BAB 2
LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam penelitian yang dilakukan oleh Pramana (2011), dilakukan perangkingan rangkaian teks dengan menggunakan algoritma *Rocchio Relevance Feedback*. *Keyword* yang dimasukkan oleh user diolah oleh sistem guna menentukan dokumen yang sesuai dengan menggunakan proses penghitungan TF-IDF dan *cosine similarity*. Dari hasil dokumen yang ditampilkan oleh sistem, *user* memberikan umpan balik dengan menentukan dokumen yang relevan. Umpan balik tersebut digunakan kembali untuk memberikan hasil yang lebih Relevan. Penghitungan Umpan balik nantinya akan menggunakan metode *Rocchio*. Hasil akhir dari peneilitian ini umpan balik berupa dokumen relevan dari *user* memberikan kenaikan nilai *precision* sekitar 3% hingga 9% pada tiap tingkatan *recall.*

Penelitian mengenai analisis sentimen pada media sosial berbahasa Indonesia telah dilakukan oleh Naradipha dan Purwarianti (2012). Penelitian ini fokus di pengklasifikasian sentimen terhadap data yang diperoleh dari halaman *facebook* suatu perusahaan. Tantangan dalam proses pengklasifikisian ini adalah mengubah kata-kata tidak baku dan penulisan yang tidak tepat yang sering muncul dalam sebuah komentar. Tahap transformasi kata yang dilakukan dalam penelitian ini adalah CleanNumber, ConvertNumber, RemoveRepeat, dan *Translate* dengan menggunakan KamusAlay. Hasil dari pengubahan akan diklasifikasi dengan menggunakan dua metode yaitu SVM dan *Maximum Entropy.* Dari hasil akhir ditemukan bahwa SVM merupakan metode terbaik dengan tingkat akurasi 83,5%

Penggunaan algoritma *Rocchio Classification* dalam pengkategorian suatu dokumen telah dilakukan oleh Widjojo (2013). Pada penelitiannya, Widjojo meneliti

keakuratan alogritma *Rocchio Classification* dalam melakukan pengkategorian renungan harian Kristen ke dalam kategori Berkat, Motivator, Iman, dan Hikmat, serta menghitung *precision* dan *recall* untuk masing-masing kategori. Tahap *preprocessing* yang dilakukan adalah *lowercase,* menghilangkan karakter non-*alphabet,* tokenisasi, dan penghapusan *stopword.* Hasil akhir dari penelitian ini memberikan akurasi yang cukup tinggi untuk *feature selection* 20%, yaitu sebesar 73,33%

Pengaplikasian Penelitian mengenai pengaplikasian algoritma *Rocchio classification* juga telah dilakukan Wusana (2015). Pada penelitiannya, Wusana melakukan mengkategorikan sentimen komentar pengunjung *youtube* ke dalam dua kelas yaitu sentiment positif dan negatif. Tahap-tahap ekstraksi fitur yang dilakukan pada penelitian ini meliputi: *grabbing* komentar pada *youtube,* pendeteksian teks berbahasa Indonesia dengan menggunakan unigram, *casefolding,* tokenisasi, *stopword removal,* dan *feature selection.* Dengan menekankan penggungaan TF-IDF *weighting,* Metode *Rocchio* dinilai mampu mengklasifikasikan sentiment pengunjung *youtube* ketika menggunakan nilai *feature selection 20%* dengan akurasi 75,754627%.

Dalam penelitian yang dilakukan Antonius Rachmat dan Yuan Lukito (2016), dilakukan pembangunan dataset yang dikumpulkan dari status dan komentar terhadap calon presiden Indonesia pada masa kampanye pemilu tahun 2014 dari halaman *facebook,* dan dinamakan dataset SentiPol. Data status dan komentar diambil dari Facebook menggunakan Facebook API untuk kemudian disimpan dalam basis data lokal. Proses berikutnya dilakukan dengan pemberian label sentimen (positif, negatif, atau netral) untuk setiap data secara *crowdsourced labelling* menggunakan aplikasi web secara *online*. Hasil akhir label pada dataset ditentukan secara otomatis menggunakan metode *Weighted Majority Voting* berdasarkan bobot terbesarnya. Peneltian ini menghasilkan dataset sejumlah 3400 komentar dari 68 status dalam format CSV dengan label positif lebih dominan dari lebel negatif dan netral sehingga

dapat dingunakan dalam pembelajaran sistem *supervised learning* lainnya. Dataset Sentipol ini juga telah dicoba digunakan dalam klasifikasi sentimen analisis menggunakan metode *Naïve Bayes* dan *Support Vector Machine* dengan tingkat akurasi masing-masing 82.23% dan 84.82%.

Berdasarkan beberapa penelitian di atas, penulis akan melakukan pengklasifikasian dataset SentiPol dengan menggunakan algoritma *Rocchio Classification.* Dalam penelitian ini, penulis fokus dalam melihat keakuratan algoritma dalam mengklasifikasikan sentimen dataset SentiPol yang telah melalui tahap *preprocessing* sebagai berikut: tokenisasi, *casefolding, stopword removal, stemming, feature selection,* dan pembobotan TF-IDF.

2.2 Landasan Teori

2.2.1 Text Mining

Menurut Lumbanraja (2013), *text mining* merupakan salah satu aplikasi dari bidang *data mining* yang khusus mengolah data dalam bentuk teks. Tujuan *text mining* adalah mencari informasi implisit dari data teks sehingga bisa digunakan oleh pengguna untuk mengambil keputusan. Menurut Februariyanti dan Zuliarso (2012) *text mining* dapat diartikan sebagai penemuan informasi yang baru dan tidak diketahui sebelumnya oleh komputer, dengan secara otomatis mengekstrak informasi dari sumber-sumber yang berbeda. Kunci dari proses ini adalah menggabungkan informasi yang berhasil diekstraksi dari berbagai sumber. *Text mining* dapat didefinisikan secara luas sebagai proses pengetahuan intensif, dan penggunaannya adalah dengan berinteraksi dengan sekumpulan dokumen dan menggunakan rangkaian alat analisis (Feldman, 2007). Tahap *text mining* adalah dengan mempersiapkan dokumen (*preprocessing*), mentransformasi, memilih fitur, dan menemukan pola yang dimaksud.

2.2.3 Text Preprocessing

Menurut Februariyanti dan Zuliarso (2012) Tahapan awal dari *text mining* adalah *text preprocessing* yang bertujuan untuk mempersiapkan teks menjadi data yang akan mengalami pengolahan pada tahapan berikutnya. Beberapa contoh tindakan yang dapat dilakukan pada tahap ini, mulai dari tindakan yang bersifat kompleks seperti *part of speech (pos) tagging*, *parse tree*, hingga tindakan yang bersifat sederhana seperti proses *parsing* sederhana terhadap teks, yaitu memecah suatu kalimat menjadi sekumpulan kata. Dalam penelitian Feldman (2007), proses *parsing* teks menjadi kalimat dan kata-kata disebut sebagai tokenisasi. Pada proses tokenisasi dilakukan pemiliahan karakter yang merupakan karakter pemisah dan memiliki makna. Selain itu pada tahapan ini biasanya juga dilakukan *case folding*, yaitu pengubahan karakter huruf menjadi huruf kecil.

2.2.4 Text Transformation

Februariyanti dan Zuliarso (2012) menjelaskan pada tahap ini hasil yang diperoleh dari tahap *text preprocessing* akan melalui proses tranformasi. Adapun proses transformasi ini dilakukan dengan mengurangi jumlah kata-kata yang ada dengan penghilangan *stopword (stopword removal)*. *Stopword* adalah kata-kata yang bukan merupakan ciri (kata unik) dari suatu dokumen seperti kata sambung, kata kepunyaan.

2.2.5 Stopword Removal

Menurut Sunni dan Widyantoro (2012), *Stopword removal* merupakan proses penghilangan kata-kata yang sering muncul namun tidak memiliki pengaruh apapun dalam ekstraksi sentimen dari hasil tokenisasi. *Stopword removal* akan menghilangkan kata yang sering muncul yang terdapat di *stopword list*. Contoh dari stopword adalah “yang”, “dan”, “di”, “dari”, dan seterusnya. Kumpulan kata-kata ini akan dikumpulkan terlebih dahulu dalam sebuah *stoplist*. *Stoplist* akan berisi kata-kata yang sering muncul yang dinamakan dengan *stopword* yang nantinya akan

dihilangkan di dalam tahap *preprocessing* ini. Tahap ini dilakukan dengan membandingkan dokumen teks dengan *stoplist* yang sudah ada.

2.2.6 Stemming

Menurut Agusta (2009), Stemming merupakan suatu proses yang terdapat dalam sistem *information retrieval* yang mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (*root word*) dengan menggunakan aturan-aturan tertentu. Proses *stemming* pada teks berbahasa Indonesia memerlukan penghilangan sufiks, prefiks, dan konfiks. Sebagai contoh, kalimat "Rakyat memenuhi halaman gedung untuk menyuarakan isi hatinya" dapat diubah menjadi hanya kata dasar menjadi: "rakyat penuh halaman gedung suara isi hati".

2.2.7 *Feature Selection* dan Pembobotan Menggunakan Algoritma TF-IDF

Menurut Uchyigit dan Clark (2008), kesulitan besar dalam proses pengklasifikasian adalah banyaknya kata dari kumpulan dokumen yang ada. Agar dapat memaksimalkan algoritma pengklasifikasian, diperlukan metode *Feature Selection* yang bertujuan untuk mengurangi beban kerja tanpa mengurangi tingkat akurasi algoritma pengklasifikasian. Dalam tahap ini akan dilakukan pembobotan kata-kata tersebut. Tujuan dari pembobotan ini untuk mempermudah pengklasifikasian nantinya. Dalam penelitian ini, proses pembobotan akan menggunakan algoritma TF-IDF *(Term Frequency/Inverse Document Frequency).*

Algortima TF-IDF *(Term Frequency Inverse Document Frequency)* merupakan algoritma yang menghitung bobot masing-masing dokumen terhadap kata-kata yang menjadi isi dokumen. Frekuensi kemunculan kata menentukan bobot dan seberapa penting kata tersebut terhadap suatu dokumen. *TF (Term Frequency)* menyatakan banyak suatu kata muncul dalam sebuah dokumen, N adalah jumlah dokumen, *DF (Document Frequency)* menyatakan banyaknya dokumen yang mengandung suatu kata, sedangkan *TF-IDF* sendiri adalah nilai bobot dari suatu kata

yang diambil dari nilai *TF* dan nilai *inverse DF*, dapat didefinisikan dengan Rumus 2.1 :

$$TF - IDF\ Weight_{(w,d)} = TF_{(w,d)} \times \log(N/DF_{(w)})$$

[2.1]

Keterangan:

TF-IDF Weight(w,d) : bobot suatu kata dalam keseluruhan dokumen

w : suatu kata (*word*)

d : sebuah dokumen (*document*)

TF(w,d) : frekuensi kemunculan suatu kata *w* dalam sebuah dokumen *d*

N : jumlah dokumen secara keseluruhan

DF(w) : jumlah dokumen yang mengandung kata *w*

Contoh Kasus perhitungan *TF-IDF* :

Diketahui 3 buah dokumen (N=3)

D1 : informasi harga smartphone

D2 : masyarakat menerima pembagian beras bulog

D3 : harga *smartphone* masyarakat

Q : harga beras bulog

diperoleh hasil seperti yang disajikan pada Tabel 2.1

						IDF	*TF-IDF*			
kata (hasil token)	**D1**	**D2**	**D3**	**Query**	***DF***	**N**	**Log(N/*DF*)**	**D1**	**D2**	**D3**	**Query**
informasi	1	0	0	0	1	3	0,47712125	0,47712125	0	0	0
harga	1	0	1	1	2	3	0,17609126	0,17609126	0	0,17609126	0,17609126
smartphone	1	0	1	0	2	3	0,17609126	0,17609126	0	0,17609126	0
masyarakat	0	1	1	0	2	3	0,17609126	0	0,17609126	0,17609126	0

terima	0	1	0	0	1	3	0,47712125	0	0,47712125	0	0
bagi	0	1	0	0	1	3	0,47712125	0	0,47712125	0	0
beras	0	1	0	1	1	3	0,47712125	0	0,47712125	0	0,47712125
bulog	0	1	0	1	1	3	0,47712125	0	0,47712125	0	0,47712125
	TOTAL							**0,82930377**	**2,08457626**	**0,52827378**	**1,13033376**

Tabel 2.1 Tabel hasil perhitungna *TF* dan *IDF*

Dari hasil pembobotan yang didapat akan dlakukan proses normalisasi agar bobot token berada pada rentang 0 – 1. Menurut Wusana (2015) Proses normalisasi dimulai dengan memasukkan semua daftar TF-IDF ke dalam array sejumlah daftar TF-IDF tersebut. Setiap nilai TF-IDF akan dibagi dengan nilai akar dari penjumlahan semua nilai TF-IDF dari dalam daftar yang sudah dikuadratkan sebelumnya. Berikut merupakan contoh perhitungan normalisasi nilai TF-IDF dari Dokumen 1:

$$Normalisasi\ D1 = \frac{0{,}82930377}{\sqrt{(0{,}82930377)^2 + (2{,}08457626)^2 + (0{,}52827378)^2}}$$

$$Normalisasi\ D1 = \frac{0{,}82930377}{\sqrt{0{,}687744743 + 4{,}345458184 + 0{,}279073187}}$$

$$Normalisasi\ D1 = \frac{0{,}82930377}{\sqrt{0{,}687744743 + 4{,}345458184 + 0{,}279073187}}$$

$$Normalisasi\ D1 = \frac{0{,}82930377}{\sqrt{5{,}312276}}$$

$$Normalisasi\ D1 = \frac{0{,}82930377}{2{,}304838}$$

$$Normalisasi\ D1 = 0{,}359810075$$

Dari hasil penghitungan yang sama, nilai normalisasi D2 adalah 0,94435223 dan nilai normalisasi D3 adalah 0.229202176.

