BAB 3
METODOLOGI PENELITIAN

3.1 Bahan dan Perangkat Penelitian

Dalam penelitian ini, peneliti menggunakan 1 unit perangkat *Laptop* dengan spesifikasi yang terdapat pada poin di bawah ini :

3.1.1 Perangkat Keras (Hardware)

Spesifikasi perangkat keras yang digunakan dalam memenuhi penelitian ini adalah sebagai berikut:

1. Laptop HP EliteBook 8460p
2. Processor Intel® Core™ i5- 2520M CPU @ 2.50GHz 2.5 GHz,
3. Windows 10 Pro 64-bit.
4. RAM 16GB
5. Kamera CCTV Wifi Inforce 2MP Full HD 1080P

3.1.2 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan untuk Menyusun aplikasi dan laporan ini yaitu:

1. Balsamiq Mockup3, digunakan untuk perancangan antarmuka.
2. Matlab R2020a, digunakan untuk menguji variable yang digunakan dan membuat aplikasi akhir.

3.2 Obyek Penelitian

Obyek penelitian ini adalah gambar wajah memakai masker dan tidak memakai masker yang didapat dari dataset yang diambil menggunakan CCTV sebanyak 30 orang per kategori kelas. selain pengambilan menggunakan CCTV digunakan juga dataset berupa gambar sebanyak 853 yang menunjukkan pemakaian masker yang benar dan salah, serta pemakaian masker yang salah . Dataset diperoleh dari https://makeml.app/datasets/mask dan https://www.kaggle.com.

Tabel 3.1 *Dataset* yang dibutuhkan pada obyek penelitian

Type	Jumlah Data		
	Total Data Train	**Total Data Test**	**Dataset**
Masker Wajah Medis	175	75	250
Masker Wajah non medis	175	75	250
Masker Posisi Salah	245	105	350
Tidak Memakai Masker	315	135	450
Jumlah Dataset	1300 citra		

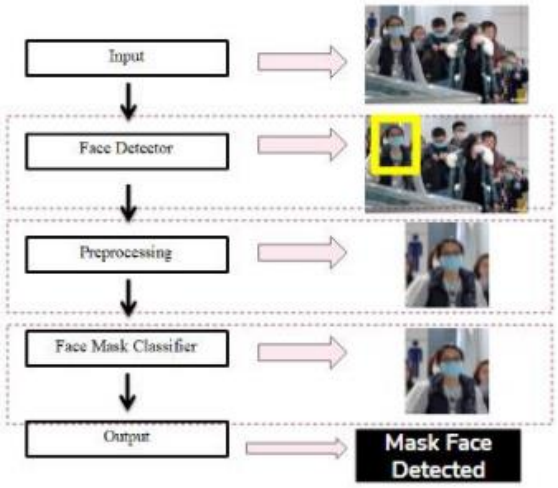
Pada Tabel 3.1 memperlihatkan jumlah dataset keseluruhan yang dibutuhkan pada penelitian deteksi pemakaian masker.

3.3 Tahap Penelitian

Pada tahap penelitian terdapat alur sistem deteksi pemakaian masker yang dimulai dari pengenalan wajah hingga proses deteksi masker, detail rancangan dari sistem deteksi pemakaian masker diuraikan pada subbab ini.

3.3.1 Deskripsi Sistem

Rancangan pada sistem deteksi pemakaian masker yaitu, pertama adalah mengumpulkan dataset berupa wajah memakai masker dan tidak memakai masker yang nantinya akan dilakukan proses deteksi masker sesuai dengan kategori kelas pemakaian masker medis, pemakaian masker non medis, pemakaian masker posisi kurang tepat, dan tidak memakai masker. Dan dilanjutkan pada proses pengenalannya.

