SISTEM PENGENALAN WAJAH MAHASISWA PRAKTIKUM DI LABORATORIUM TEKNIK ELEKTRO DAN INFORMATIKA POLMANBABEL MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK (CNN)*

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Sarjana Terapan Politeknik Manufaktur Negeri Bangka Belitung

Disusun oleh :

Fauzan Akbar Nugraha NIM : 1051909

Hanum Ayu Fazira NIM : 1051910

POLITEKNIK MANUFAKTUR NEGERI

BANGKA BELITUNG

TAHUN 2023

LEMBAR PENGESAHAN

SISTEM PENGENALAN WAJAH MAHASISWA PRAKTIKUM DI LABORATORIUM TEKNIK ELEKTRO DAN INFORMATIKA POLMANBABEL MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK (CNN)*

Oleh :

Fauzan Akbar Nugraha /1051909

Hanum Ayu Fazira /1051910

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Sarjana Terapan Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1	Pembimbing 2
(Muhammad Iqbal Nugraha, M.Eng)	(Aan Febriansyah, M.T)
Penguji 1	Penguji 2
(I Made Andik Setiawan, M.Eng, Ph.D)	(Indra Dwisaputra, M.T)

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan dibawah ini :

Nama Mahasiswa 1 : Fauzan Akbar Nugraha NIM : 1051909
Nama Mahasiswa 2 : Hanum Ayu Fazira NIM : 1051910

Dengan Judul : Sistem Pengenalan Wajah Mahasiswa di Laboratorium Teknik Elektro dan Informatika Polmanbabel Menggunakan *Convolutional Neural Network* (CNN)

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 07 Februari 2023

Nama Mahasiswa	Tanda Tangan
1. Fauzan Akbar Nugraha	……..
2. Hanum Ayu Fazira	……..

ABSTRAK

Pengenalan wajah termasuk ke dalam jenis biometrik yang didasarkan pada fitur wajah manusia. Pengenalan wajah ini sangat penting untuk sebuah instansi seperti perguruan tinggi yang memiliki banyak mahasiswa, sehingga para karyawan sulit untuk mengenali mahasiswa satu per satu. Penelitian ini menggunakan mahasiswa sebagai objek dan difokuskan pada satu ruang laboratorium dengan tujuan untuk mengidentifikasi siapa saja yang berada dalam ruangan laboratorium yang tujuannya untuk meningkatkan fungsi keamanan dan pengawasan. Metode yang digunakan dalam proyek akhir ini adalah Convolutional Neural Network (CNN). Berdasarkan hasil pengujian menggunakan 7000 data uji dan proses training sebanyak 5 epoch, sistem pengenalan wajah ini dapat berjalan secara real-time akan tetapi belum bekerja secara optimal. Sistem dapat mengenali wajah pada jarak optimum 4 meter dengan rata-rata pencahayaan 375 lux. Hasil pengujian juga menunjukkan bahwa sistem ini dapat mengenali lebih dari satu wajah dalam satu waktu, sehingga dapat membuat sistem bekerja secara efektif dan efisien. Hasil prediksi yang diperoleh dari pengujian menggunakan confusion matrix didapatkan akurasi sebesar 99,7%.

Kata kunci: Confusion matrix, Convolutional Neural Network (CNN), jarak, pengenalan wajah.

ABSTRACT

Facial recognition is a type of biometric based on human facial features. Face recognition is very important for an institution such as a university that has many students, so it is difficult for employees to recognize students one by one. This study uses students as objects and is focused on one laboratory room with the aim of identifying anyone who is in the laboratory room whose purpose is to improve security and surveillance functions. The method used in this final project is a Convolutional Neural Network (CNN). Based on the test results using 7000 test data and a training process of 5 epochs, this facial recognition system can run in real-time but has not worked optimally. The system can recognize faces at an optimum distance of 4 meters with an average lighting of 375 lux. The test results also show that this system can recognize more than one face at a time, so that the system can work effectively and efficiently. The prediction results obtained from testing using the confusion matrix obtained an accuracy of 99.7%.

Keywords: Confusion matrix, Convolutional Neural Network (CNN), distance, face recognition.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT atas karunia dan limpahan rahmat-Nya, sehingga penulis dapat menyelesaikan laporan proyek akhir ini yang berjudul "**Sistem Pengenalan Wajah Mahasiswa Praktikum di Laboratorium Teknik Elektro dan Informatika Polmanbabel Menggunakan *Convolutional Neural Network* (CNN)**". Shalawat beserta salam selalu tersampaikan kepada Nabi Muhammad SAW yang telah membawa umat manusia ke dunia yang damai, terang dan penuh dengan ilmu pengetahuan. Tujuan penulis membuat laporan proyek akhir ini adalah untuk memenuhi salah satu syarat kelulusan pendidikan Sarjana Terapan di Politeknik Manufaktur Negeri Bangka Belitung. Dalam laporan proyek akhir, penulis membahas tentang penelitian yang penulis laksanakan selama proyek akhir berlangsung. Dengan sistem pengenalan wajah menggunakan *Convolutional Neural Network* (CNN) ini diharapkan dapat meningkatkan fungsi pengawasan dan keamanan fasilitas khususnya laboratorium serta memudahkan karyawan dalam mengenali dan mengidentifikasi mahasiswa secara efektif dan efisien.

Pada kesempatan ini, ucapan terimakasih juga disampaikan kepada pihak yang telah banyak membantu serta memberikan motivasi, saran dan kritik yang tentunya sangat diharapkan dalam menyelesaikan proyek akhir ini. Berikut ini adalah pihak-pihak yang ikut membantu baik secara langsung maupun tidak langsung, diantaranya :

1. Orangtua penulis, Bapak Marianto, Bapak Darmadi, Ibu Indrawati, dan Ibu Sumarni yang telah banyak memberikan sumbangsih moril dalam bentuk doa dan dukungan kepada penulis.
2. Saudara penulis yang telah memberikan banyak nasihat serta motivasi dan dukungannya.
3. Bapak Muhammad Iqbal Nugraha, M.Eng. selaku Kepala Jurusan Teknik Elektro dan Informatika dan dosen pembimbing 1 proyek akhir penulis di Politeknik Manufaktur Negeri Bangka Belitung.

4. Bapak Aan Febriansyah, M.T. selaku dosen pembimbing 2 proyek akhir penulis di Politeknik Manufaktur Negeri Bangka Belitung.
5. Bapak I Made Andik Setiawan, M.Eng., Ph.D. selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
6. Bapak Indra Dwisaputra, M.T. selaku Kepala Prodi D-IV Teknik Elektronika Politeknik Manufaktur Negeri Bangka Belitung.
7. Seluruh dosen dan PLP yang telah banyak membantu penulis dalam menyelesaikan proyek akhir ini.
8. Teman-teman UKKI Al-Farisi Politeknik Manufaktur Negeri Bangka Belitung Angkatan 2019 yang telah membersamai selama hampir 4 tahun ini.
9. Teman-teman UKKI Al-Farisi Politeknik Manufaktur Negeri Bangka Belitung Angkatan 2020-2022 yang telah memberikan semangat dan motivasi sehingga penulis dapat menyelesaikan proyek akhir ini.
10. Seluruh rekan kerja kepengurusan FSLDK Kepulauan Bangka Belitung yang telah banyak memberikan semangat dan motivasi kepada penulis.
11. Seluruh rekan-rekan kelas 4 TE yang telah banyak membantu dan membersamai hampir 4 tahun ini.
12. Seluruh pihak-pihak bersangkutan yang tidak dapat penulis sebutkan satu per satu.

Penulis sangat berharap makalah ini memberikan manfaat dalam rangka menambah wawasan dan ilmu pengetahuan yang terdapat dalam proyek akhir ini. Penulis menyadari bahwa dalam penulisan laporan proyek akhir ini masih terdapat banyak kekurangan dan jauh dari kata sempurna. Oleh sebab itu, penulis mengharapkan kritik dan saran yang membangun demi perbaikan laporan penulis di masa yang akan datang. Mudah-mudahan laporan proyek akhir ini dapat dipahami dan bermanfaat untuk semua orang khususnya bagi para pembaca makalah ini.

Sungailiat, 07 Februari 2023

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1. Latar Belakang

Pengenalan wajah termasuk kedalam jenis biometrik yang didasarkan pada fitur wajah manusia. Teknik yang digunakan dalam pengenalan wajah ini termasuk kedalam sistem yang memiliki tingkat konkurensi tinggi, ramah pengguna juga tidak perlu dilakukan kontak langsung [1]. Apabila dibandingkan dengan jenis biometrik yang lain seperti iris mata, vena, telapak tangan, ataupun sidik jari, wajah memiliki potensi yang jauh lebih baik jika digunakan untuk mengenali identitas seseorang. Hal ini lah yang mendasari pengenalan wajah banyak digunakan di berbagai aplikasi pengawasan, juga pengontrolan [2].

Pengenalan wajah ini dirasa sangat penting untuk sebuah instansi apalagi perguruan tinggi yang memiliki banyak mahasiswa sehingga para karyawan sulit untuk mengenali mahasiswa satu per satu. Akses keluar masuk laboratorium yang sangat mudah ini, akan mengakibatkan keamanan yang ada di dalam laboratorium tidak terkontrol dengan baik. Dengan adanya pengenalan wajah mahasiswa ini, maka akan memudahkan karyawan mengontrol keamanan dan pengawasan mahasiswa yang ada di dalam laboratorium.

Berdasarkan keterangan diatas, perlu adanya sistem yang dapat mengidentifikasi wajah seseorang sebagai bukti autentik penggunaan fasilitas di dalam kampus khususnya laboratorium. Oleh karena itu, proyek akhir kami yang berjudul **"Sistem Pengenalan Wajah Mahasiswa Praktikum di Laboratorium Teknik Elektro dan Informatika Polmanbabel Menggunakan *Convolutional Neural Network* (CNN)"** ini kami angkat untuk meningkatkan fungsi pengawasan dan pengamanan di kampus khususnya laboratorium. Sasaran dari sistem pengawasan ini untuk memudahkan dan mengenali objek deteksi wajah manusia dengan indikasi output berupa identitas umum dari mahasiswa tersebut.

1.2. Rumusan Masalah

Rumusan masalah yang terdapat pada proyek akhir ini adalah sebagai berikut:

1. Bagaimana cara yang efektif dan efisien untuk mengolah *image* yang bergerak sebelum digunakan untuk pengenalan wajah?
2. Bagaimana algoritma *Convolutional Neural Network (CNN)* yang optimal untuk mengenali wajah manusia dengan cepat dan akurat?

1.3. Tujuan Penelitian

Tujuan yang ingin penulis capai dalam proyek akhir yang berjudul **Sistem Pengenalan Wajah Mahasiswa Praktikum di Laboratorium Teknik Elektro dan Informatika Polmanbabel Menggunakan *Convolutional Neural Network* (CNN)** ini adalah sebagai berikut :

1. Menerapkan sistem pengenalan wajah menggunakan *Convolutional Neural Network* (CNN)
2. Mengetahui tingkat arukasi yang didapatkan menggunakan *Convolutional Neural Network* (CNN)
3. Mengenali identitas umum mahasiswa dengan deteksi objek berupa wajah
4. Meningkatkan fungsi pengawasan penggunaan fasilitas kampus secara efektif dan efisien

1.4. Batasan Masalah

Agar masalah dalam proyek akhir ini lebih terarah dan sesuai dengan apa yang sudah direncanakan dan mempermudah mendapatkan data serta informasi yang diperlukan, maka penulis menetapkan beberapa batasan masalah, sebagai berikut:

1. Objek deteksi hanya mahasiswa Jurusan Teknik Elektro dan Informatika Polmanbabel.
2. Objek deteksi dapat mendeteksi wajah mahasiswa yang tidak menggunakan masker dan aksesoris wajah.

3. Objek deteksi juga dapat mendeteksi wajah mahasiswa yang menggunakan aksesoris berupa kacamata.
4. Indikator output yang dihasilkan hanya berupa identitas mahasiswa yaitu nama.

