IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* UNTUK *FACIAL RECOGNITION*

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat Kelulusan
Program Pendidikan Sarjana

Oleh :
Andikha Dwi Putra
2017130016

JURUSAN INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER – LIKMI
BANDUNG
2020

IMPLEMENTASI *CONVOLUTIONAL NEURAL NETWORK* UNTUK *FACIAL RECOGNITION*

Oleh :
Andikha Dwi Putra
2017130016

Bandung, 27 Agustus 2020
Menyetujui,

Dhanny Setiawan, S.T., M.T.	Dhanny Setiawan, S.T., M.T.
Pembimbing	Ketua Jurusan

JURUSAN INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER – LIKMI
BANDUNG
2020

ABSTRAK

Face recognition merupakan salah satu teknik *biometric*. Teknik yang dapat disebut juga pengenalan wajah ini telah menjadi topik yang cukup diminati untuk diteliti. Pada tugas akhir ini dilakukan proses pengenalan wajah dengan menggunakan metode *CNN* (*Convolutional Neural Network*). Penelitian ini memiliki tujuan untuk mengimplementasikan metode *CNN* ke dalam pengenalan wajah dengan menggunakan *library Tensorflow*. Metode ini digunakan karena proses pembelajaran dilakukan dengan mendalam (*deep learning*). Metode *CNN* yang digunakan memiliki beberapa lapisan pada proses *training* yang dilakukan, yaitu lapisan *Conv2D, MaxPooling2d, Flatten,* dan *Dense*. *Face recognition* yang dihasilkan terdapat pendeteksi wajah menggunakan *Haarcascade* dengan bantuan *library Opencv* di dalamnya. Jumlah data set juga diketahui dapat mempengaruhi hasil pengenalan dan proses pengenalan wajah dengan *CNN* juga memerlukan dataset yang besar. Adapun jumlah citra wajah yang digunakan dalam penelitian ini sebanyak 90.000 gambar wajah yang berasal dari 36 himpunan gambar dan menghasilkan tingkat akurasi sebesar 97%.

KATA PENGANTAR

Segala puji syukur penulis panjatkan pada Tuhan yang Maha Esa, dengan penyertaan-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **"Implementasi *Convolutional Neural Network* Untuk *Facial Recognition*".** Sebagai salah satu syarat untuk menyelesaikan program studi S1 Jurusan Teknik Informatika STMIK LIKMI. Penulis jugan mengucapkan terima kasih kepada:

1. Bapak Dhanny Setiawan, S.T., M.T. sebagai dosen wali dan dosen pembimbing.
2. Orang tua penulis atas dukungan yang diberikan kepada penulis.
3. Rekan-rekan penulis atas bantuan dan kerjasamanya selama penulisan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini tidak luput dari kekurangan. Oleh sebab itu, penulis mengharapkan kritik dan saran yang membangun sehingga Tugas Akhir ini menjadi lebih baik dan bermanfaat.

Bandung, 27 Agustus 2020

Penulis

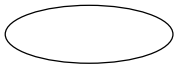
DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SIMBOL

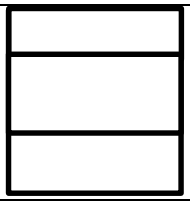
Use Case diagram

Nama Simbol	Simbol	Keterangan
Use Case		Sebuah *behavior* yang dilakukan sistem perangkat lunak
Aktor		Entitas atau roll diluar sistem yang berhubungan dengan sistem
Asosiasi		Hubungan antata aktor dan *use case*
Extend		Hubungan antara case dengan case yang bersifat pilihan
include		Hubungan antara case dengan case yang bersifat wajib melewati case

Activity diagram

Nama Simbol	Simbol	Keterangan
Kondisi awal		Menunjukan kondisi awal aktivitas sebuah sistem
Kondisi akhir		Menunjukan kondisi akhir aktivitas berakhirnya sistem
Action state		Kegiatan sistem lakukan
Decision		Percabangan untuk memilih cabang keputusan
garis		Menujukan arah action stateselanjutnya

Class diagram

Nama Simbol	Simbol	Keterangan
Class		Kelas (Class) memilki tiga bagian yang di atas nama kelas, kedua atribut kelas, dan proses di dalam kelas
asosiasi		Garis penghubung atar kelas

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sistem untuk pengenalan identitas perlu mengalami peningkatan tingkat akurasi yang lebih baik untuk menangani kejahatan dan pemalsuan identitas. Sistem yang telah terlebih dahulu digunakan seperti *token-based system* dan *knowledge-based. Token-based* / berbasis token memiliki tingkat resiko yang tinggi seperti pencurian identitas. *Knowledge-based* / berbasis ingatan juga memilki tingkat resiko yang tinggi seperti tidak ingat terhadap kata sandi. Sistem *token-based system* dan *knowledge-based* masih memiliki kekurangan muncul sistem *biometric*, tidak bisa dipindahkan karena unik dan spesifik. *Biometric* memiliki dua sistem perilku dan fisiologis. Berdasarkan perilaku memilki contoh suara. Untuk fisiologis seperti tanda tangan, sidik jari dan wajah. Wajah sampai sekarang masih menarik untuk diteliti.

Face recognition memiliki masalah seperti berbagai macam posisi gambar wajah mengakibatkan mata dan hidung tidak terlihat secara keseluruhan. Permasalahan tambahan dengan penambahan aksesoris / tambahan seperti kacamata, jenggot pada gambar wajah. Faktor yang lain dengan tingkat pencahayaan. Ukuran gambar dan ekspresi wajah. Adanya iluminasi perubahan dari persebaran cahaya karena sifat memantulkan cahaya di beberapa bagian pada wajah yang dapat mempengaruhi.

Sistem *face recognition* diperlukan untuk membantu mengenali seseorang. Pengenalan menggunakan algoritme *CNN (Convolutional Neural Network)* yang merupakan pengembangan dari *MLP Multi Layer Perceptron. MLP* salah satu model *Articial Neural Network* yang diperkenalkan pada tahun 1969 oleh S Papert dan M.Minsky. (Graupe, 2019: 24) *MLP* memiliki permasalahan yaitu menimbulkan parameter yang secara berlebihan.