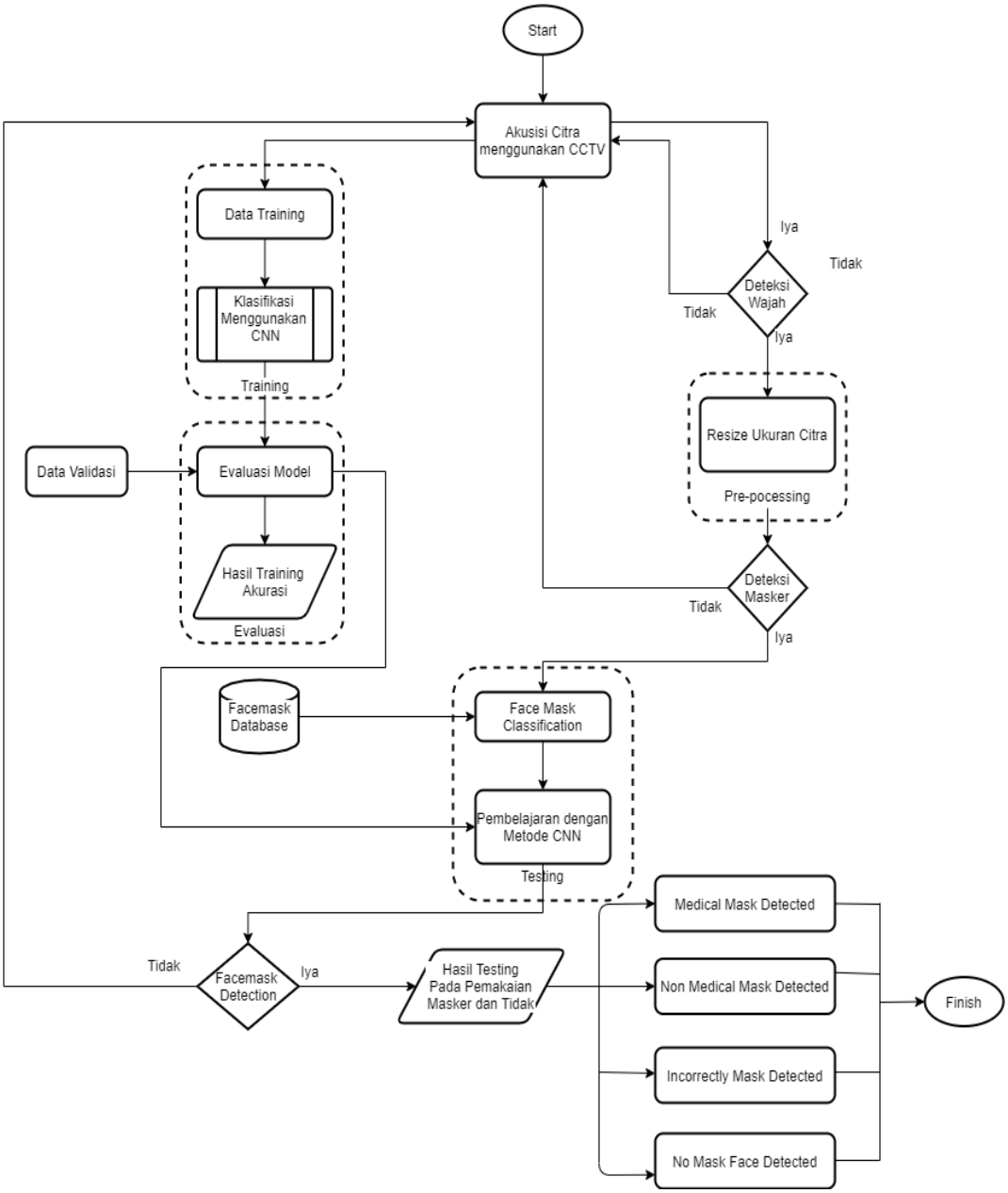


Gambar 3.1 Blok Diagram Sistem Deteksi Pemakaian Masker

Gambar 3.1 merupakan alur kerja dari sistem deteksi pemakaian masker. Pertama-tama yang dilakukan adalah melakukan tahap deteksi wajah terlebih dahulu sebelum melakukan proses pengenalan deteksi masker wajah. Pada saat melakukan deteksi wajah berguna untuk menerapkan model wajah dan selanjutnya proses deteksi pada wajah memakai masker dan tidak memakai masker. Setelah citra wajah terdeteksi tersebut terdeteksi.

Maka pada tahap selanjutnya yang dilakukan adalah pada sebuah image yang telah diberi ROI *label* akan dilakukan *processing* yaitu *cropping*, yang nantinya *imag*e dari hasil *cropping* akan dilanjutkan pada proses deteksi pemakaian masker. Pada wajah menggunakan metode *Viola Jones Object Detection* dan selanjutnya akan dilakukan proses training menggunakan metode *pretrained* CNN.

Untuk mengetahui nilai akurasi yang dihasilkan, maka proses selanjutnya adalah melakukan pengenalan pada wajah yang memakai masker dan tidak memakai masker sesuai dengan kategori kelas yang ada yaitu pemakaian masker medis, pemakaian masker non medis, pemakaian masker kurang tepat, dan tidak memakai masker. Pada proses klasifikasi dari tahapan citra menggunakan CNN. Seluruh data yang dapat terdeteksi akan memberikan output pada sistem bahwa wajah seseorang memakai masker layak untuk dipakai dan wajah seseorang tidak memakai masker.



Gambar 3.2 Blok diagram Sistem Deteksi Pemakaian Masker

Gambar 3.2 merupakan blok diagram Sistem Deteksi Pemakaian Masker mulai dari tahap akuisisi pada citra, proses pembelajaran terhadap dataset wajah, deteksi masker dan pengenalan pemakaian masker pada wajah, proses klasifikasi pada citra dibawah ini:

3.3.2 Deteksi Wajah

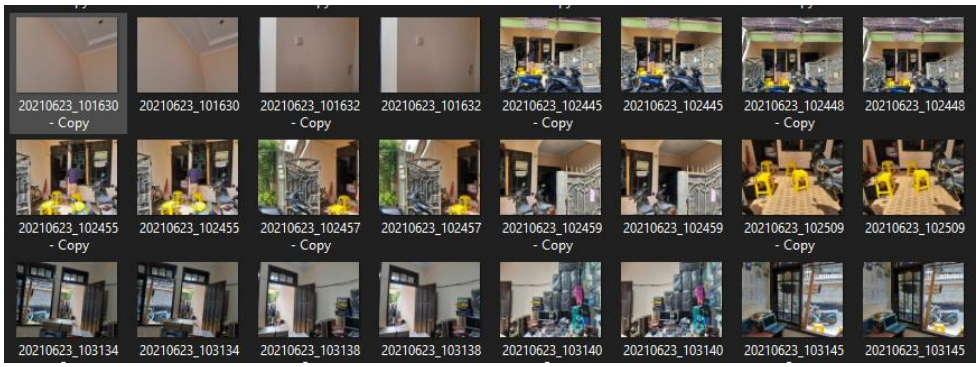
Wajah merupakan obyek utama yang akan dikenali didalam penelitian ini. Sebelum melakukan proses deteksi masker wajah, data wajah akan dilakukan proses pengenalan terlebih dahulu untuk dilakukan proses pembelajaran dan memberi *label* sebagai wajah dan terdapat beberapa tahapan yang akan dilakukan.

3.3.2.1 Tahap Pengumpulan Data

Jumlah data yang diperlukan pada deteksi masker ada dua folder, yaitu *positive* citra dan *negative* citra pada dataset wajah. Cara pengumpulan data wajah dilakukan secara manual.

Gambar 3.3 *Positive Images*

Gambar 3.3 merupakan dataset pada *positive image* yang memiliki jumlah data sebanyak 370 citra wajah. Pada citra tersebut akan dilakukan tahap labeling pada bagian area wajah saja.



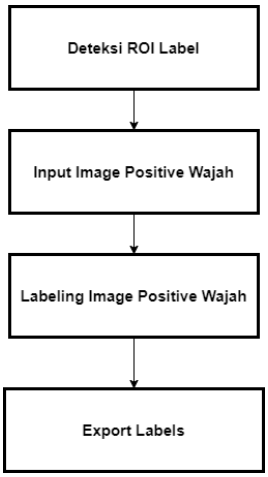
Gambar 3.4 *Negative Images*

Gambar 3.4 merupakan dataset pada *negative image* yang memiliki jumlah

data sebanyak 500 citra wajah berupa gambar yang tidak terdapat area wajah.

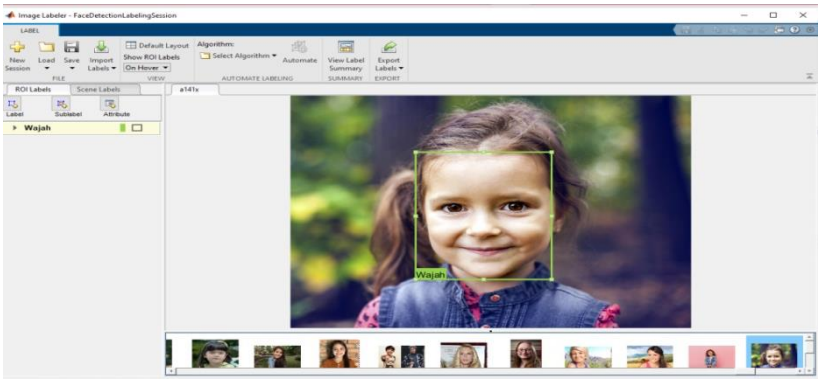
3.3.2.2 Tahap Pengolahan Data

Dalam tahapan pengolahan data melakukan sebuah proses *labeling* pada *Positive Image* wajah sebelum melanjutkan ke tahap selanjutnya yaitu proses *training,* hal ini bertujuan untuk mendapatkan ROI yang akan di deteksi. Pada *Image Labeling* ini menggunakan *tool* pada Matlab yaitu *Image Labeler*.



Gambar 3.5 Blok Diagram Tahap *Image Labeler* Pada Wajah

Gambar 3.5 merupakan tahapan *image labeling* pada dataset wajah, proses pertama yang dilakukan adalah mendefinisikan sebuah ROI berupa "*Wajah*" sesuai dengan objek yang akan dilabeli yaitu wajah, langkah selanjutnya yang dilakukan yaitu memberi ROI label sebanyak 370 *Positive Image* pada objek yang akan dilabeli yaitu wajah dengan *Bounding Box*. Proses terakhir adalah mengekspor ROI label yang telah di definisikan dalam bentuk tabel *groundtruth.*



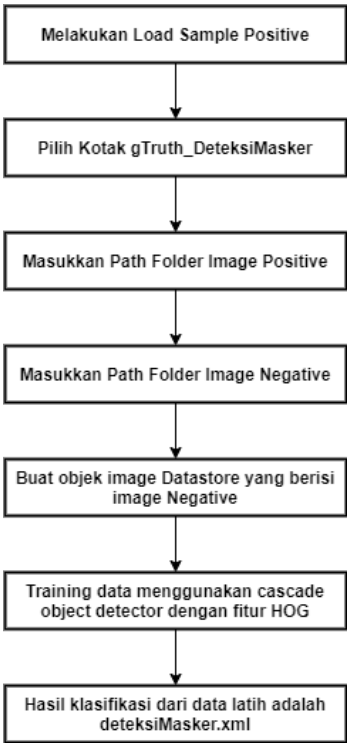
Gambar 3.6 *Bounding Box* Wajah

Gambar 3.6 adalah *Bounding Box* wajah yang sudah diberi ROI label, hasil dari *image label* yang akan digunakan pada proses *training* menggunakan fungsi `cascade object detector` pada Matlab.