BAB II
DASAR TEORI

2.1. Pengenalan Wajah (*Face Recognition*)

Deteksi wajah merupakan proses untuk mengenali berbagai bentuk wajah dengan mencocokan lekuk wajah dengan gambar yang tersimpan. Identifikasi wajah banyak diteliti pada *computer* vision sebagai penerapan pada aplikasi keamanan [3]. Wajah menjadi objek utama dari bagian tubuh manusia saat berinteraksi. Wajah memiliki peran yang sangat penting dalam menunjukkan identitas seseorang. Sering kita jumpai ratusan bahkan ribuan wajah manusia saat berinteraksi dengan waktu yang cukup lama. Akan tetapi sangat sulit membedakan ataupun mengenali seseorang dengan rentang waktu yang bersamaan terutama pada kerumunan ataupun masa yang banyak. Hal inilah yang mendasari wajah menjadi indikasi dalam mengenali seseorang yang biasa disebut dengan *face recognition.*

Terdapat beberapa metode yang digunakan dalam pengklasifikasian dan ekstraksi citra dalam pengenalan wajah seperti, *Support Vector Machine* (SVM), *Face Features, Principal Component Analysis* (PCA), *Eigenface, Local Binary Pattern Histogram*(LBPH) dan *Convolutional Neural Network* (CNN). CNN merupakan salah satu bagian dari model *deep learning* yang mampu mengenali wajah dengan tingkat akurasi yang tinggi secara *real-time* dan statik [4].

Berdasarkan penelitian sebelummya yang telah membuat sistem pengenalan wajah manusia dengan tingkat akurasi menggunakan proses *dropout* sebesar 89.73% dan pengujian data *testing* dengan tingkat akurasi sebesar 75.79%. Maka dapat disimpulkan bahwa peneliti sebelumnya cukup berhasil membuat sistem pengenalan wajah manusia. Akan tetapi, sistem tersebut belum menampilkan identitas seseorang [5].

Penelitian sebelumnya yang meneliti tentang pengenalan wajah dengan *Haar Cascade Classifier* dengan *Local Binary Pattern Histogram.* Data *training* wajah

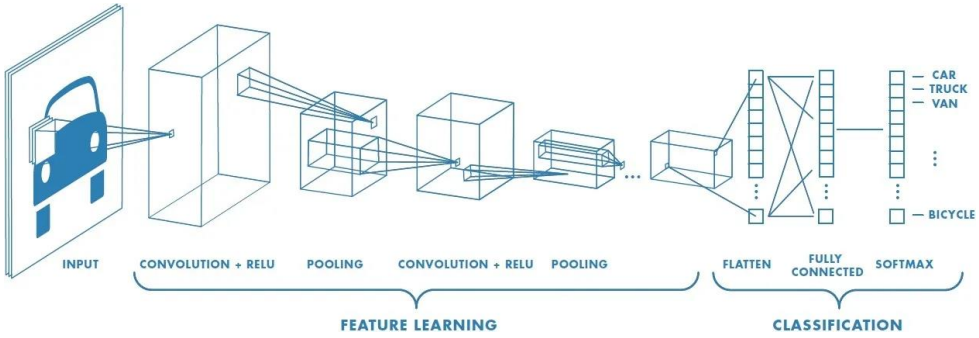
yang digunakan untuk mendeteksi wajah secara berkelompok ataupun sendiri dengan sistem komputasi yang terbilang cukup efektif pada jarak 50 – 100 cm [6].

Selain itu, penelitian mengenai pengenalan wajah dengan *Convolutional Neural Network* (CNN) yang menggunakan *framework* Open CV dengan dataset yang digunakan adalah data gambar wajah yang dilakukan secara real-time. Penelitian ini menghasilkan sebuah model *Convolutional Neural Network* (CNN) dengan kedalaman jaringan sebanyak 7 layer dengan tingkat akurasi sebesar 87% [7].

2.2. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan pengembangan dari *Multilayer Perceptron* (MLP). *Convolutional Neural Network* (CNN) termasuk kedalam jenis *Deep Neural Network* yang didesain untuk mengolah data dua dimensi karena memiliki kedalaman jaringan yang tinggi dan banyak sekali diaplikasikan pada data *image*. CNN digunakan untuk mendeteksi dan mengenali berbagai objek gambar, menganalisis gambar visual, yang merupakan vektor berdimensi tinggi yang akan melibatkan banyak parameter untuk mencirikan jaringan [8].

Convolutional merupakan matriks yang difungsikan untuk melakukan penyaringan pada gambar ataupun menggabungkan dua buah deret angka yang akan menghasilkan deret angka ketiga. Dalam proses pembuatan model terdapat 3 tahapan yang perlu dilakukan, yakni *Convolutional layer, Pooling layer,* dan *Fully connected layer* [9].

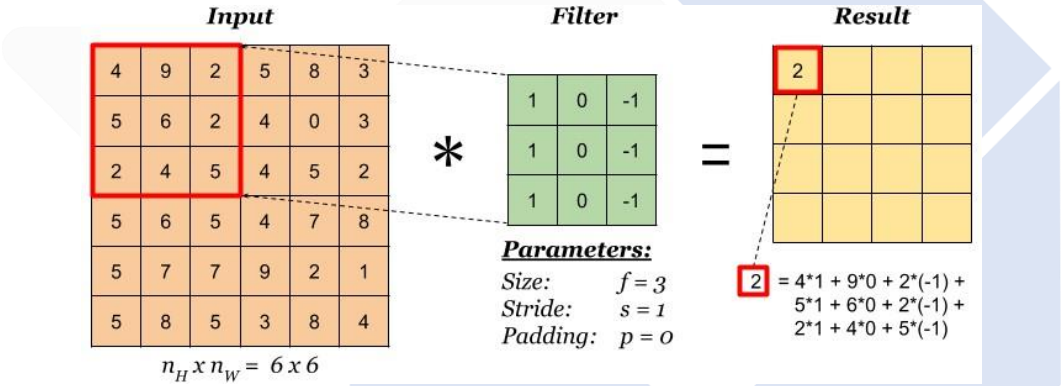


Gambar 2.1 Model CNN [9]

2.2.1. Convolutional Layer

Pada lapisan *convolutional*, data akan mengalami proses konvolusi, yang mana seluruh lapisan yang masuk dikonversi setiap *filter* ke dalam data masukan yang akan menghasilkan sebuah *activition map* atau *feature map*. Panjang, tinggi dan tebal pada *Filter* yang tersedia sudah sesuai dengan data masukannya. Setiap *filter* akan mengalami pergeseran antara data input dan nilai *filter* yang bergerak dari sudut kiri atas ke kanan bawah [9].

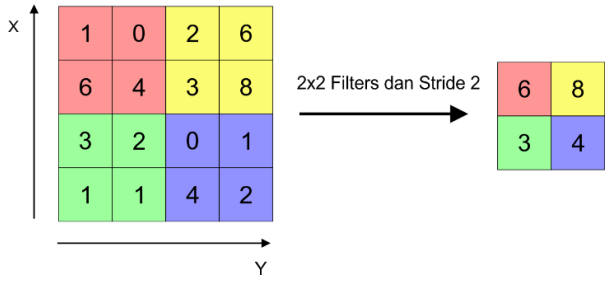
Proses konvolusi dilakukan pada data citra dengan tujuan untuk memisahkan fitur dari *input* yang akan menghasilkan transformsi linear sesuai dengan informasi yang terdapat pada data [10].



Gambar 2.2 *Convolutional Layer* [10]

2.2.2. Pooling Layer

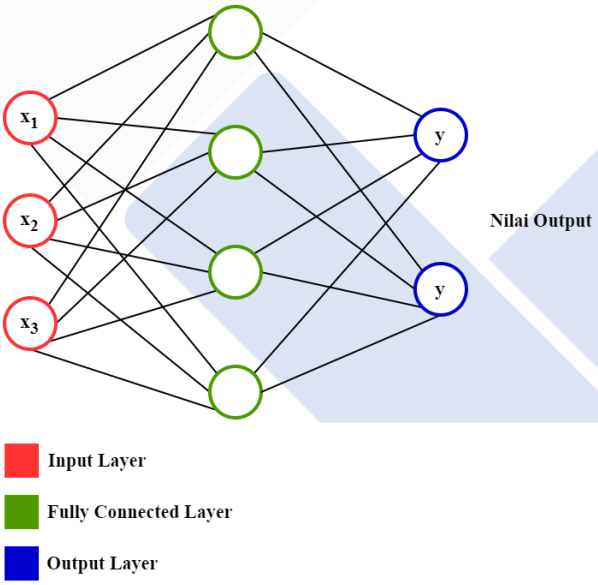
Pooling Layer adalah proses lanjutan dari *Convolutional Layer*. Di dalam *Pooling Layer* terdapat sebuah filter yang memiliki ukuran dan *stride* tertentu. Proses pergeseran ditentukan dari jumlah *stride* yang akan digeser pada area *feature map* atau *activation map*. *Pooling layer* terdiri dari dua jenis, yaitu *average pooling* dan *max pooling*. Pada jenis *max pooling* yang diambil adalah nilai yang terbesar, sedangkan *average pooling* diambil nilai rata-rata [9].



Gambar 2.3 *Pooling Layer* [9]

2.2.3. *Fully Connected Layer*

Multidimensional array merupakan bentuk dari *feature map* yang dihasilkan dari tahap sebelumnya. Sebelum masuk ke tahap *fully connected* layer ini, *feature map* harus melalui proses "*flatten*" atau *reshape* terlebih dahulu, yang fungsinya untuk menghasilkan sebuah vektor yang nantinya akan dijadikan sebagai input [9].



Gambar 2.4 *Fully Connected Layer* [9]

2.3. *Python*

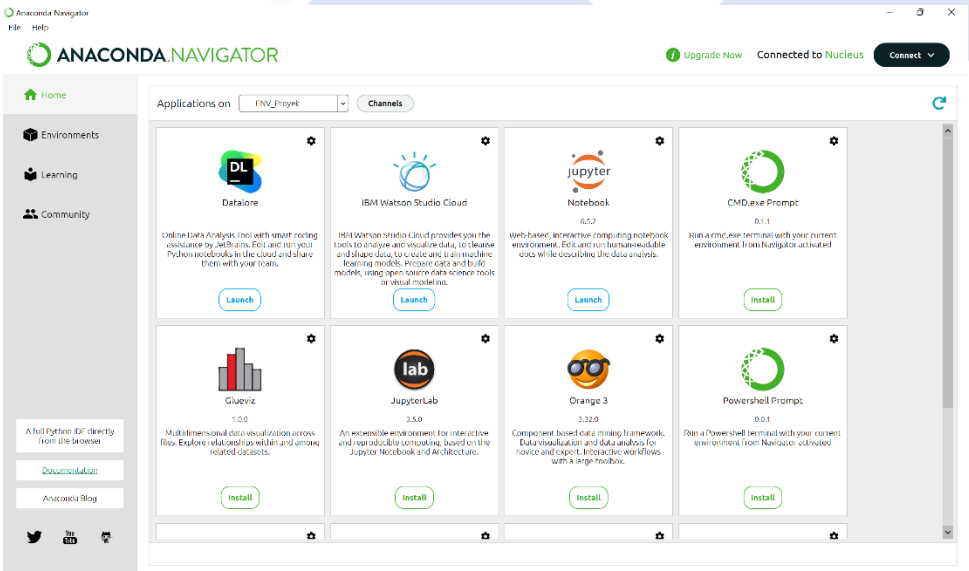
Bahasa Pemograman *Python* memiliki beberapa keunggulan salah satu diantaranya ialah memiliki *library* yang komprehensif.

Beberapa *library* pada *Python* yang digunakan dalam pengerjaan proyek akhir ini yaitu:

1. *OpenCV* – Library yang difungsikan untuk pemrograman yang ditujukan ke *computer vision* [7].
2. *Tensorflow* – Digunakan diberbagai bidang, salah satunya *object detection* yang didalamnya terdapat *framework tensorflow object detection API*, yang digunakan untuk mempermudah proses *constructing*, *training* dan *deployment* pada model *object detection* [11].
3. *Keras* – merupakan bahasa pemrograman tingkat tinggi yang memudahkan pada pengguna untuk melakukan eksperimentasi *neural network*.

2.3.1 *Anaconda*

Anaconda merupakan salah satu aplikasi yang digunakan untuk mendistribusikan bahasa pemrograman *Python* yang memiliki sifat *open source*. *Anaconda* bertujuan untuk dapat menyederhanakan proses manajemen *package* atau *deployment*. *Anaconda* memiliki lebih dari 1500 *package* jumlah distribusi yang dapat diakses dari berbagai sistem operasi seperti *MacOS*, *Windows* dan *Linux* [12].