CNN dapat mengatasi yang dihadapi *MLP* karena *Convolutional neural Network* dapat mengurangi parameter bebas dan dapat menangani deformasi gambar input.

Berdasarkan latar belakang yang telah disampaikan, maka tugas akhir ini dengan judul **"*Implementasi Convolutional Neural Network Untuk Facial Recognition*".**

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dibahas. Hal yang menjadi masalah untuk diselesaikan adalah sebagai berikut:

Bagaimana pengimplemtasian algoritme *Convolutional Neural Network* pada *Face Recognition*?

1.3 Tujuan

Berdasarkan rumusan masalah yang ada, maka tujuan dari penelitian ini adalah mengenali wajah seseorang dengan menggunakan algoritme CNN.

1.4 Batasan Masalah

Adapun batasan masalah dari tugas akhir sebagai berikut:

1. Perancangan sistem dalam penulisan ini menggunakan Bahasa pemrograman *Python* dikarenakan meiliki kelbeihan yaitu mudah dibaca dan dipahami (*readibility*) dan memiliki *library* yang banyak sehingga akan menjadi lebih simple (efesien). (Wahyono, 2018: 29)
2. Untuk deteksi wajah menggunakan dengan *library OpenCV* dan untuk pengenalan wajah dengan menggunakan *library Tensorflow*.
3. Pengenalan wajah tidak bisa menggunakan kartu pengenal seperti Kartu Tanda Mahasiswa dan dari gambar wajah di *handphone*.

1.5 Kegunaan Hasil

Kegunaan hasil penelitian adalah untuk mewujudkan sistem *face recognition*. *Face recognition* dapat dikembangkan untuk keamanan dan autentikasi.

1.6 Sistematika Penulisan

Untuk menyusun tugas akhir ini dibutuhkan sistem penyajian sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, tujuan, batasan masalah, kegunaan hasil, dan sistematika penulisan tugas akhir.

BAB II LANDASAN TEORI

Bab ini akan membahas rangkuman atas teori-teori terdapat penelitian terdahulu, *machine learning, artificial neural network, biometric, deep learning,* diagram *UML,* bahasa pemrograman *Python*, *library Tensodflow dan openCV* yang digunakan untuk mendasari penyelesaian tugas akhir.

BAB III ANALISIS DAN RANCANGAN SISTEM

Bab ini menjelaskan tentang analisa untuk penerapan algoritme pada *Face Recognition.*

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Pada bab ini, akan menjelaskan bentuk implementasi, pengujian dan analisa terhadap kasus yang telah ditentukan. Laporan dari pengujian tiap skenario yang dibuat dan hasil dari setiap uji coba.

BAB V KESIMPULAN

Bab ini berisi simpulan dan saran untuk penelitian ini yang dapat digunakan untuk perkembangan pada penelitian selanjutnya.

BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Face recognition merupakan bagian dari pendekatan *biometric* sebelumnya ada dua pendekatan tradisional. Pedekatan tradisional untuk pengenalan orang yaitu *token-based* (berbasis token) dan *knowledge-based* (berbasis ingatan). *Knowledge-based* untuk mengecek pengenalan terhadap orang dengan memasukan *password* dan *pin*. *Token-based* dengan menggunakan token seperti *QR Code, Cips set* pada kartu pengenal*,* dan *Barcode*.

Pada penelitian terdahulu dalam jurnal *Integrasi Login Tanpa Mengetik Password pada WordPress*, *token-based* dengan *QR Code* memiliki kelebihan dari *knowledge based* sebagai berikut:

"*sistem integrasi login menggunakan QR Code juga dapat mengurangi risiko man-in-the-middle attack dan juga key logger pada saat pengaksesan akun menggunakan perangkat publik.*" (Arifin, Bejo, & Najib, 2017:166)

Pada jurnal *Implementasi QRCode Untuk Absensi Perkuliahan Mahasiswa Berbasis Paperless Office*, memiliki kesimpulan bahwa:

"*Penerapan QR (Quick Response) Code untuk absensi mahasiswa berbasis mobile dapat meningkat efisiensi dan efektivitas proses absensi*" (Labolo, 2019:102)

Penerapan *token-based* masih mempunyai kendala yaitu pemalsuan identitas atau dapat diwakilkan. Beberapa kekurangan dari *knowledge-based* dan *Token-based* dapat diselesaikan dengan menggunakan *biometric. Biometric* tidak dapat terwakilkan seperti sidik jari dan wajah.

Algoritme *Convolutional Neural Network* (*CNN*) diuji menggunakan beberapa arsitektur yaitu *VGGNet, ResNet, GoogleNet,* dan *AlexNet*. Pada jurnal *Facial Expression Recognition Using Convolutional Neural Network* dengan melakukan penyederhanaan terhadap arsitektur tersebut dan dibandingkan hasilnya, maka diperoleh hasil arsitektur *AlexNet* memiliki tingkat akurasi yang tinggi, sebagai berikut:

*"We get 0.6424 by using AlexNet (epochs = 20 and lrate = 0.001) as our best result which is close to the best results of FER2013. The best accuracy is around 65% of all the previous results of FER2013."(*Gan, 2018:4)

Sebuah jurnal lain juga melakukan penelitian yang diaplikasikan untuk *face recognition* dengan menggunakan *CNN* dengan *Local Binary Pattern* (LBC) layer. Pada jurnal *A comprehensive analysis of Local Binary Convolutional Neural Network for fast face recognition in surveillance video*, metode tersebut disebut LB*CNN*, yang mampu menghasilkan sistem dengan kompleksitas model yang lebih rendah dan tidak mudah terkena *over-fiting*. Hasil penelitian tersebut sebagai berikut:

"*We can conclude that the LBCNN is very sensitive to noise and we can get better results when the noisy images are inserted in the training set. LBCNN had a good performance in terms of the processing time and accuracy.*"(Feraz, 2018:268)

Perbandingan *CNN* dengan metode lain, yaitu *SRC*, dilakukan untuk mengatasi kekurangan algoritme *CNN* yang membutuhkan *database* lebih besar dan memerlukan biaya yang tinggi. Metode yang dilakukan pada jurnal *Face recognition: Sparse Representation vs. Deep Learning*, menggunakan Algoritme *SRC* yang dibandingkan dengan *CNN*. Hasil menunjukan bahwa alogritme *SRC* merupakan metode yang kuat dan mengungguli *CNN*, tetapi perlu dikembangkan dengan kombinasi *CNN* karena ada beberapa hal yang belum dapat diatasi oleh metode berbasis *SRC* yang digunakan, sebagai berikut:

"*We will further improve face recognition by combining the locality of CNNs and linearity of SRC in large face variations. Investigating errors of face recognition in pose, illumination and facial expression is still an open problem*."(Alskeini, 2018:36)

2.2 *Machine Learning*

Pengembangan *machine learning* dapat dikelompokkan menjadi 3 masa, yaitu masa pertama tahun sebelum 1980 di mana pada masa ini pembelajaran *machine learning* untuk menghasilkan *linear decision surfaces.* Masa kedua 1980 sampai 1989, pada masa ini

Decision trees dan *ANN* menjadi pelopor pembelajaran *non-linier*. Masa ketiga 1990 sampai sekarang, pembelajaran dikembangkan secara *non-linier* yang lebih efektif.

Machine learning merupakan bagian dari *artificial intelligence* yang memiliki fokus pada kemampuan belajar sendiri.

"Machine learning is all about determining patterns - analyzing training data in such a manner that the trained algorithm can perform tasks that the developer didn't originally program it to do." (Mueller & Massaron, 2016:25)

Machine learning dalam buku *Learn Computer Vision Using OpenCV* sebagai berikut:

"Machine learning is a way of building intelligence into a machine so it will be able to learn over time and do better using its own experience." (Gollapudi, 2019:11)

Dari kutipan tersebut, maka dapat digambarkan bahwa *machine Learning* adalah cara membangun kecerdasan sehingga mesin mampu belajar dengan pengelaman sendiri.

Machine learning berfokus pada pengaplikasian yang praktis. Bertugas mengajarkan untuk mempelajari pemasalahan untuk dipecahkan pada buku yang berjudul *Machine Learning with R* sebagai berikut (Lantz, 2013:5):

Reality, machine learning is focused on more practical applications. The task of teaching a computer to learn is tied more closely to a specific problem that would be a computer that can play games, ponder philosophy, or answer trivial questions. Machine learning is more like training an employee than raising a child.

Ada banyak untuk mengklasifikasikan *machine learning,* seperti dampak berdasarkan yang diharapkan *user,* Diskrit atau Kontinu, mudah atau sulit interpretasikan, dan induktif atau deduktif. Pada dampak yang diharapkan *user,* dibagi menjadi 6 kelompok yaitu:

1. *Supervised Learning*

   Pembelajaran tergantung pada kesusuaian, *input* diberi label dan output diberikan. Oleh karena itu *user* sangat berperan pada *supervised learning.* Dapat digunakan untuk klasifikasi maupun regresi.

2. *Unsupervised Learning*

   Pembelajaran ini berbeda dengan *supervised learning, input* akan secara otomatis yang berupa output yang diinginkan. Input pada *unsupervised learning* tidak diberi label pada *input.* Dapat digunakan untuk klasterisasi.

3. *Semi-supervised Learning*

   Merupakan pembelajaran merupakan kombinasi dari *supervised learning* dan *unsupervised learning.* Pada input ada yang di beri label dan tidak berlabel. Dapat digunakan untuk pengklasifikasikan semua input yang diberikan.

4. *Reinforcement Learning*

   Pembelajaran ini berdasarkan pada bagaimana melakukan aksi berdasarkan pengamatan yang ada. Setiap aksi memberikan akan berdampak dan umpan balik untuk memandu pembelajar ini.

5. *Transduction*

   Pembelajaran ini berlatih memprediksi *output* baru berdasarkan *training inputs, training outputs,* dan *testing inputs* tersedia selama tahap pembelajaran.

6. *Learning to learn*

   Pembelajaran ini dengan melakukan pembelajaran bias induktifnya sendiri berdasarkan pengalaman yang sudah terlewati.

Machine learning berdasarkan pada induktif atau deduktif. Induktif merupakan penalaran yang menghasilkan kesimpulan berdasarkan observasi-observasi ganda. Deduktif merupakan penalaran yang yang menghasilkan kesimpulan secara logis dari premis-premis tertentu (Suyanto, 2018:8). Maka dapat disimpulkan induktif menarik kesimpulan dari hal-hal yang khusus untuk menuju kesimpulan yang bersifat umum. Deduktif merupakan kebalikan dari induktif. Contoh metode induktif *Deep Learning*

didalamnya terdapat *Convolutiona Neural Network, Deep Blotzman Machine, Stacked Auto -Encoders.*

2.3 *Artificial Neural Network* dan *Multi Layer Perceptron*

Artificial Neural network merupakan bagian dari *machine learning* yang bermetode induktif. Di dalam buku *Machine Learning* Tingkat Dasar dan Lanjut, makna *ANN* sebagai berikuti:

“ANN terinspirasi oleh sistem manusia (neuron) dalam mengklasifikasi data.“(Suyanto, 2018:77)

Dari kutipan tersebut, maka dapat digambarkan bahwa *ANN* merupakan sebuah algoritme yang terinspirasi dari sistem saraf manusia hanya sistem pada *ANN* lebih sederhana. Banyak model *ANN* dengan pemodelan *supervised* dan *unsupervised*.

Didalam buku *Learn Computer Vision Using OpenCV* penelitian dalam *neural network* dimulai sebagi upaya pembelajaran *simulate multilayered learning* yang dapat sebagai berikut:

