# BAB 3
# ANALISIS DAN PERANCANGAN

## 3.1. Analisis Sistem

Analisis sistem merupakan tahap penguraian konsep menjadi lebih sederhana, sehingga struktur logisnya menjadi lebih jelas. Analisis sistem terdiri dari beberapa tahapan, diantaranya analisis masalah,analisis proses, analisis metode, analisis kebutuhan non-fungsional, dan analisis kebutuhan fungsional.

### 3.1.1 Analisis Masalah

Mengantuk banyak mengakibatkan kecelakaan lalu lintas yang meyebabkan korban cedera, kerugian materi dan bahkan kematian. Oleh karena itu dibuat aplikasi simulasi keselamatan aktif untuk mendeteksi kantuk dengan mengamati pergerakan mata seseorang. Tujuan utaman,ya adalah untuk membuat keputusan yang berguna tentang kondisi mata berdasarkan *image* yang didapat dari sensor. Penerapan *Computer Vision* cukup banyak digunakan dalam sistem keamanan maupun proteksi. Salah satunya pengenalan pola pada objek mata. Misalnya, penerapan pengenalan pola pada objek mata ini dapat digunakan sebagai sistem proteksi dengan menjadikan mata sebagai objek *scanning* untuk sistem sensor. Sistem pendeteksi kantuk ini memanfaatkan *webcam camera* sebagai sensornya. Selanjutnya data kamera diproses dengan menggunakan CPU (Central Processing Unit) untuk mengolah dan mengeksekusi data yang telah ditangkap. Metode yang digunakan untuk mendeteksi mata mengantuk adalah *Haar Cascade classifier*, yaitu metode yang dapat melihat dari adanya mata seseorang tertutup dan metode ini memiliki keakuratan yang tinggi dalam mendeteksi suatu objek.

### 3.1.2 Analisis mata mengantuk

Prangkat lunak ini menggunakan mata sebagai objek observasi. Pemilihan mata sebagai objek observasi dikarenakan mata merupakan faktor yang sangat penting untuk mendeteksi keadaan pengemudi, apakah pengemudi dalam keadaan mengantuk atau tidak. Citra mata diambil dengan menggunakan *webcam* sebagai citra *real time.* Dengan penggunaan *webcam* sebagai input secara *real time* dapat

menyebabkan masalah seperti waktu proses dan keakuratan dalam proses pendeteksian objek.

Penelitian kali ini menggunakan mata dalam keadaan tertutup 100% sebagai objek penelitiannya. Menurut Mark A.W. Andrews, profesor fisiologi dari Seton Hill University, Greensburg, Pennsylvania, kelopak mata bisa menutup sendiri sering disebabkan oleh mata yang sudah terlalu lelah. Andrews menjelaskan bahwa saat terlalu lama berkendara, membaca atau bekerja dengan komputer, kekuatan otot di area kelopak mata bisa menurun, Andrews membandingkan hal tersebut seperti ketika seseorang berjalan kaki seharian tanpa berhenti.

Kelopak mata bisa terasa sangat berat dan tertutup 100% ketika seseorang mengerahkan konsentrasi dan memaksa mata tetap terbuka dalam jangka waktu yang lama. Apalagi bila seseorang bekerja di lingkungan yang agak gelap, seperti halnya berkendara di malam hari.

Menurut studi yang dilakukan oleh Phillip .P. Caffier [10], mengelompokkan tingkatan kantuk berdasarkan durasi kedipan mata. Umumnya durasi kedipan rata-rata adalah kurang dari 400 Ms dan 75 Ms untuk minimum. Berdasarkan alasan ini, maka digunakan 400 Ms sebagai waktu kantuk (T_kantuk) dan 800 Ms sebagai waktu telah tertidur (T_tidur).

### 3.1.3 Analisis Lingkungan

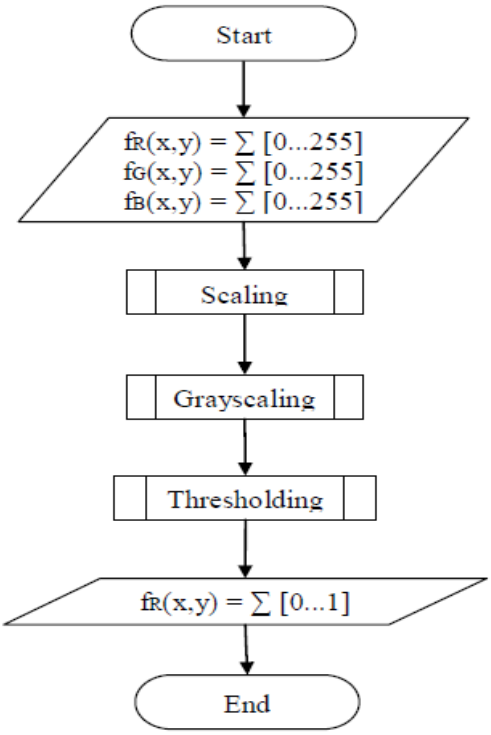
Citra mata diambil dengan menggunakan *webcam* sebagai citra *real time*. Dengan penggunaan *webcam* sebagai input secara *real time* dapat menyebabkan beberapa masalah seperti kondisi lingkungan, serta waktu proses dan keakuratan dalam proses pendeteksian objek.

Kondisi lingkungan dan keakuratan dalam melakukan pendeteksian objek menjadi pertimbangan utama dalam pemilihan metode. Hal ini dikarenakan program berlangsung secara *real time* maka dibutuhkan lingkungan yang memiliki intisitas cahaya yang cukup, selain itu mengingat objek yang akan dideteksi cukup kecil berupa mata maka tingkat keakuratan harus diperhatikan.

### 3.1.4 Analisis Proses Pengenalan Pola Mata pada Haar Classifier

Dalam analisis processing image (Pengenalan pola mata) ini dibagi menjadi 3 tahap[5]:

1. penskalaan(*scaling*)
2. *grayscaling*
3. *tresholding*




**Gambar 3.1 Alur proses pengenalan pola mata**

1. **Tahap Penskalaan (*Scaling)***

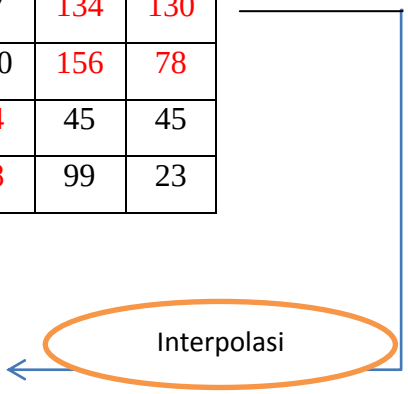
Citra digital yang diambil secara *real time* oleh *webcam* akan diperkecil dengan menggunakan metode interpolasi. Metode ini menggunakan nilai rata – rata suatu region untuk mewakili region tersebut.

**Citra asli**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 124 | 122 | 126 | 233 | 97 | 34 | 23 | 64 |
| 135 | 101 | 179 | 245 | 141 | 123 | 121 | 119 |
| 167 | 143 | 108 | 243 | 148 | 67 | 134 | 130 |
| 211 | 124 | 203 | 222 | 248 | 120 | 156 | 78 |
| 112 | 167 | 225 | 123 | 223 | 34 | 45 | 45 |
| 189 | 190 | 221 | 189 | 156 | 88 | 99 | 23 |

**Citra Hasil**

| | | | |
|---|---|---|---|
| 120,5 | 195,75 | 98,75 | 81,75 |
| 161,25 | 194 | 145,75 | 124,5 |
| 164,5 | 189,5 | 125,25 | 53 |




**Gambar 3.2 Metode interpolasi untuk memperkecil gambar**

Nilai piksel pada koordinat pada citra hasil interpolasi diperoleh dengan menghitung nilai rata – rata dari 4 nilai piksel pada citra asli, yaitu :

**Tabel 3.1 Rata-rata dari 4 piksel pada citra**

| Nilai Piksel Citra Asli | Nilai Piksel Citra Hasil Interpolasi |
|---|---|
| 124+122+135+101/4 | 120,5 |
| 126+233+175+145/4 | 195,75 |
| 97+34+141+123/4 | 98,75 |
| 23+64+121+119/4 | 81,75 |
| 167+143+211+124/4 | 161,25 |
| 108+234+203+222/4 | 194 |
| 148+67+248+120/4 | 145,75 |
| 134+130+156+78/4 | 124,5 |
| 112+167+189+190/4 | 164,5 |
| 225+123+221+189/4 | 189,5 |
| 223+34+156+88/4 | 125,25 |

2. **Tahap *Grayscaling***

Representasi nilai RGB (Red, Green, Blue) diubah menjadi gambar yang terdiri dari warna putih dan gradiasi warna hitam yang biasa disebut *greyscale*. Untuk mengubah RGB menjadi *greyscale* dapat digunakan rumus sebagai berikut:

$$Greyscale = 0.299R + 0.587G + 0.114B \quad \text{.....(3.1)}$$

Misalkan R=124,8 G=145,8 B=109,3

*Greyscal*: (0.299x124,8) + (0.587x145.8) + (0.114x109.3) = 135.36

Berikut ini adalah citra hasil *grayscaling* :

Citra Asli Cirtra *grayscaling*





**Gambar 3.3 Proses pengubahan citra RGB menjadi citra Grayscale**

3. **Tahap *Tresholding***

Selanjutnya adalah tahap *tresholding* yang digunakan untuk mengubah gambar hasil *grayscale* menjadi gambar biner. Nilai *Treshold* dihitung dengan membagi nilai hasil *grayscaling* pada tahap sebelumnya dengan nilai jumlah deraja keabuan (0 sampai 255 = 256) dibagi dengan 256 (nilai derajat keabuan yang diinginkan). Proses penghitungan nilai *treshold* untuk citra mata adalah sebagai berikut :

$$X = \frac{w}{b} = \frac{135.36}{1} = 135.36 \quad \text{.....(3.2)}$$

Dimana :

x = nilai pembanding *threshold*

w = nilai hasil *grayscaling*

b = 256/a ( a = 256)

Untuk mengubah citra RGB menjadi citra biner menggunakan aturan sebagai berikut:

1. Jika nilai piksel citra ≥ x maka nilai piksel menjadi 1
2. Jika nilai piksel citra ≤ x maka nilai piksel menjadi 0

Berikut ini adalah proses pengubahan citra RGB menjadi citra biner :

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 124 | 122 | 126 | 233 | 97 | 34 | 23 | 64 |
| 135 | 101 | 179 | 245 | 141 | 123 | 121 | 119 |
| 167 | 143 | 108 | 243 | 148 | 67 | 134 | 130 |
| 211 | 124 | 203 | 222 | 248 | 120 | 156 | 78 |
| 112 | 167 | 225 | 123 | 223 | 34 | 45 | 45 |
| 189 | 190 | 221 | 189 | 156 | 88 | 99 | 23 |

**Gambar 3.4 Citra Hasil**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

**Gambar 3.5 Citra Biner**

Berikut ini adalah gambar hasil pengubahan citra grayscale menjadi citra biner :

**Gambar Grayscale** **Gambar Biner**



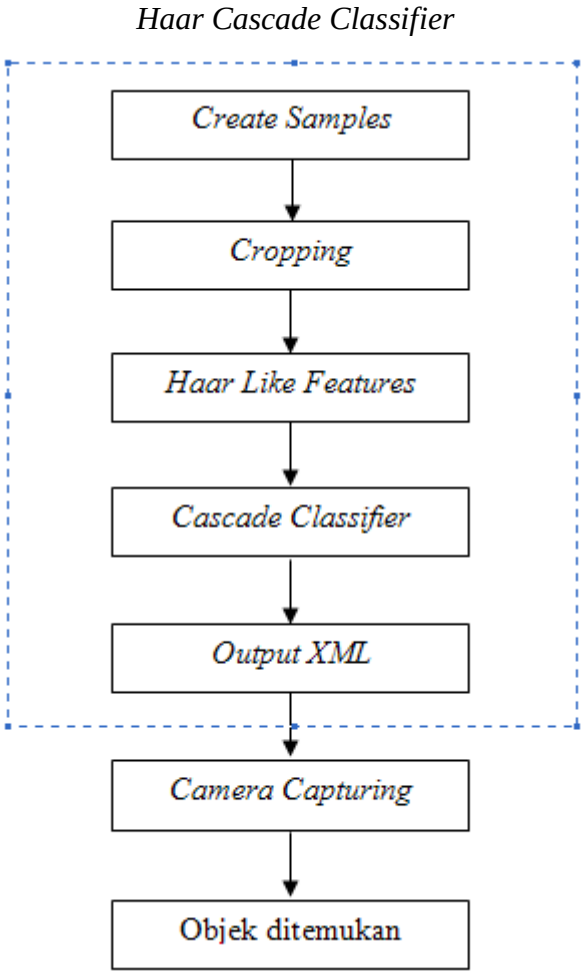
**Gambar 3.6 Proses pengubahan citra Grayscale ke Citra Biner**

Ketiga tahapan ini merupakan tahapan pengenalan pola mata yang digunakan oleh aplikasi pendeteksi mata mengantuk. Ketiga tahapan ini dibangun

dengan mengunakan emgu cv. Emgu cv telah mempasilitasi proses ini sehinga dengan mudah dapat direalisasikan di aplikasi mata mengantuk ini.