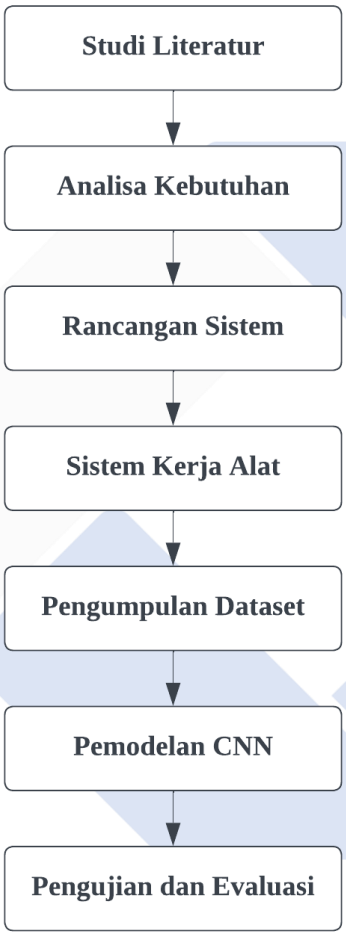


Gambar 2.5 Tampilan *Anaconda Navigator*

Pada saat menggunakan *Anaconda*, maka beberapa *package* sudah dapat digunakan seperti *pandas*, *numpy*, *jupyter* dan lain sebagainya. Kelebihan dari *Anaconda* ini sendiri adalah pengguna dapat memilih versi pemrograman *python* yang kita inginkan [12].

BAB III
METODE PELAKSANAAN

Dalam proses pengerjaan proyek akhir ini, terdapat beberapa tahapan dan dirancang secara sistematis agar mempermudah selama pengerjaan proyek akhir ini.



Gambar 3.1 Diagram Metodologi

Adapun tahapan dalam pengerjaan proyek akhir ini diuraikan sebagai berikut :

3.1. Studi Literatur

Studi Literatur merupakan tahap awal dalam pengerjaan proyek akhir ini dengan melakukan konsultasi dengan dosen pembimbing dan mengumpulkan referensi-

referensi yang berkaitan dengan permasalahan pokok yang akan diatasi, serta mengkaji dari berbagai penelitian terdahulu terkait pengenalan wajah dari jurnal, artikel dan sumber yang lainnya. Pada tahap ini juga mengkaji terkait model *Convolutional Neural Network* (CNN), Open CV dan pengolahan citra. Sehingga tujuan dari studi literatur ini untuk dapat mengidentifikasi masalah serta menyusun rencana kerja dari masalah yang dihadapi dapat tercapai.

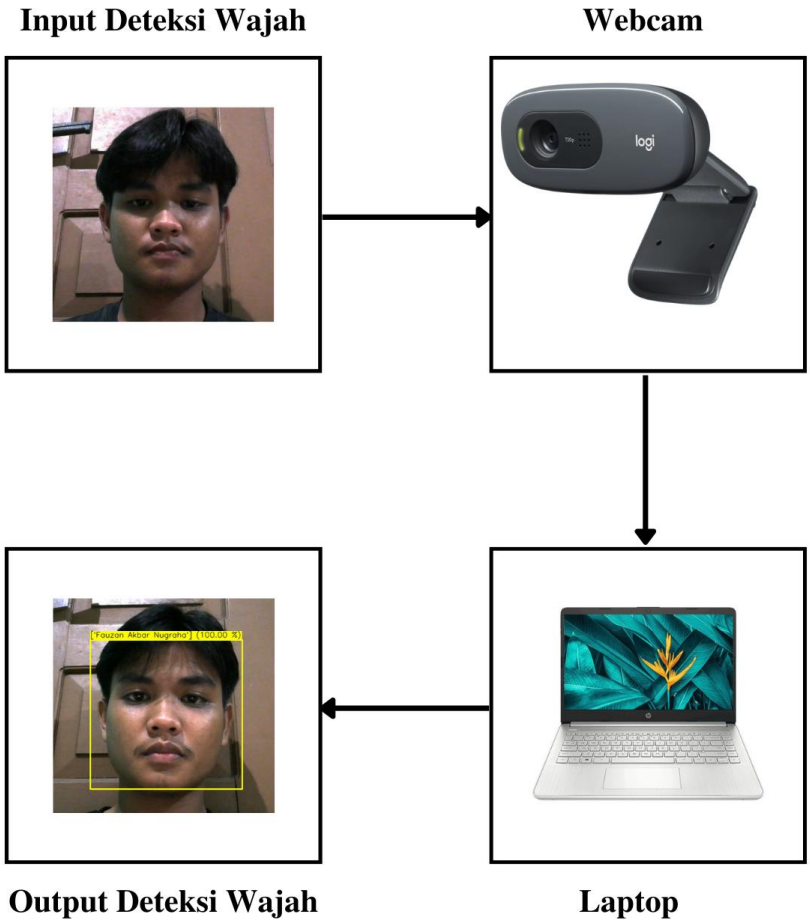
3.2. Analisis Kebutuhan

Analisis kebutuhan sistem ini merupakan tahap kedua sebagai acuan untuk pengerjaan Proyek Akhir ini:

1. Sistem dapat mendeteksi wajah mahasiswa dengan berbagai kondisi sesuai dengan yang terdapat dalam batasan masalah.
2. Sistem dapat mendeteksi wajah mahasiswa secara *real-time* berdasarkan data yang sudah dimasukkan.
3. Sistem dapat bekerja secara optimal pada cahaya dan jarak yang sesuai dengan spesifikasi *webcam* yang digunakan.

3.3. Rancangan Sistem

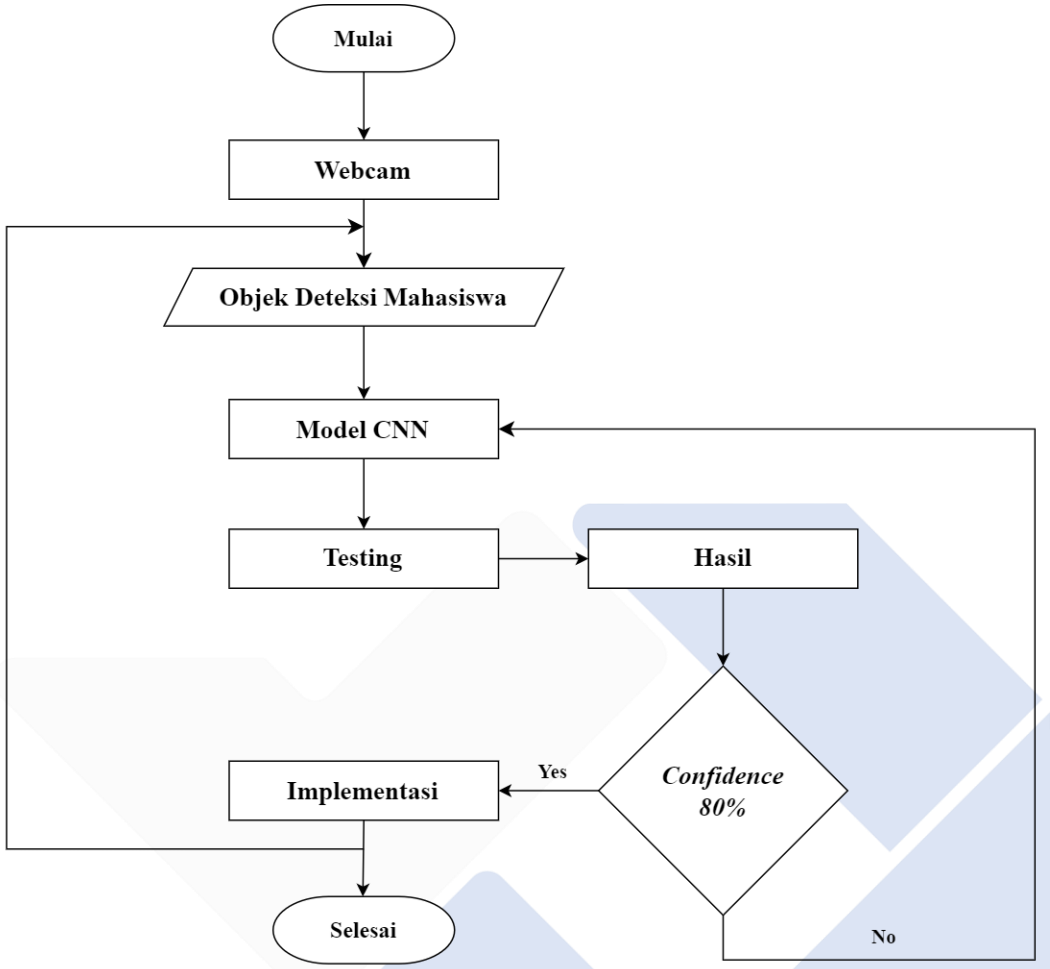
Sistem ini dikembangkan dengan bahasa pemrograman *Python* dan menggunakan *Library OpenCV*. *Library OpenCV* yang digunakan akan menghubungkan *webcam* yang selanjutnya akan mengimplementasikan deteksi wajah. *Webcam* yang digunakan pada proyek akhir ini yaitu *Logitech C270 HD Webcam* dengan resolusi 720p 30 fps. *Input* dari sistem ini berupa gambar yang sudah diambil yang kemudian dibuatkan kelas folder sesuai dengan nama masing – masing. Gambar yang sudah dimasukkan kedalam kelas folder kemudian diproses dengan metode *Convolutional Neural Network* (CNN). Selanjutnya, gambar yang telah diproses akan menghasilkan *output* berupa nama sebagai identitas dari mahasiswa tersebut. Ilustrasi ditunjukkan pada Gambar 3.3.



Gambar 3.2 Blok Diagram Rancangan Sistem

3.4. Sistem Kerja Alat

Pada pengerjaan proyek akhir ini penulis menyertakan *flowchart* tentang cara kerja dari alat yang dibuat. Adapun sistem kerja alat ditunjukkan pada Gambar 3.2.



Gambar 3.3 *Flowchart* Sistem Pengenalan Wajah

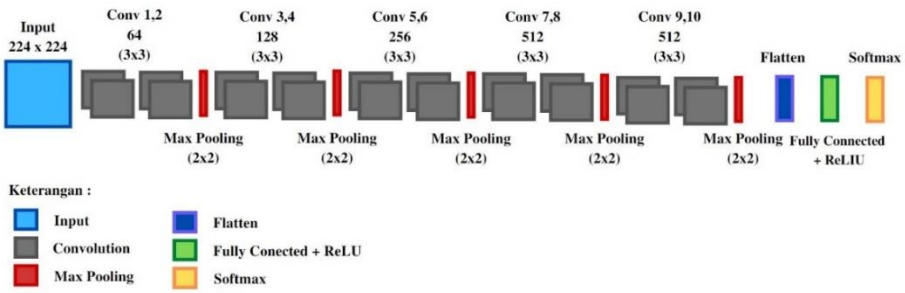
Pada bagian *flowchart* ini dimulai dengan objek deteksi mahasiswa yang diarahkan ke *webcam* untuk dikenali. Selanjutnya model di-*training* dengan model CNN yang akan melatih dataset untuk kemudian mendapatkan besaran akurasi dari hasil training model CNN ini. Tahap selanjutnya masuk ke bagian *testing*, proses *testing* ini adalah proses untuk mengenali. Jika hasil dari *testing* ini didapatkan *Confidence* lebih dari atau sama dengan 80% maka wajah mahasiswa dapat dikenali atau diimplementasikan, ditandai dengan indikator nama mahasiswa dan persentase *Confidence*-nya, sebaliknya jika *Confidence* kurang dari 80% maka wajah mahasiswa tidak dapat dikenali yang ditandai dengan indikator "Tidak Diketahui".

3.5. Pengumpulan Dataset

Sistem pengenalan wajah ini menggunakan dataset wajah mahasiswa Polmanbabel jurusan Teknik Elektro dan Informatika. Dataset yang diambil berupa foto wajah dalam beberapa kondisi seperti wajah tampak depan, tampak samping dan pada saat menggunakan kacamata bagi mahasiswa yang menggunakan kacamata, namun data foto yang penulis ambil lebih dominan pada tampak depan. Setelah melakukan pengambilan data foto sebanyak 28 folder kelas data dengan 27 nama mahasiswa dan 1 untuk yang tidak diketahui. Folder kelas data tidak diketahui ini diambil secara acak ataupun diluar dari mahasiswa yang ada di 27 kelas data folder lainnya, hal ini agar saat pendeteksian mahasiswa yang tidak termasuk kedalam dataset tidak akan muncul identitas namanya namun *output*-nya akan berupa "Tidak Diketahui". Dataset yang terkumpul 28 kelas data folder berisi 50 foto untuk masing – masing kelas data, sehingga jumlah data yang terkumpul sebanyak 1.400 foto.

