BAB 3 ANALISIS DAN PERANCANGAN

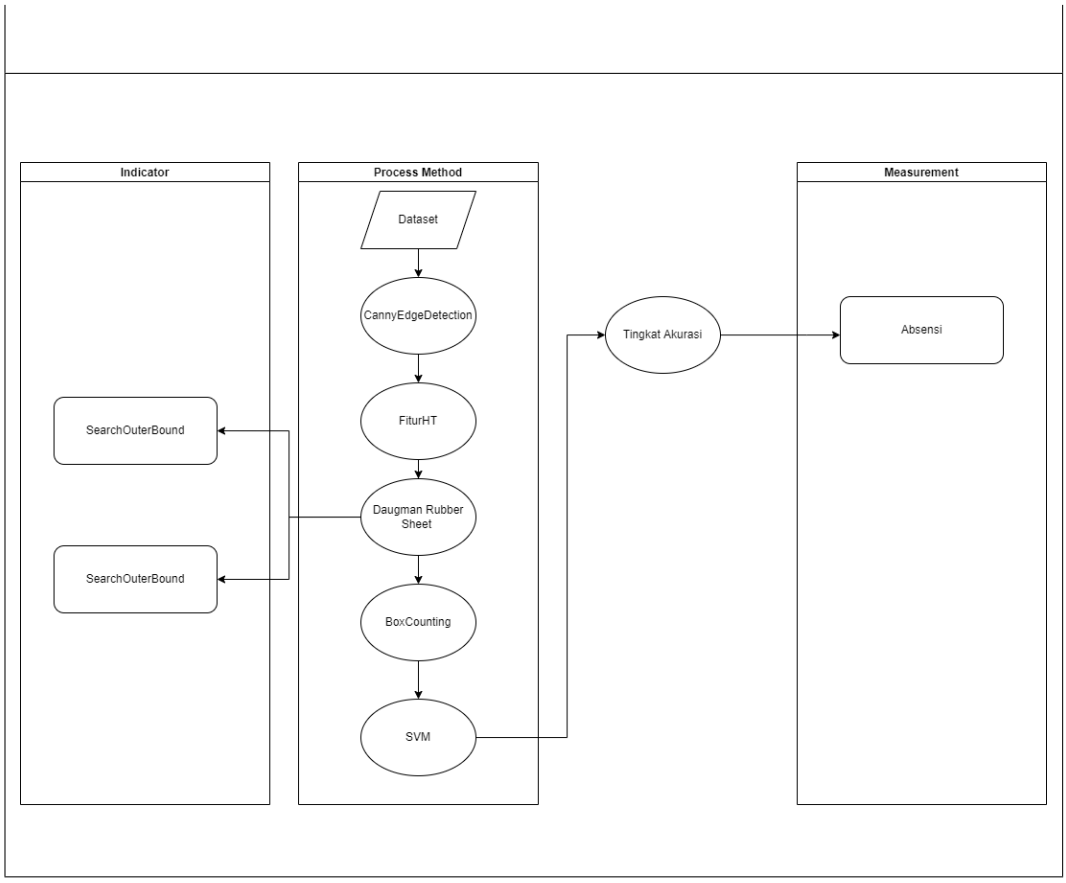
Pada bab ini menjelaskan mengenai analisis dan perancangan system yang akan dikembangkan oleh penguji

3.1 Analisis Masalah

Iris atau selaput pelangi pada mata dapat dijadikan sebagai basis sistem biometrik. Setiap iris memiliki tekstur yang amat rinci dan unik untuk setiap orang serta tidak akan berubah selama hidup seseorang.Penelitian ini membahas tentang bagaimana penerapan metode SVM untuk melakukan pemodelan serta mengindentifikasi iris mata. Masukan yang digunakan berupa gambar[2].Ekstraksi fitur pada penelitian ini menggunakan method Box-Counting yang terdapat di Tabel 2.6. Penelitian lain menyebutkan bahwa metode K-NN tersebut sudah terbukti dapat menghasilkan akurasi yang tinggi.Irish yang didapat dari perhitungan menggunakan metode Box-counting akan dibandingkan dengan yang ada didatabase dan setelah itu akan ada counter yang bertambah,sehingga tidak dapat terjadi kecurangan saat mengabsensi.

3.2 Kerangka Pemikiran

Kerangka pemikiran untuk ekstraksi fitur, pemodelan, pengklasifikasian dapat dilihat pada Gambar kerangka pemikiran ini menunjukkan apa saja yang dipakai untuk setiap tahap yang dilakukan, Box-counting akan menghasilkan fitur. Fitur dari method Box-Counting tersebut juga dipakai untuk pemodelan yang nantinya akan dipakai untuk menghasilkan fitur fitur yang terdapat di iris.



Gambar 3.1 Kerangka Pemikiran

Gambar 3.1 menunjukkan kerangka pemikiran dari metode yang diusulkan untuk sistem klasifikasi penutur. Lebih jelasnya akan dijelaskan sebagai berikut :

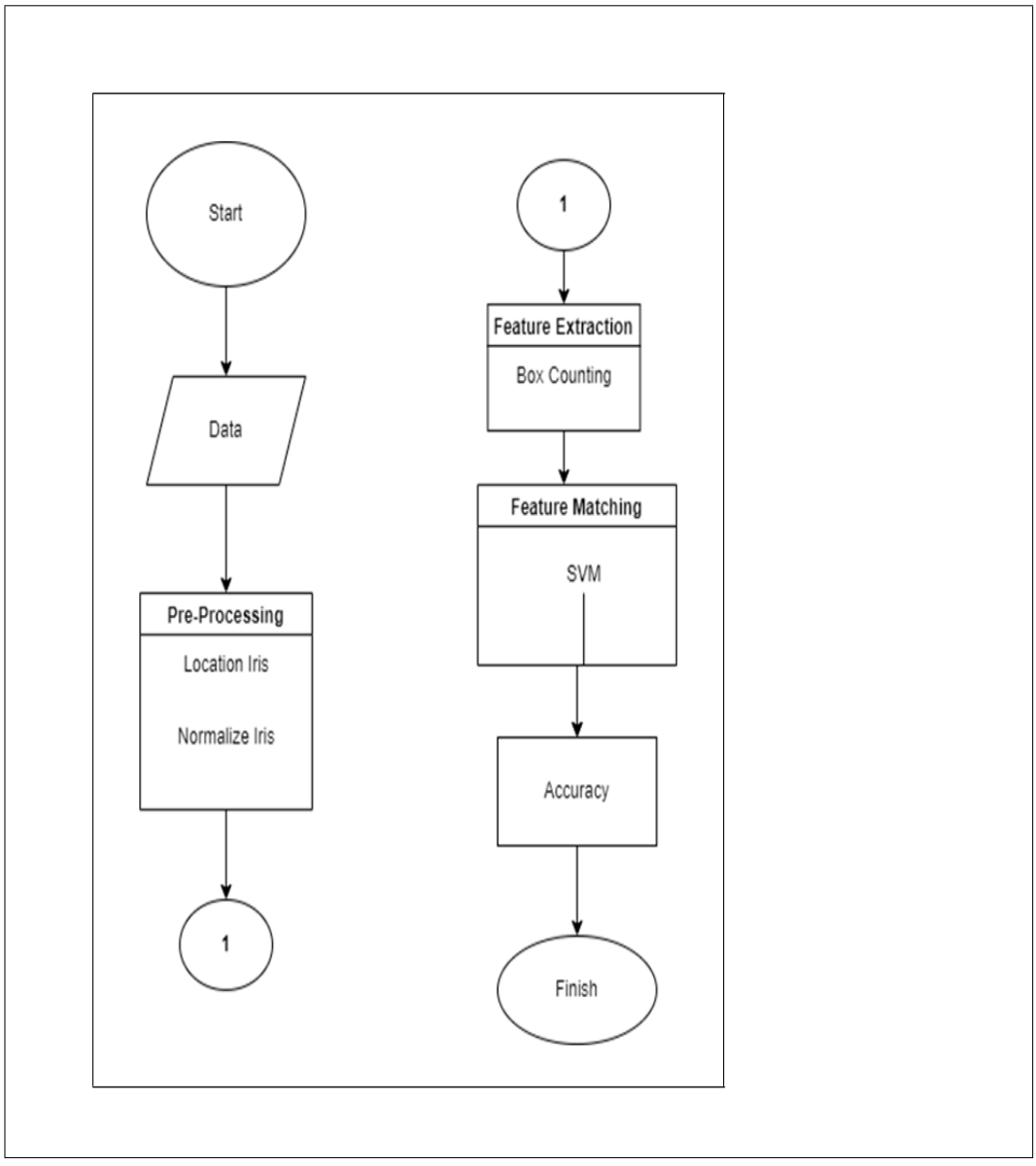
1. SearchOuterBound = digunakan untuk mencari titik terluar dari gambar
2. CircleCoordinate = letak koordinat dimana iris berada

