggunakan fungsi aktivasi *sigmoid bipolar.* Jumlah *hidden neuron* yang digunakan pada pengujian ini sebanyak 5000. Nilai hasil akurasi pengujian pengaruh *ensemble features* terhadap akurasi ditunjukkan pada Tabel 5.4.

**Tabel 5.4 Pengujian pengaruh *ensemble feature***

| Percobaan ke | *Ensemble features* | *Bag-of-word* | *Lexicon based feature* dan bag of word |
|---|---|---|---|
| 1 | 42,857% | 47,619% | 42,857% |
| 2 | 42,857% | 42,857% | 47,619% |
| 3 | 42,857% | 42,857% | 42,857% |
| 4 | 42,857% | 42,857% | 47,619% |
| 5 | 42,857% | 47,619% | 47,619% |
| Rata-rata | 42,857% | 44,762% | 45,714% |

Grafik nilai hasil akurasi pengujian *ensemble features* ditunjukkan pada Gambar 5.4.




**Gambar 5.4 Grafik Pengujian pengaruh ensemble features**

Dari Tabel 5.4 dan Gambar 5.4 Grafik Pengujian pengaruh ensemble features didapatkan hasil bahwa nilai rata-rata akurasi terendah diperoleh dengan menggunakan *ensemble feature.* Nilai akurasi tertinggi diperoleh menggunakan *lexicon based features* dan *bag-of-words* serta hanya menggunakan *bag-of-word saja* dengan nilai akurasi sebesar 47,619%. Walaupun hasil akurasi tertinggi diperoleh menggunakan *lexicon based feature* dan *bag-of-word* serta hanya menggunakan *bag-of-word* saja, namun hasil tersebut cenderung tidak stabil, karena pada percobaan yang berbeda hasil akurasi yang didapatkan berbeda tergantung pada *input weight* serta *bias* yang didapatkan secara random. Dengan menggunakan e*nsemble feature* didapatkan rata-rata hasil akurasi terendah namun hasil tersebut stabil pada lima kali percobaan dan memiiki simpangan baku sebesar 0. Sehingga hasil terbaik didapatkan menggunakan *ensemble feature.* Analisis sentimen menggunakan *ensemble feature cocok* dikarenakan Twitter memiliki beberapa fitur yang berpengaruh terhadap analisis sentimen, namun rendahnya nilai akurasi yang dihasilkan dikarenakan pada penelitian ini

beberapa fitur pada *ensemble features* tidak relevan dengan data sehingga muncul banyak *sparse data*.

## 5.5 *Confusion matrix* dari parameter yang paling optimal

Pengujian *confusion matrix* bertujuan untuk mengukur kemampuan teknik klasifikasi sentimen. Pada Pengujian ini menggunakan perbandingan data *training* dan data *testing* sebesar 70:30, serta parameter yang paling optimal hasil dari Pengujian sebelumnya yaitu menggunakan fungsi aktivasi *sigmoid bipolar* dengan jumlah *hidden neuron* sebesar 5000 dan menggunakan *Ensemble features* serta nilai akurasi sebesar 42,857%. *Confusion matrix* pada kelas positif ditunjukkan pada Tabel 5.5 *Confusion matrix* kelas positif.

**Tabel 5.5 *Confusion matrix* kelas positif**

| | | Nilai Prediksi | |
|---|---|---|---|
| | | *TRUE* | *FALSE* |
| Nilai Aktual | *TRUE* | 4 | 3 |
| | *FALSE* | 5 | 9 |

$$Precision = \frac{4}{4+5} x\ 100\% = \frac{4}{9} x\ 100\% = 44{,}444\%$$

$$Recall = \frac{4}{4+3} x\ 100\% = \frac{4}{7} x\ 100\% = 57{,}143\%$$

$$F - measure = \frac{2 \times 44{,}444\% \times 57{,}143\%}{44{,}444\% + 57{,}143\%} = 50{,}00\%$$

*Confusion matrix* pada kelas negatif ditunjukkan pada Tabel 5.6.

**Tabel 5.6 *Confusion matrix* kelas negatif**

| | | Nilai Prediksi | |
|---|---|---|---|
| | | *TRUE* | *FALSE* |
| Nilai Aktual | *TRUE* | 3 | 4 |
| | *FALSE* | 2 | 12 |

$$Precision = \frac{3}{3+2} x\ 100\% = 60\%$$

$$Recall = \frac{3}{3+4} x\ 100\% = \frac{3}{7} x\ 100\% = 42{,}857\%$$

$$F - measure = \frac{2 \times 60\% \times 42{,}857\%}{60\% + 142{,}857\%} = 50{,}000\%$$

*Confusion matrix* pada kelas netral ditunjukkan pada Tabel 5.7.