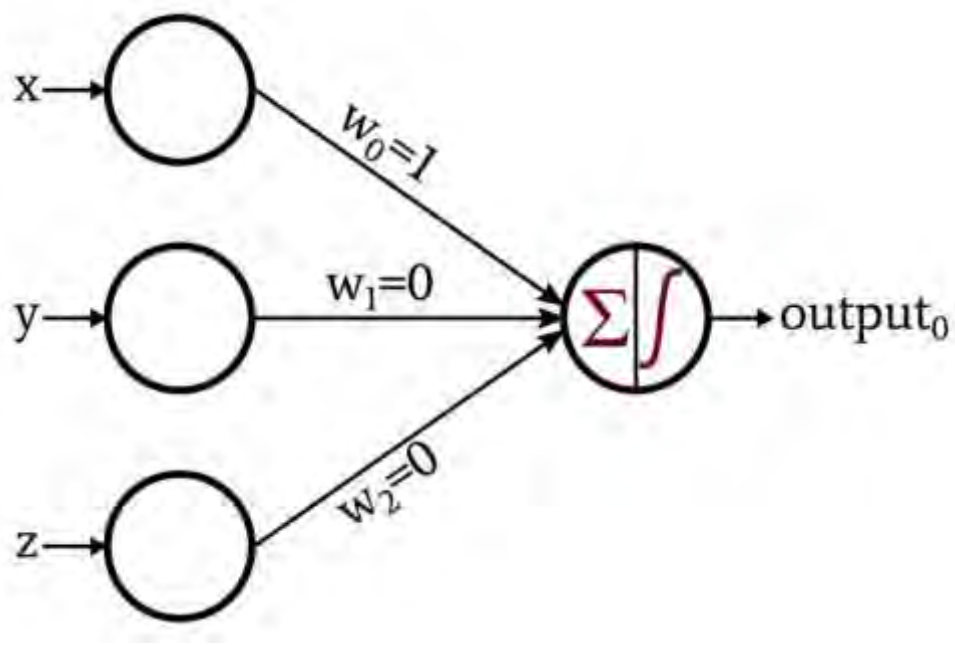
“This definitely requires feeding input to the model a large amount of input variations of handwritten digits or text or object images from which the interpretation rules can be inferred and applied for prediction on a new image input.” (Gollapudi, 2019:58)

Artificial neural network didalam buku *Artificial Neural Network Modelling* adalah sebagai berikut:

“An Artificial Neural Network (ANN) in simple terms is a biologically inspired computational model, which consists of processing elements (called neurons), and connections between them with coefficients (weights) bound to the connections.” (Shanmuganathan & Samarasinghe, 2016:4)

Artificial Neural Network dapat diterjemahkan menjadi Jaringan Saraf Tiruan dan dapat di singkat menjadi *ANN* atau JST yang dapat digunakan untuk memecahkan suatu masalah. JST dimodelkan berdasarkan kemiripan dengan otak manusia yang memiliki banyak node yang terhubung dan dapat mengirim informasi.

JST memiliki beberapa metode antara lain *Single Layer Perceptron*, *Multi Layer Perceptron (MLP),* dan *Probabilitastic Neural Network (PNN).* MLP dan PNN memiliki sifat yang bertolak belakang. Dapat dilihat pada Gambar 2.1 merupakan contoh Single-layer perceptron.

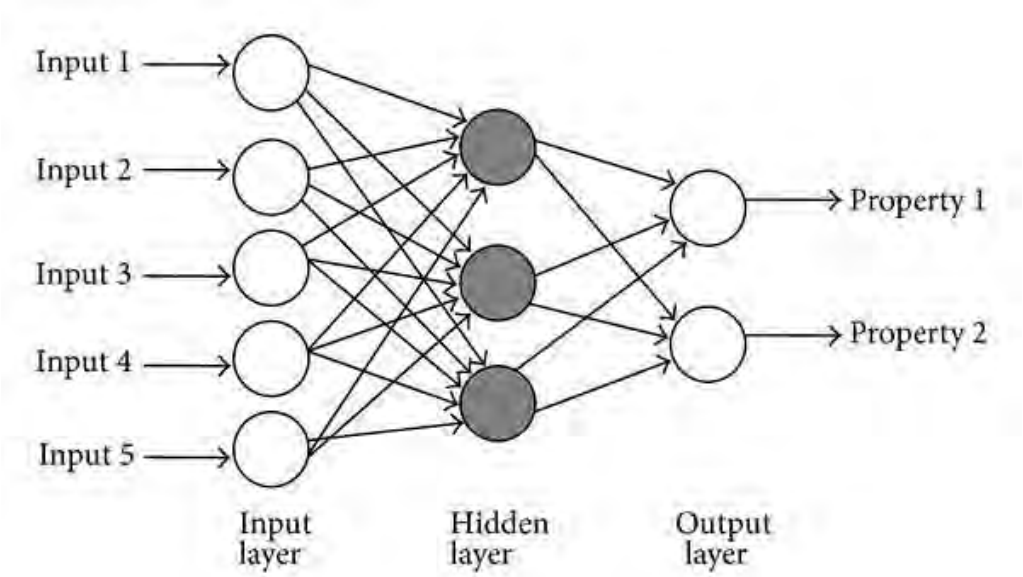


Gambar 2.1
Single-layer perceptron
(Sumber: https://www.allaboutcircuits.com/technical-articles/how-to-train-a-basic-perceptron-neural-network/)

Multi Layer Perceptron merupakan model ANN yang banyak digunakan. Terdiri dari beberapa lapisan. Di dalam buku *Artificial Intelligence Applications and Innovation* didefinisikan sebagai:

"Consider a neural model, which consists of an input layer, one or several hidden layers and an output layer." (Iliadis, Maglogiannis, dkk., 2012:13)

Dari kutipan tersebut, maka dapat digambarkan bahwa MLP memiliki tiga lapisan yaitu lapisan masukan (*input*), lapisan tersembunyi (*hidden*), dan lapisan pengeluaran (*output*). Dapat dilihat pada Gambar 2.2 memiliki input 5, setiap input akan masukan atau diuji masukan ke tahap *Hidden layer* yang terdapat 3 *perceptron.* Pada hasil akhir dibuat pada *output layer* memiliki *2 perceptron.*



Gambar 2.2
Multi Layer Perceptron
(Sumber : https://www.researchgate.net/figure/Schematic-drawing-of-multilayer-perceptron-neural-networks-16_fig1_230735806)

Algorime pelatihan untuk melatih MLP yang paling populer, salah satu adalah *Back Prooagation.* Algoritme ini melakukan pelatihan pembagian kedalam dua tahap. Didalam buku *Machine Learning Tingkat dasar dan Lanjut* sebagai berikut:

"Perhitungan maju untuk menghitung galat (atau biasa disebut *loss function*) antara keluaran aktual dan target; dan perhitungan mundur yang mempropagasikan balik galat untuk memperbaiki bobot-bobot sinaptik pada semua *neuron* yang ada".(Suyanto, 2018:84)

2.4 *Biometric*

Ada banyak teknik untuk mendukung pengaturan keamanan informasi. Banyak permasalahan untuk mengenali seseorang atau identitas menggunakan *token-based system* dan *knowledge-based* kelemahan dari metode ini orang lain dapat menggunakan atau memalsukan akses yang tidak sah atau mudah lupa.