3.6. Pemodelan *Convolutional Neural Network* (CNN)

Pada tahap ini dilakukan proses pemodelan atau yang biasa dikenal dengan *modelling*. Proses ini dimulai dengan menentukan kedalaman jaringan, susunan *layer* dan pemilihan jenis *layer*. Model CNN yang digunakan sebagai berikut:



Gambar 3.4 Arsitektur CNN

Penjelasan terkait arsitektur pada pemodelan CNN diuraikan seperti dibawah ini:

1) Input citra 224 x 224 x 1

Apabila citra input memiliki ukuran gambar yang berbeda maka dilakukan tahap *reshape* menjadi ukuran 224 x 224 x 1.

2) *Convolution* 64 3 x 3
   Proses konvolusi pertama ini dengan ukuran kernel 3 x 3 sebanyak 64 kali akan menangkap input citra.
3) *Convolution* 64 3 x 3
4) *Max Pooling* 2 x 2
5) *Convolution* 128 3 x 3
   Proses konvolusi ini dilakukan pada tahap *convolution* 2 dan 3 ini dengan ukuran kernel 3 x 3 sebanyak 128 kali akan menangkap input citra.
6) *Convolution* 128 3 x 3
7) *Max Pooling* 2 x 2
8) *Convolution* 256 3 x 3
   Proses konvolusi pertama ini dengan ukuran kernel 3 x 3 sebanyak 256 kali akan menangkap input citra.
9) *Convolution* 256 3 x 3
10) *Max Pooling* 2 x 2
11) *Convolution* 512 3 x 3
    Proses konvolusi pertama ini dengan ukuran kernel 3 x 3 sebanyak 512 kali akan menangkap input citra.
12) *Convolution* 512 3 x 3
13) *Max Pooling* 2 x 2
14) *Convolution* 512 3 x 3
15) *Convolution* 512 3 x 3
16) *Flatten*
    Flatten difungsikan untuk membentuk ulang data kedalam bentuk vektor.
17) *Fully Connected Layer*
    Data input pada layer ini adalah data dalam bentuk vektor yang merupakan hasil dari proses *flatten.*
18) ReLU
19) *Softmax*
    Softmax digunakan sebagai aktifasi yang sering dipakai untuk *multiclass classification* karena keluarannya memiliki lebih dari satu *neuron.*

3.7. Pengujian dan Evaluasi

Tahap pengujian ini dilakukan untuk mengetahui keberhasilan sistem sesuai dengan analisa kebutuhan dan rancangan serta *output* yang sudah dirancang. Tahap uji coba ini dilakukan untuk menjalankan implementasi dari sistem yang sudah dirancang sesuai dengan kebutuhan. Dalam proses pungujian ini didasarkan pada inputan data pelatihan dan *labeling* untuk memjustifikasi proses pemodelan.

Apabila saat pengujian sistem mengalami gangguan atau *error*, maka proses selanjutnya ialah melakukan evaluasi sebagai tahap perbaikan pada sistem. Namun, apabila dalam pengujian tidak terdapat gangguan ataupun sudah sesuai dengan rancangan dan kebutuhan maka pembuatan sistem selesai.

BAB IV
PEMBAHASAN

Pada bab ini penulis membahas dan menguraikan terkait Tahapan pembuatan sistem yang dilakukan selama proses pengerjaan proyek akhir ini. Berikut tahapan yang penulis lakukan dan uraikan pada bab ini tentang :

1. Pengumpulan Dataset
2. Pengolahan Citra
3. Pemodelan CNN
4. Pengujian
5. Hasil Pengujian

4.1. Pengumpulan Dataset

Dataset yang penulis ambil sebanyak 1.400 foto dengan 50 foto untuk setiap mahasiswa. Dataset ini kemudian akan dikumpulkan berdasarkan kelas data folder yang sudah dibedakan sesuai nama dari masing – masing mahasiswa sebanyak 28 kelas data. Dataset ini kemudian akan mengalami proses pengolahan citra yang nantinya akan dijadikan sebagai *input* dalam proses sistem pengenalan wajah. Data yang ada akan dibagi menjadi data *training* dan *testing*.

Gambar 4.1 Dataset

4.2. Pengolahan Citra

Setelah pengumpulan dataset dilakukan, selanjutnya dataset masuk ke proses pengolahan citra yang melalui berbagai tahapan. Tahapan dalam pengolahan citra sebagai berikut :

1. *Resize*

Ukuran semula dari gambar yang diambil secara manual dengan kamera *handphone* berukuran 3456 x 3456 piksel. Ukuran ini sangat besar untuk dijadikan sebagai data *input.* Kemudian penulis melakukan proses *resize* gambar menjadi 500 x 500 piksel sebagai *input* dataset. Proses ini dilakukan karena ukuran semula sangat besar dan akan berpengaruh terhadap proses *training.* Proses *training* akan berjalan semakin lama apabila ukuran gambar semakin besar.

2. *Cropping*

Cropping merupakan proses pemotongan beberapa sisi pada gambar untuk mendapatkan gambar wajah yang utuh, proses ini dilakukan agar membantu sistem untuk lebih mudah mengenali wajah. Proses *cropping* ini menggunakan *haar cascade* untuk pemotongannya. Data hasil dari *cropping* ini kemudian di-*resize* kembali menjadi 224 x 224 piksel.

Berikut program untuk proses *cropping* dataset:

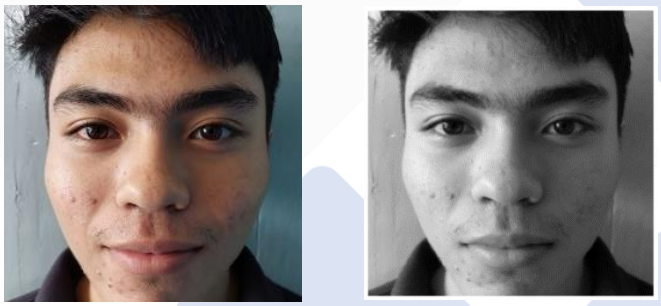
```
face_cascade=cv2.CascadeClassifier('haarcascades/haarcascade
    _frontalface_default.xml')
def detect_face(img):
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces=face_cascade.detectMultiScale(img,1.1,5)
       try :
            x, y, w, h = faces[0]
             img = img[y:y+h, x:x+w]
             img = cv2.resize(img, (224, 224))
      except :
      print("Face not found in image index", i)
            img = None
 return img
```

Keterangan :

Program diatas digunakan untuk mendefinisikan *Haarcascade Classfier* untuk memotong atau *cropping* bagian wajah dan *resize* gambar dengan ukuran 224 x 224 pixel.

3. *Grayscale*

Proses yang selanjutnya dilakukan ialah proses *grayscale*, dimana akan memproses dataset yang sebelumnya sudah di-*resize* dengan citra yang masih RGB yang kemudian akan di konversi ke *grayscale*. Proses ini dilakukan untuk mempermudah dan mempercepat proses pemodelan. Hasil sebelum dan setelah dilakukan proses *grayscale* dapat dilihat pada gambar 4.2.



(a) (b)

Gambar 4.2 (a) Sebelum di *grascale* (b) Setelah di *grayscale*

4. *Augmentation*

Pengenalan wajah memerlukan *input* dataset dengan jumlah data yang banyak. Dikarenakan dataset yang sudah penulis siapkan kurang beragam, maka penulis melakukan proses augmentasi untuk memperbanyak jumlah dataset dengan memperbanyak sebanyak 20 kali dari jumlah keseluruhan dataset yang sudah disiapkan dari awal. Augnmentasi ini tidak hanya memperbanyak jumlah dataset akan tetapi juga membuat dataset menjadi lebih beragam. Beberapa contoh dari proses augmentasi ialah membalik, memperbesar, memperkecil dan juga

mempertajam gambar. Proses ini akan menghasilkan akurasi model yang lebih tinggi.

Berikut program yang digunakan untuk proses augmentasi:

1. Pembuatan Augmentasi

```
def img_augmentation(img):
    h, w = img.shape
    center = (w // 2, h // 2)
    M_rot_5 = cv2.getRotationMatrix2D(center, 5, 1.0)
    M_rot_neg_5  =  cv2.getRotationMatrix2D(center,  -5,
    1.0)
    M_rot_10 = cv2.getRotationMatrix2D(center, 10, 1.0)
    M_rot_neg_10 = cv2.getRotationMatrix2D(center, -10,
    1.0)
    M_trans_3 = np.float32([[1, 0, 3],[0, 1, 0]])
    M_trans_neg_3 = np.float32([[1, 0, 3],[0, 1, 0]])
    M_trans_6 = np.float32([[1, 0, 6],[0, 1, 0]])
    M_trans_neg_6 = np.float32([[1, 0, -6],[0, 1, 0]])
    M_trans_y3 = np.float32([[1, 0, 0],[0, 1, 3]])
    M_trans_neg_y3 = np.float32([[1, 0, 0],[0, 1, 3]])
    M_trans_y6 = np.float32([[1, 0, 0],[0, 1, 6]])
    M_trans_neg_y6 = np.float32([[1, 0, 0],[0, 1, -6]])

    imgs = []
    imgs.append(cv2.warpAffine(img,  M_rot_5,  (w,  h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img, M_rot_neg_5, (w, h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_rot_10,  (w,  h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_rot_neg_10,  (w,
    h), borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_trans_3,  (w,  h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_trans_neg_3,  (w,
    h), borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_trans_6,  (w,  h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,  M_trans_neg_6,  (w,
    h), borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img, M_trans_y3, (w, h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img, M_trans_neg_y3, (w,
    h), borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img, M_trans_y6, (w, h),
    borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img, M_trans_neg_y6, (w,
    h), borderValue=(255,255,255)))
    imgs.append(cv2.add(img, 10))
```

```
        imgs.append(cv2.add(img, 30))
        imgs.append(cv2.add(img, -10))
        imgs.append(cv2.add(img, -30))
        imgs.append(cv2.add(img, 15))
        imgs.append(cv2.add(img, 45))
        imgs.append(cv2.add(img, -15))
        imgs.append(cv2.add(img, -45))

       return imgs
```

Keterangan:

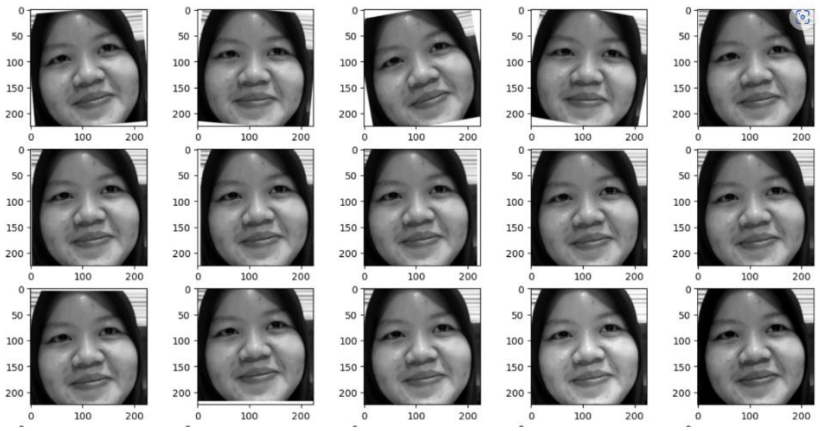
Program tersebut digunakan untuk mengatur posisi dari dataset yang akan diperbanyak nantinya. Data dibuat terbalik, miring dan juga mempertajam pencahayaan

2. Memperbanyak dataset

```
augmented_images = []
augmented_names = []
for i, img in enumerate(images):
    try :
augmented_images.extend(img_augmentation(img))
augmented_names.extend([names[i]] * 20)
    except :
        print(i)
```

Keterangan:

Program di atas digunakan untuk memperbanyak jumlah dataset sebanyak 20 kali lebih banyak dari dataset awal, sehingga total dataset setelah dilakukan proses augmentasi sebanyak 28.000.



Gambar 4.3 Hasil Augmentasi

Pada gambar di atas merupakan hasil dari proses augmentasi untuk memperbanyak dataset dengan mengatur posisi dan tata letak gambar serta pencahayaannya.