Gambar menunjukkan kerangka pemikiran dari metode yang diusulkan untuk sistem identifikasi iris. Tahapannya akan dijelaskan sebagai berikut :

1. Masukan sample citra mata.
2. Proses *Pre-Processing* Localize iris.Tujuannya adalah menemukan tempat iris berada menggunakan *Daugman Rubber Sheet*.
3. Lakukan ekstraksi fitur menggunakan method Box-Counting().
4. *Support Vector Machine* agar hasil ekstraksi fitur yang digunakan untuk proses klasifikasi seragam.

3.3 Urutan Proses Global

Urutan proses global yang dilakukan meliputi proses *training* dan proses *testing*. Proses *training* akan menyimpan model yang telah diberikan label, sementara proses *testing* akan mencocokkan masukan dengan model yang telah disimpan oleh proses *training*. Urutan proses global dari metode yang diusulkan untuk proses *training* dapat dilihat pada Gambar 3.2



Gambar 3.2 Urutan Proses Global

Gambar 3.2 menunjukkan urutan proses global *train* dari sistem klasifikasi penutur, lebih jelasnya akan dijelaskan sebagai berikut :

1. Masukan sample citra mata.
2. Proses Pre-Processing Localize iris. Tujuannya adalah menemukan tempat iris berada menggunakan Hough Transform.
3. Normalisasi menggunakan Daugman Rubber Sheet untuk mengidentifikasi lingkaran mata
4. Support Vector Machine agar hasil ekstraksi fitur yang digunakan untuk proses klasifikasi seragam.
5. Lakukan ekstraksi fitur menggunakan method Box-Counting().

3.4 Proses Perhitungan Ekstraksi Fitur iris

Pada bagian ini akan dibahas mengenai proses perhitungan ekstraksi fitur pada citra iris mulai dari tahapan pre-processing hingga mendapatkan fitur morfologi dan fitur tekstur.

3.5 *Preprocessing*

Proses Canny Edge Detection Canny Edge Detection digunakan untuk mencari batas batas dari object gambar lalu menggunakan Hough transform untuk mencari bentuk lingkaran yang terdapat pada object setelah itu menggunakan method box counting untuk mencari fitur fitur yang ada di objek tersebut

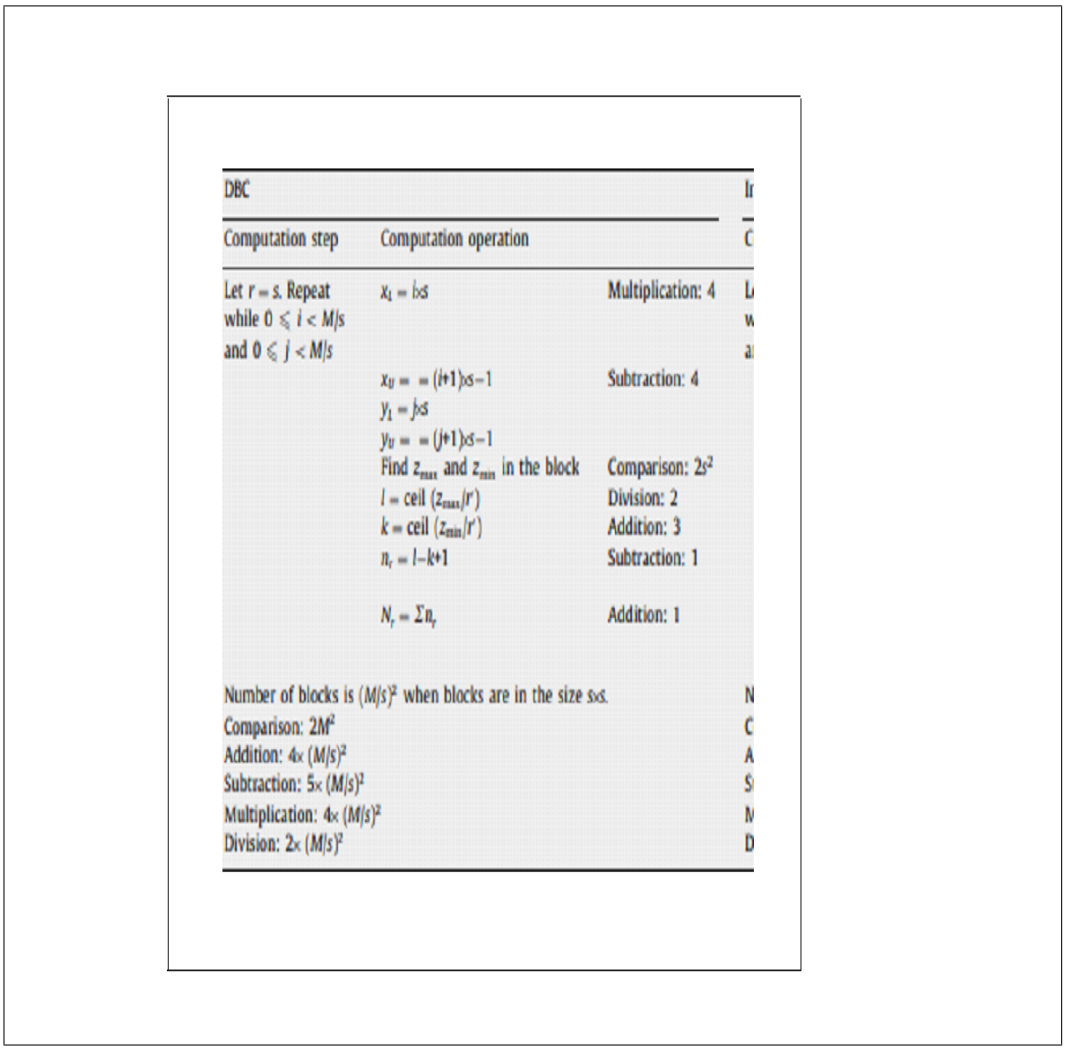
Hough Transform Circle Bekerja dengan cara yang hampir sama dengan Transformasi Garis Hough yang dijelaskan dalam tutorial sebelumnya.Dalam kasus pendeteksian garis, sebuah garis didefinisikan oleh dua parameter (r,).Dalam kasus lingkaran, kita membutuhkan tiga parameter untuk mendefinisikan lingkaran C : (xcenter, ycenter, r) di mana (xcenter, ycenter) menentukan posisi pusat (titik hijau) dan r adalah jari-jarinya, yang memungkinkan kita untuk sepenuhnya mendefinisikan lingkaran. nilai radius yang didapat disini adalah lebarnya bagian mata. Hough radius yang digunakan adalah sebesar 64 dan 320