3.3.2.3 Tahap *Training*

Tahapan *training* ini melakukan proses *training* data pada *Bounding Box* wajah yang sudah diberi ROI label menggunakan `train cascade object detector`. `Train cascade object detector` menggunakan algoritma dari *Viola-Jones* untuk mendeteksi wajah, hidung, mata, mulut, atau bagian tubuh lainnya.

`cascade object detector` digunakan untuk melatih data berupa *object* lainnya, menggunakan *Image Labeler* pada *object* yang ingin di deteksi, contohnya adalah mendeteksi wajah pada sistem ini.



Gambar 3.7 Blok Diagram Tahap *train Cascade Object Detector*

Gambar 3.7 merupakan tahapan melatih data wajah menggunakan `train cascade object detector`. Sebelum dilakukan klasifikasi terhadap *object* yang akan di deteksi adalah wajah. Proses pertama yang dilakukan adalah memuat

Bounding Box wajah yang sebelumnya diberi label dengan format *gTruthWajah2*.mat, kemudian proses selanjutnya pilih kotak pada *Ground Truth* sebagai pertanda untuk dilakukan proses eksekusinya tabel pada kolom yang sudah dipilih, selanjutnya menambahkan path pada folder *Positive Image* dan di tampung ke dalam variabel untuk diproses, sama halnya juga dengan *Negative Image* menentukan path pada folder yang sudah ditentukan, tahap selanjutnya buat *object image data store* yang berisi gambar *negative* untuk memberi data sebagai citra yang tidak akan dilakukan proses deteksi pada saat wajah di deteksi sebagai *object* yang diinginkan. Ada Tahapan yang paling penting yaitu men-*train* data menggunakan `train cascade object detector` dengan menentukan *hyperparameter* berupa *FalseAlarmRate* bernilai 0.4, *NumCascadeStages* bernilai 20 (sebagai nilai *default*), dan *FeatureType* dengan fitur HOG untuk mendeteksi *objek* wajah. Selanjutnya akan menghasilkan *Output* berupa deteksiWajah.xml dari proses klasifikasi ini, file tersebut digunakan sebagai parameter dalam *function Viola-Jones*. Gambar 3.8 merupakan source code yang dipakai untuk melakukan proses *training* pada dataset wajah.

```
% Muat sampel positif.
load('gTruthWajah2.mat');
positiveInstances = gTruthWajah2(:,1:2);

% Tambahkan folder gambar ke dalam path MATLAB.
imDir = fullfile('D:\TUGAS AKHIR RESTIN\PROGRAM COBA\Deteksi
Wajah\PositiveImage');
addpath(imDir);

% Tentukan folder untuk gambar negatif.
negativeFolder = fullfile('D:\TUGAS AKHIR RESTIN\PROGRAM
COBA\Deteksi Wajah\NegativeImage');

% Buat objek imageDatastore yang berisi gambar negatif.
negativeImages = imageDatastore(negativeFolder);

% Latih detektor objek berjenjang yang disebut
'hasilDeteksiWajah.xml' menggunakan fitur HOG.
trainCascadeObjectDetector('DeteksiWajah2.xml',positiveInstanc
es, ...

negativeFolder,'FalseAlarmRate',0.4,'NumCascadeStages',20,'Fea
tureType','Haar');
```

Gambar 3.8 *Source code* proses *learning* deteksi wajah

3.3.3 Pengenalan Wajah Memakai Masker

Pada tahapan pengenalan wajah memakai masker merupakan tahapan terakhir pada sistem deteksi pemakaian masker dipenelitian ini, dengan dilakukan proses pembelajaran pada dataset wajah memakai masker sesuai dengan kategori kelas yang diberikan yaitu memakai masker wajah medis, masker wajah non medis, masker wajah posisi salah, dan tidak memakai masker yang menggunakan metode pretrained *Convolutional Neural Networks (CNN)* sampai dengan proses klasifikasi pada citra, dengan melalui beberapa tahapan, yaitu:

3.3.3.1 Tahap Pengumpulan Dataset Wajah Memakai Masker

Pada tahap ini dilakukan pengumpulan Dataset sebanyak 1300 gambar yang akan dipilah sesuai dengan kelasnya, yaitu terdiri dari kelas masker medis, masker non medis, masker salah, dan tidak memakai masker. Hal ini dilakukan untuk memudahkan Dataset sesuai dengan keinginan yang akan dicapai. Sebanyak 910 image yang akan dimasukkan ke dalam data train dan 390 dimasukkan ke dalam data test.

Gambar 3.9 Dataset Wajah Memakai Masker Medis

Gambar 3.9 merupakan dataset wajah masker medis dengan jumlah keseluruhan yang dimiliki adalah sebanyak 250 datasets.

Gambar 3.10 Dataset Wajah Memakai Masker Non Medis

Gambar 3.10 merupakan dataset wajah masker non medis dengan jumlah keseluruhan yang dimiliki adalah sebanyak 250 datasets.