### 3.1.5 Tahap Deteksi Mata

Dalam skema penelitian ini akan diperlihatkan urutan dari saat *training data* hingga dapat terdeteksi, penelitian yang dilakukan adalah untuk membuat sebuah perangkat lunak yang dapat digunakan untuk mendeteksi mata yang mengantuk secara *real time* dengan menggunakan *Haar Cascade Classifier*, daerah mata yang terdeteksi dapat digunakan untuk mengetahui apakah seseorang dalam keadaan mata tertutup atau tidak, secara cepat dan efisien. Skema deteksi mata tertutup pada penelitian ini dapat dilihat pada Gambar 3.1.

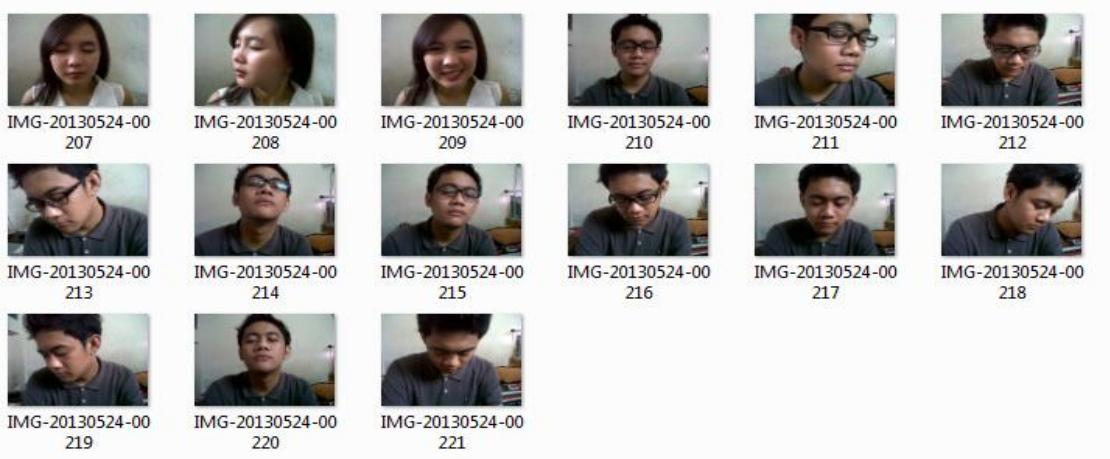



**Gambar 3.7 Skema deteksi mata mengantuk**

**1. *Create Samples***

Pada bagian ini diperlukan 2 tipe gambar objek dalam proses *training* yang akan dilakukan, yaitu :

a. *Positive samples* merupakan gambar objek yang ingin dideteksi, apabila ingin mendeteksi wajah mengantuk maka *positive samples* ini berisi dengan gambar wajah dengan kondisi mata mengantuk, begitu pula dengan objek lain yang ingin dikenali. Contoh gambar *positive samples* pada Gambar 3.8




**Gambar 3.8 Contoh gambar positif**

b. *Negative samples* merupakan gambar selain objek, umunya berupa gambar *background* (tembok, pemandangan, lantai, dan gambar lainnya). Contoh gambar *negative samples* dapat dilihat pada Gambar 3.9

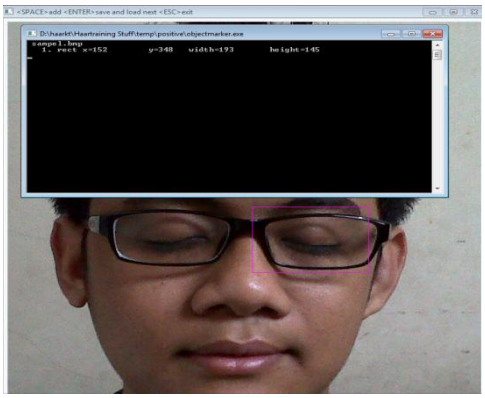



**Gambar 3.9 Contoh gambar negatif**

Resolusi untuk *positive samples* dan *negative samples* bernilai sama dengan resolusi kamera yaitu 320x240 piksel. Ini dilakukan untuk mengurangi *error* dalam proses pengklasifikasiannya.

**2. *Cropping Image***

Proses c*ropping* ini adalah untuk menentukan objek-objek yang akan digunakan dalam proses *Haar training.* Dari sekumpulan gambar positif tersebut akan dilakukan *cropping* secara manual terhadap objek spesifik yang akan dideteksi. Pada proses *Haar training*, objek yang telah di *cropping* tersebut akan dikonversikan ukurannya menggunakan teknik *resize* (umumnya menggunakan citra berukuran 24x24 piksel) pada keseluruhan objek dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti mata mengantuk atau tidak. Teknik *resize* ini menunjukkan bahwa setiap citra yang akan diklasifikasi, berapapun ukurannya, akan diubah terlebih dahulu ke dalam ukuran 24x24. Misal, terdapat gambar berukuran 40x40, maka gambar tersebut akan dikonversi menjadi berukuran 24x24. Demikian pula, jika terdapat gambar yang lebih kecil dari 24x24, misalnya 10x10, maka gambar tersebut akan diubah ke dalam ukuran 24x24 [3]. Seperti pada gambar 3.10:



**Gambar 3.10 Cropping Manual**

**3. *Cascaded Classifier***

*Cascaded classifier* merupakan suatu metode pengklasifikasian bertingkat, dimana input dari setiap tingkatan merupakan *output* dari tingkatan sebelumnya. Pada *classifier* tingkat pertama, yang menjadi inputan adalah seluruh citra. Semua citra yang berhasil melewati *classifier* pertama akan dilanjutkan ke classifer ke dua, dan seterusnya. Apabila suatu citra berhasil melewati semua tingkat

*classifier*, maka citra tersebut dinyatakan sebagai objek. Sedangkan untuk citra yang gagal melewati suatu *classifier* akan langsung dieliminasi dan dinyatakan sebagai bukan objek (tidak akan diproses lagi). Hal ini sangat mempercepat proses pengklasifikasian, karena jumlah inputan yang diterima di setiap *classifier* akan semakin berkurang [3].

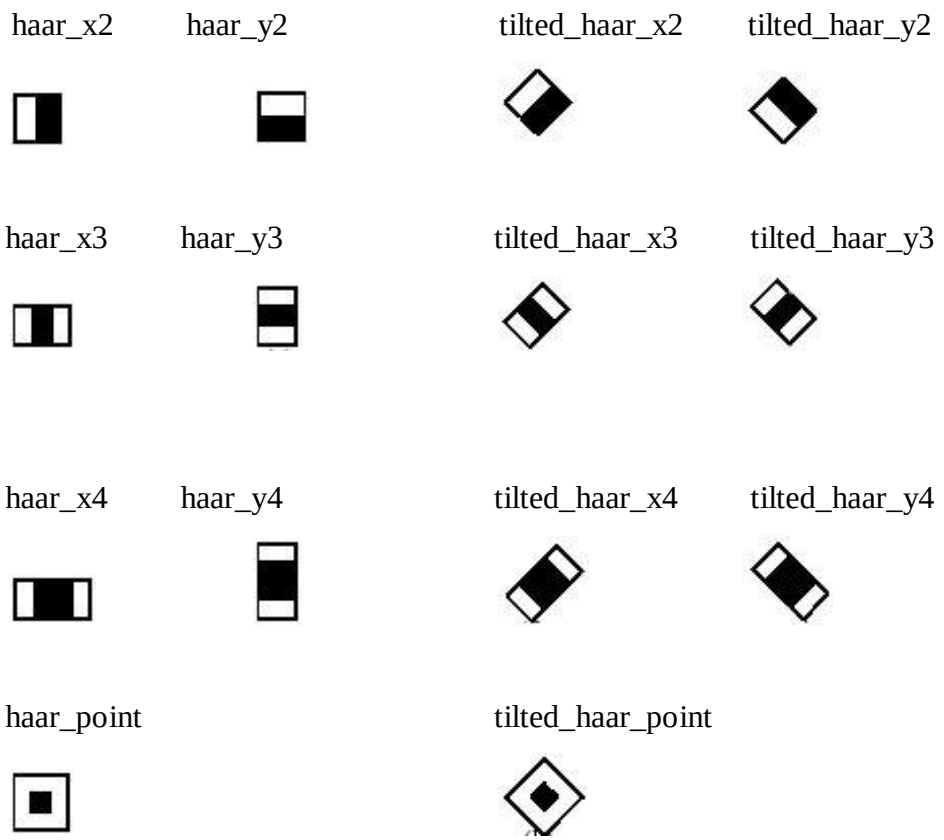
*Cascaded classifier* dirancang sedemikian rupa untuk meningkatkan tingkat pendeteksian dan mengurangi jumlah *false positive* (citra negatif yangdianggap sebagai citra positif). Setiap tingkatan *classifier* merupakan representasi hasil dari algoritma *boosting*. Jadi, disetiap tingkat *classifer* memiliki sejumlah *weak classifiers*. Setiap *weak classifier* merupakan nilai-nilai fitur Haar yang digunakan (jenis, ukuran, dan lokasi), nilai *threshold* terbaik untuk setiap fitur, serta nilai batasan setiap fitur tersebut.

Semakin tinggi tingkat *classifer*, semakin banyak pula jumlah *weak classifier* yang ada. Hal ini mengakibatkan semakin sulitnya suatu citra untuk berhasil melewati tingkatan *classifier* tersebut, sehingga jumlah citra yang dieliminasi akan semakin banyak,dan jumlah citra yang berhasil lolos ke *classifier* tingkat selanjutnya akan semakin sedikit.

Oleh karena semakin sedikit citra yang berhasil lolos ke *classifier* selanjutnya, maka semakin sedikit pula jumlah *false positive* yang berhasil lolos. Dengan berkurangnya *false positive*,tingkat keakuratan pendeteksian pun meningkat. Jadi, semakin banyak tingkat *classifier* didalam suatu *cascaded classifier*, maka semakin akurat hasil yang akan didapatkan.

**4. *Haar Like Features***

Algoritma Haar menggunakan metode statistikal dalam melakukan pendeteksian objek. Tiap *feature* dari *Haar-like feature* didefinisikan pada bentuk dari *feature*, diantaranya koordinat dari *feature* dan juga ukuran dari *feature* tersebut. *Feature* yang akan di *training* adalah *feature* yang didapat dari proses *cropping*. Adapun macam-macam variasi *feature haar* :

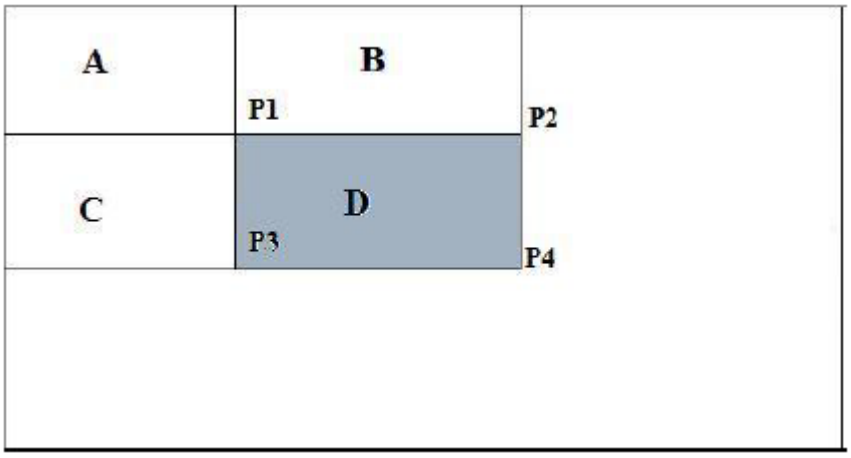



**Gambar 3.11 *Haar like feature***

Tiap *feature* terbagi atas dua bagian, yaitu bagian berwarna putih dan bagian berwarna hitam. Penggunaan fitur dilakukan karena pemrosesan fitur berlangsung lebih cepat dibandingkan pemrosesan citra perpiksel sehingga cocok untuk diterapkan pada proses pendeteksian objek secara *real time.* Pada penelitian ini untuk menghitung nilai fitur Haar pada sebuah citra dan pada skala yang berbeda secara cepat menggunakan satu teknik yang disebut *integral image.*

Secara umum integrating berarti menggabungkan unit-unit kecil menjadi satu. Dalam hal ini unit-unit kecil adalah nilai piksel. Nilai integral suatu piksel adalah jumlah dari semua piksel diatas dan di kiri piksel tersebut. Seluruh citra dapat digabungkan dengan operasi nilai yang lebih sedikit tiap pikselnya. *Integral image* sangat membantu dalam perhitungan fitur *Haar-like*. Dengan menggunakan *integral image*, perhitungan fitur *Haar-like* dapat dilakukan dengan sangat cepat.