3.6 Algoritme Box Counting

Fitur fitur yang dicari didapat dari Box-Counting didapat mengikuti langkah-langkah berikut: Algoritme 3 Box Counting

1. Menginput dari gambar mata yang sudah diproses oleh Canny Edge Detection sebelumnya
2. Masukkan ke method Hough Transform Circle untuk mencari batas-batas lingkarannya
3. Masukkan Ke dalam proses *Daugman Rubber Sheet* untuk mendapatkan lingkaran iris dan koordinat iris dari *Citra* tersebuts
4. Masukkan ke method Box Counting untuk proses pengambilan fitur

berikut merupakan gambaran psudocode dari penggunaan Box-counting 4.1



DBC

Computation step	Computation operation	
Let $r = s$. Repeat while $0 \leqslant i < M/s$ and $0 \leqslant j < M/s$	$x_L = i \times s$	Multiplication: 4
	$x_U = = (i+1) \times s - 1$	Subtraction: 4
	$y_L = j \times s$	
	$y_U = = (j+1) \times s - 1$	
	Find z_{max} and z_{min} in the block	Comparison: $2s^2$
	$l = \text{ceil}\ (z_{max}/r)$	Division: 2
	$k = \text{ceil}\ (z_{min}/r)$	Addition: 3
	$n_r = l - k + 1$	Subtraction: 1
	$N_r = \Sigma n_r$	Addition: 1

Number of blocks is $(M/s)^2$ when blocks are in the size $s \times s$.
Comparison: $2M^2$
Addition: $4 \times (M/s)^2$
Subtraction: $5 \times (M/s)^2$
Multiplication: $4 \times (M/s)^2$
Division: $2 \times (M/s)^2$

Gambar 3.3 Box Counting Algoritme

[26]

Algoritme Daugman Rubber Sheet Model Bekerja dengan cara mengambil *Citra* iris dan memprosesnya menggunakan Hough transform model dan canny edge detection agar mendapatkan koordinat dari iris tersebut.

3.7 Algoritme Support Vector Machine

Support Vektor Machine fitur yang sudah didapat dari HT dan Box-Counting akan digunakan untuk training dengan menggunakan model SVM yang sudah ada. Klasifikasi SVM dilakukan pada masing-masing round batas mata. Jika jumlah round yang memiliki mata, maka akan membatasi pencarian mata. Dengan melakukan ini, akan didapatkan estimasi akurat iris saat dalam keadaan tertentu.

Pada tahap training, nilai SVM dihitung pada setiap cell dan membutuhkan parameter c, yang pada penelitian ini bernilai 2. Setelah dihitung nilai SVMnya. Dan pada tahap testing, model SVM akan di test dengan menggunakan Java untuk mengetahui seberapa besar akurasi yang didapat. Berikut langkah-langkah algoritme Support Vector Machine :

Algoritme Support Vector Machine

1. Mencari fitur dari mata menggunakan box-counting
2. Masukkan ke method Support Vector Machine
3. Menggabung fitur yang didapat dari Box Counting dan RBF menjadi sebuah vector
4. mencari batasannya
5. jika sudah melebihi batas maka itu diluar mata
6. pada akhir maka akan didapat hasil berupa confusion matrix dan berapa besar akurasi yang didapat pada saat melakukan testing

Inputnya berupa hasil ekstraksi *Feature* dari *box-counting* lalu menghasilkan output berupa seberapa besar akurasi yang didapat.

3.8 Analisis Proses Manual

Berikut merupakan uraian proses dari data masukan, ekstraksi fitur, pemodelan, klasifikasi citra mata menggunakan metode *Support Vector Machine*.

3.8.1 Data Masukan

Data citra yang akan dianalisis manual adalah citra mata yang diambil dari depan dengan ukuran beragam. menggambarkan setiap pixel sebagai integer, nilai dari 8 bit adalah antara -255 sampai 255. Kuantisasi didapat dengan cara jumlah frame yang disarankan sekitar 77-94.

Data masukan yang digunakan untuk klasifikasi adalah berupa berkas yang berformat *Comma Separated Value (CSV)*. Berkas tersebut berupa data mata dari dataset yang dimana setiap citra memiliki 1 citra mata kiri/kanan, gambar pertama hingga gambar ke tiga akan dijadikan train sedangkan gambar ke empat dan lima akan dijadikan data test.

3.8.2 Ekstraksi Fitur

Ekstraksi fitur menggunakan *method* Box Counting(FracLac). Parameter yang digunakan dalam penelitian ini dapat terdapat *x*, *y*, *ios*,*o*,*s*

Parameter pada Tabel adalah parameter yang digunakan untuk ekstraksi fitur, dikarenakan :

1. x range block tertentu pada citraFrame .
2. y range block tertentu pada citraFrame.
3. ios range block tertentu pada citra.
4. o range block tertentu pada citra.
5. s range block tertentu pada citra.

Tabel 3.2 berikut merupakan matriks fitur dengan total fitur sebanyak 10 buah untuk contoh perhitungan.

Tabel 3.1 Hasil Ekstraksi Fitur Metode Box Counting

No.	**F1**	**F2**	**F3**	**...**	**F7**	**F8**	**F9**	**F10**
A1	1.6178	0.9927	0.0544	**...**	0.9927	0.0544	5.1789	0.0056
A2	1.5544	0.9865	0.066	**...**	0.9865	0.066	4.8734	0.0076

Tabel 3.1 Hasil Ekstraksi Fitur Metode Box Counting

No.	**F1**	**F2**	**F3**	**...**	**F7**	**F8**	**F9**	**F10**
A3	1.5092	0.9823	0.0701	**...**	0.9823	0.0701	4.7666	0.0085
A4	1.4343	0.9805	0.0626	**...**	0.9805	0.0626	4.8744	0.0076

Tabel 3.2 menunjukkan hasil ekstraksi fitur yang dilakukan oleh *BoxCounting* yang dapat dilihat pada Tabel diatas. Hasil ekstraksi memiliki 10 fitur.

3.8.3 Klasifikasi SVM

Proses klasifikasi dilakukan dengan metode *Support Vector Machine* dengan tujuan dapat memisahkan penutur berdasarkan bahasa yang digunakan oleh penutur tersebut.Langkah untuk membuat *hyperplane* yang dapat memisahkan fitur yang ada dalam lebih dari dua dimensi adalah dengan membuat kernel *Radial Basis Function*, membuat persamaan linier untuk mendapatkan nilai α dan nilai bias, lalu proses klasifikasi. Tahapan modeling dan klasifikasi yang dilakukan oleh metode SVM dapat dilihat dari *Pseudo-code* berikut :
Setelah mendapatkan fitur proses klasifikasi diperlukan untuk membuat matriks fitur. Berdasarkan perhitungan yang telah dilakukan, dari tekstur terdapat 10 fitur yaitu *fractal dimension* ,*radius(r)*, *se*,*Y-INT*,*prefactor*,*fractal dimension* ,*radius(r)*, *se*,*Y-INT*, *prefactor*, dan *homogeneity*.