Gambar 3.11 Dataset Wajah Memakai Masker Salah

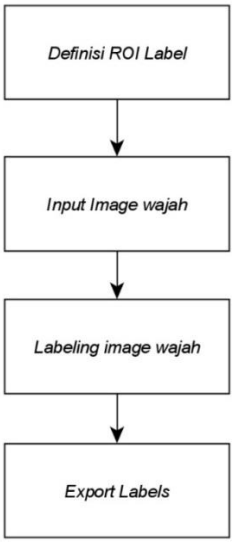
Gambar 3.11 merupakan dataset wajah masker salah atau pemakaian masker dengan posisi kurang tepat, dengan jumlah keseluruhan yang dimiliki adalah sebanyak 350 datasets.

Gambar 3.12 Dataset Wajah Tidak Memakai Masker

Gambar 3.12 merupakan dataset wajah tidak memakai masker dengan jumlah keseluruhan yang dimiliki adalah sebanyak 450 datasets.

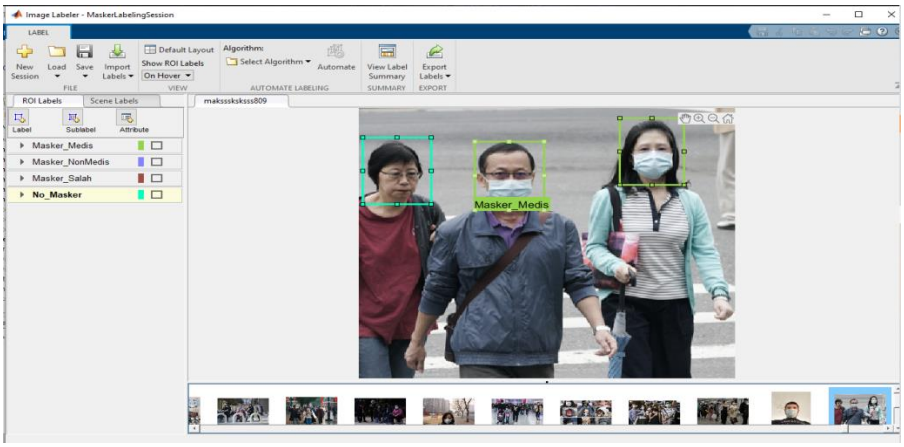
3.3.3.2 Tahap Pengolahan Dataset Wajah

Tahap yang dilakukan adalah sebuah citra akan diberi label sebelum masuk ke tahap *preprocessing* dan proses *training* yang bertujuan untuk mendapapatkan ROI *Lab*el yang akan dideteksi. Pada tahap *Image Labeling* menggunakan sebuah *tool* yang terdapat pada apps Matlab yaitu *Image Labeler*.



Gambar 3.13 Blok Diagram Tahap *Image Labeler*

Gambar 3.13 merupakan tahapan dari *image labeling* pada Dataset wajah memakai masker dan tidak memakai masker. Proses pertama yang dilakukan adalah mendefinisikan ROI wajah sesuai dengan objek wajah yang akan dilabeli yaitu wajah yang mengenakan masker medis, masker non medis, posisi masker yang salah, dan tidak memakai masker. Kemudian proses selanjutnya adalah memasukkan semua image, proses ketiga yaitu memberi ROI *label* pada objek yang akan dilabeli dan proses terakhir yang dilakukan adalah mengekspor ROI *label* yang telah didefinisikan dalam bentuk tabel *groundtruth*.



Gambar 3.14 Proses *Image Label*

Gambar 3.14 adalah gambar wajah yang mengenakan masker dan tidak yang telah diberi ROI *label*, hasil dari *image label* ini yang akan digunakan pada proses *pre-processing* yaitu *cropping*. Dibawah ini merupakan source code yang digunakan dalam melakukan proses looping pada *cropping* sesuai dengan *Bounding Box* yang sudah diberi.

```
%% load gTruth pada hasil labeling
load('gTruthmaskerbaru.mat');
for i=183:699
    imageFilename=gTruthmaskerbaru.imageFilename{i};
    filename=imageFilename;
    masker=gTruthmaskerbaru.Masker_NonMedis{i};
    for j=1:size(masker,1)
        I=imread(filename);
        I3=imcrop(I,masker(j,:));
        namafile=['nonmedis',int2str(i),' ',
int2str(j),'.png'];
        imwrite(I3,fullfile(namafile));
    end
 end
```

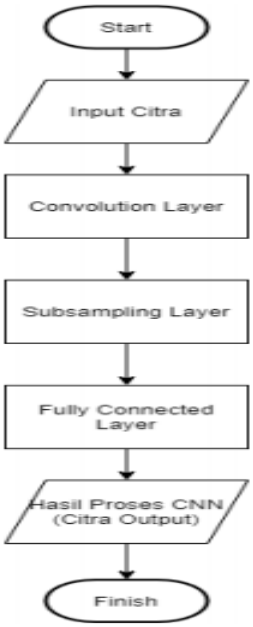
Gambar 3.15 *Source code cropping* pada *Bounding Box* wajah