4.3. Pemodelan CNN

Setelah melewati proses pengolahan citra, proses selanjutnya ialah melakukan pemodelan untuk mempelajari pola yang mana akan menghasilkan suatu pengenalan objek yang sesuai dengan pola yang sudah dilatih. Proses ini dimulai dengan menentukan kedalaman jaringan kemudian dilakukan pembuatan layer dan dilanjutkan dengan proses *training.*

Berikut program pemodelan CNN:

1. Pembuatan *layer*

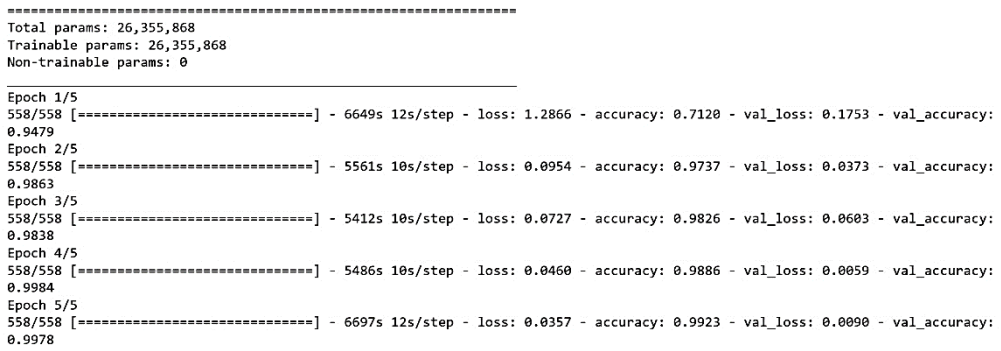
```
def cnn_model(input_shape):
  model = Sequential()

  model.add(Convolution2D(64,
                    (3, 3),
                    padding = 'valid',
                    activation = 'relu',
                          input_shape = input_shape))
  model.add(ZeroPadding2D((1, 1)))

  model.add(Convolution2D(64,
                          (3, 3),
                          padding = 'valid',
                          activation = 'relu',
                          input_shape = input_shape))
  model.add(MaxPooling2D((2, 2), strides = (2, 2)))
  model.add(ZeroPadding2D((1, 1)))

  model.add(Convolution2D(128,
                          (3, 3),
                          padding = 'valid',
                          activation = 'relu'))
  model.add(ZeroPadding2D((1, 1)))

  model.add(Convolution2D(128,
                          (3, 3),
                          padding = 'valid',
                          activation = 'relu'))
  model.add(MaxPooling2D((2, 2), strides = (2, 2)))
  model.add(ZeroPadding2D((1, 1)))

  model.add(Convolution2D(256,
                          (3, 3),
                          padding = 'valid',
```

```
                            activation = 'relu'))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(256,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(MaxPooling2D((2, 2), strides = (2, 2)))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(512,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(512,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(MaxPooling2D((2, 2), strides = (2, 2)))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(512,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(512,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
     model.add(MaxPooling2D((2, 2), strides = (2, 2)))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Flatten())
    model.add(Dense(512, activation="relu"))
    model.add(Dense(256, activation="relu"))
    model.add(Dense(128, activation="relu"))
    model.add(Dense(64, activation="relu"))
    model.add(Dense(len(labels)))
     model.add(Activation('softmax'))

    model.summary()

    model.compile(optimizer='adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

  return model
```

Keterangan:
Program diatas digunakan untuk membuat layer pada CNN, yang mana *layer* yang

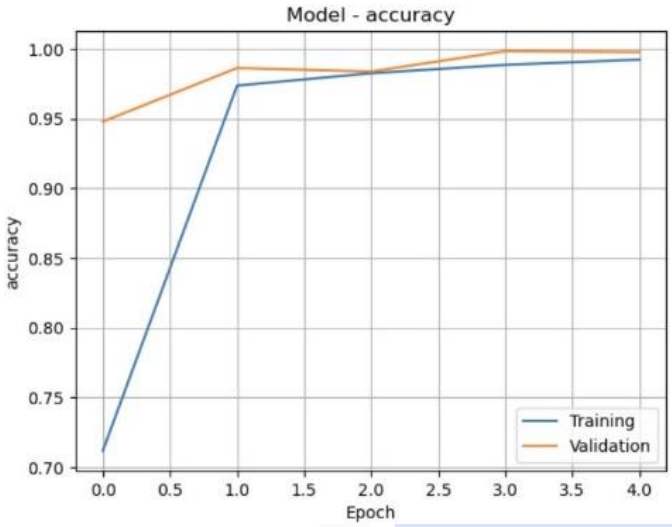
digunakan terdiri dari 10 *Convolutional Layer*, 5 *Max Pooling Layer*, 1 *Flatten Layer*, 5 *Fully Connected Layer*, dan 1 *Activation Layer* (*Softmax*).

2. Proses *training*



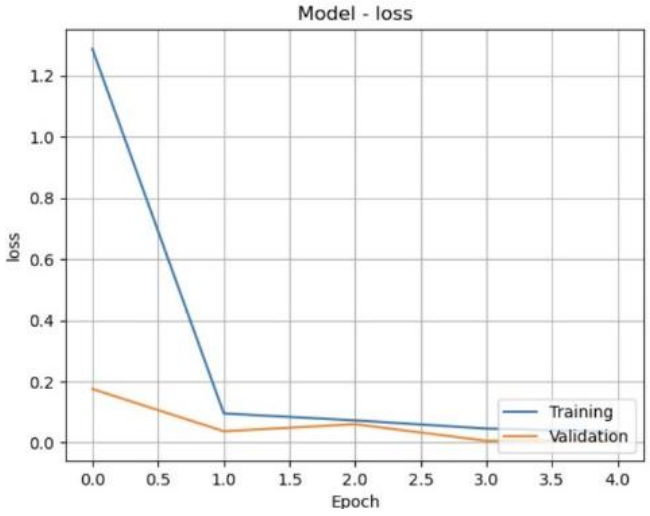
Gambar 4.4 Hasil proses *training* data

Pada proses pemodelan CNN ini data *training* sebanyak 21.000 (75%) dari total keseluruhan dataset. Proses ini dilakukan dengan jumlah *epoch* yang digunakan sebanyak 5 *epoch* yang kurang lebih memakan waktu *training* sebanyak 12 jam. Cara kerja dari *epoch* ini ialah melakukan proses training secara keseluruhan dari awal hingga akhir yang kemudian akan menunjukkan nilai *accuracy* dan *loss.* Semakin besar nilai *accuracy* dan semakin kecil nilai *loss* maka semakin bagus model yang dihasilkan. Begitu pula sebaliknya, apabila nilai *accuracy* kecil dan nilai *loos* tinggi maka model yang dihasilkan juga tidak bagus. Grafik *accuracy* dan *loss* lebih jelasnya dapat dilihat pada Gambar 4.5 dan Gambar 4.6.



Gambar 4.5 Grafik *acc* model *training*

Pada gambar 4.5 merupakan grafik *accuracy* yang didapatkan dari hasil *training* data sebanyak 5 *epoch* dan didapatkan hasil pada *epoch* pertama tingkat akurasi sebesar 71,2 % dan nilai validasi akurasi sebesar 94,79%. Pada *epoch* kedua terjadi kenaikan nilai akurasi dengan tingkat akurasi sebesar 97,37% dan nilai validasi akurasi sebesar 98,63%. Pada epoch ketiga terjadi kenaikan nilai akurasi dengan tingkat akurasi sebesar 98,26% dengan nilai validasi akurasi sebesar 98.38%. Pada epoch keempat terjadi kenaikan nilai akurasi dengan tingkat akurasi sebesar 98,86% dan nilai validasi akurasi sebesar 99,84%. Pada epoch kelima terjadi kenaikan nilai akurasi dengan tingkat akurasi sebesar 99,23% dan nilai validasi akurasi sebesar 99,78%.



Gambar 4.6 Grafik *loss* model *training*

Pada gambar 4.6 merupakan grafik *loss* yang didapatkan dari hasil *training* data sebanyak 5 *epoch* dan didapatkan hasil pada *epoch* pertama tingkat *loss* sebesar 128% dan nilai validasi *loss* sebesar 17,53%. Pada *epoch* kedua terjadi penurunan nilai *loss* dengan tingkat loss sebesar 9,54% dan nilai validasi *loss* sebesar 3,73%. Pada *epoch* ketiga terjadi penurunan nilai *loss* dengan tingkat *loss* sebesar 7,27% dengan nilai validasi *loss* sebesar 6,03%. Pada *epoch* keempat terjadi penurunan nilai *loss* dengan tingkat *loss* sebesar 4,6% dan nilai validasi *loss* sebesar 0,59%. Pada *epoch* kelima terjadi penurunan nilai *loss* dengan tingkat *loss* sebesar 3,57% dan nilai validasi *loss* sebesar 0,9%.

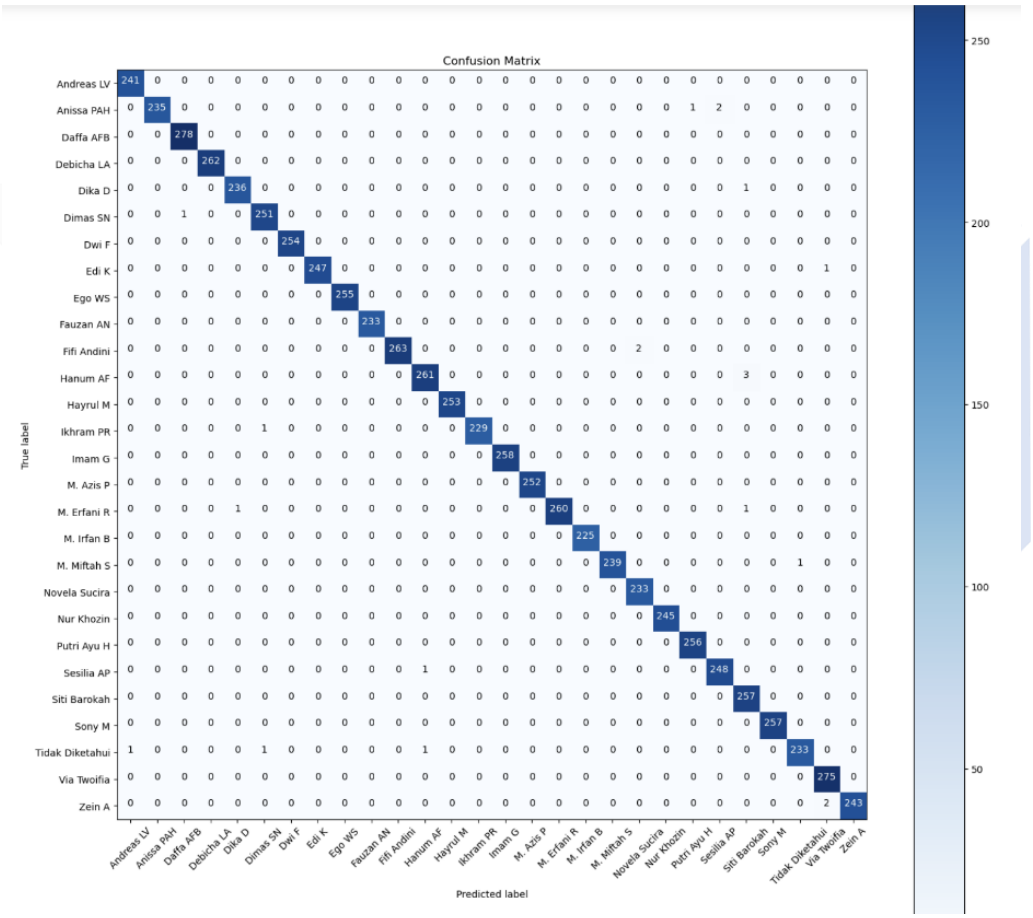
4.4. Pengujian

Setelah proses pengolahan citra dan proses *training* dilakukan maka dilanjutkan dengan proses pengujian. Pengujian dilakukan secara langsung menggunakan *webcam,* penulis mengambil *sample* mahasiswa kelas 4 Teknik Elektronika Polmanbabel untuk dijadikan dataset dan pengujian untuk dideteksi atau dikenali. Pada proses ini data *testing* digunakan sebanyak 7.000 (25%) dari total keseluruhan dataset. Proses pengujian ini penulis lakukan dengan beberapa skenario pengujian karena mempertimbangkan berbagai faktor yang memungkinkan dapat mempengaruhi dalam proses pengenalan wajah. Pengujian pertama dilakukan dengan pendeteksian berdasarkan jarak wajah dengan kamera serta intensitas cahaya. Penulis melakukan pengujian pada jarak 1 meter sampai dengan jarak yang sudah tidak bisa lagi mendeteksi dan pada intensitas pencahayaan normal saat lampu dinyalakan dan pencahayaan gelap saat lampu dimatikan. Pengujian kedua dilakukan berdasarkan jumlah wajah yang dapat dikenali dalam satu kamera. Hal ini dilakukan dengan tujuan mengukur seberapa jauh dan seberapa banyak sistem dapat mengenali wajah. Selain itu penulis juga melakukan pengujian data menggunakan *confusion matrix* untuk melihat hasil prediksi model agar dapat mengetahui seberapa besar akurasi yang diperoleh dari pengoperasian sistem ini.