Dalam pengklasifikasian ini, dokumen dipandang sebagai seexbuah vektor yang memiliki jarak dan arah. Menurut Lumbanraja (2013), Representasi vektor dapat menggunakan *boolean* (teknik *Naive Bayes*) atau angka numerik untuk

merepresentasikan isi dokumen teks. Setiap dokumen dipandang sebagai vektor berdimensi n, dimana n adalah jumlah *term* yang ada pada himpunan dokumen.

BAB III
ANALISIS DAN PERANCANGAN SISTEM

3.1 Spesifikasi Kebutuhan

3.1.1 Kebutuhan Fungsional

Sistem perhitungan TF-IDF pada dataset SentiPol yang akan dibangun akan memiliki kebutuhan fungsional sebagai berikut:

1. *Text Preprocessing*
    a. Sistem dapat melakukan proses awal terhadap dataset SentiPol yaitu:
    b. Mengubah semua karakter huruf yang ada menjadi huruf kecil (*casefolding*).
    c. Mereduksi huruf yang keluar secara beruntun
    d. Menghilangkan karakter bukan huruf yang sering muncul, namun tidak memiliki pengaruh dalam proses pembobotan, yaitu tanda baca, dan karakter spesial lainnya.
    e. Menghilangkan karakter bukan huruf yang berupa angka
    f. Melakukan proses tokenisasi
2. *Text Transformation*
    a. Mengganti kata-kata/token yang tidak baku atau berupa singkatan menjadi kata yang baku, dengan mencocokkan token yang diuji terhadap database yang memuat data kata tidak baku atau singkatan serta kata bakunya
    b. Melakukan proses *stemming* untuk mengembalikan suatu kata ke dalam bentuk dasarnya. Dalam proses ini akan digunakan *library* PHP yaitu sastrawi yang berasal dari https://github.com/sastrawi/sastrawi.

c. Menghapus *stopword*, yaitu menghilangkan kata atau token yang termasuk dalam *stoplist* dan tidak digunakan dalam proses pembobotan. Proses ini dilakukan setelah mencocokkan token dengan stoplist yang ada didatabase. *Stoplist* yang digunakan diperoleh dari http://www.illc.uva.nl/Research/Reports/MoL-2003-02.text.pdf

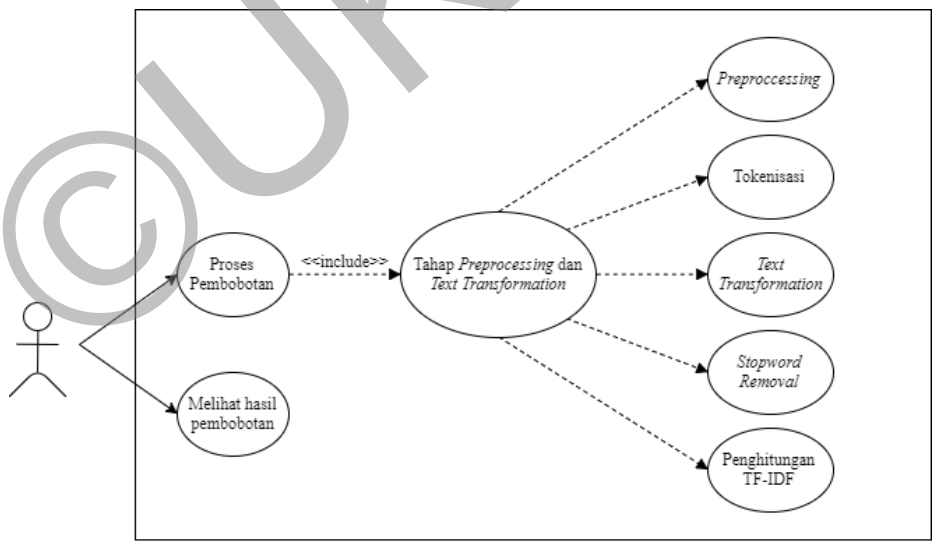
3. Pembobotan dengan Algoritma TF-IDF

   Sistem dapat melakukan pembobotan setiap kata/token yang diperoleh dari proses *preprocessing* dan *transformation*, yaitu :

   a. Pembobotan dengan algoritma TF-IDF *(Term Frequency/Inverse Document Frequency).*
   b. Normalisasi bobot setiap token menjadi berada pada rentang 0-1 untuk mempermudah proses *feature selection.*

3.1.2 Use Case

Sistem perhitungan TF-IDF pada dataset SentiPol ini melibatkan 1 aktor yaitu *user*. User merupakan aktor yang menggunakan sistem ini.



Gambar 3.1 *Use Case Diagram*

Penjelesan mengenai fungsionalitas sistem adalah sebagai berikut:

- Proses pembobotan, adalah proses dimana *user* memberi perintah kepada sistem untuk memulai proses pembobotan.
- Proses pembobotan mencakup Tahap *Preprocessing* dan *Text Transformation.* Tahap ini merupakan proses pengolahan data sebelum akan pembobotan.
- Tahap *Preprocessing* dan *Text Transformation* terdiri dari *casefolding*, mereduksi huruf yang keluar secara beruntun, menghilangkan karakter non huruf, tokenisasi, mengganti kata yang tidak baku menjadi baku, *stemming*, *stopword removal*, penghitungan TF-IDF, dan normalisasi bobot.
- Melihat hasil pembobotan, adalah proses dimana sistem sudah menyelesaikan proses pembobotan dengan menggunakan data yang telah diproses, kemudian ditampilkan kepada *user.*

3.1.3 Spesifikasi Perangkat

Dalam proses pembangunan sistem ini, perangkat yang digunakan meliputi spesifikasi sebagai berikut :

A. Perangkat Keras *(Hardware)*

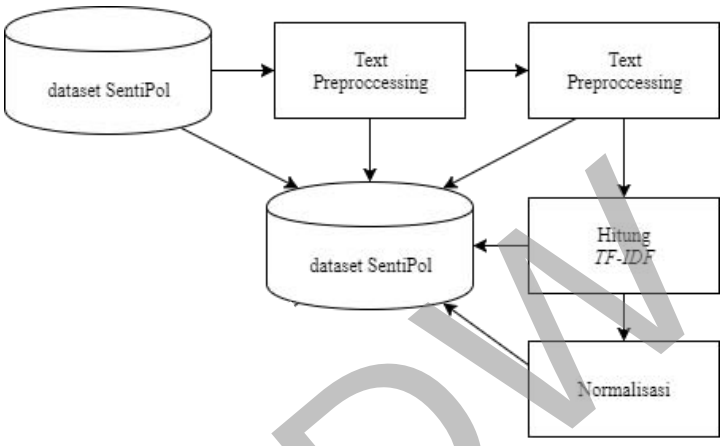
- *Processor* : Intel(R) Core(TM) i3-5005U CPU @ 2.00GHz 2.00GHz
- RAM : 4 GB
- Monitor : 14”, resolusi 1366 x 768
- *Keyboard*
- *Mouse*

B. Perangkat Lunak *(Software)*

- Sistem Operasi : *Windows 10 Enterprise 64 Bit*
- Tools : sublime text 3126, XAMPP v5.6.30, MySQL

3.2 Blok Diagram Sistem

Blok diagram dari proses pembobotan dataset SentiPol ini ditunjukkan pada Gambar 3.2.



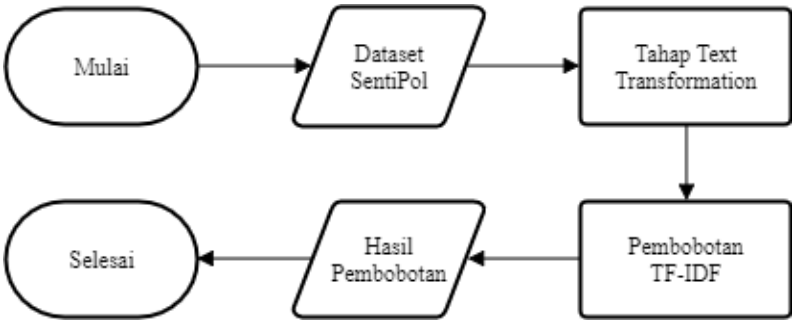
Gambar 3.2 Blok Diagram Sistem

Pada tahap pertama, sistem mengambil daftar komentar di dataset Sentipol dan disimpan di dalam *database.* Tahap kedua, komentar yang terdapat pada dataset Sentipol diolah pada tahap *Text Preprocessing* yang meliputi *casefolding*, pereduksian huruf yang keluar secara beruntun, penghilangan karakter non huruf, hingga didapatkan token. Proses sleanjutnya adalah *Text Transformation.* Pada proses ini token ditransormasi menjadi token yang baku, dikembalikan ke kata asalnya (*stemming*), hingga penghapusan *stopword*. Proses selanjutnya adalah penghitungan bobot token dengan menggunakan algoritma TF-IDF, yang kemudian hasil pembobotan tersebut dinormalisasi.

3.3 Rancangan Proses

3.3.1 Proses perhitungan TF-IDF pada dataset Pemilu

Secara umum, proses perhitungan sentiment dataset SentiPol dengan menggunakan algoritma *Rocchio Classifier* dapat dijelaskan pada Gambar 3.3.

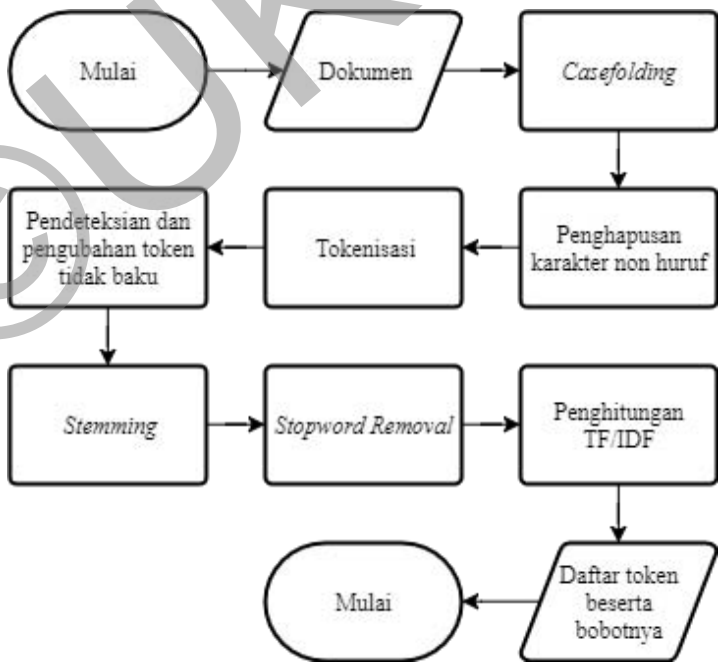


Gambar 3.3 *Flowchart* Proses perhitungan TF-IDF

Komentar yang ada akan melewati tahap *preprocessing*, *text transformation*, dan pembobotan TF-IDF. Namun, dikarenakan proses agar dapat diolah dalam proses klasifikasi, menggunakan bobot dengan rentang 0 – 1 saja, maka hasil pembobotan TF-IDF harus melewati tahap normalisasi.

3.3.2 Text Preprocessing dan Text Transformation

Penjelasan tahapan *text preprocessing* dan *text transformation* dari sistem ini, ditunjukkan dalam gambar 3.4.