1. Tentukan nilai sigma yang akan digunakan untuk proses pemodelan.
2. Tentukan jumlah fitur yang digunakan dalam *dataset* yang telah dipilih.
3. Tentukan *path* dimana nantinya model akan disimpan.
4. Buat matriks RBF(*Radial Basis Function*) berdasarkan rumus perhitungan matriks RBF 2.2 yang ada di bab 2.
5. Buat *LinearEquation* matriks.
6. Lakukan perhitungan persamaan linear berdasarkan *LinearEquation* matriks yang telah terbentuk menggunakan rumus perhitungan 2.4, 2.5, dan 2.6 yang ada di bab 2 untuk mendapatkan nilai *alpha* dan *bias*.
7. simpan nilai *alpha* dan *bias* yang sudah didapat, nilai ini akan digunakan sebagai model untuk proses pengklasifikasian.

Langkah-langkah di atas menunjukkan proses *Training* untuk metode SVM, tujuan utama dari proses *Training* tersebut adalah mendapatkan model yang berupa nilai *alpha* dan *bias*. Sementara *Pseudo-code* untuk proses *Testing* dapat dilihat pada

langkah-langkah berikut :

1. Tentukan data tes.
2. Tentukan nilai sigma yang akan digunakan untuk proses klasifikasi.
3. Tentukan jumlah fitur yang digunakan dalam *dataset* yang telah dipilih.
4. Tentukan *path* dimana model disimpan.
5. Buat matriks RBF(*Radial Basis Function*) untuk data test yang telah dipilih menggunakan rumus 2.2.
6. Lakukan perhitungan perkalian nilai *alpha* dengan nilai dari matriks RBF tes yang telah dibentuk sebelumnya. Lalu tambahkan hasil perkalian tersebut dengan nilai bias yang disimpan dalam model menggunakan rumus 2.3.
7. Nilai 1 berarti data tes masuk ke dalam kelas yang sedang dituju, sementara nilai lain berarti data tes masuk ke dalam kelas yang lain.

Contoh perhitungan manual dari metode SVM dapat dilihat sebagai berikut : Sampel masukan untuk data latih yang akan digunakan untuk perhitungan manual didapat dari fitur metode BoxCounting, sampel yang akan digunakan berjumlah 4 dengan fitur sebanyak 10 buah. Sampel untuk data latih dapat dilihat di Tabel 3.2.

Tabel 3.2 Hasil Ekstraksi Fitur Metode Box Counting

No.	**F1**	**F2**	**F3**	**...**	**F7**	**F8**	**F9**	**F10**
A1	1.6178	0.9927	0.0544	**...**	0.9927	0.0544	5.1789	0.0056
A2	1.5544	0.9865	0.066	**...**	0.9865	0.066	4.8734	0.0076
A3	1.5092	0.9823	0.0701	**...**	0.9823	0.0701	4.7666	0.0085
A4	1.4343	0.9805	0.0626	**...**	0.9805	0.0626	4.8744	0.0076

Di mana:

1. x range block tertentu pada citraFrame .
2. y range block tertentu pada citraFrame.
3. ios range block tertentu pada citra.
4. o range block tertentu pada citra.

5. s range block tertentu pada citra.

Pembuatan RBF kernel menggunakan rumus yang terdapat pada 2.2, hasil perhitungan menggunakan rumus 2.2 dapat dilihat pada perhitungan berikut :

$$K(A1,A1) = \exp\left(-\frac{|A1-A1|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.6178-1.6178|^2+|0.9927-0.9927|^2+|...+|5.225-3.935|^2+|-14.0-6.34|^2}{2*(10)^2}\right) = 1$$

$$K(A1,A2) = \exp\left(-\frac{|A1-A2|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.5544-1.5544|^2+|0.9927-0.9865|^2+...+|5.17894.8734|^2+|0.00560.0076|^2}{2*(10)^2}\right) = 0.05870163$$

$$K(A1,A3) = \exp\left(-\frac{|A1-A3|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.61781.5092|^2+|0.99270.9823|^2+...+|5.17894.7666|^2+|0.00560.0085|^2}{2*(10)^2}\right) = 0.109051345$$

$$K(A1,A4) = \exp\left(-\frac{|A1-A4|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.61781.4343|^2+|0.99270.9805|^2+...+|5.17894.8744|^2+|0.00560.0076|^2}{2*(10)^2}\right) = 0.08110975$$

Dari hasil perhitungan *dot product* di atas, akan didapatkan matriks RBF yang dapat dilihat dari Tabel 3.3.

Tabel 3.3 Hasil Matriks RBF

	R	**A**	**T**	**B**	**Class**
A1	-1.0	-0.942986200567	-0.896680605507	-0.922090619005	1.0
A2	-0.942986200567	-1.0	-0.991829818842	-0.9821812307	1.0
A3	-0.896680605507	-0.991829818842	-1.0	-0.982313107212	1.0
A4	-0.922090619005	-0.9821812307	-0.982313107212	-1.0	1.0

Dari matriks fitur yang ada akan dibuat matriks RBF yang berukuran sebanyak fitur. Misal terdapat 10 fitur yang digunakan maka ukuran matriks RBF

adalah 10 × 10. Matriks RBF digunakan untuk menghitung dot product dari masing-masing data. Pada perhitungan ini nilai yang digunakan adalah 1. Contoh perhitungan adalah sebagai berikut:

$(1)\alpha1 + (1)\alpha2 + (1)\alpha3 + (1)\alpha4 + (0)\omega = 0$

$(1)\alpha1 + (0.162676539)\alpha2 + (-0.001022397)\alpha3 + (-0.03214562)\alpha4 + \omega = 1$

$(0.162676539)\alpha1 + (1)\alpha2 + (-0.025839689)\alpha3 + (-0.0002056361)\alpha4 + \omega = 1$

$(0.001022397)\alpha1 + (0.025839689)\alpha2 + (-1)\alpha3 + (-0.00015745)\alpha4 + \omega = -1$

$(0.03214562)\alpha1 + (0.0002056361)\alpha2 + (-0.00015745)\alpha3 + (-1)\alpha4 + \omega = -1$

Untuk mendapatkan solusi untuk (α1, α2, α3, α4, ω) dibutuhkan pustaka yang bernama *Apache Commons*. Solusi yang didapat dari perhitungan persamaan linear di atas dapat dilihat pada Tabel 3.4 berikut.