Banyak peneliti dan lembaga untuk meningkatkan sistem data keamanan berdasarkan karakter tubuh atau perilaku.

Dalam buku *Face Detection and Recognition Theory and Practice biometric* adalah sebagai berikut:

"Biometric approaches are concerned with identifying an individual by his unique physical characteristics and biological traits." (Datta, Madhura, dkk., 2015:1)

Perkembangan pendekatan *biometric* seperti *face recognition* dan *voice recognition* untuk mengatasi permasalahan yang dihadapi *token-based dan knowledge-based.* Melalui pendekatan teknik *biometric* dapat meningkatkan keamanan dalam pemalsuan identitas.

Dengan perkembangan teknologi yang pesat khususnya dalam ilmu komputer, sistem *biometric* yang dapat diaplikasikan menurut Sébastien Marcel dan tim dalam buku *Handbook of Biometric Anti-Spoofing* adalah sebagai berikut:

"Nowadays, they are present in a high number of scenarios like border access control, surveillance, smartphone authentication, forensics, and online services like e-learning and e-commerce." (Marcel, Nixon, dkk., 2019:188)

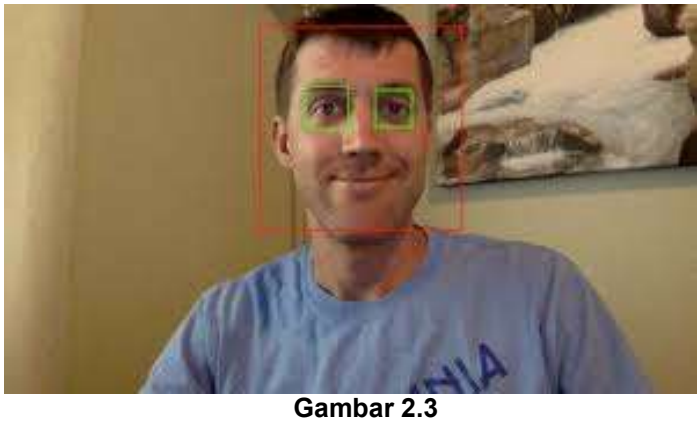
Biometric wajah pada saat ini menjadi materi untuk menjadi objek penelitian setelah *fingher print*. Pengaplikasian untuk pengenalan wajah seperti menerapkan teknologi *ID* wajah.

2.4.1 *Face Detection*

Menurut Asit Kurmar Datta dan tim dalam buku *Face Detection and Recognition Theory and Practice face detection* adalah sebagai berikut:

"*Face detection is a necessary first step in face recognition systems, with the purpose of localizing and extracting the face region from the background*." (Datta, Madhura, dkk., 2015:19)

Jika pada kamera tertangkap gambar wajah, maka sistem akan mengenali bentuk wajah untuk kemudian mengenali dan melokalisasi wajah dengan tanda kotak, sedangkan bila pada kamera tidak tertangkap bentuk wajah, maka tidak ada proses pengenalan. Pada Gambar 2.3 ditampilkan proses *face detection* berupa wajah yang diberi kotak.



Gambar 2.3
Deteksi Wajah
(Sumber : https://www.stoltzmaniac.com/facial-recognition-in-r/)

Adapun tantangan dalam proses pendeteksian wajah sebagai berikut:

1. Oklusi wajah

   Wajah terhalangi oleh benda lain seperti aksesoris.

2. Ekspresi wajah

   Penampilan wajah secara langsung dapat dipengaruhi oleh ekspresi.

3. Penerangan

   Tingkat pencahayaan akan mengaruhi penampilan wajah

2.4.2 *Face Recognition*

Menurut Asit Kumar Datta dan tim dalam buku *Face Detection and Recognition Theory and Practice* menyatakan sebagai beriut:

"Systems and techniques of face recognition and detection are a subset of an area related to information security, and information security is concerned with the assurance of condentiality, integrity and availability of information in all forms." (Datta, Madhura, dkk., 2015,1)

Penelitian mengenai pengenalan wajah secara otomatis atau dapat disebut *Automated Face Recognition* (*AFR*) sudah berlangsung dari tahun 1960. Namun, *AFR* baru popular pada tahun 1990 dan kemudian setelah itu ditemukan Eigenfaces. Beragam model dibangun menggunakan bermacam pendekatan, teknik, dan metode. Tantangan dalam melakukan pengenalan wajah antara lain dari kemiringan wajah, pencahayaan, intensitas cahaya, dan usia. Tahun 2015 menjadi tahun yang penting bagi perkembangan

sistem pengenalan wajah, dimana terbentuknya sebuah model pengenalan wajah yang melampaui manusia.