Gambar 3.16 Hasil proses *pre-processing* pada citra wajah

Gambar 3.16 merupakan tahapan *pre-processing* pada Dataset wajah mengenakan masker dan tidak memakai masker yang telah dilakukan *image labeling*, proses pertama yang dilakukan adalah me-load file *groundtruth* dari hasil *image labeling*, kemudian dilakukan lopping sebanyak *image* data yang telah diberi label, proses kedua yaitu dilakukan *cropping* dengan menggunakan parameter dari kolom pertama dan kolom kedua dari tabel groundtruth agar image dapat resize sesuai dengan ROI wajah yang telah diberi label, dan proses terakhir adalah menyimpan hasil *cropping* sesuai dengan kelas masing-masing image untuk dilanjutkan ke proses *training*.

3.3.3.3 Tahap Training Menggunakan Metode CNN

Tahap yang akan dilakukan adalah training untuk dataset wajah mengenakan masker dan tidak pada tiap class dengan pretrained CNN, data tersebut terdapat 1300 *image* yang telah dibagi untuk diuji berapa akurasi dari dataset tersebut. Proses yang dilakukan pertama adalah membagi dataset masker wajah menjadi data train dan data test, dengan persentasi 70% data train dan 30% data test, proses selanjutnya adalah training menggunakan jaringan yang sudah di *training* atau di latih sebelumnya (*pretrained*) untuk mengetahui akurasi pada dataset tersebut menggunakan transfer learning yaitu alexnet dengan layer sebagai berikut. Selanjutnya dilakukan proses pembelajaran atau yang biasa disebut dengan *training* pada dataset wajah yang telah diolah sebelumnya.



Gambar 3.17 Diagram Proses CNN

Gambar 3.17 Proses CNN terdapat ratusan *layer* untuk memprosesnya, tetapi di CNN terdapat tiga *layer* penting yang nantinya dilakukan untuk mengklasifikasi citra, layer pertama yaitu *Convoluttion Layer*, pada lapisan *convolutional* dilakukan proses konvolusi gambar input melalui serangkaian filter konvolusional, konvolusi dilakukan pada input data dengan menggunakan filter atau kernel, kemudian menghasilkan output yang disebut *activation map* atau *feature map*. Proses pergeseran pada sebuah filter ditentukan melalui parameter yang disebut *stride* dan *activation map* atau *feature map*. Dalam proses pergeseran juga dipengaruhi oleh *stride* dan *padding* yang dimiliki. Stride bergerak setiap kali dengan ukuran langkah biasanya 1 yang artinya filter bergeser piksel demi piksel. Dengan meningkatkan ukuran langkah (stride), filter akan berjalan di atas input dengan interval yang lebih besar dan dengan demikian memiliki lebih sedikit tumpang tindih antara sel. Selanjutnya menentukan parameter yang ada dipengaruhi oleh jumlah *pixels* yang selanjutnya akan dilakukan proses penambahan disetiap input dari setiap sisi. Tujuan dari proses tersebuat karena ukuran pada *Feature Map* selalu lebih kecil dari input, maka dari itu kita harus melakukan sesuatu untuk mencegah penyusutan *Feature Map* dengan menggunakan *padding*, dalam hal ini bisa memakai *convolutional layer* yang lebih dalam dan menjaga ukuran spasial konstan setelah melakukan konvolusi. Selanjutnya proses *subsampling layer* pada proses ini dilakukan metode *max pooling.* Pada bagian *Max pooling* melakukan pembagian output dari sebuah *convolution* untuk dijadikan sebagai beberapa bagian mini grid dan selanjutnya dilakukan proses pengambilan nilai secara maksimal dari setiap grid untuk disusun menjadi sebuah matriks citra yang sudah terjadi proses direduksi, sehingga bertujuan untuk memungkinkan jaringan saraf *convolutional* mendeteksi objek ketika citra disajikan dalam berbagai posisi atau gambar dengan cara apapun.

Layer terakhir yaitu *fully connected layer* pada layer ini masing-masing *neuron* akan ditransformasi menjadi sebuah data satu dimensi terlebih dahulu sebelum dilanjutkan pada proses pemasukkan ke dalam sebuah *fully connected layer*, dengan demikian dapat menyebabkan hilangnya sebuah data informasi spasial dan tidak reversible, karena pada *fully connected layer* lapisan yang sepenuhnya ini hanya dapat menerima data satu dimensi, sehingga perlu mengatur data tiga dimensi menjadi vektor satu dimensi dengan proses *flattening*. Proses terakhir yaitu fungsi aktivasi, pada tahap ini komputer akan menghitung nilai matriks dan dicocokan dengan data latih, data latih yang memiliki nilai paling tinggi dari matriks maka itu yang akan dipilih oleh komputer untuk menentukan data apa yang ada pada matriks tersebut.