Tabel 3.4 Tabel nilai α dan ω

α1	**α2**	**α3**	**α4**	ω
2.8E-14	-9.85E-14	8.14E-14	-1.13E-14	-1

Setelah mendapatkan hasil dari α dan ω, kita sudah bisa melakukan proses klasifikasi. Misal terdapat data uji yang memiliki kelas Sunda sebagai berikut :

Tabel 3.5 Tabel Data Uji

D	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**	**f9**	**f10**	**L**
T1	1.587	0.9875	0.0674	0.6005	0.5486	1.587	0.9875	0.0674	4.998	0.0068	-1

Pembuatan RBF kernel menggunakan rumus yang terdapat pada 2.2, hasil perhitungan menggunakan rumus 2.2 dapat dilihat pada perhitungan berikut :

$K(T1,A1) = \exp\left(-\frac{|A1-A1|^2}{2\sigma^2}\right)$

$= \exp\left(-\frac{|1.5871.6178|^2+|0.98750.09927|^2+|...+|4.9985.1789|^2+|0.00680.0056|^2}{2*(10)^2}\right) = 1.045777994$

$$K(T1,A2) = \exp\left(-\frac{|A1-A2|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.5871.5544|^2+|0.98750.9865|^2+...+|4.9984.8734|^2+|0.00680.0076|^2}{2*(10)^2}\right) = 1.018531027$$

$$K(T1,A3) = \exp\left(-\frac{|A1-A3|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.5871.5092|^2+|0.98750.9823|^2+...+|4.9984.7666|^2+|0.00680.0085|^2}{2*(10)^2}\right) = 1.0712389545$$

$$K(T1,A4) = \exp\left(-\frac{|A1-A4|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{|1.5873.112|^2+|0.98750.9805|^2+...+|4.9984.8744|^2+|0.00680.0076|^2}{2*(10)^2}\right) = 1.067511533$$

Tabel 3.6 Tabel Matriks RBF Uji

	R	**A**	**T**	**B**
T1	1.0	0.942986200567	0.896680605507	0.92209061900

Setelah mendapatkan matriks RBF untuk data uji, proses klasifikasi dapat dilakukan dengan menggunakan persamaan (2.3). Proses perhitungan klasifikasi akan dijabarkan sebagai berikut :

$$f(x) = sign[(2.8296012464107698E14 * 1.045777994 * 1.0) + (9.8529372743253282E - 14 * 1.018531027 * 1.0) + (8.143562343007364E14 * 1.0712389545 * 1.0) + (1.1283402057632718E - 14 * 1.067511533 * 1.0) + -1.00000000000000002]$$

f(x)= 1

Dari hasil perhitungan yang telah dilakukan maka data uji terhadap data latih nilai yang diperoleh adalah -1, artinya data uji masuk ke dalam kelas -1. Jika hasil perhitungan bernilai 1 maka data uji masuk ke dalam kelas 1. Misal dalam pengujian di atas kelas -1 adalah IrisA dan kelas 1 adalah jenis IrisB, maka data uji merupakan kelas IrisA. Karena dalam penelitian ini kelas yang ada lebih dari dua, maka jika hasil perhitungan tadi menghasilkan nilai 1, dilakukan pengujian kembali terhadap model SVM lainnya untuk menentukan data uji masuk ke dalam kelas apa.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

4.1 Lingkungan Implementasi

Pada lingkungan implementasi, akan dijelaskan mengenai perangkat-perangkat keras dan perangkat-perangkat lunak yang akan digunakan untuk membangun sistem identifikasi pembicara.

4.1.1 Spesifikasi Perangkat Keras

Spesifikasi komputer yang digunakan dalam pembangunan sistem identifikasi pembicara menggunakan *Mel Frequency Cepstral Coefficient*, dan *Support Vector Machine* adalah sebagai berikut:

1. Laptop dengan *processor* Intel(R) Core(TM) i5-5200U CPU 2.20GHz (4 CPUs)
2. *Harddisk* dengan kapasitas 500 MB
3. RAM 12 GB

4.1.2 Lingkungan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pembangunan sistem identifikasi pembicara adalah sebagai berikut:

1. Sistem Operasi: Windows 10 Pro 64-bit
2. IDE: Netbeans IDE 8.1

4.2 Penggunaan Data

Penggunaan data untuk klasifikasi per 5 data untuk setiap σ menggunakan $forward_look(278), Forward_look(288)$.

4.3 Daftar atribut

Pada bagian ini, akan dijelaskan mengenai atribut yang digunakan dalam penelitian ini. Setiap atribut yang digunakan akan dijelaskan dalam Tabel 4.1 berikut :

Tabel 4.1 Daftar *Attribute* pada *class SVMResult*

No	**Tipe**	**Nama Variabel**
1	*public* int	*index*
2	*public* double	*value*

4.4 Daftar *class*

Pada bagian ini, akan dijelaskan mengenai *class* yang digunakan dalam penelitian ini. Setiap *class* akan dijelaskan atribut dan *method* nya.

4.4.1 *Class* Normalization

Class Normalization merupakan sebuah *class* yang berisi fungsi untuk membentuk *confusion matrix*. Metode yang terdapat pada *class* Normalization dapat dilihat pada Tabel 4.2.

Tabel 4.2 Daftar *Method* pada *class* Normalization

No	**Metode**	***Masukan***		***Keluaran***	**Keterangan**
		Tipe	**Variabel**		
1	getDataFromTxt	String, int	path, n	double[]	Digunakan untuk membaca data dari file berekstensi .txt
2	findSVMResult	double[]	data	SVMResult	Digunakan untuk mendapatkan nilai tertinggi dari hasil klasifikasi SVM.
3	createConfusion Matrix	double[][], int, String[]	data, countLabel, tempLabel _test	int[][]	Digunakan untuk membentuk {confusion Matrix}.

4.4.2 *Class* SupportVectorMachine

class SupportVectorMachine merupakan sebuah *class* yang berisikan semua metode SVM yang digunakan untuk melakukan penghitungan baik pada proses pelatihan untuk membuat model dan proses pencocokan pada saat pengujian. Daftar metode yang terdapat pada *class* SupportVectorMachine dapat dilihat pada Tabel 4.3.

Tabel 4.3 Daftar *method* pada *class* SupportVectorMachine

No	Metode	*Masukan*		*Keluaran*	Keterangan
		Tipe	**Variabel**		
1	calculateRBF Kernel	double[][], double[], int, int	data, sigma, classSource, classTarget	double	Digunakan untuk menghitung nilai kernel RBF.
2	createRBF Matrix	double[][], double	data, sigma	double[][]	Digunakan untuk membentuk matriks RBF.
3	createLinear Equation Matrix	double[][], double[]	rbfMatrix, classList	double[][]	Digunakan untuk membentuk matriks hasil perkalian matriks RBF dengan classList.
4	getSolution	double[][], double[]	maxFeatures, linearEquation Matrix	Real Vector	Digunakan untuk mendapatkan solusi berupa nilai *alfa* dan *bias* untuk kelas tertentu.
5	createRBFTest Matrix	double[][], double, double[]	data, sigma, test	double[]	Digunakan untuk membentuk matriks RBF untuk pengujian.
6	classify	double[], double[], double[]	solutions, rbfTest, classList	double	Digunakan untuk melakukan perhitungan matriks RBF test dengan model yang telah dibentuk sebelumnya.

4.5 Implementasi Perangkat Lunak

Pada implementasi perangkat lunak akan dilakukan perancangan berdasarkan analisis yang telah disusun pada BAB III.

4.5.1 Implementasi Penggunaan Data Citra Mata

Jumlah data citra mata yang digunakan pada penelitian ini adalah 46 orang citra mata. citra mata dari nomor 1-3 yang merupakan citra mata dengan kemiringan yang berbeda akan digunakan untuk proses *Training*. citra dari nomor 4 beserta 5 yang merupakan citra mata yang memiliki kemiringan berbeda akan dijadikan *Testing*.