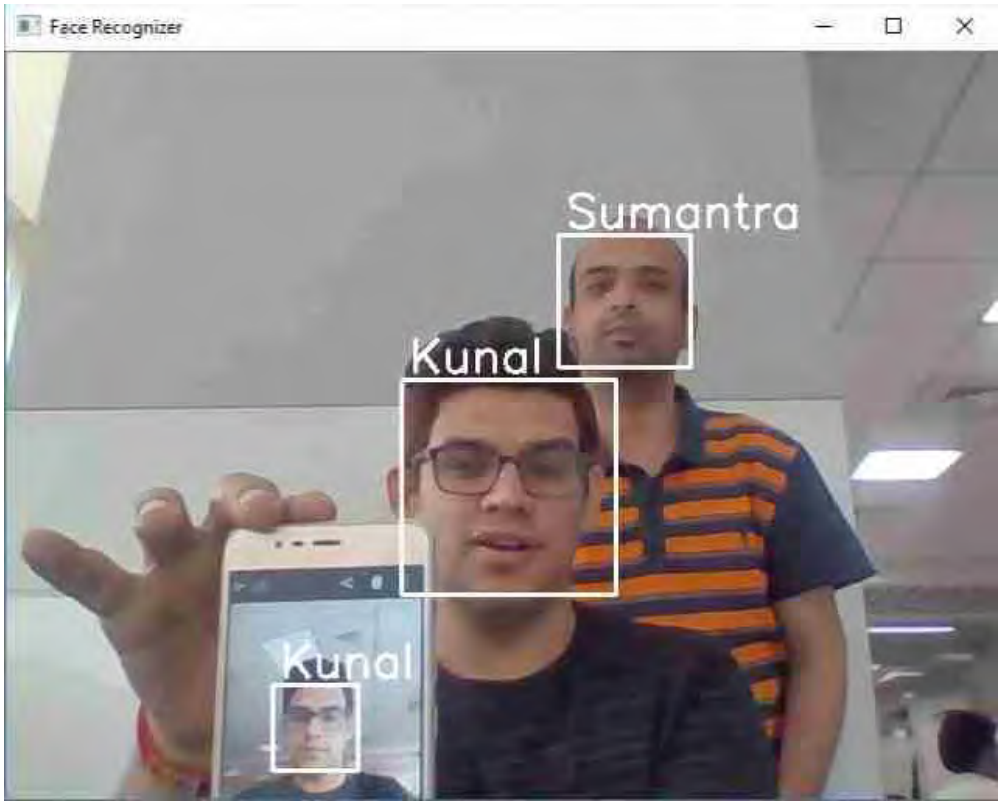
Permasalahan pada pengenalan wajah dibagi menjadi dua bagian, yaitu identifikasi dan verifikasi. Identifikasi diberi masukan sebuah citra wajah dan harus mampu mengidentifikasi citra yang bersal dari wajah yang terdapat pada basis data. Sedangkan verifikasi memasukkan dua citra wajah dan harus memutuskan apakah keduanya berasal dari wajah yang sama.

Biometric dapat dikategorikan menjadi tiga teknik, yaitu *physical biometrics, behavioral biometrics* dan *chemical biometrics*. *Physical biometrics* adalah seperti karakteristik wajah dan sidik jari. B*ehavioral biometrics* adalah hasil pengukuran kerja seseorang seperti tanda tangan, gaya berjalan, dan bicara. *Chemical biometrics* merupakan pengukuran kimia seperti bau dan komposisi kimia dari keringat manusia.

Pengenalan wajah adalah pengaplikasian *machine learning* yang berbasis gambar yang memiliki dua pendekatan yaitu *geometric* dan *photometric* berikut (Suyanto, 2018:413):

> Pendekatan *geometric* menggunakan fitur-fitur geometric seperti panjang dan lebar hidung, posisi mata, posisi mulut, bentuk dagu. Pendekatan *photometric* melakukan penyulingan citra wajah secara keseluruhan kedalam nilai-nilai dan membandingkan nilai-nilai tesebut dengan *template*.

Secara historis, fokus utama penelitian dalam pengenalan wajah diberikan peningkatan kinerja pada tugas verifikasi dan identifikasi. Pada gambar 2.4 ditampilkan contoh *face recognition* sebagai berkut:



Gambar 2.4
Face Recognition
(Sumber : https://mc.ai/face-recognizer-application-using-a-deep-learning-model-python-and-keras/)

Pengenalan wajah pada saat ini sudah memasuki tahap ke empat dengan maraknya penggunaan *Deep learning* seperti penggunaan *Convolutional Neural Network, Deep Belief Network,* dan *Stacked Auto Encoders.*

2.5 *Deep Learning* dan *Convolutional Neural Network*

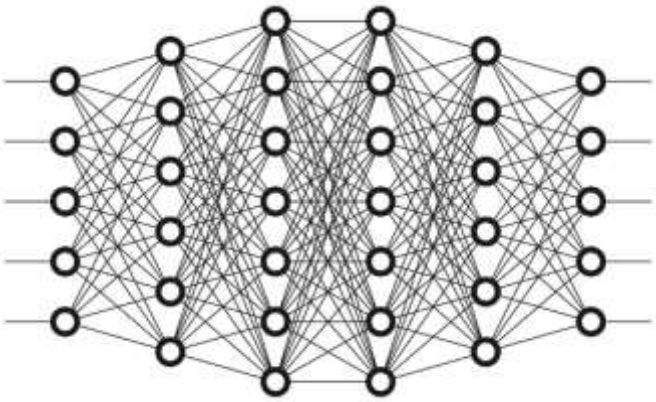
Deep learning muncul pada tahun 2012, *machine learning* dikejutkan dengan munculnya *deep learning*. Pada buku *Machine Learning Tingkat dasar dan Lanjut* sebagai berikut:

“Pada sebuah komputesi pengenalan citra *ImageNet Large Scale Visual Recognition Compotision* (ILSVRC) tahun 2012, teknik ini menjadi pemenang pertama dengan tingkat akurasi yang tinggi.”(Suyanto, 2018:305)

Deep learning dalam buku *Learning Computer Vision Using OpenCV* dapat diartikan sebagai berikut:

"Deep learning is an assemblage of techniques from an artificial neural network (ANN), which is a subfield of machine learning."(Gollapudi, 2019:58)

Regresi, *Decision Tree Learning, Naïve Bayes, Artificial Neural Network* melakukan pembelajaran secara dangkal (*shallow learning)* atau dengan jumlah lapisan yang terbatas. *Shallow learning* memaksakan pengguna untuk membangun model ekstrasi yang tepat untuk meningkatkan akurasi. Walaupun jumlah data latih ditambahkan, hal tersebut tidak akan meningkatkan akurasinya secara baik, dapat dilihat pada gambar di bawah ini (Gambar 2.5).