Rumus *integral image* : $P(x,y) = \sum_{x' \le x, y' \le y} i(x', y')$ .....(3.3)




**Gambar 3.12 *Integral image[3]***

$P_1 = A, P_2 = A + B, P_3 = A + C, P_4 = A + B + C + D$

$P_1 + P_4 - P_2 - P_3 = A + A + B + C + D - A - B - A - C = D$

Contoh perhitungan nilai fitur menggunakan *integral image* adalah sebagai berikut:

Diberikan sebuah citra A =

| | | |
|---|---|---|
| A | D | G |
| B | E | H |
| C | F | I |

| | | |
|---|---|---|
| 3 | 9 | 18 |
| 12 | 20 | 36 |
| 17 | 26 | 50 |

**Gambar 3.13 contoh citra 3x3**

Jumlah nilai piksel pada daerah hitam = F+A-(C+D) = 26+3-(17+9) = 3

Contoh perhitungan nilai fitur 2x4 dalam sebuh citra 5x5 menggunakan *integral image* :

| | | | | |
|---|---|---|---|---|
| 3 | 9 | 18 | 26 | 33 |
| 12 | 20 | 36 | 45 | 61 |
| 17 | 26 | 50 | 66 | 86 |
| 19 | 32 | 58 | 79 | 108 |
| 26 | 45 | 73 | 95 | 128 |

**Gambar 3.14 contoh citra 5x5**

Nilai fitur = | (total piksel hitam) – (total piksel putih) |

= | [20+0-(0+0)] – [45+0-(20+0)] |

= | 20 – 25 |

= 5

| | | | | |
|---|---|---|---|---|
| 3 | 9 | 18 | 26 | 33 |
| 12 | 20 | 36 | 45 | 61 |
| 17 | 26 | 50 | 66 | 86 |
| 19 | 32 | 58 | 79 | 108 |
| 26 | 45 | 73 | 95 | 128 |

**Gambar 3.15 contoh citra 5x5**

Nilai fitur = | (total piksel hitam) – (total piksel putih) |

= | [73+3-(26+18)] – [128+18-(73+33)] |

= | 32 – 40 |

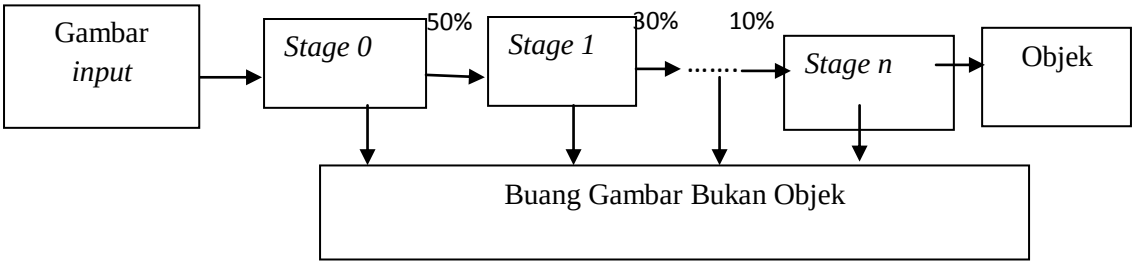
= 8

### 3.1.6 Algoritma AdaBoost

Metode *machine learning* yang disebut AdaBoost digunakan dalam memilih fitur Haar yang spesifik. Hal ini dilakukan dengan cara mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut.

Algoritma Adaboost adalah algoritma yang berusaha membangun *strong classifier* dengan mengkombinasikan sejumlah *simple* atau *weak classifier* secara

linier. *Stage classifier* dibangun dengan menggunakan algoritma *adaptive-boos*t (AdaBoost). Di dalam *stage classifier* terdapat proses pengklasifikasian fitur objek menggunakan algoritma AdaBoost. Algoritma AdaBoost digunakan untuk meningkatkan performa pengklasifikasian fitur. Algoritma tersebut mengkombinasikan performance banyak *weak classifier* untuk menghasilkan *strong classifier*. *Weak classifier* dalam hal ini adalah nilai dari *haar-like feature*. Alur proses pendeteksian pada *Haar cascade classifier* dapat dilihat pada Gambar 3.16




**Gambar 3.16 Alur proses *Haar cascade classifier***

Algoritma AdaBoost yang dipakai untuk pengklasifikasian adalah sebagai berikut:

Bobot awal $w_{1,y_i} = \frac{1}{2m}, w_{1,y_i} = \frac{1}{2l},$ .....(3.4)

dengan *m* = jumlah gambar negatif dengan $y_i = 0$ untuk gambar negatif

dengan *l* = jumlah gambar positif dengan $y_i = 1$ untuk gambar positif.

Contoh : *l* = 5, m = 10

$$w_{1,0} = \frac{1}{2 \times 10} = 0.05, w_{1,1} = \frac{1}{2 \times 5} = 0.1 \quad \text{.....(3.5)}$$

Untuk t = 1,2,3…T , dimana t adalah iterasi ke t untuk gambar positif.

Untuk j = 1,2,3…J , dimana j adalah iterasi ke j untuk gambar negatif.

$h_t(x)$ merupakan nilai fitur gambar positif.

$h_j(x)$ merupakan nilai fitur gambar negatif.

Untuk mendapatkan nilai *error rate* setiap *weak classifier*. Maka untuk setiap fitur dilakukan proses sebagai berikut:

Untuk gambar positif : $\in_t = (\sum_t^T w_{t,i}) \mid h_t(x) - y_i \mid$ .....(3.6)

Untuk gambar negatif : $\in_j = (\sum_j^J w_{j,i}) \mid h_j(x) - y_i \mid$ .....(3.7)

Misal diberikan citra positif (24x24)piksel yang memiliki nilai-nilai sebagai berikut:

1. Citra positif ke-1

| | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 33 | 34 | 34 | 34 | 35 | 36 | 37 | 37 | 37 | 38 | 40 | 42 | 43 | 45 | 47 | 49 | 54 | 56 | 57 | 57 | 59 | 60 | 59 | 56 |
| 30 | 30 | 32 | 32 | 33 | 34 | 35 | 34 | 35 | 35 | 35 | 35 | 35 | 34 | 39 | 42 | 48 | 54 | 58 | 59 | 59 | 61 | 61 | 61 |
| 29 | 29 | 30 | 30 | 30 | 31 | 31 | 31 | 30 | 30 | 31 | 30 | 30 | 29 | 32 | 35 | 41 | 49 | 57 | 59 | 58 | 60 | 59 | 59 |
| 28 | 26 | 27 | 27 | 27 | 27 | 26 | 26 | 26 | 25 | 25 | 25 | 25 | 27 | 26 | 28 | 34 | 41 | 51 | 54 | 57 | 59 | 59 | 58 |
| 27 | 25 | 24 | 24 | 24 | 23 | 24 | 23 | 22 | 21 | 20 | 20 | 22 | 24 | 23 | 24 | 26 | 31 | 38 | 42 | 50 | 54 | 58 | 59 |
| 25 | 24 | 21 | 21 | 21 | 21 | 20 | 20 | 19 | 18 | 18 | 18 | 21 | 22 | 23 | 23 | 25 | 26 | 31 | 34 | 43 | 50 | 55 | 59 |
| 23 | 22 | 20 | 19 | 19 | 18 | 18 | 16 | 17 | 17 | 19 | 20 | 22 | 23 | 25 | 26 | 26 | 28 | 31 | 33 | 37 | 46 | 54 | 59 |
| 23 | 22 | 21 | 20 | 20 | 19 | 20 | 18 | 21 | 21 | 24 | 25 | 26 | 27 | 25 | 28 | 28 | 29 | 32 | 34 | 37 | 46 | 52 | 58 |
| 22 | 22 | 23 | 22 | 22 | 23 | 23 | 23 | 24 | 25 | 27 | 28 | 29 | 29 | 28 | 30 | 30 | 29 | 30 | 31 | 35 | 42 | 49 | 55 |
| 25 | 25 | 26 | 26 | 24 | 24 | 23 | 22 | 26 | 27 | 28 | 30 | 31 | 32 | 33 | 32 | 31 | 30 | 33 | 34 | 33 | 41 | 47 | 54 |
| 26 | 26 | 26 | 25 | 24 | 24 | 23 | 23 | 26 | 27 | 29 | 31 | 32 | 33 | 33 | 34 | 33 | 31 | 33 | 33 | 34 | 40 | 47 | 54 |
| 27 | 26 | 23 | 23 | 23 | 23 | 23 | 24 | 24 | 25 | 28 | 29 | 30 | 32 | 34 | 36 | 37 | 34 | 34 | 34 | 34 | 40 | 46 | 54 |
| 26 | 26 | 23 | 23 | 23 | 22 | 23 | 24 | 24 | 25 | 27 | 28 | 30 | 32 | 35 | 38 | 41 | 38 | 37 | 37 | 35 | 42 | 47 | 55 |
| 26 | 25 | 24 | 24 | 23 | 23 | 23 | 23 | 25 | 26 | 28 | 29 | 30 | 31 | 33 | 36 | 39 | 38 | 38 | 37 | 37 | 42 | 48 | 55 |
| 28 | 26 | 25 | 24 | 23 | 22 | 21 | 21 | 24 | 23 | 24 | 25 | 25 | 27 | 28 | 31 | 33 | 34 | 35 | 37 | 37 | 43 | 48 | 55 |
| 29 | 28 | 28 | 26 | 21 | 18 | 18 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 23 | 25 | 26 | 27 | 31 | 32 | 39 | 47 | 52 | 59 |
| 34 | 29 | 28 | 27 | 23 | 22 | 21 | 20 | 18 | 17 | 19 | 19 | 20 | 20 | 19 | 22 | 26 | 29 | 34 | 36 | 41 | 47 | 53 | 59 |
| 40 | 32 | 29 | 28 | 27 | 27 | 26 | 24 | 23 | 23 | 23 | 24 | 24 | 24 | 23 | 25 | 28 | 31 | 36 | 39 | 44 | 50 | 55 | 62 |
| 47 | 36 | 30 | 28 | 27 | 27 | 27 | 25 | 26 | 26 | 26 | 26 | 26 | 26 | 25 | 29 | 32 | 35 | 40 | 43 | 47 | 52 | 57 | 62 |
| 53 | 44 | 35 | 33 | 30 | 29 | 29 | 28 | 27 | 27 | 28 | 28 | 29 | 29 | 29 | 33 | 37 | 41 | 48 | 52 | 50 | 54 | 59 | 65 |
| 58 | 49 | 40 | 38 | 31 | 31 | 30 | 30 | 30 | 29 | 31 | 32 | 34 | 34 | 36 | 38 | 41 | 44 | 49 | 51 | 53 | 56 | 60 | 66 |
| 59 | 53 | 47 | 44 | 37 | 34 | 35 | 34 | 33 | 32 | 31 | 31 | 33 | 36 | 36 | 38 | 41 | 45 | 50 | 51 | 54 | 57 | 58 | 63 |
| 63 | 58 | 52 | 48 | 40 | 37 | 37 | 36 | 36 | 34 | 33 | 33 | 34 | 35 | 35 | 36 | 40 | 43 | 48 | 50 | 52 | 55 | 58 | 63 |
| 64 | 61 | 57 | 53 | 45 | 42 | 41 | 39 | 38 | 37 | 35 | 34 | 34 | 35 | 34 | 35 | 37 | 42 | 47 | 47 | 50 | 53 | 57 | 62 |

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

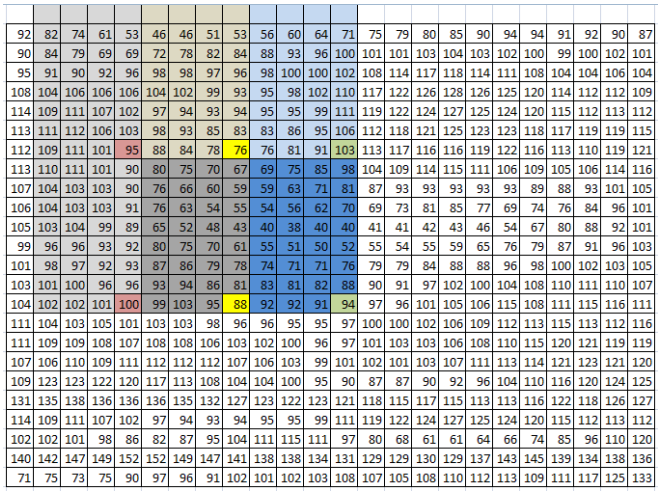
$h_t(x)$ = | (26+21-23-18)-(31+18-26-22) |

= 5

Maka nilai *error rate* yang didapatkan adalah sebagai berikut :