4.5.2 Hasil Biner

Berikut merupakan Hasil biner dari citra dengan menggunakan method *Daugman* dengan menggunakan radius sebesar 64 dan 320

Gambar 4.1 Hasil Biner

4.5.3 Implementasi Pelatihan Data SVM

Setelah mendapatkan fitur dari ekstraksi fitur, kemudian fitur-fitur tersebut akan dijadikan model dengan menggunakan SVM untuk proses pelatihan, tahapan yang dilakukan dalam prsoses pelatihan akan dijelaskan sebagai berikut :

1. Inisialisasi

   (a) Pemilihan data set yang akan digunakan untuk membuat model.
   (b) Menentukan nilai sigma yang akan digunakan

2. Perhitungan Data Belajar

   (a) Melakukan perhitungan untuk membentuk matriks RBF.
   (b) Melakukan perhitungan untuk membentuk persamaan *hyperplane* SVM.
   (c) Melakukan dekomposisi *solver* menggunakan *Singular Value Decomposition*.
   (d) Melakukan perhitungan untuk mendapatkan nilai alfa dan bias.

3. Penyimpanan Data

   (a) Nilai *alfa* dan bias dari setiap kelas akan disimpan ke dalam file berbentuk .txt untuk nantinya digunakan dalam tahap pengujian.

4.5.4 Implementasi Pengujian Data SVM

Pada bagian ini akan dilakukan pengujian SVM. Berikut adalah tahapan dalam pengujian :

1. Baca Data

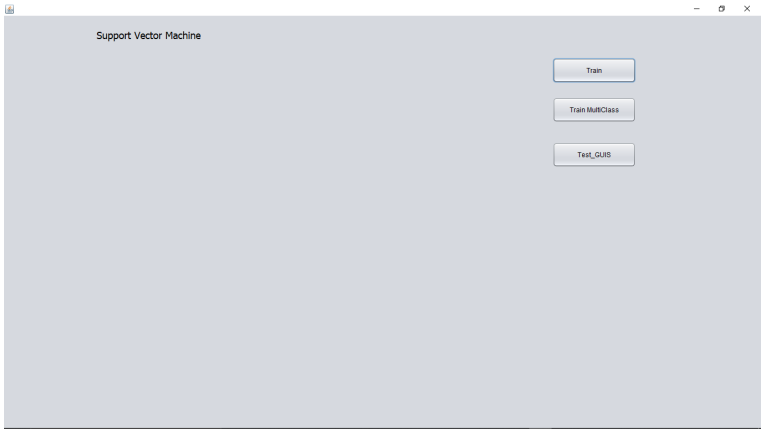
   (a) Menentukan model yang akan digunakan untuk pengujian.
   (b) Menentukan nilai sigma yang akan digunakan untuk proses pengujian.

2. Perhitungan Data Uji

(a) Melakukan pengujian menggunakan pendekatan one versus many.
(b) Melakukan perhitungan untuk membentuk matriks RBF setiap baris dari data uji *hyperplane* SVM.
(c) Mengambil nilai klasifikasi tertinggi untuk menentukan data uji masuk ke dalam kelas mana.
(d) Membuat *confusion matrix* untuk setiap data yang diuji.

4.6 Tampilan Aplikasi

Pada bagian ini akan dijelaskan mengenai tampilan aplikasi yang digunakan baik untuk pelatihan, dan pengujian. Gambar 4.2 adalah halaman utama yang digunakan untuk memilih proses pelatihan atau pengujian.

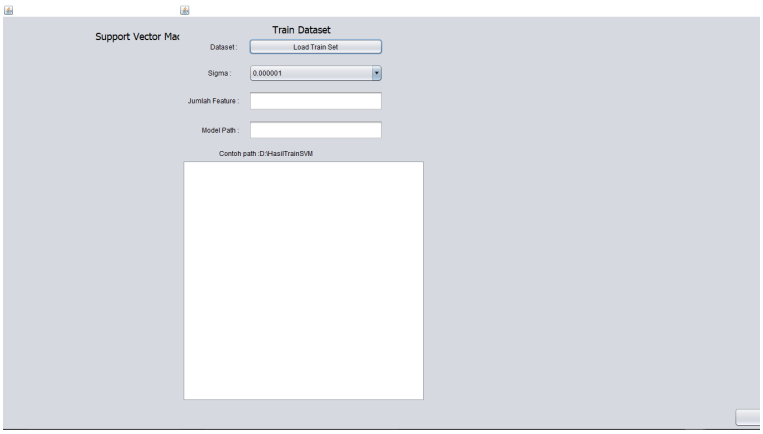


Gambar 4.2 Tampilan awal aplikasi

Pada Gambar 4.2 terdapat tombol *Train*dan *Test*. Kegunaan dari masing-masing *item* akan dijelaskan sebagai berikut :

1. *Train*, tombol ini adalah tombol untuk memilih mode train yang akan digunakan untuk membuat model
2. Test adalah tombol yang digunakan untuk memilih mode uji yang akan digunakan untuk pengujian.

4.6.1 Tampilan Train

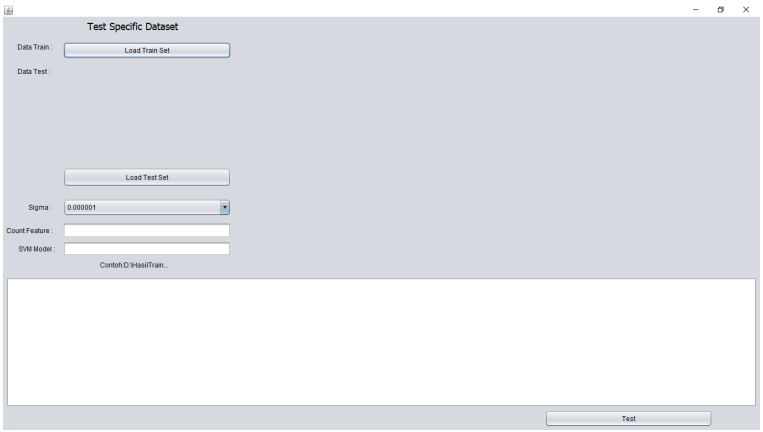


Gambar 4.3 Tampilan Train

Pada Gambar 4.3 terdapat tombol *Load Train Set*,*Sigma*,Jumlah Feature,danModel PathKegunaan dari masing-masing *item* akan dijelaskan sebagai berikut :

1. *Load Train SEt*, tombol ini adalah tombol untuk memilih data latih yang nantinya akan dijadikan model.
2. *sigma* adalah nilai yang digunakan untuk perhitungan membuat *rbfMatrixTrain* dan *rbfMatrixTest*.
3. jumlah fitur adalah jumlah fitur yang digunakan dalam data *train* yang telah dipilih dalam *Load Train Set*.
4. *Model_Path* adalah dimana model dari hasil latih SVM akan disimpan.
5. *Train* adalah tombol untuk melatih 46 kelas *pegawai1-pegawai46*.

4.6.2 Tampilan Test



Gambar 4.4 Tampilan Test

Pada Gambar 4.4 terdapat tombol *Load Train Set*,*Load Test Set*,*Sigma*,Jumlah Feature,danModel Path.Kegunaan dari masing-masing *item* akan dijelaskan sebagai berikut :

1. *Load Train Set* adalah tombol yang digunakan untuk memilih data Train yang akan digunakan untuk pengujian.
2. *Load Test Set* adalah tombol yang digunakan untuk memilih data uji yang akan digunakan untuk pengujian.
3. *sigma* adalah nilai yang digunakan untuk perhitungan membuat *rbfMatrixTrain* dan *rbfMatrixTest*.
4. *Test* adalah tombol yang digunakan untuk klasifikasi data uji *Load Test Set*.
5. *box* putih besar di sebelah kiri tampilan adalah *textbox* yang digunakan untuk menampilkan hasil klasifikasi dari data uji, menampilkan direktori dari setiap file yang dipilih dalam *Load Train Set*, dan *Load Test Set*.