Gambar 2.5
contoh *deep learning*
(Sumber : https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063)

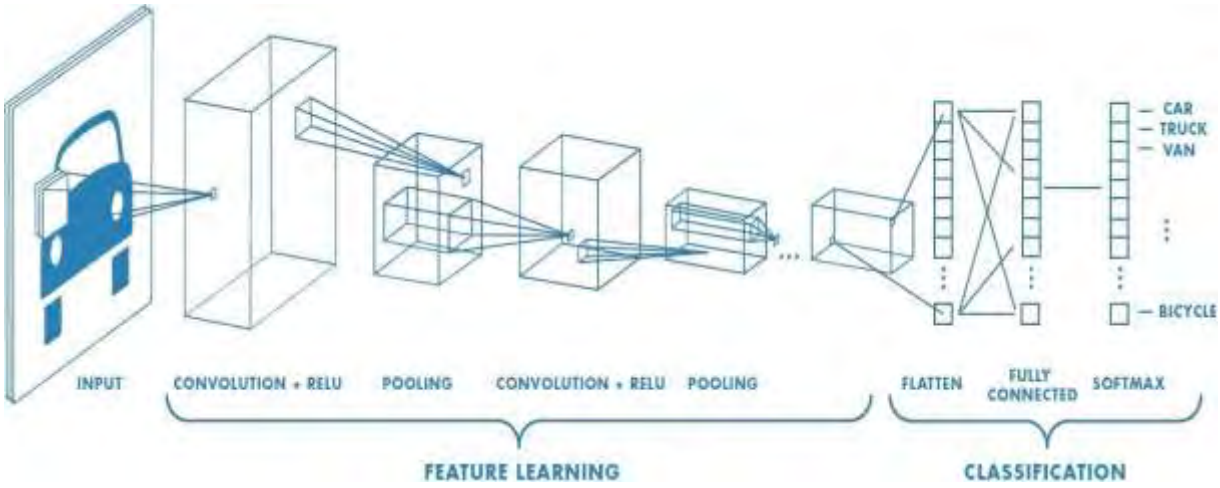
Deep learning dapat merekayasa fitur secara otomatis sehingga pengguna tidak perlu membangun model eksraksi yang sering sulit dan rumit. *Deep leraning dan Artificial Neural Network* memiliki perbedaan ANN dengan arsitektur MLP pembelajaran dengan kosep jaringan terhubung penuh (*fully connected neural network*). Sedangkan *deep learning* menggunakan skema jaringan yang sederhana karena menggunakan filter.

Deep learning dalam buku *Learning Computer Vision Using OpenCV* dapat diterapkan sebagai beriut:

"Deep learning is applied in the classification, detection, segmentation, and generation of images and videos in computer vision applications." (Gollapudi, 2019:51)

MLP memiliki perbedaan dengan *CNN*, MLP yang berarsitektur disusun secara dua dimensi, sedangkan arsitektur *CNN* menggunakan tiga dimensi yaitu lebar, tinggi, dan

kedalaman yang dapat dilihat pada Gambar 2.6. Gambar tersebut merupakan contoh *Convolutional Neural Network*.



Gambar 2.6
Convolutional Neural Network
(Sumber : https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53)

CNN dalam buku *Learning Computer Vision Using OpenCV CNN* sebagai berikut: *"This helps handle complex variations in images with higher accuracy."* (Gollapudi, 2019:63)

Setiap lapisan *CNN* mentransformasikan volume masukan tiga dimensi ke dalam volume keluaran tiga dimensi melalui aktivasi-aktivasi sel saraf pada gambar tersebut. Masukan berupa citra berwarna, dimana lebar dan tinggi menyatakan dimensi citra tersebut, sedangkan dalam depthnya adalah 3 yang menyatakan kanal *red, green,* dan *blue.*

Untuk MLP pada citra 32x32x3 RGB dikalikan 100 *neurons* menjadi 307.200 *fully connected.* Sedangkan *CNN* pada citra yang sama 32x32x3 RGB menggunakan 100 *filters* yang berukuran 5x5x3 menjadi 7.500 koneksi.

Convolutional Neural Network menurut Alom et al yang disertai dalam buku *Machine Learning Tingkat Dasar dan Lanjut* adalah sebagai berikut:

"*CNN* adalah salah satu kelas *deep feed-forward artificial neural networks* yang banyak diaplikasikan pada analisi citra." (Suyanto, 2018:308)

CNN terdiri dari lapisan masukan, lapisan tersembunyi, dan lapisan pengeluaran. Lapisan tesembunyi pada *CNN*, menurut Alom et al yang disertai dalam buku *Machine Leraning Tingakat Dasar dan Lanjut* adalah sebagai berikut:

"Lapisan tersembunyi berisi *convolution layer, pooling layer, normalization layer, ReLu Layer, fully connection layer,* dan *Loss layer.*" (Suyanto, 2018:309)

Gambar2.7 di bawah ini merupakan salah satu contoh arsitektur dalam algoritme *Convolutional Neural Network*:

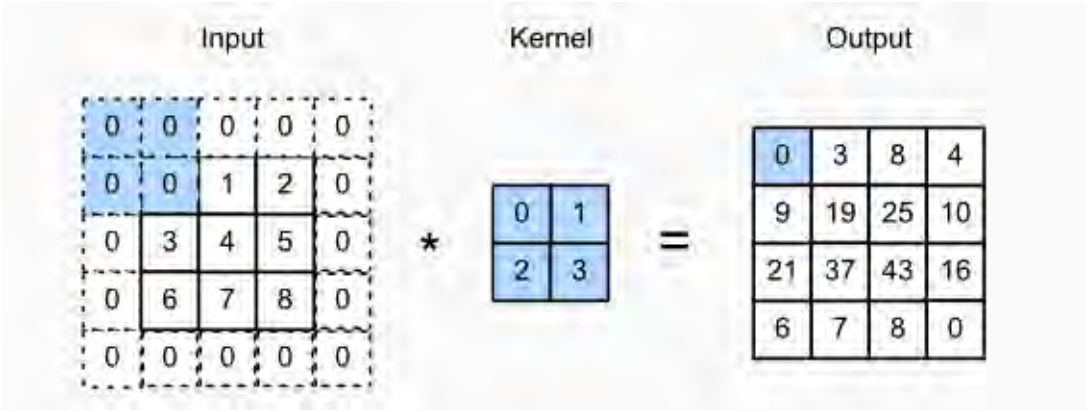
Gambar 2.7
Arsitektur *AlexNet*
(Sumber: Alon,M.Z. et al., 2018)

1. *Convolution Layer*

   Convolution Layer merupakan bagian terpenting pada *Convolutional Neural Network*. *CNN* umumnya menggunakan *stride* = 1 menggunakan zero padding sebesar:

   $$\boldsymbol{P = \frac{(F-1)}{2}}$$

   P adalah Padding, lalu F merupakan ukuran bidang reseptif yang sama dengan ukuran filter. Contoh diilustrasikan proses *convolution* dengan stride = 1 pada Gambar 2-3 dengan perhitungan 0 × 0 + 0 × 1 + 0 × 2 + 0 × 3 = 0. Contoh pada proses convolution dapat di lihat pada Gambar 2.8



Gambar 2.8
Proses *Convolution*
(Sumber: https://d2l.ai/chapter_convolutional-neural-networks/padding-and-strides.html)

Ukuran dimensi masukan dapat dinyatakan

$$W_1 \times H_1 \times D_1 \tag{2-1}$$

*W*1 dan *H*1 adalah lebar dan tinggi dari citra. *D*1 adalah jumlah kanal *Red, Green*, dan *Blue* dapat disebut *depth*.