4.5. Hasil Pengujian

4.5.1. Pengujian data menggunakan *confusion matrix*

Tahap pengujian ini dilakukan dengan menguji model yang sebelumnya sudah dilatih melalui proses *training* dengan menggunakan data yang tidak termasuk kedalam data *training*. Pengujian ini dilakukan untuk memprediksi kemiripan citra dengan model yang nanti akan diklasifikasikan secara langsung data tersebut masuk kedalam kelas data yang mana. Gambar 4.6 dibawah ini menampilkan *confusion matrix* dari pengujian data.



Gambar 4.7 Pengujian Data Dengan *confusion matrix*

Dari gambar diatas didapatkan hasil prediksi model dari data *testing*. Dapat dilihat pada pada jumlah data wajah dengan *True Label* “Andreas LV” sejumlah 241 data *testing* mendapatkan hasil prediksi sebesar 241 benar, dapat dilihat dari nilai 0 pada *Predicted Label* di kelas yang lain. Pada *True Label* “Anissa PAH” sejumlah 238

data *testing* dan mendapat hasil prediksi sebesar 235 benar, dapat dilihat dari nilai 1 pada *Predicted Label* “Putri Ayu H” dan 2 pada “Sesilia AP”. Begitu seterusnya sampai dengan kelas dataset “Zein A” yang nilainya dapat dilihat pada Tabel 4.1 dibawah ini.

Tabel 4.1 Hasil Prediksi

Kelas Dataset	**Data *Testing***	**Hasil Prediksi**	**Persentase Akurasi(%)**
Andreas LV	241	241	100
Anissa PAH	238	235	98,7
Daffa AFB	278	278	100
Debicha LA	262	262	100
Dika D	237	236	99,6
Dimas SN	252	251	99,6
Dwi F	254	254	100
Edi Kuswara	248	247	99,6
Ego WS	255	255	100
Fauzan AN	233	233	100
Fifi Andini	265	263	99,2
Hanum AF	264	261	98,9
Hayrul M	253	253	100
Ikhram PR	230	229	99,6
Imam G	258	258	100
M. Azis P	252	252	100
M. Erfani R	262	260	99,2
M. Irfan B	225	225	100
M. Miftah S	240	239	99,6
Novela Sucira	233	233	100
Nur Khozin	245	245	100
Putri Ayu H	256	256	100
Sesilia AP	249	248	99,6

Kelas Dataset	Data *Testing*	Hasil Prediksi	Persentase Akurasi(%)
Siti Barokah	257	257	100
Sony M	257	257	100
Tidak Diketahui	236	233	98,7
Via Twoifia	275	275	100
Zein A	245	243	99,2
Total	7000	6979	99,7

Nilai pada tabel hasil prediksi merupakan hasil yang didapatkan dari perbandingan hasil prediksi yang benar dengan jumlah total keseluruhan data *testing* [13]. Persamaan untuk menghitung akurasi dari hasil prediksi *confusion matrix* di atas adalah sebagai berikut:

$$\text{Akurasi Klasifikasi} = \frac{Total\ Hasil\ Prediksi\ Benar}{Total\ Data\ \text{Testing}} \text{ x } 100\% \qquad (1)$$

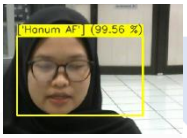
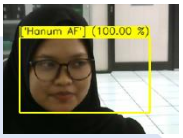
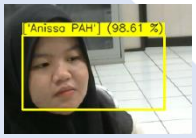
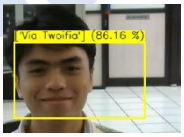
$$\text{Akurasi Klasifikasi} = \frac{6979}{7000} x\ 100\% = 99{,}7\%$$

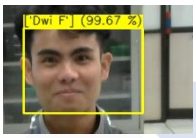
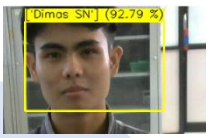
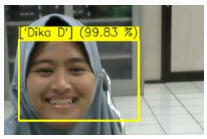
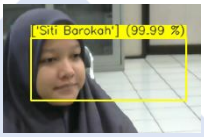
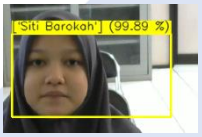
Jadi, hasil akurasi yang didapatkan dari pemodelan CNN dengan *input* dataset 224 x 224 piksel dan proses *training* dengan 5 *epoch* sebesar 99,7 %.

4.5.2. Pengujian secara *real-time* berdasarkan jarak dan intensitas cahaya

Implementasi sistem pengenalan wajah ini diuji coba secara *real-time* dengan parameter jarak dan intensitas cahaya yang memungkinkan dapat mempengaruhi sistem dalam mengenali wajah dan juga ditujukan untuk melihat seberapa optimal sistem dapat bekerja.

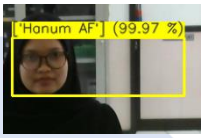
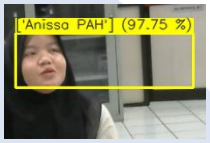
Tabel 4.2 Pengenalan wajah pada jarak 1 meter

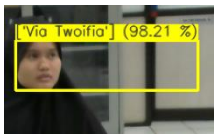
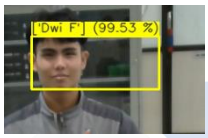
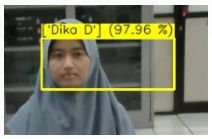
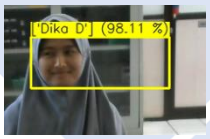
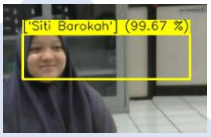
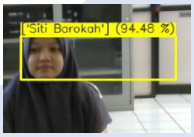
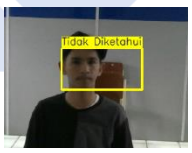
No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	**44 *lux***
1.	Fauzan A N		
			
2.	Hanum A F		
			
3.	Ikhram P R		
			
4.	Anissa P A H		
			
5.	Dimas S N		
			
6.	Sesilia A P		
			

No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	44 *lux*
7.	Via T		
			
8.	Dwi F		
			
9.	Dika D		
			
10.	Siti B		
			
11.	Tidak Diketahui		
			

Berdasarkan dari pengujian, hasil pengenalan wajah pada jarak 1 meter terdapat beberapa wajah tidak dikenali ataupun wajah dikenali sebagai orang lain. Dapat dilihat pada Tabel 4.2, saat kondisi cahaya dalam ruangan rata – rata 375 *lux* sistem dapat mengenali wajah sebanyak 9 benar dan 2 salah dalam pengklasifikasiannya. Saat cahaya dalam ruangan rata – rata 44 *lux* sistem dapat mengenali wajah sebanyak 5 benar dan 6 salah.

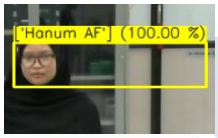
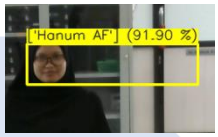
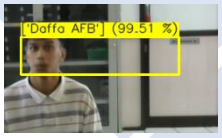
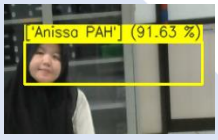
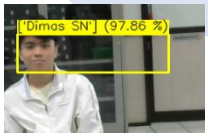
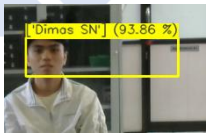
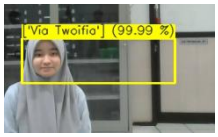
Tabel 4.3 Pengenalan wajah pada jarak 2 meter

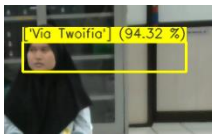
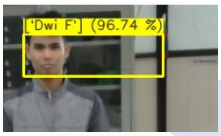
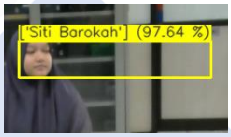
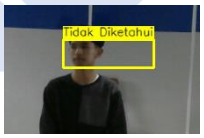
No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	**44 *lux***
1.	Fauzan A N		
			
2.	Hanum A F		
			
3.	Ikhram P R		
			
4.	Anissa P A H		
			
5.	Dimas S N		
			
6.	Sesilia A P		
			

No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	44 *lux*
7.	Via T		
			
8.	Dwi F		
			
9.	Dika D		
			
10.	Siti B		
			
11.	Tidak Diketahui		
			

Pada pengujian yang dilakukan di jarak 2 meter yang mana saat pencahayaan ruangan rata-rata 375 *lux* dengan 9 benar dan 2 salah dalam pengklasifikasiannya. Kemudian pada cahaya rata-rata ruangan 44 *lux* didapatkan hasil 6 benar dan 5 salah, dapat dilihat pada Tabel 4.3.

Tabel 4.4 Pengenalan wajah pada jarak 3 meter

No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	**44 *lux***
1.	Fauzan A N			
2.	Hanum A F			
3.	Ikhram P R			
4.	Anissa P A H			
5.	Dimas S N			
6.	Sesilia A P			

No	Dataset Wajah	Hasil Pengenalan	
		375 *lux*	44 *lux*
7.	Via T		
			
8.	Dwi F		
			
9.	Dika D		
			
10.	Siti B		
			
11.	Tidak Diketahui		
			

Pada pengujian dengan jarak 3 meter, dilihat pada Tabel 4.4 dengan kondisi pencahayaan ruangan yang sama dengan rata – rata 375 *lux* didapatkan hasil pengenalan 9 benar dan 2 salah, pada pencahayaan ruangan 44 *lux* didapatkan hasil 4 benar dan 7 salah, dapat dilihat pada Tabel 4.3.

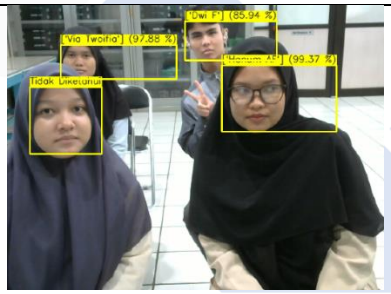
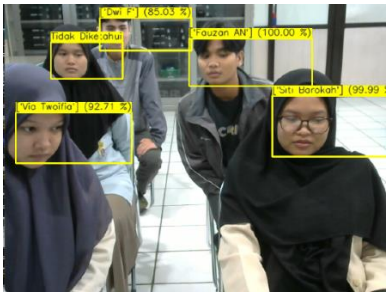
Tabel 4.5 Pengenalan wajah pada jarak lebih dari 3 meter

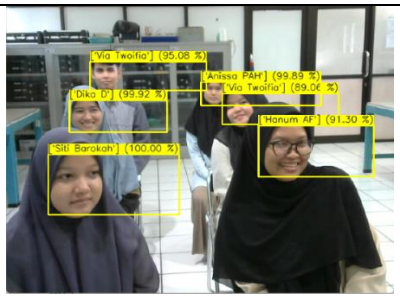
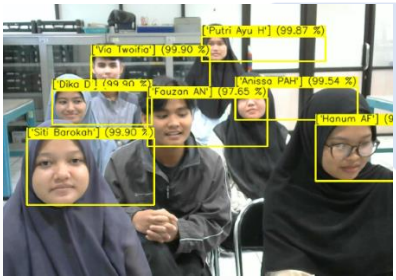
No	Jarak	Hasil Pengenalan	
		375 *lux*	44 *lux*
1.	4 Meter		

Pada jarak 4 meter sistem masih dapat mengenali wajah dengan pencahayaan ruangan yang cukup dengan rata-rata 375 *lux*, akan tetapi pada saat cahaya dalam ruangan minim dengan rata-rata 44 *lux*, sistem kesulitan untuk mengenali wajah.