$\in_t = (0,1) \mid 5-1 \mid = 0,4$

2. Citra positif ke-2



| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 92 | 82 | 74 | 61 | 53 | 46 | 46 | 51 | 53 | 56 | 60 | 64 | 71 | 75 | 79 | 80 | 85 | 90 | 94 | 94 | 91 | 92 | 90 | 87 |
| 90 | 84 | 79 | 69 | 69 | 72 | 78 | 82 | 84 | 88 | 93 | 96 | 100 | 101 | 101 | 103 | 104 | 103 | 102 | 100 | 99 | 100 | 102 | 101 |
| 95 | 91 | 90 | 92 | 96 | 98 | 98 | 97 | 96 | 98 | 100 | 100 | 102 | 108 | 114 | 117 | 118 | 114 | 111 | 108 | 104 | 104 | 106 | 104 |
| 108 | 104 | 106 | 106 | 106 | 104 | 102 | 99 | 93 | 95 | 98 | 102 | 110 | 117 | 122 | 126 | 128 | 126 | 125 | 120 | 114 | 112 | 112 | 109 |
| 114 | 109 | 111 | 107 | 102 | 97 | 94 | 93 | 94 | 95 | 95 | 99 | 111 | 119 | 122 | 124 | 127 | 125 | 124 | 120 | 115 | 112 | 113 | 112 |
| 113 | 111 | 112 | 106 | 103 | 98 | 93 | 85 | 83 | 83 | 86 | 95 | 106 | 112 | 118 | 121 | 125 | 123 | 123 | 118 | 117 | 119 | 119 | 115 |
| 112 | 109 | 111 | 101 | 95 | 88 | 84 | 78 | 76 | 76 | 81 | 91 | 103 | 113 | 117 | 116 | 116 | 119 | 122 | 116 | 113 | 110 | 119 | 121 |
| 113 | 110 | 111 | 101 | 90 | 80 | 75 | 70 | 67 | 69 | 75 | 85 | 98 | 104 | 109 | 114 | 115 | 111 | 106 | 109 | 105 | 106 | 114 | 116 |
| 107 | 104 | 103 | 103 | 90 | 76 | 66 | 60 | 59 | 59 | 63 | 71 | 81 | 87 | 93 | 93 | 93 | 93 | 93 | 89 | 88 | 93 | 101 | 105 |
| 106 | 104 | 103 | 103 | 91 | 76 | 63 | 54 | 55 | 54 | 56 | 62 | 70 | 69 | 73 | 81 | 85 | 77 | 69 | 74 | 76 | 84 | 96 | 101 |
| 105 | 103 | 104 | 99 | 89 | 65 | 52 | 48 | 43 | 40 | 38 | 40 | 40 | 41 | 41 | 42 | 43 | 46 | 54 | 67 | 80 | 88 | 92 | 101 |
| 99 | 96 | 96 | 93 | 92 | 80 | 75 | 70 | 61 | 55 | 51 | 50 | 52 | 55 | 54 | 55 | 59 | 65 | 76 | 79 | 87 | 91 | 96 | 103 |
| 101 | 98 | 97 | 92 | 93 | 87 | 86 | 79 | 78 | 74 | 71 | 71 | 76 | 79 | 79 | 84 | 88 | 88 | 96 | 98 | 100 | 102 | 103 | 105 |
| 103 | 101 | 100 | 96 | 96 | 93 | 94 | 86 | 81 | 83 | 81 | 82 | 88 | 90 | 91 | 97 | 102 | 100 | 104 | 108 | 110 | 111 | 110 | 107 |
| 104 | 102 | 102 | 101 | 100 | 99 | 103 | 95 | 88 | 92 | 92 | 91 | 94 | 97 | 96 | 101 | 105 | 106 | 115 | 108 | 111 | 115 | 116 | 111 |
| 111 | 104 | 103 | 105 | 101 | 103 | 103 | 98 | 96 | 96 | 95 | 95 | 97 | 100 | 100 | 102 | 106 | 109 | 112 | 113 | 115 | 113 | 112 | 116 |
| 111 | 109 | 109 | 108 | 107 | 108 | 108 | 106 | 103 | 102 | 100 | 96 | 97 | 101 | 103 | 103 | 106 | 108 | 110 | 115 | 120 | 121 | 119 | 119 |
| 107 | 106 | 110 | 109 | 111 | 112 | 112 | 112 | 107 | 106 | 103 | 99 | 101 | 102 | 101 | 103 | 107 | 111 | 113 | 114 | 121 | 123 | 121 | 120 |
| 109 | 123 | 123 | 122 | 120 | 117 | 113 | 108 | 104 | 104 | 100 | 95 | 90 | 87 | 87 | 90 | 92 | 96 | 104 | 110 | 116 | 120 | 124 | 125 |
| 131 | 135 | 138 | 136 | 136 | 136 | 135 | 132 | 127 | 123 | 122 | 123 | 121 | 118 | 115 | 117 | 115 | 113 | 113 | 116 | 122 | 118 | 126 | 127 |
| 114 | 109 | 111 | 107 | 102 | 97 | 94 | 93 | 94 | 95 | 95 | 99 | 111 | 119 | 122 | 124 | 127 | 125 | 124 | 120 | 115 | 112 | 113 | 112 |
| 102 | 102 | 101 | 98 | 86 | 82 | 87 | 95 | 104 | 111 | 115 | 111 | 97 | 80 | 68 | 61 | 61 | 64 | 66 | 74 | 85 | 96 | 110 | 120 |
| 140 | 142 | 147 | 149 | 152 | 152 | 149 | 147 | 141 | 138 | 138 | 134 | 131 | 129 | 129 | 130 | 129 | 137 | 143 | 145 | 139 | 134 | 138 | 136 |
| 71 | 75 | 73 | 75 | 90 | 97 | 96 | 91 | 102 | 101 | 102 | 103 | 108 | 107 | 105 | 108 | 110 | 112 | 113 | 109 | 111 | 117 | 125 | 133 |

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

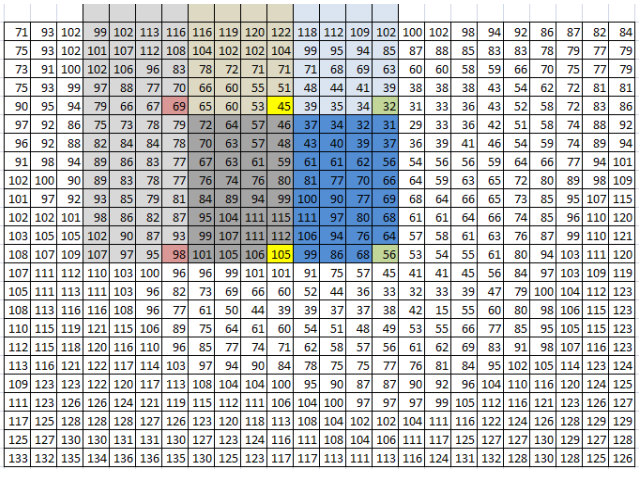
$h_t(x)$ = (88+95-100-76)-(94+76-88-103)

= 28

Maka nilai *error rate* yang didapatkan adalah sebagai berikut :

$\in_t = (0,1)\,|\,28-1|=2,7$

3. Citra positif ke-3



| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 71 | 93 | 102 | 99 | 102 | 113 | 116 | 116 | 119 | 120 | 122 | 118 | 112 | 109 | 102 | 100 | 102 | 98 | 94 | 92 | 86 | 87 | 82 | 84 |
| 75 | 93 | 102 | 101 | 107 | 112 | 108 | 104 | 102 | 102 | 104 | 99 | 95 | 94 | 85 | 87 | 88 | 85 | 83 | 83 | 78 | 79 | 77 | 79 |
| 73 | 91 | 100 | 102 | 106 | 96 | 83 | 78 | 72 | 71 | 71 | 71 | 68 | 69 | 63 | 60 | 60 | 58 | 59 | 66 | 70 | 75 | 77 | 79 |
| 75 | 93 | 99 | 97 | 88 | 77 | 70 | 66 | 60 | 55 | 51 | 48 | 44 | 41 | 39 | 38 | 38 | 38 | 43 | 54 | 62 | 72 | 81 | 81 |
| 90 | 95 | 94 | 79 | 66 | 67 | 69 | 65 | 60 | 53 | 45 | 39 | 35 | 34 | 32 | 31 | 33 | 36 | 43 | 52 | 58 | 72 | 83 | 86 |
| 97 | 92 | 86 | 75 | 73 | 78 | 79 | 72 | 64 | 57 | 46 | 37 | 34 | 32 | 31 | 29 | 33 | 36 | 42 | 51 | 58 | 74 | 88 | 92 |
| 96 | 92 | 88 | 82 | 84 | 84 | 78 | 70 | 63 | 57 | 48 | 43 | 40 | 39 | 37 | 36 | 39 | 41 | 46 | 54 | 59 | 74 | 89 | 94 |
| 91 | 98 | 94 | 89 | 86 | 83 | 77 | 67 | 63 | 61 | 59 | 61 | 61 | 62 | 56 | 54 | 56 | 56 | 59 | 64 | 66 | 77 | 94 | 101 |
| 102 | 100 | 90 | 89 | 83 | 78 | 77 | 76 | 74 | 76 | 80 | 81 | 77 | 70 | 66 | 64 | 59 | 63 | 65 | 72 | 80 | 89 | 98 | 109 |
| 101 | 97 | 92 | 93 | 85 | 79 | 81 | 84 | 89 | 94 | 99 | 100 | 90 | 77 | 69 | 68 | 64 | 66 | 65 | 73 | 85 | 95 | 107 | 115 |
| 102 | 102 | 101 | 98 | 86 | 82 | 87 | 95 | 104 | 111 | 115 | 111 | 97 | 80 | 68 | 61 | 61 | 64 | 66 | 74 | 85 | 96 | 110 | 120 |
| 103 | 105 | 105 | 102 | 90 | 87 | 93 | 99 | 107 | 111 | 112 | 106 | 94 | 76 | 64 | 57 | 58 | 61 | 63 | 76 | 87 | 99 | 110 | 121 |
| 108 | 107 | 109 | 107 | 97 | 95 | 98 | 101 | 105 | 106 | 105 | 99 | 86 | 68 | 56 | 53 | 54 | 55 | 61 | 80 | 94 | 103 | 111 | 120 |
| 107 | 111 | 112 | 110 | 103 | 100 | 96 | 96 | 99 | 101 | 101 | 91 | 75 | 57 | 45 | 41 | 41 | 45 | 56 | 84 | 97 | 103 | 109 | 119 |
| 105 | 111 | 113 | 111 | 103 | 96 | 82 | 73 | 69 | 66 | 60 | 52 | 44 | 36 | 33 | 32 | 33 | 39 | 47 | 79 | 100 | 104 | 112 | 123 |
| 108 | 113 | 116 | 116 | 108 | 96 | 77 | 61 | 50 | 44 | 39 | 39 | 37 | 37 | 38 | 42 | 15 | 55 | 60 | 80 | 98 | 106 | 115 | 123 |
| 110 | 115 | 119 | 121 | 115 | 106 | 89 | 75 | 64 | 61 | 60 | 54 | 51 | 48 | 49 | 53 | 55 | 66 | 77 | 85 | 95 | 105 | 115 | 123 |
| 112 | 115 | 118 | 120 | 116 | 110 | 96 | 85 | 77 | 74 | 71 | 62 | 58 | 57 | 56 | 61 | 62 | 69 | 83 | 91 | 98 | 107 | 116 | 123 |
| 113 | 116 | 121 | 122 | 117 | 114 | 103 | 97 | 94 | 90 | 84 | 78 | 75 | 75 | 77 | 76 | 81 | 84 | 95 | 102 | 105 | 114 | 123 | 124 |
| 109 | 123 | 123 | 122 | 120 | 117 | 113 | 108 | 104 | 104 | 100 | 95 | 90 | 87 | 87 | 90 | 92 | 96 | 104 | 110 | 116 | 120 | 124 | 125 |
| 111 | 123 | 126 | 126 | 124 | 121 | 119 | 115 | 112 | 111 | 106 | 104 | 100 | 97 | 97 | 97 | 99 | 105 | 112 | 116 | 121 | 123 | 126 | 127 |
| 117 | 125 | 128 | 128 | 127 | 126 | 123 | 120 | 118 | 113 | 108 | 104 | 102 | 102 | 104 | 111 | 116 | 122 | 124 | 126 | 128 | 129 | 129 | |
| 125 | 127 | 130 | 130 | 131 | 131 | 130 | 127 | 123 | 124 | 116 | 111 | 108 | 104 | 106 | 111 | 117 | 125 | 127 | 127 | 130 | 129 | 127 | 128 |
| 133 | 132 | 135 | 134 | 136 | 136 | 135 | 130 | 125 | 123 | 117 | 117 | 113 | 111 | 113 | 116 | 124 | 131 | 132 | 128 | 130 | 128 | 125 | 126 |

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_t(x)$ = (105+69-98-45)-(56+45-105-32)

= 67

Maka nilai *error rate* yang didapatkan adalah sebagai berikut :