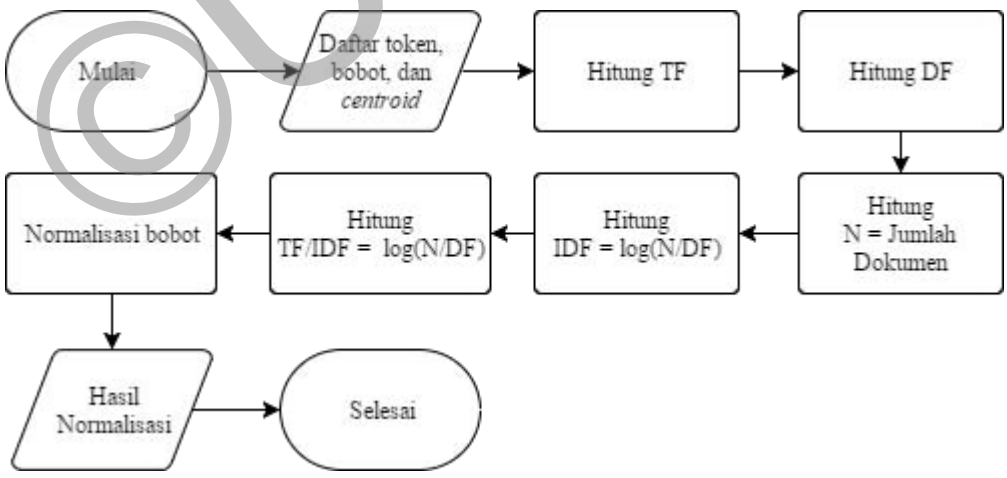


Gambar 3.4 *Flowchart* Proses *Text Preprocessing* dan *Text Transformation*

Berdasarkan gambar 3.4, tahap pertama pengklaifikasian ini dimulai dari dokumen harus melalui tahap *casefolding* untuk menyamakan jenis hurufnya. Setelah itu, kata-kata yang memiliki pengulangan huruf secara beruntun dalam penulisannya, akan direduksi sampai menjadi satu huruf saja. Kemudian, dilakukan penghapusan karakter non huruf yang ada dalam dokumen. Selanjutnya dilakukan tahap tokenisasi, dimana kata-kata dalam sebuah dokumen dipisah sehingga mendapatkan daftar token. Token yang sudah terdaftar akan dicek kembali penulisannya dan diubah menjadi kata yang baku. Setelah itu, dilakukan tahap *stemming*, dimana setiap token dikembalikan menjadi bentuk dasarnya. Kemudian, dengan membandingkan dengan daftar *stoplist* yang ada, token yang dianggap tidak berguna dalam proses perhitungan akan dihapus. Token yang sudah melalui tahap tersebut, merupakan token yang terpilih dan akan melalui proses pembobotan. Untuk melakukan proses pembobotan, token yang telah terpilih akan melewati penghitungan bobot dengan menggunakan *Term Frequency/Inverse Document Frequency.*

3.3.3 Pembobotan TF-IDF

Tahapan pada proses pembobotan TF-IDF yang digunakan pada sistem ini, ditunjukkan pada gambar 3.5.

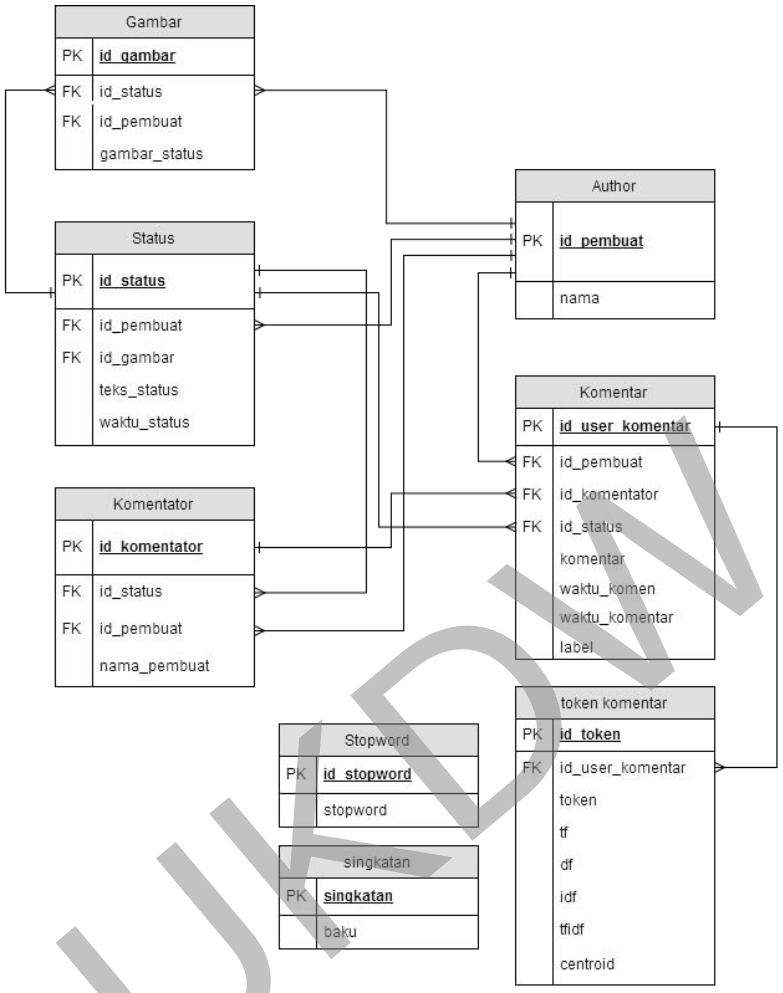


Gambar 3.P5 *Flowchart* Proses Pembobotan TF-IDF

Pada proses ini, diasumsikan bahwa proses tokenisasi sudah dilakukan. Pada tahap pertama dilakukan penghitungan *Term Frequencies* (TF) yaitu menghitung jumlah kemunculan suatu kata dari seluruh dokumen. Selanjutnya dilakukan penghitungan *Document Frequencies* (DF) yaitu jumlah dokumen yang memiliki suatu kata. Kemudian, dilakukan penghitungan jumlah dokumen. Setelah TF, DF, dan jumlah dokumen telah diketahui, dilakukan penghitungan *Inverse Document Frequencies* (IDF) dengan menggunakan rumus *log(N/DF).* Tahap selanjutnya merupakan penghitungan pembobotan TF-IDF. Hasil pembobotan TF-IDF didapatkan dari perkalian TF dan IDF.

3.4 Rancangan Database

Gambar 3.6 adalah rancangan model relasi database dalam sistem perhitungan TF-IDF pada dataset SentiPol yang akan dibangun.



Gambar 3.6 Relasi antar entitas

3.6 Rancangan Pengujian Sistem

Dalam proses perhitungan TF-IDF, akan dilakukan penganalisaan token yang berhasil didapatkan oleh sistem setelah melalui proses tahap-tahap *preprocessing* dan *transformation,* waktu yang diperlukan oleh sistem dalam memproses bobot dan data token, dan tingkat keberhasilan memproses dengan waktu yang ditentukan 3 jam.

Untuk mengevaluasi kecepatan sistem yaitu dengan menghitung jumlah dokumen yang berhasil dibobotkan, dengan terlebih dahulu melalui tahap preprocessing, tranformasi, dan pembobotan. Dari 3400 komentar yang terdapat di

dalam dataset SentiPol, 80% data akan digunakan sebagai data pelatihan dengan 2723 dokumen. Dokumen pengujian yang akan digunakan adalah 20% dari dataset SentiPol, yang berjumlah 677 dokumen,. Hasil yang didapat dari pengevaluasian ini adalah berupa persentase keakuratan sistem.

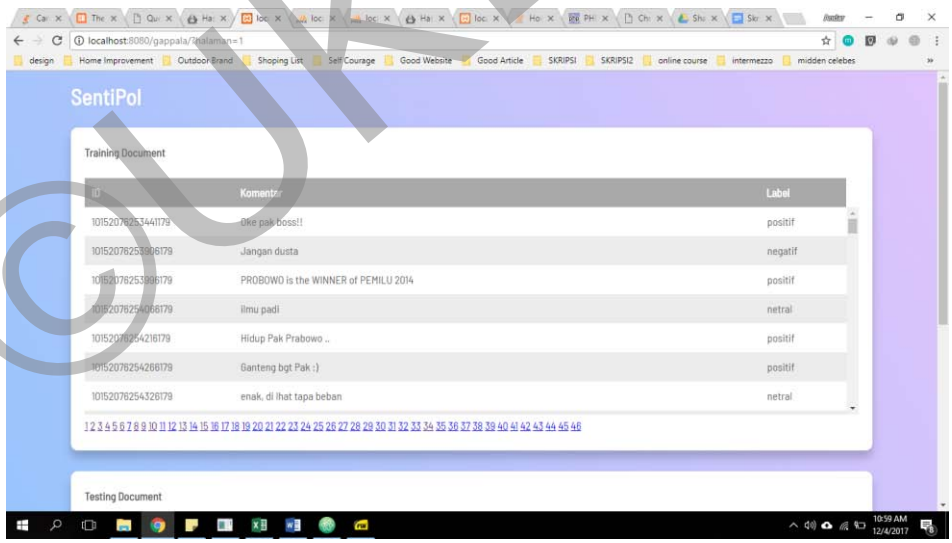
BAB IV
IMPLEMENTASI DAN ANALISIS

4.1 Implementasi Sistem

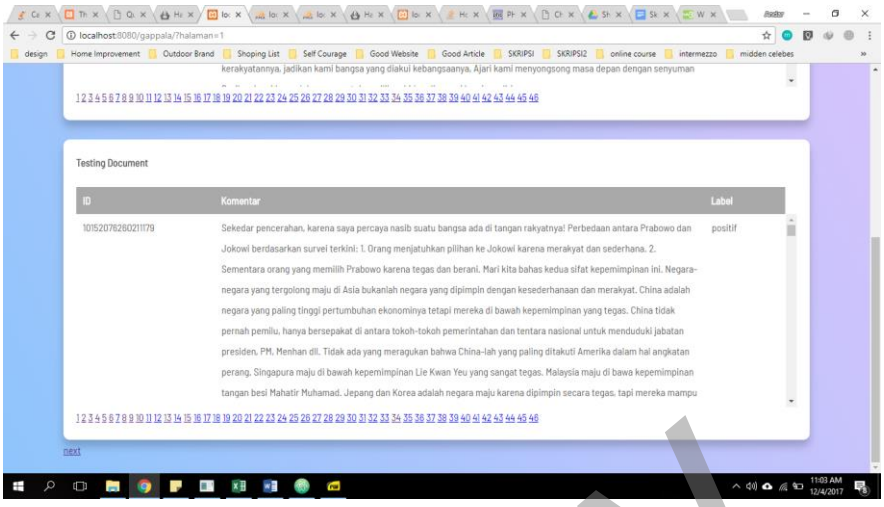
4.1.1 Antar Muka Sistem

A. Halaman Awal

Gambar 4.1 dan Gambar 4.2 merupakan halaman awal dari sistem ini. Halaman awal ini berisi daftar komentar training yang digunakan sebagai daftar acuan pengklasifikasian dan daftar komentar testing. Semua komentar teresbut telah berada pada tabel data komentar_testing dan komentar_training di dalam *database.* Pada bagian akhir daftar komentar terdapat tombol yang digunakan untuk menuju ke dalam halaman cleanse.



Gambar 4.1 Implementasi halaman awal (bagian a)

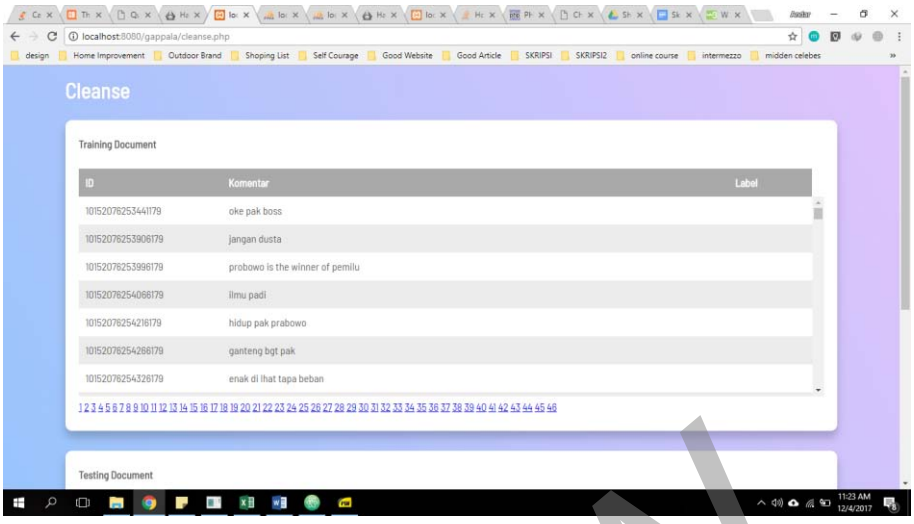


Gambar 4.2 Implementasi halaman awal (bagian b)