Pada perancangan sistem yang akan dikembangkan ini, metode CNN menggunakan jaringan yang sudah ditraining dengan teknik yang disebut dengan transfer learning, karena jauh lebih cepat dan lebih mudah dari pada melatih jaringan

dari awal, yaitu *alexnet*. Gambar 3.18 merupakan code yang digunakan untuk melakukan pelatihan dataset dengan metode CNN.

```
%%% fungsi ini untuk melakukan klasifikasi obyek menggunakan
CNN. Tranfer learning yaitu alexnet
 %% load data training dan data testing
imdsTrain =
imageDatastore('train','IncludeSubfolders',true,'LabelSource',
'foldernames');
imdsTest =
imageDatastore('test','IncludeSubfolders',true,'LabelSource','
foldernames');
 %% load pretrained
net = alexnet;
net.Layers
numClasses = 3;
%% ambil layer lama dikurangi tiga layer tearkhir
layersTransfer = net.Layers(1:end-3);
layers = [
    layersTransfer

fullyConnectedLayer(numClasses,'Name','new_fc','WeightLearnRat
eFactor',10,'BiasLearnRateFactor',10)
    softmaxLayer
    classificationLayer];
%%%PROSES TRAINING
inputSize = net.Layers(1).InputSize;
augimdsTrain =
augmentedImageDatastore(inputSize(1:2),imdsTrain);
augimdsTest =
augmentedImageDatastore(inputSize(1:2),imdsTest);

%% menspesifikasikan parameter training
options = trainingOptions('sgdm', ...
    'MiniBatchSize',40, ...
    'MaxEpochs',90, ...
    'InitialLearnRate',1e-4, ...
    'Shuffle','every-epoch', ...
    'Verbose',false, ...
    'Plots','training-progress');
%% melakukan proses training jaringan baru
trainedNet = trainNetwork(augimdsTrain,layers, options);
%% KLASIFIKASI DAN VALIDASI
YPred = classify(trainedNet,augimdsTest);
testLabels = imdsTest.Labels;
accuracy = mean(YPred == testLabels)
```

Gambar 3.18 *Source code* pelatihan dataset dengan metode CNN

3.3.3.4 Tahap Pengenalan Wajah

Setelah citra wajah terdeteksi, selanjutnya dilakukan proses pengenalan wajah dengan pencocokan data dari data set dengan proses klasifikasi citra menggunakan CNN. Data telah teridentifikasi sesuai dataset maka dilakukan pencatatan kehadiran dan menyimpannya kedalam database sesuai Id, jam, dan tanggal dilakukannya absensi.

Gambar 3.19 Blog Diagram Klasifikasi Citra

Gambar 3.19 merupakan tahap pengenalan pada citra dan hasil dari deteksi tersebut akan dilakukan pada proses klasifikasi dengan menggunakan metode CNN, proses pertama yaitu memuat model dari hasil *training* dan dilanjut ketahapan *resize* ukuran pada citra dengan 227 x 227 piksel, lalu dilakukan klasifikasi pada citra dengan menggunakan metode CNN. Hasil klasifikasi tadi ditampung kedalam sebuah variabel yang mana variabel tersebut akan ditampilkan pada *text*. Gambar 3.20 merupakan source code yang digunakan untuk melakukan tahap pengenalan dengan metode CNN.

```
%% PENGENALAN MASKER WAJAH

%% Load hasil train
load(cobatraining(84).mat);

%% menyesuaikan ukuran image pada layer input
I=imresize(faces,[227 227]);
trainedNet = handles.trainedNet;

%% trainedNet adalah jaringan yg sudah ditraining
YPred = classify(trainedNet,I);
nama=string(YPred);
str = strcat(nama);
set(handles.edit1,'string',str);
```

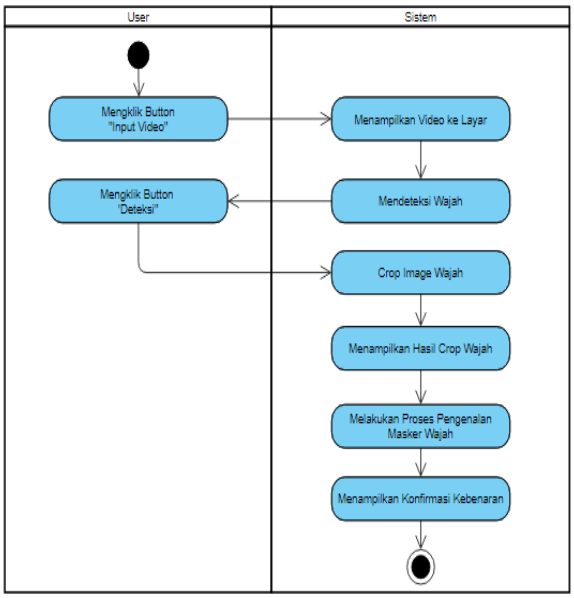
Gambar 3.20 *Source code* pelatihan dataset dengan metode CNN

3.4 Perancangan Sistem

Sistem deteksi pemakaian masker ini memiliki sebuah perrancangan yang akan dibuat yaitu berupa proses deteksi wajah terlebih dahulu dengan memberikan input video yang didapat dari kamera CCTV. Setelah mendapatkan video, akan dilakukan proses pengenalan apakah terdapat obyek wajah. Ketika terdeteksi wajah akan dilanjutkan proses pengenalan masker wajah. Dengan adanya perancangan sistem ini dapat mengetahui alur dari sistem deteksi pemakaian masker dengan jelas dan mudah dipahami.