$\in_t = (0,1) \mid 67 - 1 \mid = 6,6$

4. Citra positif ke-4

| | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 95 | 98 | 101 | 103 | 105 | 106 | 107 | 108 | 109 | 111 | 112 | 119 | 122 | 122 | 123 | 119 | 121 | 118 | 121 | 127 | 127 | 136 | 136 | 127 |
| 99 | 100 | 102 | 105 | 107 | 110 | 113 | 114 | 114 | 114 | 117 | 122 | 127 | 129 | 129 | 123 | 127 | 121 | 118 | 126 | 131 | 130 | 124 | 118 |
| 102 | 102 | 104 | 108 | 109 | 114 | 120 | 122 | 120 | 119 | 122 | 124 | 129 | 131 | 127 | 121 | 123 | 119 | 119 | 124 | 125 | 123 | 119 | 111 |
| 103 | 102 | 106 | 111 | 115 | 120 | 126 | 126 | 121 | 117 | 123 | 122 | 125 | 126 | 118 | 113 | 113 | 111 | 115 | 120 | 116 | 113 | 112 | 103 |
| 94 | 100 | 106 | 112 | 117 | 121 | 115 | 105 | 107 | 120 | 110 | 113 | 122 | 118 | 113 | 109 | 110 | 110 | 107 | 107 | 109 | 112 | 109 | 96 |
| 96 | 99 | 103 | 105 | 110 | 102 | 93 | 87 | 82 | 78 | 78 | 80 | 89 | 91 | 90 | 101 | 102 | 100 | 99 | 99 | 101 | 93 | 93 | 87 |
| 96 | 92 | 84 | 79 | 72 | 72 | 72 | 72 | 69 | 65 | 62 | 57 | 58 | 57 | 56 | 58 | 57 | 57 | 55 | 56 | 59 | 58 | 61 | 68 |
| 96 | 93 | 87 | 80 | 73 | 81 | 83 | 78 | 74 | 76 | 65 | 55 | 50 | 44 | 40 | 39 | 37 | 36 | 36 | 38 | 41 | 42 | 47 | 56 |
| 99 | 97 | 90 | 82 | 80 | 85 | 85 | 81 | 76 | 76 | 70 | 58 | 49 | 39 | 34 | 34 | 32 | 31 | 30 | 33 | 37 | 40 | 45 | 51 |
| 98 | 89 | 81 | 80 | 83 | 87 | 90 | 90 | 86 | 86 | 80 | 71 | 69 | 71 | 64 | 58 | 50 | 46 | 45 | 44 | 50 | 48 | 55 | 61 |
| 101 | 95 | 89 | 86 | 85 | 90 | 97 | 99 | 98 | 97 | 95 | 98 | 101 | 98 | 90 | 85 | 74 | 66 | 63 | 59 | 63 | 69 | 75 | 80 |
| 106 | 103 | 99 | 95 | 99 | 100 | 105 | 109 | 109 | 107 | 114 | 112 | 106 | 99 | 94 | 84 | 75 | 68 | 66 | 64 | 69 | 73 | 84 | 90 |
| 109 | 110 | 109 | 103 | 106 | 100 | 97 | 100 | 97 | 93 | 77 | 81 | 82 | 74 | 61 | 68 | 61 | 61 | 61 | 64 | 72 | 77 | 86 | 92 |
| 111 | 114 | 114 | 107 | 101 | 92 | 90 | 94 | 98 | 96 | 99 | 92 | 88 | 80 | 64 | 54 | 50 | 52 | 56 | 61 | 71 | 81 | 89 | 94 |
| 113 | 112 | 108 | 101 | 94 | 79 | 64 | 58 | 50 | 43 | 34 | 35 | 36 | 36 | 35 | 35 | 40 | 41 | 40 | 50 | 65 | 86 | 91 | 99 |
| 117 | 115 | 110 | 104 | 95 | 76 | 59 | 55 | 54 | 53 | 57 | 57 | 55 | 54 | 52 | 49 | 50 | 57 | 62 | 64 | 60 | 91 | 99 | 104 |
| 121 | 122 | 122 | 118 | 109 | 96 | 83 | 80 | 80 | 80 | 81 | 77 | 71 | 66 | 64 | 68 | 67 | 69 | 73 | 83 | 90 | 95 | 104 | 107 |
| 126 | 127 | 127 | 122 | 121 | 116 | 109 | 103 | 95 | 90 | 88 | 85 | 83 | 82 | 83 | 87 | 94 | 97 | 92 | 93 | 102 | 102 | 107 | 111 |
| 128 | 133 | 136 | 133 | 125 | 125 | 124 | 119 | 114 | 112 | 111 | 106 | 104 | 104 | 106 | 106 | 105 | 103 | 100 | 104 | 111 | 110 | 110 | 117 |
| 131 | 135 | 138 | 136 | 136 | 136 | 135 | 132 | 127 | 123 | 122 | 123 | 121 | 118 | 115 | 117 | 115 | 113 | 113 | 116 | 122 | 118 | 126 | 127 |
| 137 | 137 | 139 | 144 | 140 | 149 | 142 | 140 | 136 | 133 | 129 | 128 | 126 | 123 | 120 | 120 | 119 | 119 | 121 | 125 | 129 | 131 | 134 | 132 |
| 138 | 136 | 139 | 146 | 149 | 148 | 147 | 145 | 139 | 136 | 129 | 129 | 125 | 124 | 122 | 122 | 123 | 127 | 130 | 134 | 134 | 133 | 136 | 133 |
| 140 | 142 | 147 | 149 | 152 | 152 | 149 | 147 | 141 | 138 | 138 | 134 | 131 | 129 | 129 | 130 | 129 | 137 | 143 | 145 | 139 | 134 | 138 | 136 |
| 135 | 141 | 148 | 144 | 151 | 151 | 151 | 150 | 146 | 143 | 140 | 137 | 132 | 132 | 131 | 136 | 135 | 143 | 150 | 150 | 143 | 137 | 142 | 140 |

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_t(x)$ = | (35+72-58-57)-(35+57-35-58) ||

= 7

Maka nilai *error rate* yang didapatkan adalah sebagai berikut :

$\in_t = (0,1) \mid 7 - 1 \mid = 0,6$

5. Citra positif ke-5

| | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 33 | 34 | 34 | 34 | 35 | 36 | 37 | 37 | 37 | 38 | 40 | 42 | 43 | 45 | 47 | 49 | 54 | 56 | 57 | 57 | 59 | 60 | 59 | 56 |
| 30 | 30 | 32 | 32 | 33 | 34 | 35 | 34 | 35 | 35 | 35 | 35 | 35 | 34 | 39 | 42 | 48 | 54 | 58 | 59 | 59 | 61 | 61 | 61 |
| 29 | 29 | 30 | 30 | 30 | 31 | 31 | 31 | 30 | 30 | 31 | 30 | 30 | 29 | 32 | 35 | 41 | 49 | 57 | 59 | 58 | 60 | 59 | 59 |
| 28 | 26 | 27 | 27 | 27 | 27 | 26 | 26 | 26 | 25 | 25 | 25 | 25 | 27 | 26 | 28 | 34 | 41 | 51 | 54 | 57 | 59 | 59 | 58 |
| 27 | 25 | 24 | 24 | 24 | 23 | 24 | 23 | 22 | 21 | 20 | 20 | 22 | 24 | 23 | 24 | 26 | 31 | 38 | 42 | 50 | 54 | 58 | 59 |
| 25 | 24 | 21 | 21 | 21 | 21 | 20 | 20 | 19 | 18 | 18 | 18 | 21 | 22 | 23 | 23 | 25 | 26 | 31 | 34 | 43 | 50 | 55 | 59 |
| 23 | 22 | 20 | 19 | 19 | 18 | 18 | 16 | 17 | 17 | 19 | 20 | 22 | 23 | 25 | 26 | 26 | 28 | 31 | 33 | 37 | 46 | 54 | 59 |
| 23 | 22 | 21 | 20 | 20 | 19 | 20 | 18 | 21 | 21 | 24 | 25 | 26 | 27 | 25 | 28 | 28 | 29 | 32 | 34 | 37 | 46 | 52 | 58 |
| 22 | 22 | 23 | 22 | 22 | 23 | 23 | 23 | 24 | 25 | 27 | 28 | 29 | 29 | 28 | 30 | 30 | 29 | 30 | 31 | 35 | 42 | 49 | 55 |
| 25 | 25 | 26 | 26 | 24 | 24 | 23 | 22 | 26 | 27 | 28 | 30 | 31 | 32 | 33 | 32 | 31 | 30 | 33 | 34 | 33 | 41 | 47 | 54 |
| 26 | 26 | 26 | 25 | 24 | 24 | 23 | 23 | 26 | 27 | 29 | 31 | 32 | 33 | 33 | 34 | 33 | 31 | 33 | 33 | 34 | 40 | 47 | 54 |
| 27 | 26 | 23 | 23 | 23 | 23 | 23 | 24 | 24 | 25 | 28 | 29 | 30 | 32 | 34 | 36 | 37 | 34 | 34 | 34 | 34 | 40 | 46 | 54 |
| 26 | 26 | 23 | 23 | 23 | 22 | 23 | 24 | 24 | 25 | 27 | 28 | 30 | 32 | 35 | 38 | 41 | 38 | 37 | 37 | 35 | 42 | 47 | 55 |
| 26 | 25 | 24 | 24 | 23 | 23 | 23 | 23 | 25 | 26 | 28 | 29 | 30 | 31 | 33 | 36 | 39 | 38 | 38 | 37 | 37 | 42 | 48 | 55 |
| 28 | 26 | 25 | 24 | 23 | 22 | 21 | 21 | 24 | 23 | 24 | 25 | 25 | 27 | 28 | 31 | 33 | 34 | 35 | 37 | 37 | 43 | 48 | 55 |
| 29 | 28 | 28 | 26 | 23 | 18 | 18 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 23 | 25 | 26 | 27 | 31 | 32 | 39 | 47 | 52 | 59 |
| 34 | 29 | 29 | 27 | 23 | 22 | 21 | 20 | 18 | 17 | 19 | 19 | 20 | 20 | 19 | 22 | 26 | 29 | 34 | 36 | 41 | 47 | 53 | 59 |
| 40 | 32 | 29 | 28 | 27 | 27 | 26 | 24 | 23 | 23 | 23 | 24 | 24 | 24 | 23 | 25 | 28 | 31 | 36 | 39 | 44 | 50 | 55 | 62 |
| 47 | 36 | 30 | 28 | 27 | 27 | 27 | 25 | 26 | 26 | 26 | 26 | 26 | 26 | 25 | 29 | 32 | 35 | 40 | 43 | 47 | 52 | 57 | 62 |
| 53 | 44 | 35 | 33 | 30 | 29 | 29 | 28 | 27 | 27 | 28 | 28 | 29 | 29 | 29 | 22 | 37 | 41 | 48 | 52 | 50 | 54 | 59 | 65 |
| 58 | 49 | 40 | 38 | 31 | 31 | 30 | 30 | 30 | 29 | 31 | 32 | 34 | 34 | 36 | 28 | 41 | 44 | 49 | 51 | 53 | 56 | 60 | 66 |
| 59 | 53 | 47 | 44 | 37 | 34 | 35 | 34 | 33 | 32 | 31 | 31 | 33 | 36 | 36 | 28 | 41 | 45 | 50 | 51 | 54 | 57 | 58 | 63 |
| 63 | 58 | 52 | 48 | 40 | 37 | 37 | 36 | 36 | 34 | 33 | 33 | 34 | 35 | 35 | 36 | 40 | 43 | 48 | 50 | 52 | 55 | 58 | 63 |
| 64 | 61 | 57 | 53 | 45 | 42 | 41 | 39 | 38 | 37 | 35 | 34 | 34 | 35 | 34 | 35 | 37 | 42 | 47 | 47 | 50 | 53 | 57 | 62 |

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_t(x)$ = (20+20-18-24)-(23+24-20-25)= 4

Maka $\in_t = (0,1) \mid 4-1 \mid = 0,3$

Jadi:

*Error rate* masing-masing citra adalah 0,4; 2,7; 6,6; dan 0,3. Sehingga untuk memudahkan pemilihan fitur yang terbaik digunakan klasifikasi data *training* menggunakan *error rate* terkecil yang telah didapatkan yaitu 0,3 dimana

$$\beta_t = \frac{\in_t}{1- \in_t} = \frac{0,3}{1-0,3} = 0,428$$

*Update* bobot: $w_{t+1,i} = w_{t,i}.\beta_t$

Maka bobot setelah iterasi 1 : $w_{2,1} = 0,1 \times 0,428 = 0,0428$

$\in_t = (0,1+0,0428) \mid 4-1 \mid = 0,4284$

$$\beta_t = \frac{0,4284}{1-0,4284} = 0,7494$$

Maka bobot setelah iterasi 2 : $w_{3,1} = 0,0428 \times 0,7494 = 0,032$

$\in_t = (0,1+0,0428+0,032) \mid 4-1 \mid = 0,5244$

$$\beta_t = \frac{0,5244}{1-0,5244} = 1,1026$$

Maka bobot setelah iterasi 3: $w_{4,1} = 0,032 \times 1,1026 = 0,035$

$\in_t = (0,1+0,0428+0,032+0,035) \mid 4-1 \mid = 0,6294$

$$\beta_t = \frac{0,6294}{1-0,6294} = 1,67$$

Maka bobot setelah iterasi 4: $w_{5,1} = 0,035 \times 1,67 = 0,058$

$\in_t = (0,1+0,0428+0,032+0,035+0,058) \mid 4-1 \mid = 0,8033$

$$\beta_t = \frac{0,803}{1-0,803} = 4,076$$

Maka bobot setelah iterasi 5: $w_{6,1} = 0,058 \times 4,076 = 0,236$

$\in_t = (0,1 + 0,0428 + 0,032 + 0,035 + 0,058 + 0,236) \mid 4 - 1 \mid = 1,511$ ;

$\beta_t = \frac{1,511}{1 - 1.511} = -2,957$

Maka bobot setelah iterasi 6: $w_{7,1} = 0,236 \times (-2,957) = -0,697$

Hasil akhir klasifikasi yang diharapkan pada citra positif adalah sebagai berikut :

$$H(x) = \begin{cases} 1 & \sum_{j=1}^{J} \alpha_j h_j(x) \geq \frac{1}{2} \sum_{t}^{T} \alpha_t \\ 0 & \text{lainnya.} \end{cases} \quad \text{.....(3.8)}$$

Dimana:

$\alpha_j = \log \frac{1}{\beta_j}$ $\alpha_t = \log \frac{1}{\beta_t}$ .....(3.9)

Jika posisi H(x) = Ketentuan 1 maka citra tersebut merupakan objek

Jika posisi H(x) = Ketentuan 0 maka citra tersebut merupakn bukan objek

H(x) = Strong Classifier atau kelasifikasi yang menyatakan objek atau bukan

αj = Tingkat pembelajaran gambar positif.