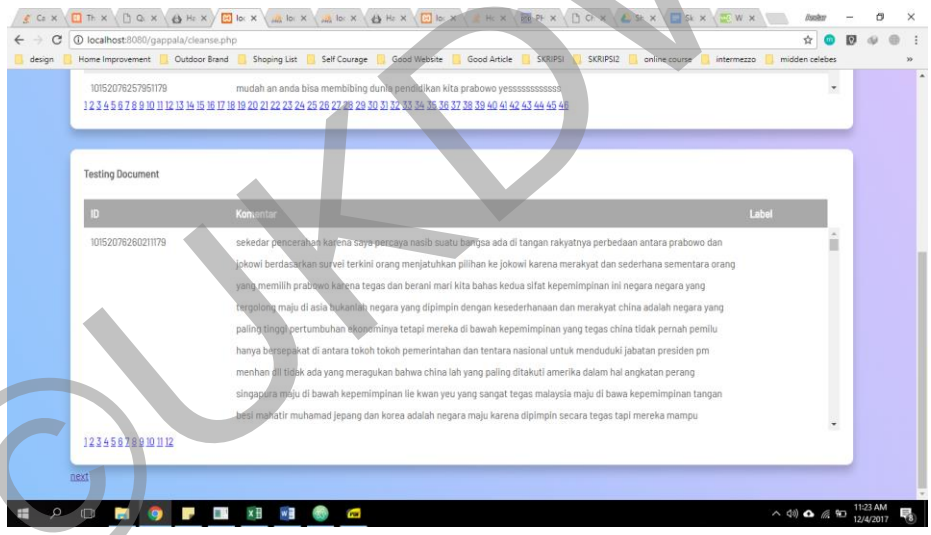
B. Halaman *Testing*

a. Halaman *Cleanse*

Gambar 4.3 dan Gambar 4.4 adalah halaman yang memuat komentar yang telah melalui proses *cleanse.* Pada proses *cleanse,* komentar mengalami dua proses pembersihan. Proses pertama, karakter non huruf (angka, tanda baca, dan karakter special lainnya) yang berada di dalam komentar dihapus, hingga menyisakan komentar saja. Proses kedua sistem melakukan proses *casefolding*. Pada proses ini, setiap huruf di dalam komentar diganti dari huruf kapital dan menjadi huruf kecil. Hasil pada proses ini disimpan pada tabel prep_testing dan prep_training. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



Gambar 4.3 Implementasi halaman *cleanse* (bagian a)

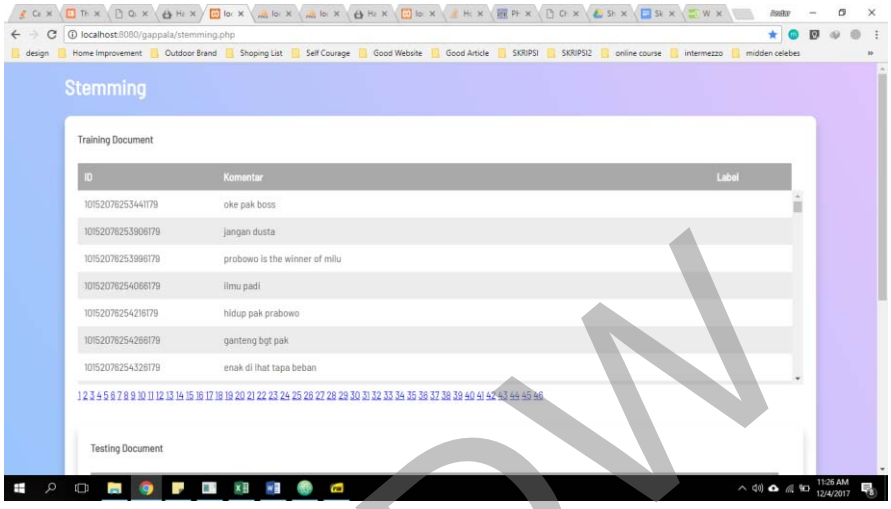


Gambar 4.4 Implementasi halaman *cleanse* (bagian b)

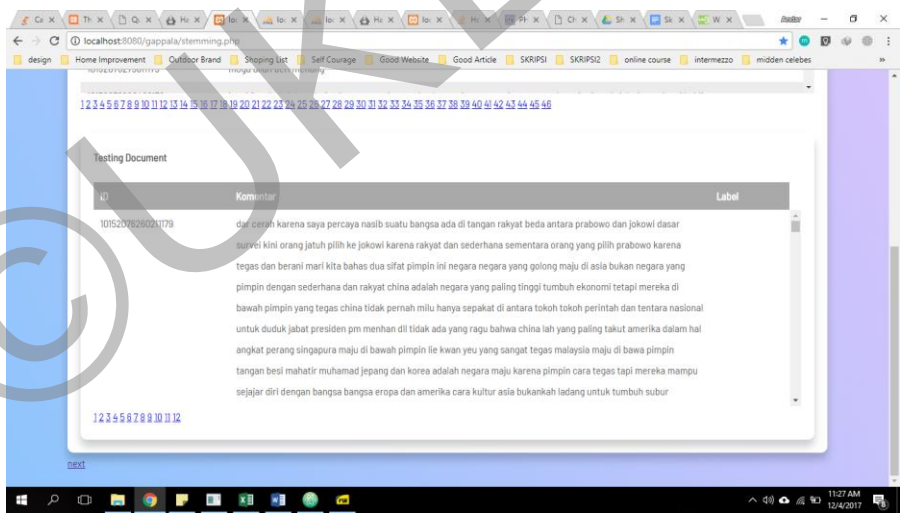
b. Halaman daftar hasil *Stemming*

Gambar 4.5 dan Gambar 4.6 adalah implementasi halaman hasil *stemming* dengan menggunakan *library* dari Sastrawi-1.1.0. Hasil yang didapatkan adalah daftar komentar yang di dalamnya terdapat kata berimbuhan, menjadi kata dasar. Komentar hasil proses *stemming* ini akan

disimpan pada tabel stem_testing dan stem_training di dalam *database.* Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



Gambar 4.5 Implementasi halaman *stemming* (bagian a)

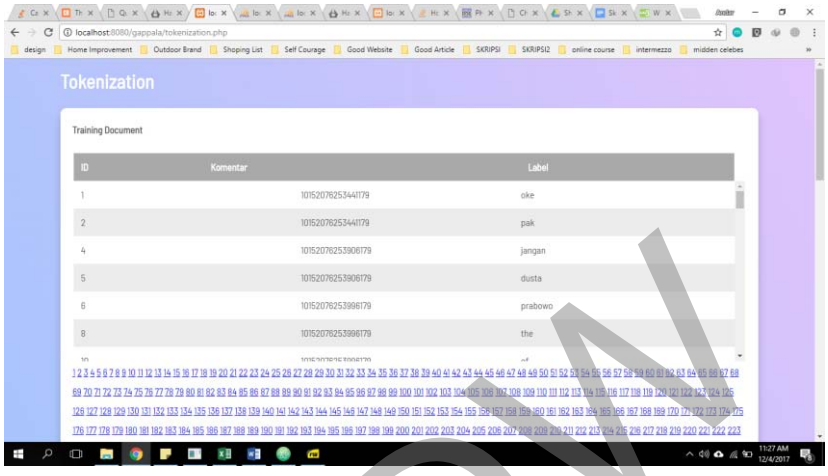


Gambar 4.6 Implementasi halaman *stemming* (bagian b)

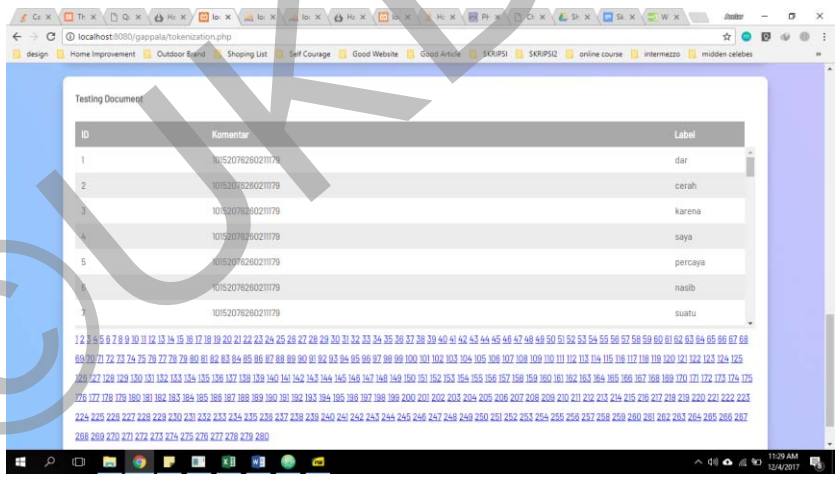
c. Halaman daftar hasil tokenisasi

Gambar 4.7. dan Gambar 4.8 adalah implementasi halaman hasil tokenisasi. Hasil yang didapatkan adalah daftar token dari seluruh komentar dataset

SentiPol. Token yang didapatkan akan disimpan pada tabel tokenization_training, dan tabel tokenization_testing. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



Gambar 4 7 Implementasi halaman tokenisasi (bagian a)

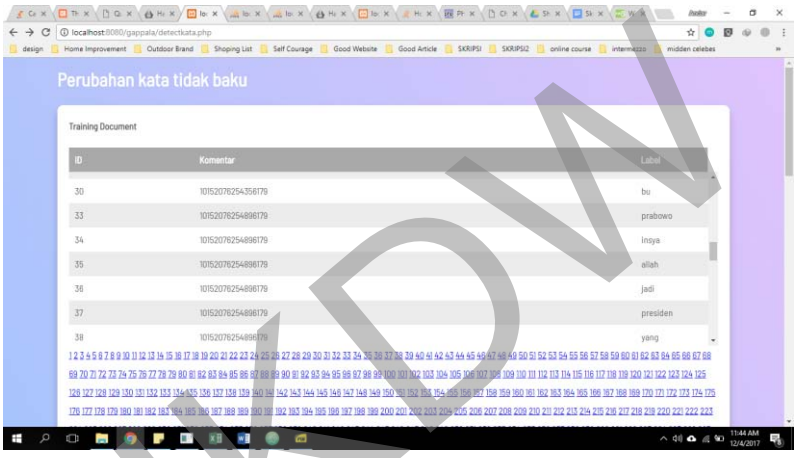


Gambar 4 8 Implementasi halaman tokenisasi (bagian b)

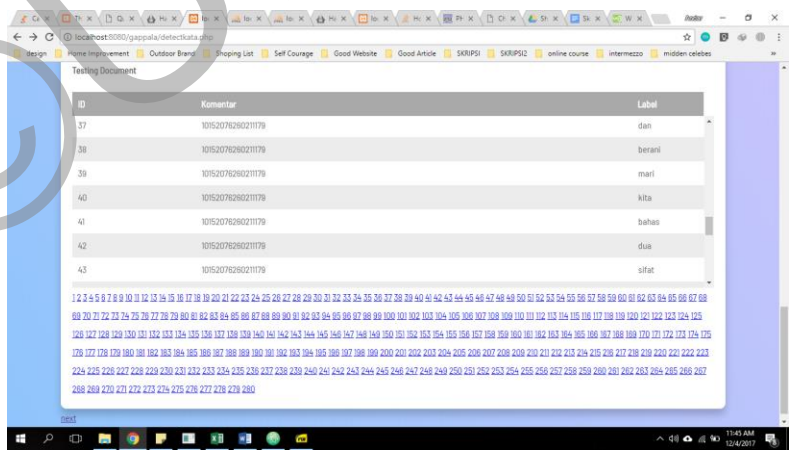
d. Halaman daftar token hasil Perubahan kata tidak baku

Gambar 4.9 dan Gambar 4.10 adalah implementasi halaman hasil pendeteksian kata tidak baku. Token yang telah tersimpan akan dibandingkan dengan kumpulan kata bahasa Indonesia yang tersimpan di dalam tabel

kamus. kumpulan kata yang tidak terdeteksi dalam proses perbandingan, dianggap sebagai kata tidak baku. Selanjutnya, kata yang tidak baku tersebut dicari padananannya dalam tabel tidakbaku. Token yang terdeteksi akan mengalami perubahan dan langung diupdate di dalam tabel tokenization_training, dan tokenization_testing. Kata yang masih tidak teredeteksi, dihapus dari daftar token. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



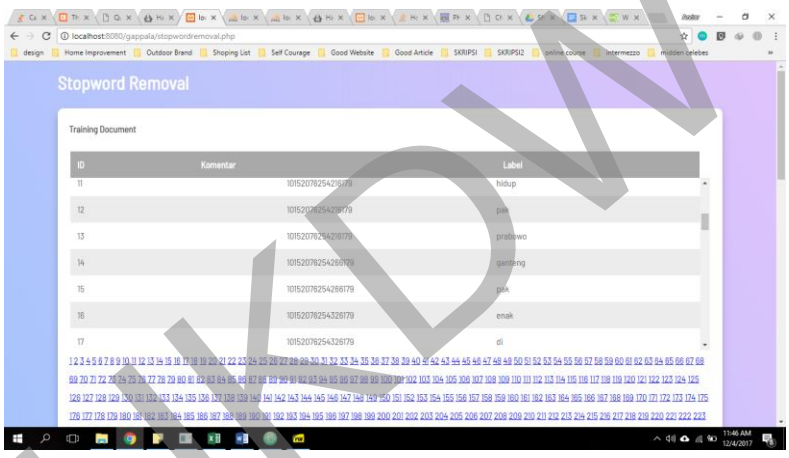
Gambar 4.9 Implementasi halaman Perubahan kata tidak baku (bagian a)