3.4.1 Activity Diagram

Activity diagram merupakan cara kerja yang dilakukan pada sistem deteksi pemakaian masker. Dengan adanya activity diagram dapat mengetahui langkah-langkah pada sistem yang dibuat.

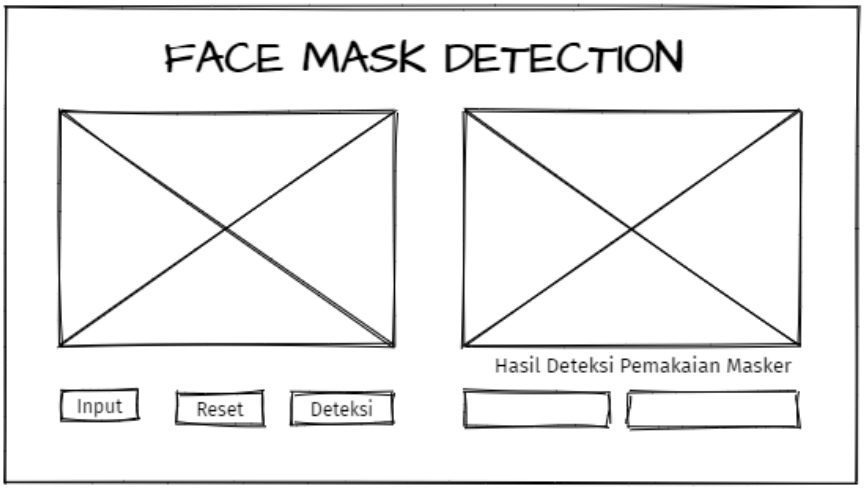


Gambar 3.21 *Activity Diagram*

Gambar 3.21 merupakan alur kerja atau proses dari sistem deteksi pemakaian masker.

3.4.2 Rancangan Antar Muka

Pada perancangan mockup atau antarmuka merupakan sebuah tahapan yang harus dilakukan untuk memudahkan proses pembuatan User Interface yang digunakan untuk mempermudahkan pengguna dalam merancang sistem deteksi masker. Desain antarmuka sistem akan disajikan pada Gambar 3.22



Gambar 3.22 Design Antarmuka Sistem Deteksi Masker

Gambar 3.22 merupakan tampilan sistem pada saat user melakukan proses deteksi pemakaian masker. Sistem deteksi pemakaian masker akan melakukan pengenalan wajah dari video secara real time melalui input berupa video CCTV. Dan proses akhirnya wajah yang terdeteksi akan dicrop dan dilakukan klasifikasi sesuai dengan kategori kelas yang telah ditentukan.

3.5 Skenario Pengujian

Pengukuran akurasi deteksi pemakaian masker dilakukan dengan menggunakan skenario sebagai berikut:

1. Menentukan posisi dan kondisi wajah yang tepat untuk mengetahui bentuk wajah pada citra dengan menguji :
    a. Posisi sudut wajah pada gambar
    b. Jarak obyek terhadap CCTV atau kamera

2. Menentukan posisi dan kondisi masker pada wajah untuk mendapatkan akurasi pengenalan wajah memakai masker dan tidak memakai masker yang baik dengan menguji:

    a. Posisi sudut masker pada wajah
    b. Perubahan posisi pemakaian masker seperti dikenakan dibawah hidung, dibawah hidung dan mulut, dan posisi pemakaian masker yang tidak sesuai.

Uji coba pada sistem deteksi pemakaian masker akan dilakukan dengan cara melakukan uji coba pada video dari kamera CCTV lalu sistem akan mengcapture

apabila terdapat bagian wajah dengan settingan yang sudah diatur pada sistem. Untuk menghitung tingkat keberhasilan pendeteksian ada dua parameter yang dapat dihitung dengan menggunakan persamaan sebagai berikut :

$$Tingkat\ Akurasi\ Pencocokan = \frac{JumlahWajahYangTerdeteksi}{JumlahSemuaWajah} \quad (3.1)$$

Tingkat akurasi ini berdasarkan pada banyaknya citra wajah yang tertangkap kamera yang dapat dikenali dengan tepat. Hal ini digunakan untuk mengukur apakah pendeteksi mampu mengenali berbagai macam wajah.

Untuk mengevaluasi dua *Bounding Box* yang dapat dihitung dengan menggunakan perhitungan akurasi sebagai berikut :

$$Precision = \frac{TP}{TP+FP} \quad (3.2)$$

Precision merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif.

$$Recall = \frac{TP}{TP+FN} \quad (3.3)$$

Recall adalah sebuah rasio yang mempredeksikan benar positif dan dibandingkan dengan jumlah keseluruhan data benar positi.
Dimana:
TP (*True Positive*) : Hasil yang benar
FP (*False Positive*) : Hasil yang tidak terduga
FN (*False Negative*) : Hasil yang hilang
TN (*True Negative*) : Tidak adanya hasil yang benar atau sesuai

Halaman ini sengaja dikosongkan