4.5.3. Pengujian berdasarkan banyaknya wajah dalam satu kamera

Tabel 4.6 Pengenalan wajah dalam satu kamera

No	Jumlah Orang	Hasil Pengenalan	
		Implementasi	Jumlah Terdeteksi
1.	4		4 dengan 3 benar
2.	5		5 dengan 2 benar

No	Jumlah Orang	Hasil Pengenalan	
		Implementasi	Jumlah Terdeteksi
3.	6		6 dengan 3 benar
4.	7		7 dengan 5 benar

Penulis juga melakukan pengujian dengan *multiclass* untuk melihat seberapa banyak sistem dapat mengenali wajah dalam satu waktu. Seperti yang terdapat dalam Tabel 4.6, sistem dapat mengenali dengan lebih dari 1 wajah dalam satu waktu, hal ini tentu sangat efektif dan efisien untuk digunakan. Akan tetapi sistem mengalami penurunan saat beroperasi ditandai dengan hasil prediksi yang tidak semuanya benar saat pengujian *multiclass.*

Jadi, dapat disimpulkan bahwa sistem dapat mengenali wajah secara baik dan sesuai dengan prediksi apabila objek berada pada jarak dan cahaya yang cukup, dengan jarak optimum 4 meter dan rata-rata pencahayaan di dalam ruangan 375 *lux*. Apabila jarak terlalu jauh dan pencahayaan di dalam ruangan minim, maka semakin sulit sistem mengenali wajah dikarenakan sistem memprediksi data sesuai dengan kemiripannya.

Banyak faktor yang memperngaruhi hal tersebut terjadi. Faktor ini penulis bedakan menjadi 2, yaitu faktor internal dan eksternal. Faktor internal yang

mempengaruhinya tentu berasal dari proses pembuatan sistemnya, dimulai dari dataset yang kurang bervariasi sehingga menyebabkan proses pengklasifikasian dalam implementasi secara *real-time* kurang baik. Kemudian, faktor pembuatan atau pemilihan layer yang kurang tepat akan mempengaruhi proses training data yang mana didalam proses training inilah data dilatih untuk dijadikan model. Proses ini juga disebabkan oleh data yang *overfitting.* Hal ini disebabkan oleh model yang berusaha mempelajari detail yang ada pada gambar atau citra secara keseluruhan, akan tetapi model tidak dapat mempelajari data dominan yang ada pada gambar atau citra sehingga *overfitting* ini menjadi salah satu penyebab sistem yang dioperasikan secara *real-time* berjalan kurang optimal. Penggunaan layer yang juga menjadi menjadi salah satu faktor *overfitting* ini terjadi dan proses *training* yang lama, sehingga harus selektif dalam memilih dan membangun *layer* CNN.

Faktor eksternal yang juga mempengaruhi proses pengenalan ini yaitu jarak peletakan kamera dengan wajah yang akan dikenali. Sistem dapat mengenali wajah pada jarak – jarak tertentu sesuai dengan kondisi ruangan, tidak terlalu dekat dan tidak pula terlalu jauh. Seperti yang terdapat pada tabel diatas sistem dapat mengenali wajah pada jarak 1 meter sampai dengan 4 meter. Akan tetapi pada jarak lebih dari 3 meter sistem tidak dapat bekerja secara optimal dikarenakan jarak objek yang terlalu jauh membuat kamera tidak dapat mengenali objek secara jelas. Hal ini selaras dengan faktor dari kualitas perangkat yang digunakan. Spesifikasi *webcam* yang digunakan tidak dapat menjangkau objek dengan jarak yang terlalu jauh secara jelas. Kemudian faktor pencahayaan di dalam ruangan sangat berpengaruh dalam proses pengenalan, sistem dapat mengenali wajah dengan baik saat ruangan mendapatkan pencahayaan yang cukup. Namun pada saat ruangan tidak mendapatkan pencahayaan yang cukup, maka ini akan menyulitkan sistem dalam mengenali objek. Faktor terakhir yang penulis simpulkan dari pengujian ini adalah kualitas atupun spesifikasi PC ataupun laptop yang digunakan. PC yang digunakan haruslah memiliki spesifikasi yang tinggi agar membantu mempercepat proses *training* pada model CNN serta proses untuk mengaktifkan kamera khususnya *webcam.*

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisa yang telah dilakukan dalam penelitian ini, maka kesimpulannya sebagai berikut :

1. Pemilihan dan pengumpulan dataset akan berpengaruh pada saat proses *training* model CNN. Pemilihan kualitas dan kuantitas pada dataset akan membuat proses *training* menjadi lebih seimbang. Apabila kualitas pada dataset bagus dan ukuran piksel besar maka proses *testing* akan lebih akurat dalam mengenali wajah. Sebaliknya, apabila kualitas dataset kurang bagus dan ukuran piksel kecil maka hasil pengenalannya akan kurang akurat.
2. Sistem dapat mengenali wajah dan *bounding box* dapat mengikuti arah gerak wajah dari mahasiswa.
3. Sistem pengenalan wajah ini dapat berjalan secara *real-time* mengikuti arah gerak wajah namun belum bekerja secara optimal.
4. Sistem dapat mengenali lebih dari 1 wajah dalam satu waktu sehingga dapat membuat sistem bekerja lebih efektif dan efisien.
5. Banyak faktor yang mempengaruhi sistem bekerja kurang optimal, seperti faktor jarak yang terlalu dekat dan jauh, pencahayaan yang minim, proses pembuatan layer dan kualitas perangkat yang digunakan.
6. Hasil nilai akurasi yang diperoleh dari pemodelan CNN yang telah diprediksi oleh *confusion matrix* sebesar 99,7%.
7. Akurasi yang dihasilkan bergantung kepada kondisi data *input*-an, pendeteksian wajah dan proses pengklasifikasian.
8. Semakin besar nilai *accuracy* dan semakin kecil nilai *loss* maka semakin bagus model yang dihasilkan. Begitu pula sebaliknya, apabila nilai *accuracy* kecil dan nilai *loos* tinggi maka model yang dihasilkan juga kurang baik.

5.2 Saran

Dari hasil proyek akhir ini ada beberapa saran yang ingin penulis sampaikan untuk pengembangan yang lebih lanjut, sebagai berikut:

1. Menambahkan *input* dataset yang lebih bervariasi agar dalam pengimplementasian secara *real-time* sistem dapat bekerja dengan baik.
2. Untuk kedepannya dibuatkan *user interface* sebagai media untuk mempermudah para pengguna dalam mengenali wajah dan menambah nilai estetika.
3. Menggunakan perangkat dengan kualitas dan spesifikasi yang tinggi.

DAFTAR PUSTAKA

[1] I. Q. Mundial, M. S. Ul Hassan, M. I. Tiwana, W. S. Qureshi, dan E. Alanazi, "Towards facial recognition problem in COVID-19 pandemic," dalam *2020 4th International Conference on Electrical, Telecommunication and Computer Engineering, ELTICOM 2020 - Proceedings*, Sep 2020, hlm. 210–214. doi: 10.1109/ELTICOM50775.2020.9230504.

[2] D. Zeng, R. Veldhuis, dan L. Spreeuwers, "A survey of face recognition techniques under occlusion," *IET Biometrics*, vol. 10, no. 6. John Wiley and Sons Inc, hlm. 581–606, 1 November 2021. doi: 10.1049/bme2.12029.

[3] R. Purwati dan G. Ariyanto, "Pengenalan Wajah Manusia Berbasis Algoritma Local Binary Pattern," *Jurnal Teknik Elektro,* vol. 2, no. 17, pp. 70-79, 2017.

[4] V. D. Win, "Pengenalan Wajah Menggunakan Convolutional Neural Network," Laporan Proyek Akhir, *Institusi Teknologi Sepuluh November*, Surabaya, 2018.

[5] H. Abhirawa, Jondri dan A. Arifianto, "Pengenalan Wajah Menggunakan Convolutional Neural Network," *e-Proceeding of engineering,* vol. 4, no. 3, p. 4907, 2017.

[6] S. Al-Aidid dan D. Pamungkas, "Sistem Pengenalan Wajah dengan Algoritma Haar Cascade dan Local Binary Pattern Histogram," *Jurnal Rekayasa Elektrika*, vol. 14, no. 1, hlm. 62–67, Apr 2018, doi: 10.17529/jre.v14i1.9799.

[7] M. Zufar, "Convolutional Neural Networks Untuk Pengenalan Wajah Secara Real-Time," Laporan Proyek Akhir, *Institut Teknologi Sepuluh Nopember*, Surabaya, 2016.

[8] I. W. S. E. Putra, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) Pada Caltech 101," dalam *Institut Teknologi Sepuluh Nopember*, Surabaya, 2016.

[9] A. Santoso dan G. Ariyanto, “Implementasi Deep Learning Berbasis Keras untuk Pengenalan Wajah,” *Jurnal Teknik Elektro,* vol. 18, no. 01, pp. 15-21, 2018.

[10] F. N. Cahya, “Perbandingan Identifikasi Wajah Dengan Ekstraksi Fitur Haralick Dan CNN,” *Informatics and Digital Expert (INDEX),* vol. 2, no. 2, pp. 60-66, 2020.

[11] D. J. Mananjang, S. R. Sompie dan A. Jacobus, “Implementasi Framework Tensorflow Object Detection Dalam Mengklasifikasi Jenis Kendaraan Bermotor,” *Jurnal teknik Informatika,* vol. 15, no. 3, pp. 171-178, 2021.

[12] Efyanntyo, “Perancangan Aplikasi Sistem Pengenalan Wajah Dengan Metode Convolutional Neural Network (CNN) Untuk Pencatatan Kehadiran Karyawan,” *Jurnal Instrumentasi dan Teknologi Informatika,* vol. 3, no. 1, pp. 1-11, 2021.

[13] W. Anggraini, “Deep Learning Untuk Deteksi Wajah Yang Berhijab Menggunakan Algoritma Convolutional Neural Network (CNN) Dengan Tensorflow,” Laporan Proyek Akhir, *Universitas Islam Negeri Ar-Raniry Darussalam Banda Aceh*, Banda Aceh, 2020.

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama Lengkap	: Fauzan Akbar Nugraha
Tempat & Tanggal Lahir	: Pangkalpinang, 26 September 2001
Alamat Rumah	: Jl. Karimata 2 No.274 RT.006/002, Karya Makmur, Pemali, Bangka
Telp.	: 085972718651
Email	: fauzangraha6@gmail.com
Jenis Kelamin	: Laki – laki
Agama	: Islam

2. Riwayat Pendidikan

SD NEGERI 28 SUNGAILIAT

SMP NEGERI 1 SUNGAILIAT

SMK NEGERI 2 PANGKALPINANG

Sungailiat, 07 Februari 2023

Fauzan Akbar Nugraha

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama Lengkap	: Hanum Ayu Fazira
Tempat & Tanggal Lahir	: Sungailiat, 07 Juli 2001
Alamat Rumah	: Jl. Tarumanegara Gg. Bukit Sekip No.178 RT.003/003, Karya Makmur, Pemali, Bangka
Telp.	: 0895604472264
Email	: hanumayu7@gmail.com
Jenis Kelamin	: Perempuan
Agama	: Islam