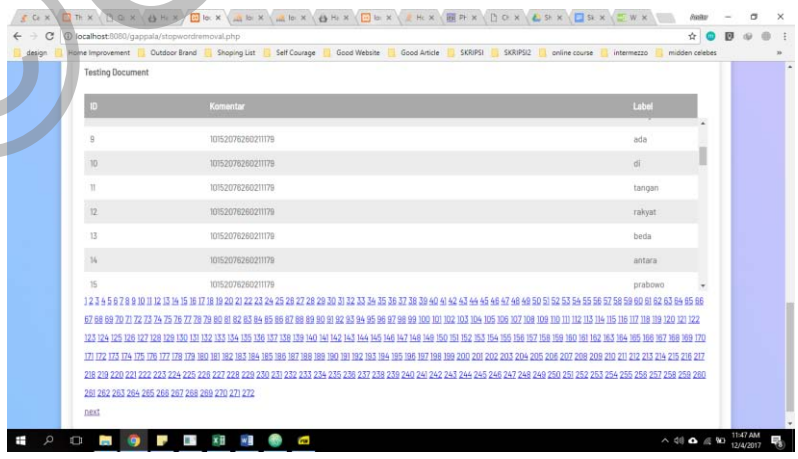
Gambar 4. 10 Implementasi halaman perubahan kata tidak baku (bagian b)

e. Halaman daftar token hasil *Stopword Removal*

Gambar 4.11 dan Gambar 4.12 merupakan implementasi halaman hasil *stopword removal*. Pada proses ini, dilakukan pengecekan token yang telah terdaftar untuk mencari token-token yang dianggap sebagai stoplist yang telah terdaftar pada tabel stoplist. Token yang terdeteksi dihapus, dan hasil dari proses ini dismpan di dalam tabel abisdistopword_testing dan abisdistopword_training. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



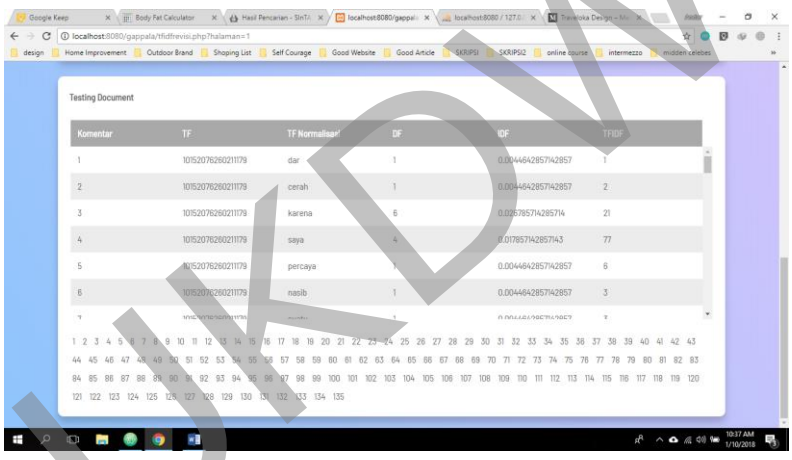
Gambar 4.11 Implementasi halaman *Stopword Removal* (bagian a)



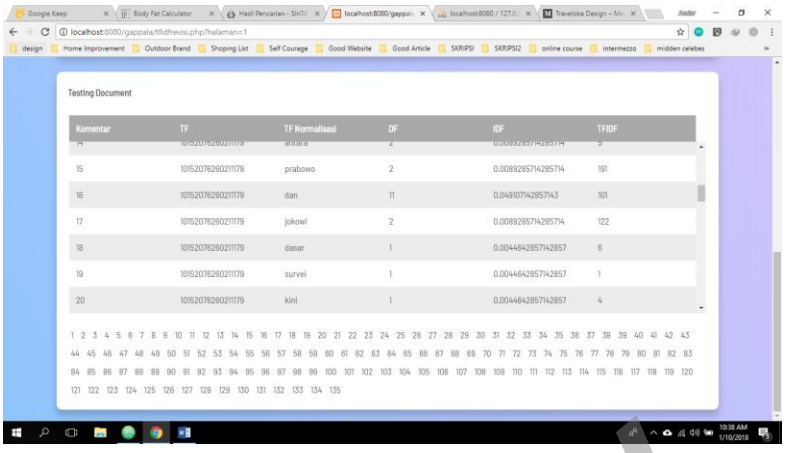
Gambar 4.12 Implementasi halaman Stopword Removal (bagian b)

f. Halaman daftar hasil penghitungan TF-IDF

Gambar 4.13 dan Gambar 4.14 merupakan implementasi halaman hasil penghitunganTF-IDF untuk komentar training dan komentar testing. Pada bagian tabel training, ditampilkan daftar token unik sekaligus dengan nilai TF, DF, IDF, *centroid* Positif, Negatif, dan Netral. Pada bagian tabel testing, ditampilkan daftar token unik beserta dengan nilai TF, DF, IDFnya. Semua hasil penghitungan disimpan pada tabel tfidf_training dan tfidf_testing. Untuk menuju ke proses berikutnya, user harus menekan tombol selanjutnya.



Gambar 4.13 Implementasi hasil penghitungan TF-IDF (bagian a)



Gambar 4.14 Implementasi hasi penghitungan TF-IDF (bagian b)

4.2 Analisis Sistem

4.2.1 Evaluasi Proses Sistem

A. Proses *Cleanse*

Pada proses *cleanse*, dokumen selesai diproses oleh sistem dengan waktu pemrosesan 4 detik. Tabel 4.1 dan Tabel 4.2 menunjukkan persentase keberhasilan sistem melakukan proses *casefolding* dan penghapusan karakter non huruf yang terdapat di dalam dataset *training* dan testing.

Tabel 4.1 Tabel presentase proses *cleanse* pada data *training*

	Frekuensi	**persentase**
berhasil	2684	98,57%
gagal	39	1,43%
Total	**2723**	**100%**

Tabel 4.2 Tabel persentase proses *cleanse* pada data *testing*

	Frekuensi	**presentase**
berhasil	667	98,52%
gagal	10	1,48%

Total	**677**	**100%**

Dari Tabel 4.1 dan Tabel 4.2 ditunjukkan bahwa persentase keberhasilan proses *cleanse* di data *training* sebesar 98,57% dan di data *testing* sebesar 98.52%. Kegagalan yang terjadi dalam melakukan proses *cleanse* dikarenakan sistem tidak dapat mendeteksi beberapa karakter non huruf baik yang terdapat dalam data *training* ataupun *testing.* Kakater tersebut antara lain : tanda petik tunggal (‘ ’), tanda petik ganda (“ ”), simbol ‘�’, simbol ‘•’, tanda kutip (` `), dan tanda ellipsis (…).

B. Proses *Stemming*

Pada proses *stemming,* dokumen selesai diproses oleh sistem dengan waktu pemrosesan 8 detik. Tabel 4.3 dan Tabel 4.4 menunjukkan persentase dokumen yang mengalami perubahan setelah dilakukan proses *stemming* pada data *training* dan data *testing.*

Tabel 4.3 Tabel persentase data *training* yang mengalami perubahan saat proses *stemming*

	Frekuensi	**Presentase**
Berubah	1584	58.17%
Tetap	1139	41.83%
Total	**2723**	**100%**

Tabel 4.4 Tabel persentase data *testing* yang mengalami perubahan proses *stemming*

	Frekuensi	**Presentase**
Berubah	321	47.42%
Tetap	356	52.58%
Total	**677**	**100%**

Dari Tabel 4.3 dan Tabel 4.4 ditunjukkan bahwa persentase data yang mengalami perubahan dalam proses *stemming* pada data *training* sebesar 58,17% dan pada data *testing* sebesar 47,42%.

C. Proses tokenisasi final

Pada proses tokenisasi final, data yang telah melalui proses *stemming*, akan melalui proses tokenisasi, *detectkata,* dan *stopwordremoval.* Pada proses tokenisasi, didapatkan 80142 token pada data *training* dan 18737 token pada data *testing* dengan waktu pemrosesan 8 detik.

Pada proses *detectkata*, sistem terlebih dahulu mendeteksi token yang tidak dianggap sebagai sbahasa indonesia yang baku. Tabel 4.5 dan Tabel 4.6 menunjukkan jumlah token yang dianggap tidak baku dalam data *training* dan data *testing.* Dari Tabel 4.5 dan Tabel 4.6 ditunjukkan jumlah token yang dianggap tidak baku dalam data *training* sejumlah 12196 token. Pada data *testing,* token yang dianggap tidak baku sejumlah 2402 token. Token yang tidak baku ini selanjutnya diproses untuk dicari padanan kata yang baku. Jika ditemukan, token yang tidak baku langsung dirubah ke dalam bentuk baku. Jika tidak ditemukan, token tersebut dihapus dari data *training.*

Tabel 4.5 Tabel jumlah deteksi token yang tidak baku pada data *training*

	Frekuensi
Token Tidak baku	12196
Token Baku	67946
Total	**80142**

Tabel 4.6 Tabel jumlah deteksi token yang tidak baku pada data *training*

	Frekuensi
Token Tidak baku	2402
Token Baku	16335
Total	**18737**

Tabel 4.7 dan Tabel 4.8 menunjukkan jumlah token tidak baku yang mengalami perubahan menjadi token yang baku dan token yang dihapus dalam *database*. Dari Tabel 4.7 dan Tabel 4.8 ditunjukkan jumlah token yang berhasil dirubah pada data *training* sejumlah 3070 token dan 9126 token yang tidak berhasil dirubah akan dihapus dari data *training*. Pada data *testing*, jumlah token yang berhasil dirubah pada data *testing* sejumlah 542 token dan 1860 token yang tidak berhasil dirubah akan dihapus dari data *testing*. Setelah melalui proses ini, token pada data *training* menjadi 71016 token dan 16877 token pada data *testing* dengan total waktu pemroresan 32 menit, 2 detik.

Tabel 4.7 Tabel jumlah token tidak baku yang ter-*update* dan terhapus dari data *training*

	Frekuensi
Token terdelete	9126
Token terupdate	3070
Total	**12196**

Tabel 4.8 Tabel jumlah token tidak baku yang ter-*update* dan terhapus dari data *testing*

	Frekuensi
Token terdelete	1860
Token terupdate	542
Total	**2402**

Pada proses *stopword removal*, sistem akan menghapus token pada data *training* dan data *testing* yang dianggap sebagai *stopword*. Tabel 4.9 dan Tabel 4.10 menunjukkan jumlah token yang terdeteksi sebagai *stopword* dalam data *training* dan data *testing.* Dari Tabel 4.9 dan Tabel 4.10 ditunjukkan jumlah token yang terdeteksi sebagai *stopword* pada data *training* sejumlah 30916. Pada data *testing,* ditemukan sejumlah 7197 token yang terdeteksi sebgai *stopword*. Dengan Token yang terdeteksi sebagai stopword dihapus dari sistem, maka token pada data *training* menjadi 40100 token dan pada data *testing* menjadi 9680 token, dengan waktu pemrosesan 1 menit 27.25 detik.

Tabel 4.9 *Tabel jumlah token terdeteksi sebagai* stopword *pada data* training

	Frekuensi
Token stopword	30916
Token final	40100
Total	**71016**

Tabel 4.10 Tabel jumlah token terdeteksi sebagai *stopword* pada data *testing*

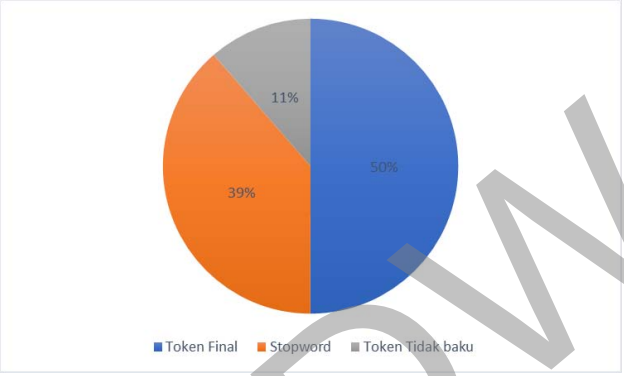
	Frekuensi
Token stopword	7197
Token final	9680
Total	**16877**

D. Grafik Hasil Proses Tokenisasi Final

Melalui Grafik 4.1 dapat terlihat bahwa dari total 80142 token yang didapat dari proses tokenisasi pada data *training*, 50% dari token merupakan

token final dengan jumlah 40100 token yang akan diklasifikasi. Persentase terbesar kedua sebesar 39% merupakan *stopword* yang harus dihapus dari data *training* dengan jumlah 30916. Dan 11% sisanya merupakan persentase kata tidak baku yang tidak dapat diterjemahkan oleh sistem.

Grafik 4.1 Grafik Persentase seluruh token pada data *training*



Melalui Grafik 4.2 dapat terlihat bahwa dari total 18737 token yang didapat dari proses tokenisasi pada data *testing*, 52% dari token merupakan token final dengan jumlah 9680 token yang akan diklasifikasi. Presentase terbesar kedua sebesar 38% merupakan *stopword* yang harus dihapus dari data *training* dengan jumlah 7197. Dan 10% sisanya merupakan presentase kata tidak baku yang tidak dapat diterjemahkan oleh sistem.