**Tabel 5.7 *Confusion matrix* kelas netral**

| | | Nilai Prediksi | |
|---|---|---|---|
| | | *TRUE* | *FALSE* |
| Nilai Aktual | *TRUE* | 2 | 5 |
| | *FALSE* | 5 | 9 |

$$Precision = \frac{2}{2+5} x\ 100\% = \frac{2}{7}\ x\ 100\% = 28{,}571\%$$

$$Recall = \frac{2}{2+5} x\ 100\% = \frac{2}{7}\ x\ 100\%\ = 28{,}571\%$$

$$F - measure = \frac{2 \times 28{,}571\% \times 28{,}571\%}{28{,}571\% + 28{,}571\%} = 28{,}571\%$$

Dari ketiga tabel di atas terlihat bahwa kemampuan sistem dengan menggunakan metode *ELM* tidak selalu baik untuk setiap kelasnya. Pada kelas negatif sistem dapat menghasilkan nilai tertingggi pada *precision* dengan nilai 60% namun nilai *recallnya* rendah yaitu 42,857%. Rendahnya nilai *precision* dan *recall* berarti tingkat ketepatan analisis sentimen ini rendah dan terjadi ketidak kesesuaian informasi yang didapat dari percobaan yang telah dilakukan. Hal tersebut kemungkinan disebabkan karena jumlah data yang terlalu sedikit jika dilakukan *undersampling* data sehingga menyebabkan terjadinya *overfitting,* dan dapat juga disebabkan oleh data kuisoner yang digunakan kurang akurat karena hanya dilakukan oleh satu pakar.

# BAB 6 PENUTUP

Pada bab ini memuat kesimpulan terhadap sistem yang telah dibuat dan usulan saran untuk penelitian yang pengembangan dari penelitian ini.

## 6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan Pengujian yang telah dibuat sebelumnya, didapatkan kesimpulan sebagai berikut:

*1.* Berdasarkan hasil Pengujian analisis sentimen Twitter Samsung Indonesia dengan perbandingan jumlah data *training* : data *testing* sebesar 70:30 serta menggunakan fungsi aktivasi *sigmoid bipolar* dengan jumlah *hidden neuron* sebanyak 5000 serta menggunakan metode *Extreme Learning Machine* dan *ensemble based features* mendapatkan tingkat akurasi sebesar 42,857%. Rendahnya nilai akurasi yang didapatkan diakibatkan terjadinya *sparse matrix,* data *testing* jauh berbeda dengan data *training.*

2. *Ensemble features* memiliki pengaruh terhadap hasil klasifikasi analisis sentimen Twitter Samsung dengan menghasilkan nilai akurasi yang stabil sebesar 42,857% walaupun jika dibandingkan dengan hanya menggunakan *bag-of-word* saja serta dengan menggunakan *lexicon based features* dan *bag-of-word* merupakan nilai rata-rata akurasi terendah. Analisis sentimen menggunakan *ensemble feature* cocok dikarenakan Twitter memiliki beberapa fitur yang berpengaruh terhadap analisis sentimen, namun pada penelitian ini mendapatkan nilai akurasi yang rendah dikarenakan pada penelitian ini beberapa fitur pada *ensemble features* tidak relevan dengan data sehingga muncul banyak data sparse.

## 6.2 Saran

Saran yang dapat diberikan pada penelitian berikutnya yang sejenis dengan analisis sentimen Twitter menggunakan *ensemble features* dan *Extreme Learning Machine,* sebagai berikut:

1. Penelitian berikutnya agar menambahkan data dengan jumlah yang lebih banyak dan lebih seimbang pada setiap kelasnya untuk mengatasi terjadinya *overfitting*.
2. Untuk mengatasi terjadinya *overfitting data,* penelitian selanjutnya dapat mencari metode yang dapat digunakan dalam mengatasi *overfitting* data yang dapat diterapkan pada metode *Extreme Learning Machine.*
3. Penelitian berikutnya dapat menggunakan seleksi fitur untuk mengurangi *sparse matrix* dan jumlah fitur yang tidak signifikan sehingga dapat mengatasi terjadinya *overfitting* agar mendapatkan nilai akurasi yang lebih baik.

# DAFTAR PUSTAKA


Ariestyani, M.C., 2017. Klasifikasi Penyimpangan Tumbuh Kembang Anak Menggunakan Metode Extreme Learning Machine ( ELM ). Universitas Brawijaya.