2. Riwayat Pendidikan

SD NEGERI 1 AIR RUAY
SMP NEGERI 1 SUNGAILIAT
SMA NEGERI 1 SUNGAILIAT

Sungailiat, 07 Februari 2023

Hanum Ayu Fazira

```
#Import Library

import os
import cv2
import itertools
import numpy as np
import matplotlib.pyplot as plt

from PIL import Image
from skimage import io
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from datetime import datetime
from  sklearn.metrics  import  accuracy_score,  confusion_matrix,
classification_report
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential,Model
from                tensorflow.keras.layers                import
ZeroPadding2D,Convolution2D,MaxPooling2D
from                tensorflow.keras.layers                import
Dense,Dropout,Softmax,Flatten,Activation,BatchNormalization
import tensorflow.keras.backend as K
from tensorflow.keras.utils import to_categorical
from datetime import datetime

import keras
from keras.models import Sequential, Model
from keras.layers import Dense, Activation, Input, Dropout

from keras.utils import to_categorical
from keras.layers import Conv2D, MaxPool2D, Flatten, ZeroPadding2D
-----------------------------------------------------
#Mendefinisikan dan Upload Dataset Foto Wajah

def show_dataset(images_class, label):
    #show data for 1 class
    plt.figure(figsize=(14,5))
    k = 0
    for i in range(1,6):
        plt.subplot(1,5,i)
        try :
            plt.imshow(images_class[k][:,:,::-1])
        except :
            plt.imshow(images_class[k], cmap='gray')
        plt.title(label)
        plt.axis('off')
        plt.tight_layout()
        k += 1
    plt.show()
```

```
#Ekstraksi Dataset Menggunakan Haarcascade (Face Detection)
```

```
face_cascade                                                    =
cv2.CascadeClassifier('haarcascades/haarcascade_frontalface_defaul
t.xml')

def detect_face(img):
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(img, 1.1, 5)
    try :
        x, y, w, h = faces[0]

        img = img[y:y+h, x:x+w]
        img = cv2.resize(img, (224, 224))
    except :
        print("Face not found in image index", i)
        img = None
    return img
-----------------------------------------------------
#Upload Dataset Foto Wajah Mahasiswa

def print_progress(val, val_len, folder, bar_size=20):
    progr = "#"*round((val)*bar_size/val_len) + " "*round((val_len
- (val))*bar_size/val_len)
    if val == 0:
        print("", end = "\n")
    else:
        print("[%s] (%d samples)\t label : %s \t\t" % (progr, val+1,
folder), end="\r")

dataset_folder = "27 Dataset 80/"

names = []
images = []
for folder in os.listdir(dataset_folder):
    files = os.listdir(os.path.join(dataset_folder, folder))[:150]
    if len(files) < 50 :
        continue
    for i, name in enumerate(files):
        if name.find(".jpg") > -1 :
            img = cv2.imread(os.path.join(dataset_folder + folder,
name))
            img = detect_face(img)
            if img is not None :
                images.append(img)
                names.append(folder)

                print_progress(i, len(files), folder)
-----------------------------------------------------
labels = np.unique(names)

labels

for label in labels:

    ids = np.where(label== np.array(names))[0]
    image_class = images[ids[0] : ids[-1] + 1]
    show_dataset(image_class, label)
```

```
print("Number of Samples :", len(names))
```

```
#Augmentasi Dataset
def img_augmentation(img):
    h, w = img.shape
    center = (w // 2, h // 2)
    M_rot_5 = cv2.getRotationMatrix2D(center, 5, 1.0)
    M_rot_neg_5 = cv2.getRotationMatrix2D(center, -5, 1.0)
    M_rot_10 = cv2.getRotationMatrix2D(center, 10, 1.0)
    M_rot_neg_10 = cv2.getRotationMatrix2D(center, -10, 1.0)
    M_trans_3 = np.float32([[1, 0, 3],[0, 1, 0]])
    M_trans_neg_3 = np.float32([[1, 0, 3],[0, 1, 0]])
    M_trans_6 = np.float32([[1, 0, 6],[0, 1, 0]])
    M_trans_neg_6 = np.float32([[1, 0, -6],[0, 1, 0]])
    M_trans_y3 = np.float32([[1, 0, 0],[0, 1, 3]])
    M_trans_neg_y3 = np.float32([[1, 0, 0],[0, 1, 3]])
    M_trans_y6 = np.float32([[1, 0, 0],[0, 1, 6]])
    M_trans_neg_y6 = np.float32([[1, 0, 0],[0, 1, -6]])

    imgs = []
    imgs.append(cv2.warpAffine(img,      M_rot_5,      (w,      h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_rot_neg_5,     (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,     M_rot_10,      (w,     h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_rot_neg_10,     (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,     M_trans_3,     (w,     h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_trans_neg_3,    (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,     M_trans_6,     (w,     h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_trans_neg_6,    (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,     M_trans_y3,     (w,     h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_trans_neg_y3,    (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,     M_trans_y6,     (w,     h),
borderValue=(255,255,255)))
    imgs.append(cv2.warpAffine(img,    M_trans_neg_y6,    (w,    h),
borderValue=(255,255,255)))
    imgs.append(cv2.add(img, 10))
    imgs.append(cv2.add(img, 30))
    imgs.append(cv2.add(img, -10))
    imgs.append(cv2.add(img, -30))
    imgs.append(cv2.add(img, 15))
    imgs.append(cv2.add(img, 45))
    imgs.append(cv2.add(img, -15))
    imgs.append(cv2.add(img, -45))

    return imgs
```

```
plt.imshow(images[1], cmap = "gray")
img.shape

img_test = images[0]

augmented_image_test = img_augmentation(img_test)

plt.figure(figsize=(15,10))
for i, img in enumerate(augmented_image_test):
    plt.subplot(4,5,i+1)
    plt.imshow(img, cmap="gray")
plt.show()

augmented_images = []
augmented_names = []
for i, img in enumerate(images):
    try :
        augmented_images.extend(img_augmentation(img))
        augmented_names.extend([names[i]] * 20)
    except :
        print(i)

len(augmented_images), len(augmented_names)

images.extend(augmented_images)
names.extend(augmented_names)

len(images), len(names)
-----------------------------------------------------
#Persentase Jumlah Masing - Masing Dataset Foto Wajah Mahasiswa

unique, counts = np.unique(names, return_counts = True)

for item in zip(unique, counts):
    print(item)

def print_data(label_distr, label_name):
    plt.figure(figsize=(12, 6))

    my_circle = plt.Circle( (0,0), 0.7, color = "white")
    plt.pie(label_distr, labels=label_name, autopct='%1.1f%%')
    plt.gcf().gca().add_artist(my_circle)
    plt.show()

unique = np.unique(names)
label_distr = {i:names.count(i) for i in names}.values()
print_data(label_distr, unique)
-----------------------------------------------------
#Balancing Dataset Foto Wajah Mahasiswa

n = 1000

def randc(labels, l):
    return np.random.choice(np.where(np.array(labels) == l)[0], n,
replace=True)
```

```
mask = np.hstack([randc(names, l) for l in np.unique(names)])

names = [names[m] for m in mask]
images = [images[m] for m in mask]

label_distr = {i:names.count(i) for i in names}.values()
print_data(label_distr, unique)

len(names), len(images)
-----------------------------------------------------
#Encoding Label Menjadi Bentuk Vector

le = LabelEncoder()

le.fit(names)

labels = le.classes_

name_vec = le.transform(names)

categorical_name_vec = to_categorical(name_vec)

print("Number of Class :", len(labels))
print(labels)

print(name_vec)

print(categorical_name_vec)
-----------------------------------------------------
#Train Test Split Dataset

x_train,          x_test,          y_train,          y_test         =
train_test_split(np.array(images, dtype=np.float32),

np.array(categorical_name_vec),
                                                    test_size=0.25,

random_state=42)

print(x_train.shape, y_train.shape, x_test.shape, y_test.shape)

x_train.shape
```

```
#Reshape Dataset Foto Wajah Mahasiswa

x_train   =   x_train.reshape(x_train.shape[0],   x_train.shape[1],
x_train.shape[2], 1)
x_test    =    x_test.reshape(x_test.shape[0],    x_test.shape[1],
x_test.shape[2], 1)

x_train.shape, x_test.shape
-----------------------------------------------------
#Modelling Convolutional Neural Network

def cnn_model(input_shape):
    model = Sequential()
```

```
model.add(Convolution2D(64,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu',
                        input_shape = input_shape))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(64,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu',
                        input_shape = input_shape))
model.add(MaxPooling2D((2, 2), strides = (2, 2)))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(128,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(128,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(MaxPooling2D((2, 2), strides = (2, 2)))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(256,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(256,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(MaxPooling2D((2, 2), strides = (2, 2)))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(512,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(512,
                        (3, 3),
                        padding = 'valid',
                        activation = 'relu'))
model.add(MaxPooling2D((2, 2), strides = (2, 2)))
model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(512,
```

```
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Convolution2D(512,
                            (3, 3),
                            padding = 'valid',
                            activation = 'relu'))
    model.add(MaxPooling2D((2, 2), strides = (2, 2)))
    model.add(ZeroPadding2D((1, 1)))

    model.add(Flatten())
    model.add(Dense(512, activation="relu"))
    model.add(Dense(256, activation="relu"))
    model.add(Dense(128, activation="relu"))
    model.add(Dense(64, activation="relu"))
    model.add(Dense(len(labels)))
    model.add(Activation('softmax'))

    model.summary()

    model.compile(optimizer='adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

    return model
-----------------------------------------------------------
#Training Model CNN

input_shape = x_train[0].shape

EPOCHS = 5
BATCH_SIZE = 32

model = cnn_model(input_shape)

history = model.fit(x_train,
                    y_train,
                    epochs=EPOCHS,
                    batch_size=BATCH_SIZE,
                    shuffle=True,
                    validation_split=0.15
                    )
-----------------------------------------------------------
#Grafik Hasil Training Validasi dan Loss Model CNN

def evaluate_model_(history):
    names = [['accuracy', 'val_accuracy'],
             ['loss', 'val_loss']]
    for name in names :
        fig1, ax_accuracy = plt.subplots()
        plt.plot(history.history[name[0]])
        plt.plot(history.history[name[1]])
        plt.xlabel('Epoch')
        plt.ylabel(name[0])
```

```
        plt.title('Model - ' + name[0])
        plt.legend(['Training', 'Validation'], loc='lower right')
        plt.grid()
        plt.show()

evaluate_model_(history)
-----------------------------------------------------
#Save Model CNN

model.save("model-cnnproyekakhir.h5")
-----------------------------------------------------
#Predict Test Data

y_pred=model.predict(x_test)
-----------------------------------------------------
#Confusion Matrix dan Classification Report

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

    plt.figure(figsize=(15, 15))

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()

    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for    i,    j    in    itertools.product(range(cm.shape[0]),
range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.show()

#Compute confusion matrix

cnf_matrix        =        confusion_matrix(y_test.argmax(axis=1),
y_pred.argmax(axis=1))
np.set_printoptions(precision=2)

#Plot non-normalized confusion matrix

plot_confusion_matrix(cnf_matrix, classes=labels,normalize=False,
```

```
                      title='Confusion Matrix')


print(classification_report(y_test.argmax(axis=1),
                            y_pred.argmax(axis=1),
                            target_names=labels))
-------------------------------------------------------
#Import Load Model dari Library Keras

from keras.models import load_model
-------------------------------------------------------
#Membuat Bounding Box dan Font

def  draw_ped(img,  label,  x0,  y0,  xt,  yt,  color=(255,127,0),
text_color=(255,255,255)):

    (w,      h),      baseline      =      cv2.getTextSize(label,
cv2.FONT_HERSHEY_SIMPLEX, 0.5, 1)
    cv2.rectangle(img,
                  (x0, y0 + baseline),
                  (max(xt, x0 + w), yt),
                  color,
                  2)
    cv2.rectangle(img,
                  (x0, y0 - h),
                  (x0 + w, y0 + baseline),
                  color,
                  -1)
    cv2.putText(img,
                label,
                (x0, y0),
                cv2.FONT_HERSHEY_SIMPLEX,
                0.5,
                text_color,
                1,
                cv2.LINE_AA)
    return img
-------------------------------------------------------
#Runnning Program Deteksi Wajah Mahasiswa
# --------- load Haar Cascade model -------------
face_cascade                                                    =
cv2.CascadeClassifier('haarcascades/haarcascade_frontalface_defaul
t.xml')

# --------- load Keras CNN model -------------
model = load_model("model-cnnproyekakhir.h5")
print("[INFO] finish load model...")

cap = cv2.VideoCapture(0)
while cap.isOpened() :
    ret, frame = cap.read()
    if ret:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.1, 5)
        for (x, y, w, h) in faces:
```

```
            face_img = gray[y:y+h, x:x+w]
            face_img = cv2.resize(face_img, (224, 224))
            face_img = face_img.reshape(1, 224, 224, 1)

            result = model.predict(face_img)
            idx = result.argmax(axis=1)
            confidence = result.max(axis=1)*100
            if confidence > 80 :
                label_text = "%s (%.2f %%)" % (labels[idx],
confidence)
            else :
                label_text = "Tidak Diketahui"
            frame = draw_ped(frame, label_text, x, y, x + w, y + h,
color=(0,255,255), text_color=(50,50,50))
        cv2.imshow('Face Recognition', frame)
    else :
        break
    if cv2.waitKey(60) == ord('q'):
        break
cv2.destroyAllWindows()
cap.release()
```