Grafik 4.2 Grafik presentase seluruh token pada data *testing*

4.2.2 Evaluasi Perhitungan TF-IDF

Proses perhitungan TF-IDF pada sistem dimulai dengan menyimpan setiap token unik di dalam setiap komentar dari data *training* dan juga data *testing*. Pada proses ini, token unik yang tersimpan pada data *training* sejumlah 30140 token. Token unik yang tersimpan pada data *testing* sejumlah 6571 token, dengan waktu memproses selama 23 menit, 23,28 detik. Setelah proses penyimpanan token unik sudah dijalankan, sistem memulai proses perhitungan bobot dari setiap token yang sudah disimpan sebelumnya. Lama waktu yang dibutuhkan untuk sistem memberikan nilai bobot kepada semua token adalah 4 jam, 19 menit, 38,42 detik.

Pada hasil perhitungan TF-IDF yang telah dinormalisasi dan masih bernilai lebih dari 1, sistem akan menyimpan nilai TF-IDF tersebut dengan nilai maksimal yaitu 1 agar TF-IDF yang telah dinormalisasi tetap berada pada rentang 0 – 1. Terdapat sejumlah 364 token pada data *training* dan 81 token pada data *testing* yang memiliki nilai TF-IDF lebih dari 1 dan mengalami perubahan dalam sistem. Pada tabel 4.11 ditunjukkan 15 daftar bobot tertinggi pada data *training* yang telah tersimpan dan mengalami perubahan bobot menjadi nilai 1. Pada tabel 4.12. ditunjukkan 15 daftar bobot tertinggi pada data *training* yang telah tersimpan dan mengalami perubahan bobot menjadi nilai 1

Tabel 4.11 Tabel daftar token dengan nilai TF-IDF tertinggi pada data *training*

ID Komentar	token	TF	TF Norm	DF	IDF	TF-IDF	TF-IDF perubahan
10152109590886100	sempak	1	1	1	3.408748606	3.408748606	1
10152114703206100	cakep	1	1	1	3.408748606	3.408748606	1
250575651816237	lawas	1	1	1	3.408748606	3.408748606	1
258326517707817	sut	1	1	1	3.408748606	3.408748606	1
268064936733975	jember	1	1	1	3.408748606	3.408748606	1
10152122496621100	kangen	1	1	1	3.408748606	3.408748606	1
10152142269101100	premium	1	1	2	3.107718611	3.107718611	1
10152149208426100	mer	1	1	2	3.107718611	3.107718611	1
10152149208086100	karawang	1	1	2	3.107718611	3.107718611	1

10152093516736100	tegal	1	1	2	3.107718611	3.107718611	1
251614281712374	pil	1	1	2	3.107718611	3.107718611	1
268064380067364	ki	1	1	2	3.107718611	3.107718611	1
265452963661839	jr	1	1	2	3.107718611	3.107718611	1
259726990901103	sus	1	1	2	3.107718611	3.107718611	1
258919037648565	subhanallah	1	1	3	2.931627351	2.931627351	1

Tabel 4.12 Tabel perubahan nilai TF-IDF daftar token dengan nilai TF-IDF tertinggi pada data *testing*

ID Komentar	token	TF	TF Norm	DF	IDF	TF-IDF	TF-IDF Perubahan
272579926282476	nek	1	1	1	2.794488047	2.794488047	1
272580629615739	mel	1	1	1	2.794488047	2.794488047	1
10152181891726100	komandan	1	1	1	2.794488047	2.794488047	1
269097949964007	labbaik	1	1	1	2.794488047	2.794488047	1
10152173257976100	top	1	1	2	2.493458051	2.493458051	1
10152169424506100	alhamdulillah	1	1	2	2.493458051	2.493458051	1
10152181931856100	kampung	1	1	3	2.317366792	2.317366792	1
275745575965911	metal	1	1	4	2.192428055	2.192428055	1
10152181894531100	laksana	1	1	4	2.192428055	2.192428055	1
276918245848644	mas	1	1	4	2.192428055	2.192428055	1
10152181891561100	milih	1	1	5	2.095518042	2.095518042	1
269107326629736	ok	1	1	6	2.016336796	2.016336796	1
276919312515204	amien	1	1	6	2.016336796	2.016336796	1
276918955848573	amien	1	1	6	2.016336796	2.016336796	1
10152181891636100	pake	1	1	6	2.016336796	2.016336796	1

4.2.3 Evaluasi Waktu Proses Keseluruhan Sistem

Tabel 4.12 menunjukkan waktu yang dibutuhkan sistem dalam memproses seluruh data yang ada pada dataset SentiPol hingga mendapatan bobot tiap tokennya. Ditunjukkan bahwa total lama waktu yang dibutuhkan oleh sistem adalah 4 jam, 53 menit, 28 detik. Dari enam proses utama dalam sistem ini, dapat terlihat bahwa waktu terlama yang dibutuhkan oleh sistem dalam menyelesaikan tiap proses mencapai empat jam.

Tabel 4.13 Tabel waktu pemrosesan

Nama Proses	Lama Waktu
Cleanse	0:00:04
Stemming	0:00:08
Tokenisasi	0:00:08
*de*tect*kata*	0:32:02
Stopword Removal	0:01:27
TF-IDF	4:19:39
TOTAL WAKTU	**4:53:28**

LAMPIRAN

Program Studi Informatika
Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana Yogyakarta
Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929

FORMULIR PERBAIKAN (REVISI) SKRIPSI

Strata-1 Program Studi Informatika

Yang bertanda tangan di bawah ini:

Nama	:	ANDAR SETIAWAN POLE
N I M	:	22104906
Judul Skripsi	:	PERHITUNGAN TF-IDF PADA DATASET SENTIPOL
Tanggal Pendadaran	:	21 Desember 2017 08:00 WIB

Telah melakukan perbaikan tugas akhir dengan lengkap.

Demikian pernyataan kami agar dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 16 Januari 2018

Dosen Pembimbing I	Dosen Pembimbing II
Antonius Rachmat C., S.Kom.,M.Cs.	Yuan Lukito, S.Kom., M.Cs.

Dicetak tanggal: *16 Januari 2018 07.52 WIB*

Kartu Konsultasi Tugas Akhir

Program Studi Teknik Informatika Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana Yogyakarta
Dr. Wahidin Sudirahusada 5-25 Yogyakarta, 55224. Telp. (0274)563929

NIM : ANDAR SETIAWAN POLE
Judul : **KLASIFIKASI SENTIMEN DATASET PEMILU MENGGUNAKAN ALGORITMA ROCCHIO**
Dosen Pembimbing I : Antonius Rachmat C., S.Kom.,M.Cs.

	Tanggal	Paraf	
1	11/10 - 2017		- Demo dan menunjukkan hasil manual program, ok - Cari info ttg centroid. - perbaiki ada fitur seleksi - tidak perlu regex..
2	8 - 11 - 2017		- Demo, konsultasi tentang query kamus - Demo, tidak perlu simpan token untuk - Konsultasi ke pricil
3	20 - 11 - 2017		- Demo program sdh sampai TF IDF - Diskusi cosine - Diskusi fitur selection
4	27 - 11 - 2017		- Demo program. - TF IDF ok.
5	30 - 11 - 2017		Revisi Bab IV.
6	4 - 12 - 2017		Acc pendadaran!
7	Tanggal:	Paraf:	
8	Tanggal:	Paraf:	

Kartu Konsultasi Tugas Akhir

Program Studi Teknik Informatika Fakultas Teknologi Informasi
Universitas Kristen Duta Wacana Yogyakarta
Dr. Wahidin Sudirahusada 5-25 Yogyakarta. 55224. Telp. (0274)563929

NIM : ANDAR SETIAWAN POLE
Judul : **KLASIFIKASI SENTIMEN DATASET PEMILU MENGGUNAKAN ALGORITMA ROCCHIO**
Dosen Pembimbing II : Yuan Lukito, S.Kom., M.Cs.

1	Tanggal: 20/11/2017	Paraf:

- Menunjukkan Hasil sampai TF-IDF
-
- Kata typo dicari kemiripannya
- 3400 Data diolah saja
- Minggu Depan Penelitian + Laporan selesai

2	Tanggal: 27/11/2017	Paraf:

- Perhitungan Klasifikasi Salah karena penghitungan TF-IDF salah
- Store procedure / sulit di query, looping manual saja.
- Pseudocode Klasifikasi, Feature selection, Centroid

3	Tanggal:	Paraf:
4	Tanggal:	Paraf:
5	Tanggal:	Paraf:
6	Tanggal:	Paraf:
7	Tanggal:	Paraf:
8	Tanggal:	Paraf:

Source Code Halaman Home

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">

    <title>
    </title>
    <link rel="stylesheet"
href="css/style.css">
  </head>
  <!-- connect database -->
  <?php
      $link =
mysqli_connect('localhost','roo
t','root','sentipol2' );
      if($link)
      {
      }
      else
      {
            echo "koneksi
database gagal.";
      }
  ?>

  <body>
    <div class="container">
      <header>
        <h1 class="text-
muted">SentiPol</h1>
      </header>
      <div class="training-
table">
        <h4>Training
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
              <th>Label</th>
            </tr>
          </thead>
          <tbody>
            <?php
              $train_halaman =
60;
              $train_page =
isset($_GET["train_halaman"]) ?
(int)$_GET["train_halaman"] :
1;
              $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
              $train_result =
mysqli_query($link, "SELECT *
FROM komentar_training");
              $train_total =
mysqli_num_rows($train_result);
              $train_pages =
ceil($train_total/$train_halama
n);
              $train_query =
mysqli_query($link, "SELECT *
FROM komentar_training LIMIT
$train_mulai, $train_halaman");
              $train_no
=$train_mulai+1;

              while($row =
mysqli_fetch_row($train_query))
{
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[3]"; ?></td>
                  <td><?php
echo "$row[4]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
komentar = ".$train_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
            <a
href="?train_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>
```

```
            <?php } ?>
          </div>
        </div>

        <div class="testing-table
clearfix">
          <h4>Testing
Document</h4>
          <table class="fixed-
headers">
            <thead>
              <tr>
                <th>ID</th>
                <th>Komentar</th>
              </tr>
            </thead>
            <tbody >
              <?php
                $test_halaman =
60;
                $test_page =
isset($_GET["test_halaman"]) ?
(int)$_GET["test_halaman"] : 1;
                $test_mulai =
($test_page>1) ? ($test_page *
$test_halaman) - $test_halaman
: 0;
                $test_result =
mysqli_query($link, "SELECT *
FROM komentar_testing");
                $test_total =
mysqli_num_rows($test_result);
                $test_pages =
ceil($test_total/$test_halaman)
;
                $test_query =
mysqli_query($link, "SELECT *
FROM komentar_testing LIMIT
$test_mulai, $test_halaman");

                while($row =
mysqli_fetch_row($test_query)){
              ?>
                  <tr>
                    <td><?php
echo "$row[0]"; ?></td>
                    <td><?php
echo "$row[3]"; ?></td>
                  </tr>
              <?php
                }
              ?>
            </tbody>
          </table>
          <?php echo "jumlah
komentar = ".$test_total; ?>
          <div
class="pagination">
            <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
            <a
href="?test_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>