αt = Tingkat pembelajaran gambar negatif.

βj=Nilai bobot setelah error rate pada gambar negatif

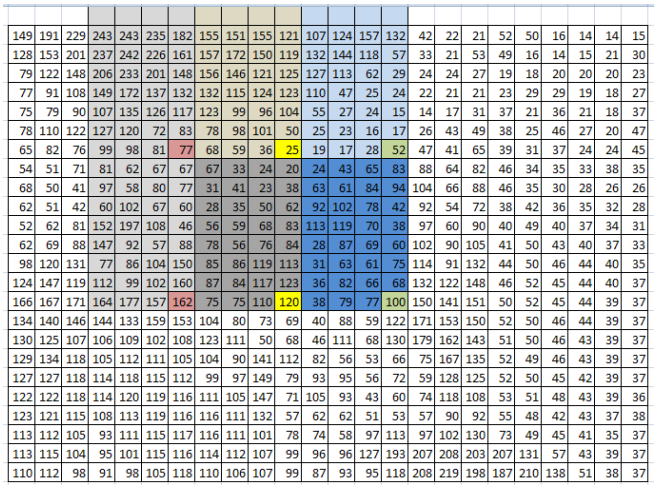
βt= Nilai bobot setelah error rate pada gambar positif

Hj= weak atau basic classifiers (awal dari klasifikasi) gambar negatif.

Ht = weak atau basic classifiers (awal dari klasifikasi) gambar positif.

Sehingga untuk citra negatif (24x24)piksel yang memiliki nilai-nilai sebagai berikut :

1. Citra negatif ke-1



| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 149 | 191 | 229 | 243 | 243 | 235 | 182 | 155 | 151 | 155 | 121 | 107 | 124 | 157 | 132 | 42 | 22 | 21 | 52 | 50 | 16 | 14 | 14 | 15 |
| 128 | 153 | 201 | 237 | 242 | 226 | 161 | 157 | 172 | 150 | 119 | 132 | 144 | 118 | 57 | 33 | 21 | 53 | 49 | 16 | 14 | 15 | 21 | 30 |
| 79 | 122 | 148 | 206 | 233 | 201 | 148 | 156 | 146 | 121 | 125 | 127 | 113 | 62 | 29 | 24 | 24 | 27 | 19 | 18 | 20 | 20 | 20 | 23 |
| 77 | 91 | 108 | 149 | 172 | 137 | 132 | 132 | 115 | 124 | 123 | 110 | 47 | 25 | 24 | 22 | 21 | 21 | 23 | 29 | 29 | 19 | 18 | 27 |
| 75 | 79 | 90 | 107 | 135 | 126 | 117 | 123 | 99 | 96 | 104 | 55 | 27 | 24 | 15 | 14 | 17 | 31 | 37 | 21 | 36 | 21 | 18 | 37 |
| 78 | 110 | 122 | 127 | 120 | 72 | 83 | 78 | 98 | 101 | 50 | 25 | 23 | 16 | 17 | 26 | 43 | 49 | 38 | 25 | 46 | 27 | 20 | 47 |
| 65 | 82 | 76 | 99 | 98 | 81 | 77 | 68 | 59 | 36 | 25 | 19 | 17 | 28 | 52 | 47 | 41 | 65 | 39 | 31 | 37 | 24 | 24 | 45 |
| 54 | 51 | 71 | 81 | 62 | 67 | 67 | 67 | 33 | 24 | 20 | 24 | 43 | 65 | 83 | 88 | 64 | 82 | 46 | 34 | 35 | 33 | 38 | 35 |
| 68 | 50 | 41 | 97 | 58 | 80 | 77 | 31 | 41 | 23 | 38 | 63 | 61 | 84 | 94 | 104 | 66 | 88 | 46 | 35 | 30 | 28 | 26 | 26 |
| 62 | 51 | 42 | 60 | 102 | 67 | 60 | 28 | 35 | 50 | 62 | 92 | 102 | 78 | 42 | 92 | 54 | 72 | 38 | 42 | 36 | 35 | 32 | 28 |
| 52 | 62 | 81 | 152 | 197 | 108 | 46 | 56 | 59 | 68 | 83 | 113 | 119 | 70 | 38 | 97 | 60 | 90 | 40 | 49 | 40 | 37 | 34 | 31 |
| 62 | 69 | 88 | 147 | 92 | 57 | 88 | 78 | 56 | 76 | 84 | 28 | 87 | 69 | 60 | 102 | 90 | 105 | 41 | 50 | 43 | 40 | 37 | 33 |
| 98 | 120 | 131 | 77 | 86 | 104 | 150 | 85 | 86 | 119 | 113 | 31 | 63 | 61 | 75 | 114 | 91 | 132 | 44 | 50 | 46 | 44 | 40 | 35 |
| 124 | 147 | 119 | 112 | 99 | 102 | 160 | 87 | 84 | 117 | 123 | 36 | 82 | 66 | 68 | 132 | 122 | 148 | 46 | 52 | 45 | 44 | 40 | 37 |
| 166 | 167 | 171 | 164 | 177 | 157 | 162 | 75 | 75 | 110 | 120 | 38 | 79 | 77 | 100 | 150 | 141 | 151 | 50 | 52 | 45 | 44 | 39 | 37 |
| 134 | 140 | 146 | 144 | 133 | 159 | 153 | 104 | 80 | 73 | 69 | 40 | 88 | 59 | 122 | 171 | 153 | 150 | 52 | 50 | 46 | 44 | 39 | 37 |
| 130 | 125 | 107 | 106 | 109 | 102 | 108 | 123 | 111 | 50 | 68 | 46 | 111 | 68 | 130 | 179 | 162 | 143 | 51 | 50 | 46 | 43 | 39 | 37 |
| 129 | 134 | 118 | 105 | 112 | 111 | 105 | 104 | 90 | 141 | 112 | 82 | 56 | 53 | 66 | 75 | 167 | 135 | 52 | 49 | 46 | 43 | 39 | 37 |
| 127 | 127 | 118 | 114 | 118 | 115 | 112 | 99 | 97 | 149 | 79 | 93 | 95 | 56 | 72 | 59 | 128 | 125 | 52 | 50 | 45 | 42 | 39 | 37 |
| 122 | 122 | 118 | 114 | 120 | 119 | 116 | 111 | 105 | 147 | 71 | 105 | 93 | 43 | 60 | 74 | 118 | 108 | 53 | 51 | 48 | 43 | 39 | 36 |
| 123 | 121 | 115 | 108 | 113 | 119 | 116 | 116 | 111 | 132 | 57 | 62 | 62 | 51 | 53 | 57 | 90 | 92 | 55 | 48 | 42 | 43 | 37 | 38 |
| 113 | 112 | 105 | 93 | 111 | 115 | 117 | 116 | 111 | 101 | 78 | 74 | 58 | 97 | 113 | 97 | 102 | 130 | 73 | 49 | 45 | 41 | 35 | 37 |
| 113 | 115 | 104 | 95 | 101 | 115 | 116 | 114 | 112 | 107 | 99 | 96 | 96 | 127 | 193 | 207 | 208 | 203 | 207 | 131 | 57 | 43 | 39 | 37 |
| 110 | 112 | 98 | 91 | 98 | 105 | 118 | 110 | 106 | 107 | 99 | 87 | 93 | 95 | 118 | 208 | 219 | 198 | 187 | 210 | 138 | 51 | 38 | 37 |

Dengan nilai fitur :

$h_j(x)$ = (120+77-162-25)-(100+25-120-52) = 57

Maka $\in_j = (0,05)\,|\,57-0\,| = 2,8$

$\beta_j = \frac{\in_j}{1-\in_j}$ Maka $\beta_j = \frac{2,85}{1-2,85} = -1,54$

Perbaharui bobot: $w_{t+1,i} = w_{t,i}.\beta_j$

Maka bobot setelah iterasi 1 : $w_{2,1} = 0,05 \times (-1,54) = -0,077$

jika bobot stelah intersi ke n jumlahnya < 0 maka iterasi berhenti. βj yang digunakan untuk mencari H(x) mengunakan $\in_t$ pada citra negatif yang nilai βj<0

$$H_{(x)} = \sum_{j=1}^{J} a_j\, h_j(x) \geq \frac{1}{2}\sum_{j}^{T} a_t = untuk\ nilai\ 1\ bernilai\ benar$$

Untuk salah bernilai = 0

Maka: $H(x) = \log\frac{1}{2,85} \times 57 \geq \frac{1}{2}\log\frac{1}{0,428}$

$= -25,99 \geq 0,184$

Karena $-25,99 \geq 0,184$ itu bernilai salah jadi citra bukan merupakan objek

*C*itra negatif ke-2

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 72 | 95 | 94 | 89 | 115 | 115 | 114 | 112 | 110 | 108 | 106 | 103 | 102 | 100 | 98 | 95 | 93 | 91 | 89 | 87 | 85 | 85 | 94 | 87 |
| 73 | 102 | 109 | 108 | 115 | 117 | 116 | 112 | 109 | 112 | 109 | 107 | 104 | 103 | 101 | 98 | 96 | 94 | 91 | 89 | 87 | 87 | 97 | 90 |
| 77 | 100 | 102 | 111 | 116 | 117 | 118 | 110 | 91 | 100 | 109 | 109 | 106 | 104 | 102 | 100 | 98 | 96 | 94 | 91 | 88 | 88 | 98 | 91 |
| 76 | 105 | 106 | 109 | 115 | 121 | 121 | 110 | 92 | 96 | 109 | 110 | 107 | 106 | 104 | 101 | 99 | 98 | 95 | 94 | 91 | 88 | 100 | 82 |
| 78 | 102 | 114 | 112 | 106 | 108 | 110 | 110 | 102 | 93 | 108 | 112 | 109 | 109 | 103 | 102 | 101 | 98 | 96 | 95 | 92 | 88 | 103 | 74 |
| 88 | 95 | 108 | 112 | 112 | 93 | 92 | 107 | 104 | 98 | 110 | 113 | 111 | 111 | 91 | 79 | 102 | 99 | 97 | 96 | 93 | 89 | 104 | 79 |
| 120 | 117 | 112 | 111 | 112 | 105 | 103 | 107 | 103 | 97 | 111 | 114 | 111 | 111 | 98 | 88 | 102 | 99 | 96 | 94 | 93 | 89 | 103 | 80 |
| 119 | 122 | 121 | 118 | 116 | 117 | 109 | 108 | 107 | 100 | 112 | 115 | 112 | 111 | 99 | 93 | 102 | 99 | 97 | 80 | 86 | 89 | 104 | 78 |
| 118 | 119 | 118 | 118 | 119 | 119 | 119 | 117 | 117 | 116 | 115 | 114 | 112 | 110 | 100 | 89 | 101 | 98 | 96 | 79 | 82 | 89 | 102 | 86 |
| 117 | 118 | 117 | 117 | 117 | 118 | 118 | 117 | 116 | 117 | 115 | 113 | 111 | 109 | 104 | 96 | 100 | 98 | 96 | 75 | 82 | 89 | 101 | 94 |
| 115 | 116 | 115 | 116 | 116 | 116 | 117 | 117 | 115 | 115 | 113 | 112 | 110 | 107 | 106 | 103 | 100 | 97 | 95 | 79 | 83 | 87 | 100 | 93 |
| 114 | 115 | 114 | 113 | 114 | 115 | 116 | 116 | 115 | 113 | 112 | 110 | 108 | 106 | 104 | 101 | 99 | 95 | 93 | 90 | 87 | 86 | 99 | 93 |
| 111 | 112 | 112 | 112 | 113 | 114 | 114 | 114 | 113 | 112 | 111 | 108 | 106 | 104 | 102 | 100 | 97 | 93 | 91 | 88 | 86 | 83 | 97 | 91 |
| 109 | 110 | 110 | 109 | 110 | 111 | 112 | 112 | 111 | 111 | 109 | 107 | 105 | 102 | 99 | 98 | 95 | 92 | 89 | 85 | 83 | 81 | 98 | 90 |
| 107 | 107 | 107 | 107 | 107 | 109 | 110 | 109 | 108 | 108 | 106 | 104 | 102 | 100 | 98 | 96 | 92 | 89 | 86 | 81 | 77 | 61 | 55 | 51 |
| 97 | 104 | 103 | 105 | 105 | 107 | 106 | 107 | 106 | 106 | 104 | 101 | 99 | 98 | 96 | 93 | 89 | 87 | 83 | 77 | 67 | 40 | 27 | 29 |
| 55 | 99 | 101 | 101 | 102 | 104 | 104 | 104 | 104 | 102 | 101 | 99 | 96 | 95 | 93 | 90 | 87 | 83 | 78 | 72 | 60 | 37 | 31 | 32 |
| 25 | 85 | 102 | 97 | 100 | 100 | 100 | 100 | 100 | 99 | 97 | 93 | 91 | 92 | 89 | 87 | 83 | 79 | 76 | 69 | 58 | 35 | 29 | 30 |
| 14 | 55 | 101 | 96 | 98 | 100 | 99 | 98 | 97 | 99 | 97 | 94 | 92 | 91 | 87 | 83 | 79 | 74 | 71 | 62 | 47 | 30 | 29 | 28 |
| 16 | 34 | 98 | 100 | 97 | 92 | 92 | 90 | 87 | 85 | 81 | 78 | 70 | 65 | 58 | 37 | 52 | 51 | 45 | 40 | 27 | 17 | 21 | 24 |
| 42 | 45 | 73 | 73 | 69 | 30 | 47 | 63 | 59 | 57 | 54 | 50 | 46 | 40 | 29 | 18 | 16 | 21 | 27 | 23 | 9 | 10 | 13 | 20 |
| 60 | 52 | 47 | 52 | 49 | 38 | 40 | 40 | 37 | 36 | 30 | 27 | 25 | 7 | 7 | 9 | 8 | 15 | 25 | 22 | 8 | 4 | 2 | 9 |
| 64 | 51 | 35 | 20 | 13 | 9 | 4 | 0 | 19 | 30 | 23 | 25 | 26 | 8 | 11 | 7 | 4 | 4 | 6 | 21 | 10 | 6 | 4 | 10 |
| 59 | 49 | 29 | 5 | 2 | 2 | 5 | 8 | 24 | 32 | 27 | 28 | 22 | 1 | 1 | 3 | 0 | 0 | 4 | 22 | 12 | 10 | 2 | 10 |