4.7 Skenario Pengujian

Pada bagian ini akan dijelaskan pengujian dengan skenario yang berbeda-beda.Pertama menggunakan nilai perhitungan sigma 1, kedua menggunakan nilai perhitungan sigma 10, ketiga menggunakan nilai perhitungan sigma 100, keempat menggunakan nilai perhitungan sigma 1000, masing-masing diuji dengan 45 data dengan kombinasi 3 train, 2 test dan 4 train, 1 test . Pengujian menggunakan sigma yang telah disebutkan di atas berdasarkan hasil uji coba dari banyak percobaan.

4.7.1 Pengujian SVM dengan nilai sigma 1

Pengujian SVM dengan sigma 1 dilakukan untuk mengklasifikasi citra mata berdasarkan data kepemilikan dari citra tersebut. Data uji yang dipakai adalah citra mata 4-5, sementara data latih yang dipakai adalah citra mata 1-3. Pengujian sigma 1 dengan 46 data dapat dilihat pada Tabel 4.4. Pengujian dengan 46 data dilakukan di bawah kondisi yang sama dengan data yang telah dijelaskan pada poin 4.3.1.

Tabel 4.4 Hasil Prediksi SVM sigma 1, 46 Data

No	**Filename**	**SVM Predict**	**Time(Ns)**	**True /False (1/0)**
1	Aeval	Aeval	16.361	1
2	Bryan	Bryan	16.62814827	1
3	Chingcy	Aeval	9.04056398	0
4	Chongpk1	Chongpk1	9.42130500	1
5	Christine	Christine	7.46211998	1
6	Chuals	Chuals	7.41333011	1
7	Eugeneho	Eugeneho	9.44262083	1
8	Fatma	Fatma	7.25604835	1
9	Fiona	Chongpk1	7.47351733	0

10	hock	Chongpk1	8.12314122	0
11	Kelvin	Kelvin	8.23424251	1
12	Lec	Lec	8.90412414	1
13	Liujw	Christine	8.23424251	0
14	Loke	Chuals	8.90412414	0
15	Lowyf	Lowyf	9.04056398	1
16	Lpj	Lpj	7.47510821	1
17	Mahsk	Eugeneho	8.23424251	0
18	Maran	Maran	7.47510821	1
19	Mas	Mas	7.47510821	1
20	Mazwan	Mazwan	7.47510821	1
21	Mimi	Mimi	7.47510821	1
22	Mingli	hock	8.23424251	0

23	Ngkokwhy	Ngkokwhy	7.47510821	1
24	Nkl	Nkl	8.23424251	1
25	Noraza	Noraza	7.47510821	1
26	Norsuhaidah	Norsuhaidah	7.47510821	1
27	Ongbl	Lec	7.47510821	0
28	Pcl	Pcl	7.47510821	1
29	Philip	Philip	7.47510821	1
30	Rosli	Rosli	8.12314122	1
31	Sala	Loke	9.04056398	0
32	Sarina	Sarina	7.47510821	1
33	Siti	Lowyf	8.23424251	0
34	Suzaili	Lpj	7.47510821	0
35	Tanwn	Tanwn	8.23424251	1

36	Thomas	Thomas	7.47510821	1
37	Tick	Tick	8.23424251	1
38	Tingcy	Maran	8.23424251	0
39	Tonghl	Tonghl	7.47510821	1
40	Vimala	Mas	8.23424251	0
41	Weecm	Weecm	7.47510821	1
42	Win	Win	7.47510821	1
43	Yann	Yann	7.47510821	1
44	Zaridah	Mimi	9.04056398	0
45	Zulaikah	Mimi	7.47510821	0

Dari hasil pengujian 4.4 mendapatkan akurasi sebesar 70.0%. Dapat dilihat juga dari Tabel semua yang berarti model yang digunakan untuk perbandingan dari data latih bukan merupakan model yang baik.

4.7.2 Pengujian SVM dengan nilai sigma 10

Pengujian SVM dengan sigma 1 dilakukan untuk mengklasifikasi citra mata berdasarkan data kepemilikan dari citra tersebut. Data uji yang dipakai adalah citra

mata 4-5, sementara data latih yang dipakai adalah citra mata 1-3, Pengujian sigma 10 dengan 46 data dapat dilihat pada Tabel 4.5. Pengujian dengan 46 data dilakukan di bawah kondisi yang sama dengan data yang telah dijelaskan pada poin 4.3.1.

Tabel 4.5 Hasil Prediksi SVM sigma 10, 46 Data

No	**Filename**	**SVM Predict**	**Time(Ns)**	**True /False (1/0)**
1	Aeval	Aeval	16.361	1
2	Bryan	Bryan	16.62814827	1
3	Chingcy	Aeval	9.04056398	0
4	Chongpk1	Chongpk1	9.42130500	1
5	Christine	Christine	7.46211998	1
6	Chuals	Chuals	7.41333011	1
7	Eugeneho	Eugeneho	9.44262083	1
8	Fatma	Fatma	7.25604835	1
9	Fiona	Fiona	7.47351733	1
10	hock	hock	8.12314122	1

11	Kelvin	Kelvin	8.23424251	1
12	Lec	Lec	8.90412414	1
13	Liujw	Liujw	8.23424251	1
14	Loke	Chuals	8.90412414	0
15	Lowyf	Lowyf	9.04056398	1
16	Lpj	Lpj	7.47510821	1
17	Mahsk	Eugeneho	8.23424251	0
18	Maran	Maran	7.47510821	1
19	Mas	Fatma	7.47510821	0
20	Mazwan	Mazwan	7.47510821	1
21	Mimi	Mimi	7.47510821	1
22	Mingli	Hock	8.23424251	0
23	Ngkokwhy	Ngkokwhy	7.47510821	1

24	Nkl	Nkl	8.23424251	1
25	Noraza	Noraza	7.47510821	1
26	Norsuhaidah	Norsuhaidah	7.47510821	1
27	Ongbl	Lec	7.47510821	0
28	Pcl	Pcl	7.47510821	1
29	Philip	Philip	7.47510821	1
30	Rosli	Rosli	8.12314122	1
31	Sala	Loke	9.04056398	1
32	Sarina	Sarina	7.47510821	1
33	Siti	Lowyf	8.23424251	0
34	Suzaili	Lpj	7.47510821	0
35	Tanwn	Tanwn	8.23424251	1
36	Thomas	Thomas	7.47510821	1

37	Tick	Tick	8.23424251	1
38	Tingcy	Maran	8.23424251	0
39	Tonghl	Tonghl	7.47510821	1
40	Vimala	Max	8.23424251	1
41	Weecm	Weecm	7.47510821	1
42	Win	Mazwan	7.47510821	0
43	Yann	Yann	7.47510821	1
44	Zaridah	Mimi	9.04056398	0
45	Zulaikah	Mimi	7.47510821	0

Dari hasil pengujian 4.5 mendapatkan akurasi sebesar 80.0%. Dapat dilihat juga dari Tabel semua yang berarti model yang digunakan untuk perbandingan dari data latih bukan merupakan model yang baik.