            <?php } ?>
          </div>
        </div>

        <footer>
          <a
href="cleanse.php">next</a>
        </footer>
      </div>
      <!-- Op tional JavaScript -
->
    </body>
    <?php mysqli_close($link); ?>
</html>
```

Source Code Halaman Cleanse

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
    </title>
    <link rel="stylesheet"
href="css/style.css">
  </head>
  <!-- connect database -->
  <?php
    $link =
mysqli_connect('localhost','roo
t','root','sentipol2' );
    if($link)
    {
    }
    else
    {
```

```
          echo "koneksi
database gagal.";
    }
  ?>
  <body>
    <?php
      //PAGINATION TRAINING
      $train_halaman = 60;
      $train_page =
isset($_GET["train_halaman"]) ?
(int)$_GET["train_halaman"] :
1;
      $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
      $train_result =
mysqli_query($link, "SELECT *
FROM komentar_training");
      $train_total =
mysqli_num_rows($train_result);
      $train_pages =
ceil($train_total/$train_halama
n);
      $train_query =
mysqli_query($link, "SELECT *
FROM komentar_training LIMIT
$train_mulai, $train_halaman");

      //PAGINATION TESTING
      $test_halaman = 60;
      $test_page =
isset($_GET["test_halaman"]) ?
(int)$_GET["test_halaman"] : 1;
      $test_mulai =
($test_page>1) ? ($test_page *
$test_halaman) - $test_halaman
: 0;
      $test_result =
mysqli_query($link, "SELECT *
FROM komentar_testing");
      $test_total =
mysqli_num_rows($test_result);
      $test_pages =
ceil($test_total/$test_halaman)
;
      $test_query =
mysqli_query($link, "SELECT *
FROM komentar_testing LIMIT
$test_mulai, $test_halaman");

      if($train_total==0 &&
$test_total==0){
        $query_parts = array();
        //cleaning non
alphabetical and casefolding
        while ($row =
mysqli_fetch_row($train_result)
){
            $words      = [];
            $delim      = "
\r\t\"*''/;:[]{}+-#$+=_|.,;-
()?></!%:@1234567890#&"."";
            $tok  =
strtok($row[3], $delim);

            while ($tok !==
false) {
              $words[] = $tok;
              $tok =
strtok($delim);
            }
            $sentences =
array_values($words);
            $panjang_array =
count($sentences);
            $bersih = implode("
", $sentences);
            $query_parts[] =
"('" . $row[0]."', '" .
strtolower($bersih). "')";
        }

        //QUERY INSERT TRAINING
DATA
        $insPrep = "INSERT INTO
prep_training
(idtraining_komentar,
komentraining_casefold) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$insPrep)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute $insPrep. "
. mysqli_error($link);
        }

        $query_parts = array();
```

```
        //cleaning non
alphabetical and casefolding
        while ($row =
mysqli_fetch_row($test_result))
{
            $words      = [];
            $delim      = "
\r\t\"*''/;:[]{}+-#$+=_
()?></~`!%:@1234567890#&"."";
            $tok  =
strtok($row[3], $delim);

            while ($tok !==
false) {
              $words[] = $tok;
              $tok =
strtok($delim);
            }
            $sentences =
array_values($words);
            $panjang_array =
count($sentences);
            $bersih = implode("
", $sentences);
            $query_parts[] =
"('" . $row[0]."', '" .
strtolower($bersih). "')";
        }
        //QUERY INSERT TESTING
DATA
        $insPrep = "INSERT INTO
prep_testing
(idtesting_komentar,
komentesting_casefold) VALUES
".implode(', ', $query_parts);

if(mysqli_query($link,
$insPrep)){
            echo "Records
updated successfully.";
          }
          else
          {
            echo "ERROR: Could
not able to execute $insPrep. "
. mysqli_error($link);
          }
      }
      else {
      }
    ?>
```

```
    <div class="container">
      <header>
        <h1 class="text-
muted">Cleanse</h1>
      </header>
      <div class="training-
table">
        <h4>Training
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
            </tr>
          </thead>
          <tbody >
            <?php
              $result =
mysqli_query($link, "SELECT *
FROM prep_training");
              while
($row=mysqli_fetch_row($result)
)
              {
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
komentar = ".$train_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
            <a
href="?train_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>

            <?php } ?>
        </div>
      </div>
```

```
      <div class="testing-table
clearfix">
        <h4>Testing
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
            </tr>
          </thead>
          <tbody >
            <?php
              $result =
mysqli_query($link, "SELECT *
FROM prep_testing");
              while
($row=mysqli_fetch_row($result)
)
              {
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
komentar = ".$test_total; ?>

        <div
class="pagination">
          <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
          <a
href="?test_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>
          <?php } ?>
        </div>
      </div>
      <footer>
        <a
href="stemming.php">next</a>
        <a
href="export.php">export
data</a>
      </footer>
    </div>
    <!-- Op tional JavaScript -
->
  </body>
  <?php mysqli_close($link); ?>
</html>
```

Source Code Stemming

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
    </title>
    <link rel="stylesheet"
href="css/style.css">

  </head>
    <!-- connect database -->
    <?php
      $link =
mysqli_connect('localhost','roo
t','root','sentipol2' );
      if($link)
      {
      }
      else
      {
        echo "koneksi database
gagal.";
      }
    ?>
```

```
  <body>
    <?php
      require_once __DIR__ .
'/sastrawi-
1.1.0/vendor/autoload.php';

      // create stemmer
      // cukup dijalankan
sekali saja, biasanya
didaftarkan di service
container
      $stemmerFactory = new
\Sastrawi\Stemmer\StemmerFactor
y();
      $stemmer  =
$stemmerFactory-
>createStemmer();

      //PAGINATION TRAINING
      $train_halaman = 60;
      $train_page =
isset($_GET["train_halaman"]) ?
(int)$_GET["train_halaman"] :
1;
      $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
      $train_result =
mysqli_query($link, "SELECT *
FROM prep_training");
      $train_total =
mysqli_num_rows($train_result);
      $train_pages =
ceil($train_total/$train_halama
n);
      $train_query =
mysqli_query($link, "SELECT *
FROM prep_training LIMIT
$train_mulai, $train_halaman");
      $train_no
=$train_mulai+1;

      //STEMMING TRAINING
      if ($train_total==0) {
        $query_parts = array();
        while ($row =
mysqli_fetch_row($train_result)
){
            // stem
            $sentence =
$row[1];
            $output   =
$stemmer->stem($sentence);
            $query_parts[] =
"('" .$row[0]."', '" .$output.
"')";

        }
        $insPrep = "INSERT INTO
stem_training (idtraining_stem,
training_stemmed) VALUES
".implode(', ', $query_parts);

if(mysqli_query($link,
$insPrep)){
            echo "Records
updated successfully.";
          }
          else
          {
            echo "ERROR: Could
not able to execute $insPrep. "
. mysqli_error($link);
          }
        }
      else {
      }

      //PAGINATION TESTING
      $test_halaman = 60;
      $test_page =
isset($_GET["test_halaman"]) ?
(int)$_GET["test_halaman"] : 1;
      $test_mulai =
($test_page>1) ? ($test_page *
$test_halaman) - $test_halaman
: 0;
      $test_result =
mysqli_query($link, "SELECT *
FROM prep_testing");
      $test_total =
mysqli_num_rows($test_result);
      $test_pages =
ceil($test_total/$test_halaman)
;
      $test_query =
mysqli_query($link, "SELECT *
FROM prep_testing LIMIT
$test_mulai, $test_halaman");
      $test_no =$test_mulai+1;

      //STEMMING TESTING
      if ($test_total==0) {
```

```
        $query_parts = array();
        while ($row =
mysqli_fetch_row($test_result))
{
            // stem
            $sentence =
$row[1];
            $output   =
$stemmer->stem($sentence);
            $query_parts[] =
"('" .$row[0]."', '" .$output.
"')";
        }
        $query = "INSERT INTO
stem_testing (idtesting_stem,
testing_stemmed) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$query)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute $query. " .
mysqli_error($link);
        }
      }
      else {
      }

     ?>

    <div class="container">
      <header>
        <h1 class="text-
muted">Stemming</h1>
      </header>

      <div class="training-
table">
        <h4>Training
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
            </tr>
          </thead>
          <tbody >
            <?php
              $result =
mysqli_query($link, "SELECT *
FROM stem_training");
              while
($row=mysqli_fetch_row($result)
)
              {
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
komentar = ".$train_total; ?>
        <div
class="pagination">
          <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
          <a
href="?train_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>

          <?php } ?>
        </div>
      </div>
      <div class="testing-table
clearfix">
        <h4>Testing
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
            </tr>
          </thead>
          <tbody >
            <?php
              $result =
mysqli_query($link, "SELECT *
FROM stem_testing");
```

```
              while
($row=mysqli_fetch_row($result)
)
              {
             ?>
                 <tr>
                   <td><?php
echo "$row[0]"; ?></td>
                   <td><?php
echo "$row[1]"; ?></td>
                 </tr>
             <?php
               }
             ?>
           </tbody>
         </table>
         <?php echo "jumlah
komentar = ".$test_total; ?>
         <div
class="pagination">
             <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
             <a
href="?test_halaman=<?php echo
$i; ?>"><?php echo $i; ?></a>

             <?php } ?>
         </div>
       </div>
     </div>
     <footer>
         <a
href="tokenization.php">next</a
>
     </footer>
   </div>

     <!-- Op tional JavaScript -
->
   </body>
</html>
```

Source Code Tokenization

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
    </title>
    <link rel="stylesheet"
href="css/style.css">
  </head>

  <!-- connect database -->
  <?php
  $link =
mysqli_connect('localhost','roo
t','root','sentipol2');
  if($link)
  {
  }
  else
  {
    echo "koneksi database
gagal.";
  }
  ?>

  <body>
    <?php
      //PAGINATION TRAINING
      $train_halaman = 500;
      $train_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
      $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
      $train_tampilin =
mysqli_query($link, "SELECT *
FROM tokenization_training");
      $train_total =
mysqli_num_rows($train_tampilin
);
      $train_pages =
ceil($train_total/$train_halama
n);
      $train_query =
mysqli_query($link, "SELECT *
FROM tokenization_training
LIMIT $train_mulai,
$train_halaman");

      //PAGINATION TESTING
      $test_halaman = 300;
      $test_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
      $test_mulai =
($test_page>1) ? ($test_page *
```

```
$test_halaman) - $test_halaman
: 0;
      $test_tampilin =
mysqli_query($link, "SELECT *
FROM tokenization_testing");
      $test_total =
mysqli_num_rows($test_tampilin)
;
      $test_pages =
ceil($test_total/$test_halaman)
;
      $test_query =
mysqli_query($link, "SELECT *
FROM tokenization_testing LIMIT
$test_mulai, $test_halaman");

      //TOKENIZATION TRAINING
DOKUMEN
      if ($train_total == 0 &&
$test_total == 0) {
        $query_parts = array();
        $train_dokumen =
mysqli_query($link, "SELECT *
FROM stem_training");
        while ($row =
mysqli_fetch_row($train_dokumen
)) {

          $words  = [];
          $delim  = " \t\n
"."";
          $tok    =
strtok($row[1], $delim);

          while ($tok !==
false) {
            $words[] = $tok;
            $tok =
strtok($delim);
          }
          $values =
array_values($words);

          foreach($values as
$value){
            $query_parts[] =
"('" .$row[0]."', '" .$value.
"')";
          }
        }
        $insPrep =  "INSERT
INTO tokenization_training
(idtraining_komentar,
training_token) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$insPrep)){
          echo "Records added
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute " .
mysqli_error($link);
        }

        //TOKENIZATION TESTING
DOKUMEN
        $query_parts = array();
        $test_dokumen =
mysqli_query($link, "SELECT *
FROM stem_testing");
        while ($row =
mysqli_fetch_row($test_dokumen)
) {
          $words      = [];
          $delim      = "
\t\n "."";
          $tok  =
strtok($row[1], $delim);

          while ($tok !==
false) {
            $words[] = $tok;
            $tok =
strtok($delim);
          }
          $values =
array_values($words);

          foreach($values as
$value){
            $query_parts[] =
"('" .$row[0]."', '" .$value.
"')";
          }
        }
        $insPrep =  "INSERT
INTO tokenization_testing
(idtesting_komentar,
testing_token) VALUES
".implode(', ', $query_parts);
```

```
if(mysqli_query($link,
$insPrep)){
            echo "Records added
successfully.";
          }
          else
          {
            echo "ERROR: Could
not able to execute " .
mysqli_error($link);
          }
      }
      else {
      }

    ?>

    <div class="container">
      <header>
        <h1 class="text-
muted">Tokenization</h1>
      </header>
      <div class="training-
table">
        <h4>Training
Document</h4>
        <table>
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
              <th>Label</th>
            </tr>
          </thead>
          <tbody >
            <?php
              while($row =
mysqli_fetch_row($train_query))
{
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                  <td><?php
echo "$row[2]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
token = ".$train_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
            <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
            <?php } ?>
        </div>
      </div>

    <div class="testing-table
clearfix">
      <h4>Testing
Document</h4>
      <table class="fixed-
headers">
        <thead>
          <tr>
            <th>ID</th>
            <th>Komentar</th>
            <th>Label</th>
          </tr>
        </thead>
        <tbody >
          <?php
            while($row =
mysqli_fetch_row($test_query)){
          ?>
              <tr>
                <td><?php
echo "$row[0]"; ?></td>
                <td><?php
echo "$row[1]"; ?></td>
                <td><?php
echo "$row[2]"; ?></td>
              </tr>
          <?php
            }
          ?>
        </tbody>
      </table>
      <?php echo "jumlah
token = ".$test_total; ?>
      <div
class="pagination">
          <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
```

```
            <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
            <?php } ?>
          </div>
        </div>
        <footer>
          <a
href="detectkata.php">next</a>
        </footer>
      </div>
      <!-- Op tional JavaScript -
->
    </body>
</html>
```

Source Code detectkata

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
    </title>
    <link rel="stylesheet"
href="css/style.css">
  </head>

  <!-- connect database -->
  <?php
  $link =
mysqli_connect('localhost','roo
t','root','sentipol2');
  if($link)
  {
  }
  else
  {
    echo "koneksi database
gagal.";
  }
  ?>

  <body>
    <?php
      //PAGINATION TRAINING
      $train_halaman = 500;
      $train_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
      $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
      $train_tampilin =
mysqli_query($link, "SELECT *
FROM tokenization_training");
      $train_total =
mysqli_num_rows($train_tampilin
);
      $train_pages =
ceil($train_total/$train_halama
n);
      $train_query =
mysqli_query($link, "SELECT *
FROM tokenization_training
LIMIT $train_mulai,
$train_halaman");