Dengan nilai fitur :

$h_j(x)$ = (101+119-104-115)-(93+115-101-100) = 6

Maka $\in_j = (0,05)\,|\,6-0\,| = 0,3$

$\beta_j = \dfrac{\in_j}{1-\in_j}$ Maka $\beta_j = \dfrac{0,3}{1-0,3} = 0,428$

*Update* bobot: $w_{t+1,i} = w_{t,i}.\beta_j$

Maka bobot setelah iterasi 1 : $w_{2,1} = 0,05 \times (0,428) = 0,021$

$\in_t = (0,05+0,021)\,|\,6-0\,| = 0,426$ ; $\beta_t = \dfrac{0,426}{1-0,426} = 0,742$

Maka bobot setelah iterasi 2 : $w_{3,1} = 0,021 \times 0,742 = 0,0155$

$\in_t = (0,05+0,021+0,0155)\,|\,6-0\,| = 0,519$ ; $\beta_t = \dfrac{0,519}{1-0,519} = 1,079$

Maka bobot setelah iterasi 3: $w_{4,1} = 0,0155 \times 1,079 = 0,0167$

$\in_t = (0,05+0,021+0,0155+0,0167)\,|\,6-0\,| = 0,619$ ; $\beta_t = \dfrac{0,619}{1-0,619} = 1,624$

Maka bobot setelah iterasi 4: $w_{5,1} = 0,0167 \times 1,624 = 0,0271$

$\in_t = (0,05+0,021+0,0155+0,0167+0,0271)\,|\,6-0\,| = 0,7818$ ;

$\beta_t = \dfrac{0,7818}{1-0,7818} = 3,582$

Maka bobot setelah iterasi 5: $w_{6,1} = 0,0271 \times 3,582 = 0,0971$

$\in_t = (0,05 + 0,021 + 0,0155 + 0,0167 + 0,0271 + 0,0970) \mid 6 - 0 \mid = 1,364$ ;

$\beta_t = \frac{1,364}{1 - 1.364} = -3,747$

Maka bobot setelah iterasi 6: $w_{7,1} = 0,0971 \times (-3,747) = -0,363$

Maka:

$H_{(x)} = \sum_{j=1}^{J} a_j \, h_j(x) \geq \frac{1}{2} \sum_t^T a_t$.....(3.10)

$$H(x) = (\log \frac{1}{0,428} + \log \frac{1}{0,742} + \log \frac{1}{1,079} + \log \frac{1}{1,624} + \log \frac{1}{3,582}) \times 6 \geq$$

$$\frac{1}{2}(\log \frac{1}{0,428} + \log \frac{1}{0,749} + \log \frac{1}{1,1026} + \log \frac{1}{1,67} + \log \frac{1}{4,076})$$

$$= 0,368 + 0,129 - 0,033 - 0,210 - 0,554$$

$$\geq \frac{1}{2}(0,368 + 0,125 - 0,042 - 0,223 - 0,610)$$

$$= -0,3 \geq -0,191$$

Maka citra tersebut bukan berupa gambar positif.

Jadi semakain banyak interasi yang bisa dilakukan oleh sebuah gambar positif dan negatif maka semakain besak kemungkinan untuk objek ditemukan. Dari analisa diatas dapat diambil kesimpulan, bahwa tiga faktor yang mempegaruhi kecepatan dan tingkat akurasi dalam pendeteksian, faktor - faktor tersebut antara lain:

1. jumlah dari tahapan classifier.
2. Jumlah semakin banyak gambar positif dimasukan semakin akurat pendeteksianya.
3. Jumlah dari data latih yang dimasukan.

### 3.1.7 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional adalah sebuah langkah untuk menganalisis sumber daya manusia yang akan menggunakan perangkat lunak yang dibangun, perangkat keras dan perangkat lunak yang dimiliki sesuai dengan kebutuhan atau permintaan, sehingga dapat ditentukan kompabilitas perangkat lunak yang dibangun terhadap sumber daya yang ada.

#### 3.1.7.1 Analisis Perangkat Keras

Analisis perangkat keras dimaksudkan untuk mengetahui spesifikasi perangkat keras yang akan digunakan oleh Aplikasi ini. Perangkat keras yang dianjurkan untuk menggunakan aplikasi ini adalah sebagai berikut :

1. Processor 2,27 GHz
2. Hard Disk 320 GB
3. Memory 2 GB
4. Monitor
5. Webcame, tipe VCR (Video Camera Recorder)
6. Mouse
7. Keyboard

#### 3.1.7.2 Analisis Perangkat Lunak

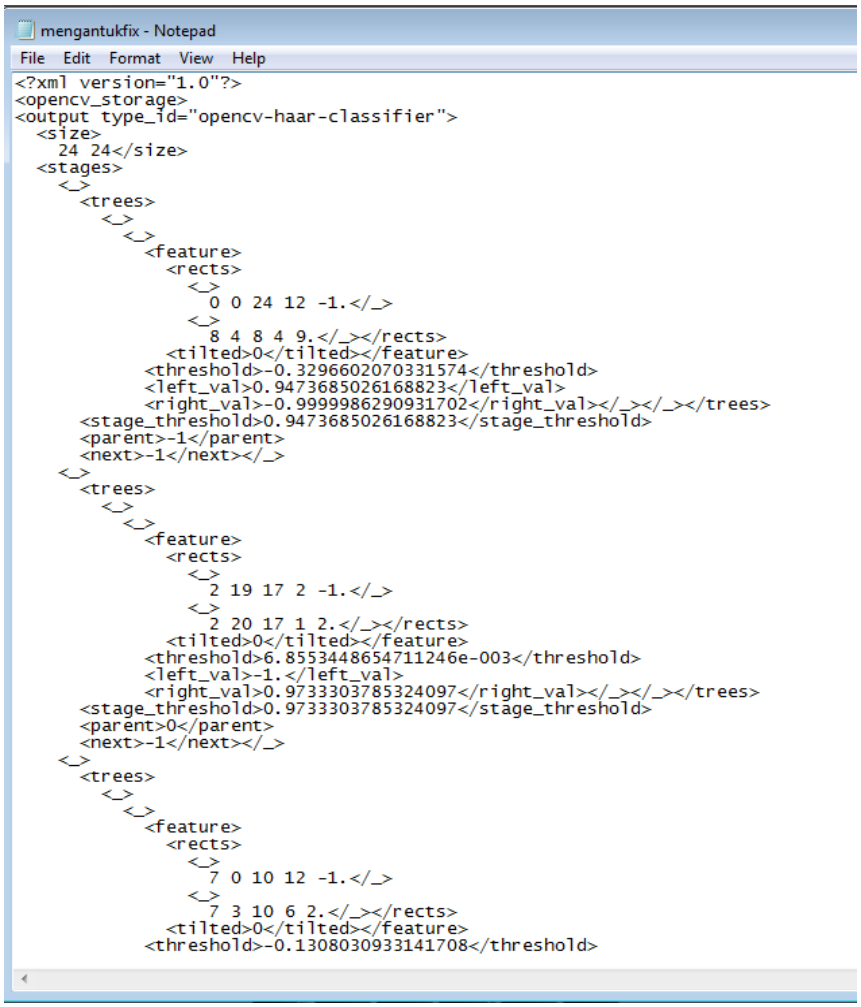
Sebelum menggunakan perangkat lunak pendeteksikantukini *user* harus terlebih dahulu menginstalasi aplikasi pendukung untuk mengaktifkan *webcam*. Adapun perangkat lunak pendukung pembangunan perangkat lunak pendeteksi kantuk ini antara lain :

1. Sistem Operasi Windows XP Profesional.
2. Microsoft Visual Visual Studio 2010, digunakan untuk pengkodean sistem.
3. OpenCV 2.2, digunakan sebagai *library* tambahan untuk pengkodean sistem.

### 3.1.8 Deskripsi Kebutuhan Data *Eksternal*

Kebutuhan data *eksternal* akan diuraikan secara rinci untuk keperluan perancangan perangkat lunak. Data *eksternal* yang digunakan pada perangkat lunak pendeteksian mata mengantuk yang akan dibangun yaitu :

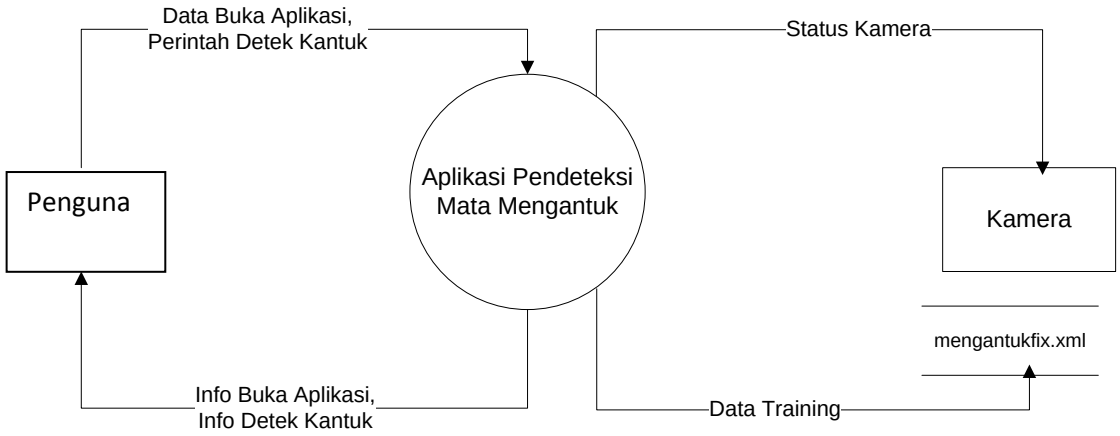
*File* mengantukfix.xml berisi data hasil dari *training* mata mengantuk yang digunakan untuk mendeteksi mata mengantuk. Adapun isi dari *file* mengantukfix.xml adalah sebagai berikut :




Kebutuhan fungsional dianalisis dengan memodelkan sistem. Pemodelan yang digunakan untuk memodelkan perangkat lunak pengendali *pointer* ini adalah pemodelan terstruktur. Perangkat lunak ini dimodelkan menggunakan DFD (*Data Flow Diagram*).*Tools* yang digunakan adalah Diagram Konteks, *Data Flow Diagram* dan Spesifikasi Proses yang dibuat menggunakan *Microsoft Visio 2007* sebagai perangkat lunak yang digunakan.

### 3.1.8.1 Diagram Konteks

Diagram konteks adalah diagram yang menggambarkan masukan, proses dan keluaran secara umum yang terjadi pada sistem. Berikut ini adalah diagram konteks perangkat lunak pendeteksi mata mengantuk.