4.7.3 Pengujian SVM dengan nilai sigma 100

Pengujian sigma 100 dengan 46 data dapat dilihat pada Tabel 4.6. Pengujian dengan 46 data dilakukan di bawah kondisi yang sama dengan data yang telah dijelaskan pada poin 4.3.1.

Tabel 4.6 Hasil Prediksi SVM sigma 100, 46 Data

No	Filename	SVM Predict	Time(Ns)	True /False (1/0)
1	Aeval	Aeval	16.361	1
2	Bryan	Bryan	16.62814827	1
3	Chingcy	Aeval	9.04056398	0
4	Chongpk1	Chongpk1	9.42130500	1
5	Christine	Christine	7.46211998	1
6	Chuals	Chuals	7.41333011	1
7	Eugeneho	Eugeneho	9.44262083	1
8	Fatma	Fatma	7.25604835	1
9	Fiona	Fiona	7.47351733	1
10	hock	hock	8.12314122	1
11	Kelvin	Kelvin	8.23424251	1

12	Lec	Lec	8.90412414	1
13	Liujw	Liujw	8.23424251	1
14	Loke	Chuals	8.90412414	0
15	Lowyf	Lowyf	9.04056398	1
16	Lpj	Lpj	7.47510821	1
17	Mahsk	Eugeneho	8.23424251	0
18	Maran	Maran	7.47510821	1
19	Mas	Fatma	7.47510821	0
20	Mazwan	Mazwan	7.47510821	1
21	Mimi	Mimi	7.47510821	1
22	Mingli	Hock	8.23424251	0
23	Ngkokwhy	Ngkokwhy	7.47510821	1
24	Nkl	Nkl	8.23424251	1

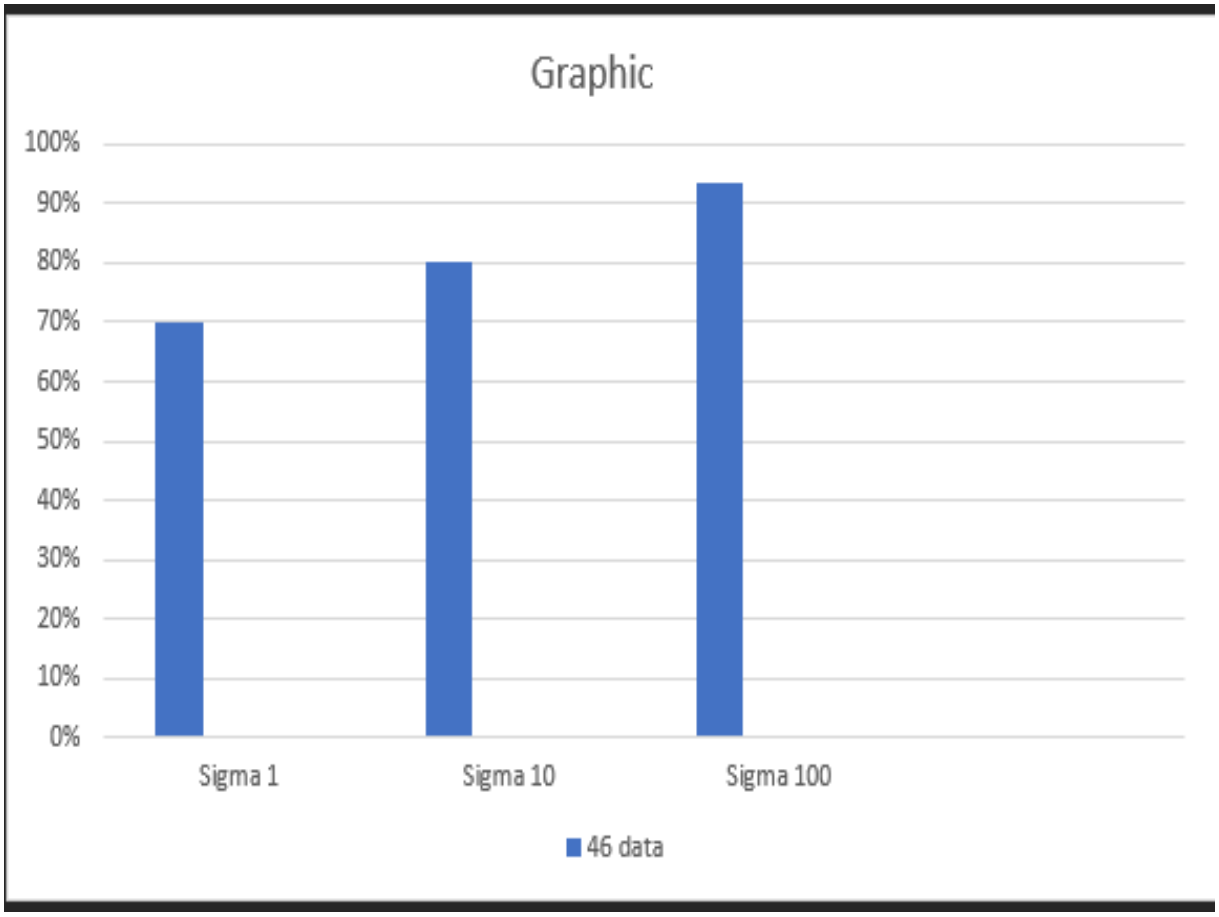
25	Noraza	Noraza	7.47510821	1
26	Norsuhaidah	Norsuhaidah	7.47510821	1
27	Ongbl	Lec	7.47510821	0
28	Pcl	Pcl	7.47510821	1
29	Philip	Philip	7.47510821	1
30	Rosli	Rosli	8.12314122	1
31	Sala	Loke	9.04056398	1
32	Sarina	Sarina	7.47510821	1
33	Siti	Lowyf	8.23424251	0
34	Suzaili	Lpj	7.47510821	0
35	Tanwn	Tanwn	8.23424251	1
36	Thomas	Thomas	7.47510821	1
37	Tick	Tick	8.23424251	1

38	Tingcy	Maran	8.23424251	0
39	Tonghl	Tonghl	7.47510821	1
40	Vimala	Max	8.23424251	1
41	Weecm	Weecm	7.47510821	1
42	Win	Mazwan	7.47510821	0
43	Yann	Yann	7.47510821	1
44	Zaridah	Zaridah	9.04056398	1
45	Zulaikah	Zaridah	7.47510821	1

Dari hasil pengujian 4.6 mendapatkan akurasi sebesar 93.3%. Dapat dilihat juga dari Tabel semua yang berarti model yang digunakan untuk perbandingan dari data latih bukan merupakan model yang baik.

4.7.4 Evaluasi Kesalahan

Dari semua pengujian yang telah dilakukan dapat dilihat bahwa semakin besar sigma yang digunakan, maka akan semakin baik model yang dibuat. Hasil klasifikasi yang dilakukan juga mendapatkan akurasi yang semakin tinggi.



Gambar 4.5 Grafik Evaluasi

Dapat dilihat pada grafik 4.5 bahwa tingkat kesalahan untuk sigma 1 adalah 30%. Tingkat kesalahan semakin mengecil dengan dilakukannya pengujian untuk sigma 10 dan 100, masing-masing dari sigma tersebut mendapatkan nilai kesalahan sebesar 20% dan 16.7%.