      //PAGINATION TESTING
      $test_halaman = 300;
      $test_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
      $test_mulai =
($test_page>1) ? ($test_page *
$test_halaman) - $test_halaman
: 0;
      $test_tampilin =
mysqli_query($link, "SELECT *
FROM tokenization_testing");
      $test_total =
mysqli_num_rows($test_tampilin)
;
      $test_pages =
ceil($test_total/$test_halaman)
;
      $test_query =
mysqli_query($link, "SELECT *
FROM tokenization_testing LIMIT
$test_mulai, $test_halaman");

      //TOKENIZATION TRAINING
DOKUMEN
      if ($train_total == 0 &&
$test_total == 0) {
        $query_parts = array();
        $train_dokumen =
mysqli_query($link, "SELECT *
FROM stem_training");
```

```
        while ($row =
mysqli_fetch_row($train_dokumen
)) {

          $words  = [];
          $delim  = " \t\n
".","";
          $tok    =
strtok($row[1], $delim);

          while ($tok !==
false) {
            $words[] = $tok;
            $tok =
strtok($delim);
          }
          $values =
array_values($words);

          foreach($values as
$value){
            $query_parts[] =
"('" .$row[0]."', '" .$value.
"')";
          }
        }
        $insPrep =  "INSERT
INTO tokenization_training
(idtraining_komentar,
training_token) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$insPrep)){
          echo "Records added
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute " .
mysqli_error($link);
        }

        //TOKENIZATION TESTING
DOKUMEN
        $query_parts = array();
        $test_dokumen =
mysqli_query($link, "SELECT *
FROM stem_testing");
          while ($row =
mysqli_fetch_row($test_dokumen)
) {
            $words      = [];
            $delim      = "
\t\n ".","";
            $tok  =
strtok($row[1], $delim);

            while ($tok !==
false) {
              $words[] = $tok;
              $tok =
strtok($delim);
            }
            $values =
array_values($words);

            foreach($values as
$value){
              $query_parts[] =
"('" .$row[0]."', '" .$value.
"')";
            }
          }
          $insPrep =  "INSERT
INTO tokenization_testing
(idtesting_komentar,
testing_token) VALUES
".implode(', ', $query_parts);

if(mysqli_query($link,
$insPrep)){
            echo "Records added
successfully.";
          }
          else
          {
            echo "ERROR: Could
not able to execute " .
mysqli_error($link);
          }
      }
      else {
      }

    ?>

    <div class="container">
      <header>
        <h1 class="text-
muted">Tokenization</h1>
      </header>
      <div class="training-
table">
```

```
        <h4>Training
Document</h4>
        <table>
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
              <th>Label</th>
            </tr>
          </thead>
          <tbody >
            <?php
              while($row =
mysqli_fetch_row($train_query))
{
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                  <td><?php
echo "$row[2]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
token = ".$train_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
            <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
            <?php } ?>
          </div>
        </div>


      <div class="testing-table
clearfix">
        <h4>Testing
Document</h4>
        <table class="fixed-
headers">
          <thead>
            <tr>
              <th>ID</th>
              <th>Komentar</th>
              <th>Label</th>
            </tr>
          </thead>
          <tbody >
            <?php
              while($row =
mysqli_fetch_row($test_query)){
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                  <td><?php
echo "$row[2]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
token = ".$test_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
            <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
            <?php } ?>
          </div>
        </div>
        <footer>
          <a
href="detectkata.php">next</a>
        </footer>
    </div>
    <!-- Op tional JavaScript -
->
  </body>
</html>
```

Source Code Stopword Removal

```
<!-- jQuery first, then
Popper.js, then Bootstrap JS --
>
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>
```

```
    </title>
    <link rel="stylesheet"
href="css/style.css">
  </head>
  <!-- connect database -->
  <?php
  $link =
mysqli_connect('localhost','roo
t','root','sentipol2' );
  if($link)
  {
  }
  else
  {
    echo "koneksi database
gagal.";
  }
  ?>
  <body>

    <?php
      //TRAINING
        $result =
mysqli_query($link,"SELECT *
FROM tokenization_training");
        $query_parts = array();
        while($row =
mysqli_fetch_row($result)){
          $cocok  =
mysqli_query($link,"SELECT
stoplist FROM stoplist WHERE
stoplist = '$row[2]'");
          $tampung      =
mysqli_num_rows($cocok);
        if ($tampung > 0) {
        }
        else {

          $query_parts[] =
"('" .$row[1]."','"
.$row[2]."')";
          }
          //bikin query buat
nyimpan data kalo isi array
tidak NULL
        }
        $insPrep = "INSERT INTO
abisdistopword_training
(flagtraining_id_komentar,
training_token) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$insPrep)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute $insPrep. "
. mysqli_error($link);
        }

      //TESTING
        $result =
mysqli_query($link,"SELECT *
FROM tokenization_testing");
        $query_parts = array();
        while($row =
mysqli_fetch_row($result)){
          $cocok  =
mysqli_query($link,"SELECT
stoplist FROM stoplist WHERE
stoplist = '$row[2]'");
          $tampung      =
mysqli_num_rows($cocok);
          if ($tampung > 0) {
          }
          else {

            $query_parts[] =
"('" .$row[1]."','"
.$row[2]."')";
          }
          //bikin query buat
nyimpan data kalo isi array
tidak NULL
        }
        $insPrep = "INSERT INTO
abisdistopword_testing
(flagtesting_id_komentar,
testing_token) VALUES
".implode(', ', $query_parts);
        if(mysqli_query($link,
$insPrep)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute $insPrep. "
. mysqli_error($link);
```

```
        }
    ?>

    <div class="container">
      <header>
        <h1 class="text-
muted">Stopword Removal</h1>
      </header>
      <div class="training-
table">
        <h4>Training
Document</h4>
         <table>
           <thead>
             <tr>
               <th>ID</th>

<th>Komentar</th>
               <th>Label</th>
             </tr>
           </thead>
           <tbody >
           <?php
             $train_halaman =
500;
             $train_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
             $train_mulai =
($train_page>1) ? ($train_page
* $train_halaman) -
$train_halaman : 0;
             $train_tampilin =
mysqli_query($link, "SELECT *
FROM abisdistopword_training");
             $train_total =
mysqli_num_rows($train_tampilin
);
             $train_pages =
ceil($train_total/$train_halama
n);
             $train_query =
mysqli_query($link, "SELECT *
FROM abisdistopword_training
LIMIT $train_mulai,
$train_halaman");

             while($row =
mysqli_fetch_row($train_query))
{
           ?>
               <tr>
                 <td><?php
echo "$row[0]"; ?></td>
                 <td><?php
echo "$row[1]"; ?></td>
                 <td><?php
echo "$row[2]"; ?></td>
               </tr>
           <?php
             }
           ?>
         </tbody>
       </table>
       <?php echo "jumlah
token = ".$train_total; ?>
       <div
class="pagination">
         <?php for ($i=1;
$i<=$train_pages ; $i++){ ?>
           <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
         <?php } ?>
       </div>
     </div>

       <div class="testing-
table clearfix">
         <h4>Testing
Document</h4>
         <table class="fixed-
headers">
         <thead>
           <tr>
             <th>ID</th>
             <th>Komentar</th>
             <th>Label</th>
           </tr>
         </thead>
         <tbody >
           <?php
             $test_halaman =
300;
             $test_page =
isset($_GET["halaman"]) ?
(int)$_GET["halaman"] : 1;
             $test_mulai =
($test_page>1) ? ($test_page *
$test_halaman) - $test_halaman
: 0;
             $test_tampilin =
mysqli_query($link, "SELECT *
FROM abisdistopword_testing");
```

```
              $test_total =
mysqli_num_rows($test_tampilin)
;
              $test_pages =
ceil($test_total/$test_halaman)
;
              $test_query =
mysqli_query($link, "SELECT *
FROM abisdistopword_testing
LIMIT $test_mulai,
$test_halaman");

              while($row =
mysqli_fetch_row($test_query)){
            ?>
                <tr>
                  <td><?php
echo "$row[0]"; ?></td>
                  <td><?php
echo "$row[1]"; ?></td>
                  <td><?php
echo "$row[2]"; ?></td>
                </tr>
            <?php
              }
            ?>
          </tbody>
        </table>
        <?php echo "jumlah
token = ".$test_total; ?>
        <div
class="pagination">
            <?php for ($i=1;
$i<=$test_pages ; $i++){ ?>
              <a
href="?halaman=<?php echo $i;
?>"><?php echo $i; ?></a>
            <?php } ?>
        </div>
      </div>
    <!-- Op tional JavaScript -
->
        <footer>
          <a
href="tfidfrevisi.php">next</a>
        </footer>
  </div>
    <!-- Op tional JavaScript -
->
  </body>
</html>
```

Source Code TF-IDF revisi

```
// NILAI-NILAI TRAINING
      $result_train     =
mysqli_query($link,"SELECT *
FROM tfidf_training");
      //NILAI N
      $total_seluruh_train =
mysqli_query($link,"SELECT
DISTINCT
flagtraining_id_komentar FROM
abisdistopword_training");
      $N_train =
mysqli_num_rows($total_seluruh_
train);

      while($row =
mysqli_fetch_row($result_train)
){
        $ambiltf_train =
mysqli_query($link,"SELECT
COUNT(training_token) AS ambil
FROM abisdistopword_training
          WHERE
training_token='$row[2]' AND
flagtraining_id_komentar='$row[
1]'"); //tf
        $ambildf_train =
mysqli_query($link,"SELECT
COUNT(DISTINCT
flagtraining_id_komentar) AS
ambil
          FROM
abisdistopword_training WHERE
training_token='$row[2]'");
//df
        $ambiltoken_train =
mysqli_query($link,"SELECT
COUNT(training_token) AS ambil
         FROM
abisdistopword_training WHERE
flagtraining_id_komentar='$row[
1]'");//ambil jumlah token di
sebuah kalimat

        $tf_train =
mysqli_fetch_assoc($ambiltf_tra
in);
        $df_train =
mysqli_fetch_assoc($ambildf_tra
in);
```

```
        $sum_token_train =
mysqli_fetch_assoc($ambiltoken_
train);

        $nilaitf_train =
$tf_train['ambil'];
        $tfnorm_train =
$nilaitf_train/$sum_token_train
['ambil'];
        $nilaidf_train =
$df_train['ambil'];
        $idf_train =
abs(log10($N_train/$nilaidf_tra
in));
        $tfidf_train =
abs($tfnorm_train*$idf_train) ;

        $insPrep_train =
"UPDATE tfidf_training SET
tf=$nilaitf_train,
tf_normalisasi=$tfnorm_train,
df=$nilaidf_train,
          idf=$idf_train,
tfidf=$tfidf_train WHERE
token='$row[2]' AND
idtrain_komentar='$row[1]'";
        if(mysqli_query($link,
$insPrep_train)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute
$insPrep_train. " .
mysqli_error($link);
        }
      }

  // NILAI-NILAI TESTING
      $result_test      =
mysqli_query($link,"SELECT *
FROM tfidf_testing");
      //NILAI N
      $total_seluruh_test =
mysqli_query($link,"SELECT
DISTINCT
flagtesting_id_komentar FROM
abisdistopword_testing");

      $N_test =
mysqli_num_rows($total_seluruh_
test);

      while($row =
mysqli_fetch_row($result_test))
{
        $ambiltf_test =
mysqli_query($link,"SELECT
COUNT(testing_token) AS ambil
FROM abisdistopword_testing
          WHERE
testing_token='$row[2]' AND
flagtesting_id_komentar='$row[1
]'"); //tf
        $ambildf_test =
mysqli_query($link,"SELECT
COUNT(DISTINCT
flagtesting_id_komentar) AS
ambil
          FROM
abisdistopword_testing WHERE
testing_token='$row[2]'"); //df
        $ambiltoken_test =
mysqli_query($link,"SELECT
COUNT(testing_token) AS ambil
          FROM
abisdistopword_testing WHERE
flagtesting_id_komentar='$row[1
]'");//ambil jumlah token di
sebuah kalimat

        $tf_test =
mysqli_fetch_assoc($ambiltf_tes
t);
        $df_test =
mysqli_fetch_assoc($ambildf_tes
t);
        $sum_token_test =
mysqli_fetch_assoc($ambiltoken_
test);

        $nilaitf_test =
$tf_test['ambil'];
        $tfnorm_test =
$nilaitf_test/$sum_token_test['
ambil'];
        $nilaidf_test =
$df_test['ambil'];
        $idf_test =
abs(log10($N_test/$nilaidf_test
));
```

```
        $tfidf_test =
abs($tfnorm_test*$idf_test) ;

        $insPrep_test = "UPDATE
tfidf_testing SET
tf=$nilaitf_test,
tf_normalisasi=$tfnorm_test,
df=$nilaidf_test,
          idf=$idf_test,
tfidf=$tfidf_test WHERE
token='$row[2]' AND
idtest_komentar='$row[1]'";

        if(mysqli_query($link,
$insPrep_test)){
          echo "Records updated
successfully.";
        }
        else
        {
          echo "ERROR: Could
not able to execute
$insPrep_test. " .
mysqli_error($link);
        }
      }
```