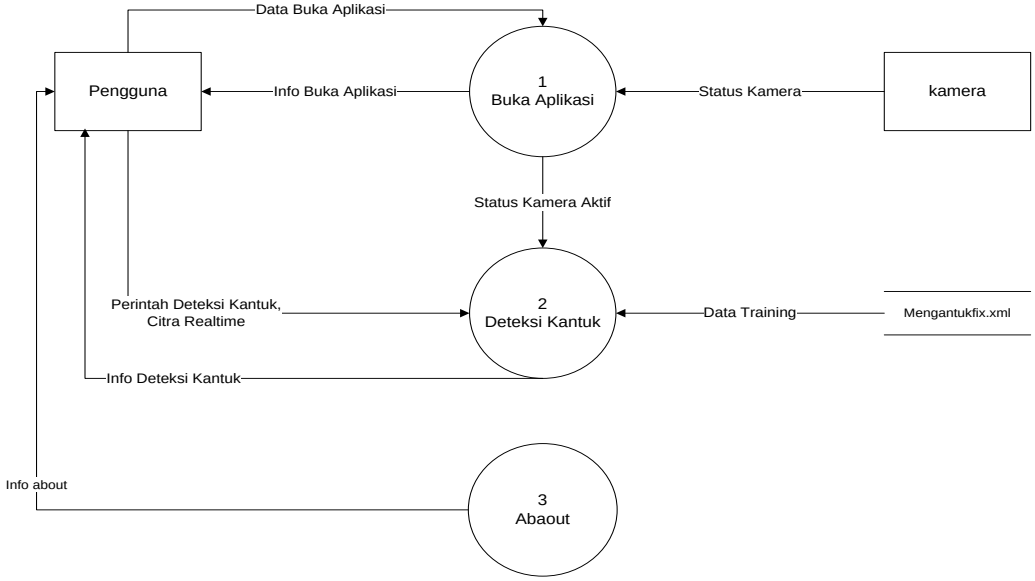

**Gambar 3.17 Diagram Konteks Aplikasi pendeteksi Mata mengantuk**

### 3.1.8.2 *Data Flow Diagram ( DFD )*

*Data flow diagram* merupakan model dari sistem untuk menggambarkan pembagian sistem ke modul yang lebih kecil. DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir atau lingkungan fisik dimana data tersebut akan disimpan. Salah satu keuntungan DFD adalah memudahkan pemakai atau *user* yang kurang menguasai bidang komputer untuk mengerti sistem yang akan dikerjakan.

1. **DFD Level 1**

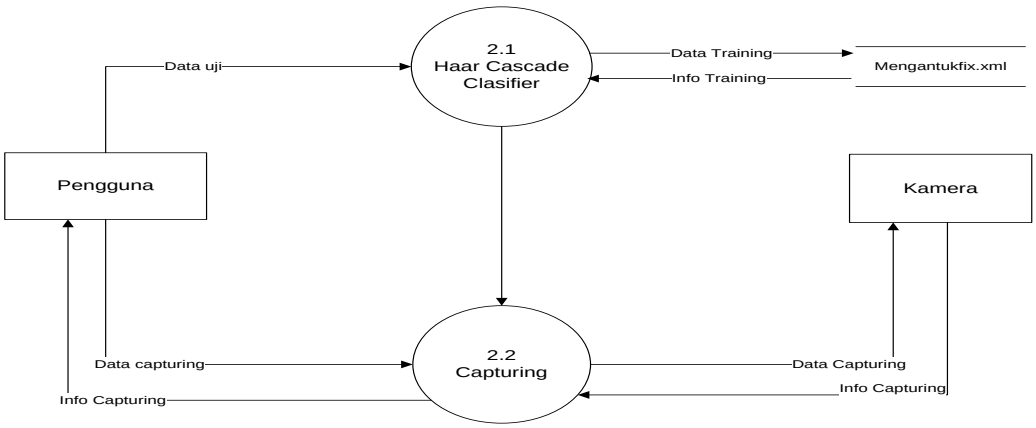
DFD Level 1 dibuat jika pada diagram konteks masih terdapat proses yang harus dijelaskan lebih rinci. Pada DFD Level 1 terdapat 2 proses yaitu buka aplikasi, deteksi kantuk. DFD level 1 pendeteksi mata mengantuk digambarkan sebagai berikut :




**Gambar 3.18 DFD Level 1 Pendeteksian mata mengantuk**

**2. DFD Level 2 Proses Deteksi kantuk**

Pada DFD Level 2 proses deteksi kantuk terdapat 2 proses yaitu Haar Cascade Clasifier,Capturing. DFD Level 2 proses deteksi kantuk digambarkan sebagai berikut :




**Gambar 3.19 DFD Level 2 Pendeteksian Mata mengantuk**

### 3.1.9 Spesifikasi Proses

**Table 3.2 Spesifikasi Proses Aplikasi mata mengantuk**

| NO | PROSES | KETERANGAN |
|---|---|---|
| 1 | Nomor Proses | 1 |
| | Nama Proses | Buka Aplikasi |
| | Deskripsi | Memeriksa koneksi kamera dengan sistem |
| | Input | Webcam |
| | Output | Tampilan awal sistem |
| | Proses | Priksa kesedian webcam |
| | Logika Proses | 1. Proses menerima permintaan buka aplikasi dari User<br>2. Proses menerima status webcam<br>3. Jika webcam aktif, menu utama terbuka<br>4. Jika webcam tidak aktif, menu utama tidak terbuka |
| 2 | Nomor Proses | 2 |
| | Nama Proses | Deteksi Kantuk |
| | Deskripsi | Mendeteksi objek mata oleh webcam |
| | Input | Citra Realtime |
| | Output | Mata mengantuk atau mata terbuka |
| | Proses | Sistem mendeteksi mata mengantuk atau terbuka |
| | Logika Proses | 1. Proses webcame terbuka dan citra realtime dari user dimasukan<br>2. Proses akan mendeteksi apakah mata mengantuk atau terbuka dengan mencocokan dengan data XML yang telah dilatih.<br>3. Jika terdeteksi mata maka proses akan |

| | | |
|---|---|---|
| | | membuat persegi disekitar area citra mata.<br>4. Proses akan membuat kota persegi di area citra mata kiri dan kanan, kemudian mendeteksi kedipan mata.<br>5. Jika user mengantuk dengan menutup mata makan aplikasi akan mendeteksi mata mengantuk<br>6. Jika user matanya terbuka aplikasi tidak akan memproses. |
| 3 | Nomor proses | 3 |
| | Nama proses | Tentang |
| | Deskripsi | Info tentang aplikasi |
| | Input | Perintah tentang |
| | Output | Informasi tentang pembuatan aplikasi |
| | Proses | Sistem menampilkan tentang aplikasi |
| | Logika Proses | 1. Proses menerima perintah tentang dari pengguna<br>2. Proses akan menampilkan Informasi tentang |

| | | |
|---|---|---|
| 4 | Nomor Proses | 2.1 |
| | Nama Proses | Haar Cascade Clasifier |
| | Deskripsi | Melakukan training data |
| | Input | Citra positif dan negatif |
| | Output | Data XML |
| | Proses | Proses Create sample,cropping,haar like feature,cassade clasifier |
| | Logika Proses | 1. Melakukan Create Sampling, memasukan gambar positif dan negatif<br>2. Melakukan cropping secara manual<br>3. Menentukan haar fetuares yang akan di pakai<br>4. Proses Cascade Classifier atau mengkelasifikasikan apakah citra positif atau negatif secara berulang ulang<br>5. Menghasilkan data latih yang berupa data XML yang nantinya menjadi Store untuk mata mengantuk. |

| | | |
|---|---|---|
| 5 | Nomor Proses | 2.2 |
| | Nama Proses | Capuring |
| | Deskripsi | Mengidentifikasi mata mengantuk atau mata terbuka |
| | Input | Citra Realtime |
| | Output | Objek mata yang mengantuk atau terbuka |
| | Proses | Mengcapturing dari hasil citra realtime |
| | Logika Proses | 1. Citra realtime yang merupakan objek dicapturing<br>2. Objek ditemukan dan di proses mengantuk atau terbuka |

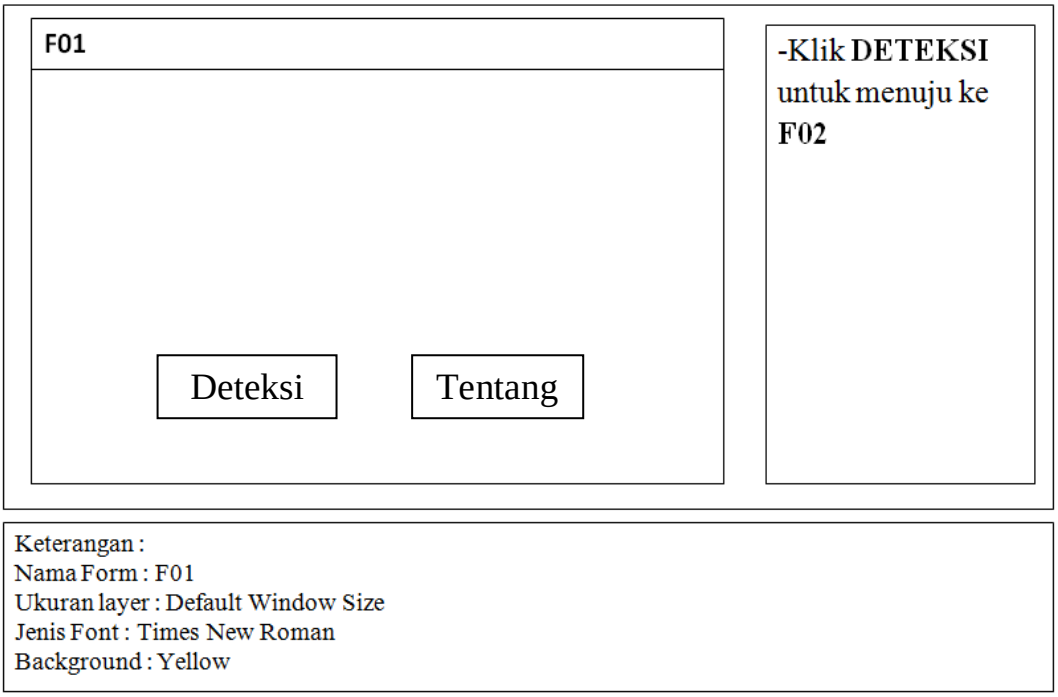
### 3.1.10 Perancangan Antarmuka

Perancangan antarmuka merupakan sebuah penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi. Adapun perancangan antarmuka perangkat lunak pendeteksi mata mengantuk adalah sebagai berikut :

### 3.1.11Perancangan Antarmuka Form MenuUtama

#### 3.1.11.1 Desain *Form* MenuUtama

*Form* menu utama merupakan *form* yang digunakan sebagai tampilan pada saat pengguna membuka aplikasi. Berikut ini adalah desain tampilan form aplikasi :




**Gambar 3.20Perancangan Antarmuka MenuUtama**

### 3.1.11.2 Deskripsi Objek *Form* Menu Utama

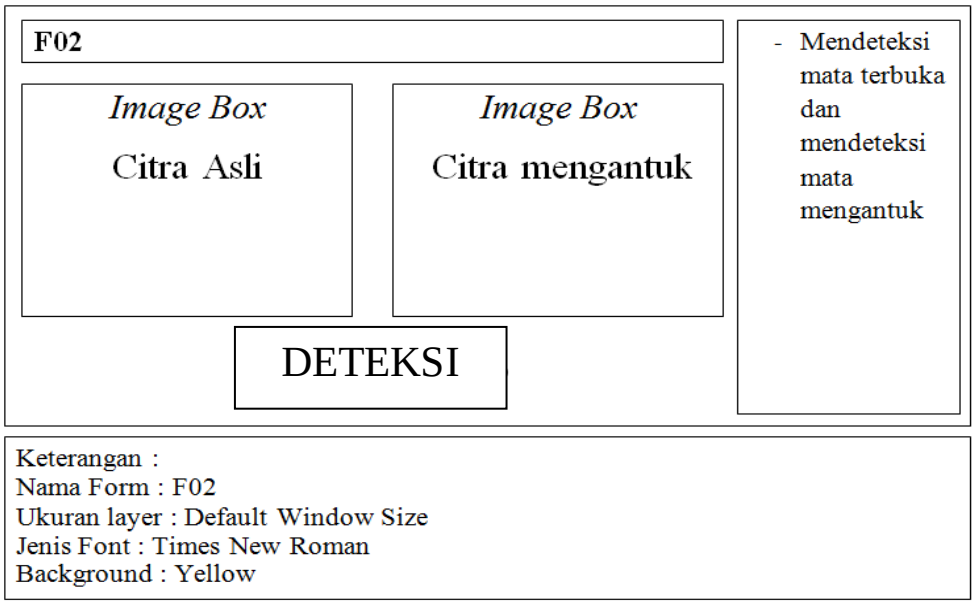
Berikut ini adalah deskripsi objek *form* MenuUtama:

**Tabel 3.3 Deskripsi Objek form menu utama**

| Objek | Jenis | Keterangan |
|---|---|---|
| **DETEKSI KANTUK** | **BUTTON** | **MENUJU FORM DETEKSI F02** |
| **TENTANG** | **BUTTON** | **MENUJU FORM TENTANG** |

### 3.1.11.3 Desain *Form* Deteksi

Form Deteksi Terdapat dua buah image box untuk menampilkan citra mata terbuka dan citra mata mengantuk. Berikut tampilan antarmuka.




**Gambar 3.21 Perancangan Antarmuka Form Deteksi**

### 3.1.11.4 Deskripsi objek *Form* Deteksi

Berikut ini adalah deskripsi objek *form* Deteksi :

**Table 3.4 Deskripsi objek form deteksi**

| Objek | Jenis | Keterangan |
|---|---|---|
| **Citra Asli** | **Image Box** | **Citra pada saat mata terbuka** |
| **Citra Mengantuk** | **Image Box** | **Citra pada saat citra mengatuk** |

### 3.1.12 Jaringan Semantik

Berikut ini adalah gambar jaringan semantik yang menggambarkan hubungan antar modul perangkat lunak Pendeteksi mata mengantuk.




**Gambar 3.22 Jaringan semantik**