Azizah, N., Ivan, M. and Budi, I., 2015. Twitter Sentiment to Analyze Net Brand Reputation of Mobile Phone Providers. Procedia - Procedia Computer Science, [online] 72, pp.519–526. Available at: <http://dx.doi.org/10.1016/j.procs.2015.12.159>.

Davidov, D., Tsur, O. and Rappoport, A., 2010. Semi-Supervised Recognition of Sarcastic Sentences in Twitter and Amazon. Fourteenth Conference on Computational Natural Language Learning, [online] (July), pp.107–116. Available at: <http://eprints.pascal-network.org/archive/00007069/>.

Faradhillah, N.Y.A., Kusumawardani, R.P. and Hafidz, I., 2016. Eksperimen Sistem Klasifikasi Analisa Sentimen Twitter pada Akun Resmi Pemerintah Kota Surabaya Berbasis Pembelajaran Mesin (Experiments on Sentiment Classification System for Tweets of the Official Account of the City Government of Surabaya based on Mach. Prosiding Seminar Nasional Sistem Informasi Indonesia 2016, [online] pp.15–24. Available at: <http://is.its.ac.id/pubs/oajis/index.php/home/detail/1645/Eksperimen-Sistem-Klasifikasi-Analisa-Sentimen-Twitter-Pada-Akun-Resmi-Pemerintah-Kota-Surabaya-Berbasis-Pembelajaran-Mesin>.

Govindasamy, P., 2017. Sentiment Analysis: A Business-Critical Need To Improve Customer Experience. [online] Forbes. Available at: <https://www.forbes.com/sites/forbestechcouncil/2017/08/03/sentiment-analysis-a-business-critical-need-to-improve-customer-experience/#34ef6778352b> [Accessed 10 Apr. 2018].

Huang, G., Member, S., Zhou, H., Ding, X. and Zhang, R., 2012. Extreme Learning Machine for Regression and Multiclass Classification. 42(2), pp.513–529.

Junaedi, H., Budianto, H. and Maryati, I., 2011. Data transformation pada data mining. 7, pp.93–99.

Liu, B., 2012. Sentiment Analysis and Opinion Mining. (May).

Manning, C & Raghavan, P., 2009. An Introduction to Information Retrieval. (c).

Medhat, W., Hassan, A. and Korashy, H., 2014. Sentiment analysis algorithms and applications : A survey. Ain Shams Engineering Journal, [online] 5(4), pp.1093–1113. Available at: <http://dx.doi.org/10.1016/j.asej.2014.04.011>.

Rashel, F., Luthfi, A., Dinakaramani, A. and Manurung, R., 2014. Building an Indonesian rule-based part-of-speech tagger. Proceedings of the International Conference on Asian Language Processing 2014, IALP 2014, pp.70–73.

Rofiqoh, U., 2017. Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexicon Based Features. Universitas Brawijaya.

Roul, R.K., Nanda, A., Patel, V. and Sahay, S.K., 2015. Extreme Learning Machines in the Field of Text Classification. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS International Conference on, pp.1–7.

Sharma, A. and Dey, S., 2012. A Document-level Sentiment Analysis Approach Using Artificial Neural Network and Sentiment Lexicons. SIGAPP Appl. Comput. Rev., [online] 12(4), pp.67–75. Available at: <http://doi.acm.org/10.1145/2432546.2432552>.

Siddiqua, U.A., Ahsan, T. and Chy, A.N., 2017. Combining a rule-based classifier with ensemble of feature sets and machine learning techniques for sentiment analysis on microblog. 19th International Conference on Computer and Information Technology, ICCIT 2016, pp.304–309.

Susilowati, E., Sabariah, M.K., Gozali, A.A., Informatika, J.T., Telkom, U. and Machine, S.V., 2015. Implementasi metode support vector machine untuk melakukan klasifikasi kemacetan lalu lintas pada twitter implementation support vector machine method for traffic jam classification on twitter. 2(1), pp.1478–1484.

Tala, F.Z., 2003. A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia. M.Sc. Thesis, Appendix D, pp, pp.39–46.

Wahid, D.H. and Azhari, S.N., 2016. Peringkasan Sentimen Esktraktif di Twitter Menggunakan Hybrid TF-IDF dan Cosine Similarity. IJCCS (Indonesian Journal of Computing and Cybernetics Systems), [online] 10(2), p.207. Available at: <https://journal.ugm.ac.id/ijccs/article/view/16625>.

You, Z.H., Li, S., Gao, X., Luo, X. and Ji, Z., 2014. Large-scale protein-protein interactions detection by integrating big biosensing data with computational model. BioMed Research International, 2014, pp.28–30.