CNN memiliki empat *hyperparrameters*, yaitu jumlah filter K, ukuran bidang reseptif atau *spatial extent*, lebar langkah atau *stride S,* dan *zero padding* P.

Output dapat dihitung dengan formula:

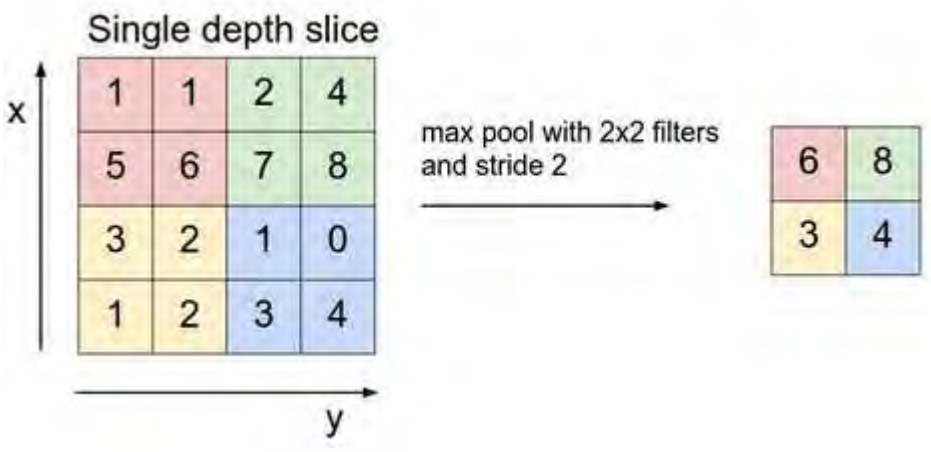
$$W_2 \times H_2 \times D_2, \tag{2-2}$$

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1, \tag{2-3}$$

$$h_2 = \frac{(h_1 - F + 2P)}{S} + 1, \tag{2-4}$$

2. Pooling Layer

Tahap selanjutnya Pooling Layer berfungsi untuk menjaga ukuran data ketika convolution, dengan melakukan *down sampling* dapat dilihat pada Gambar 2.9:



Gambar 2.9
Max Pooling
(Sumber: http://cs231n.github.io/convolutional-networks/)

Pooling Layer dapat mempresentasikan data menjadi lebih kecil supaya mudah untuk dikelola dan juga mempermudah kontrol *overfitting*. Proses Pooling yang digunakan pada umumnya menggunakan *Max Pooling* yang memiliki nilai maksimum dalam suatu area. Terlihat pada Gambar 2.9, dimana *Max pooling* dari empat nilai di bagian kiri atas akan menghasilkan satu nilai maksimum, yaitu 6 untuk mengisi data hasil *Max Pooling*. Hal yang sama digunakan pada empat nilai di bagian kanan atas, kiri bawah, dan kanan bawah. Teknik *Max Pooling* memberikan performansi yang lebih baik di banding *L2-norm Pooling* dan *Average Pooling*.

3. *Normalization Layer*

Normalization Layer pada awalnya didesain untuk mengatasi adanya perbedaan rentang nilai yang sinifikan pada citra masukan atau *input*. Para ahli telah mengusulkan banyak jenis lapisan normalisasi. Lapisan normalisasi tidak banyak digunakan secara praktis karena dampaknya tidak terlalu besar, atau tidak ada dampak.

4. ReLu layer

Rectified Linear Units biasa disebut ReLu layer. Lapisan ini menggunakan pengaplikasiaan fungsi aktivasi $f(x) = \max(0, x)$. Lapisan ini meningkatkan sifat

nonlinearitas fungsi keputusan dan jaringan secara keseluruhan tanpa mempengaruhi bidang-bidang reseptif pada *covolution layer*.

5. *Fully Connected Layer*

   Fully Connected Layer dapat disebut terhubung secara penuh, setiap *neurons* memiliki koneksi penuh ke semua aktivasi dalam lapisan sebelumnya. Hal ini sama dengan *Multi Layer Perceptron*. Komputasi menggunakan suatu perkalian matriks yang diikuti dengan bias *offset.*

6. *Loss Layer*

   Loss Layer merupakan lapisan terakhir pada *Convolutional Neural Network* yang menentukan bagaimana pelatihan dapat memberi *penalty* atas penyimpangan hasil prediksi dan label. Ada tiga variasi *loss function*, yang pertama *softmax loss* digunakan untuk memprediksi satu dari kelas yang saling eksklusif. Selanjutnya *sigmoid cross-entropy loss* yang digunakan untuk memprediksi sejumlah nilai probalitas dalam interval I $[0,1]$, dan *Euclidean loss* yang digunakan untuk regresi nilai kontinu.

2.6 Diagram *UML*

Rational Software membawa tiga pemimpin industri bersama untuk menciptakan pendekatan tunggal untuk sistem berorientasi objek. Pengembangan Grady Booch, Ivar Jacobson, dan James Rumbaugh bekerja membuat seperangkat teknik diagram yang dikenal sebagai *Unifieded Modeling Language* (*UML*).

Unified Modeling Language (*UML*) memiliki beberapa pengertian, didefinisikan sebagai:

"*The Unified Modeling Language (UML) of the Object Management Group (OMG) provides a standardized mechanism to model software systems.*" (Unhelkar, 2017:5)

"*UML* (*Unified Modeling Language*) adalah 'bahasa' pemodelan untuk sistem atau perangkat lunak yang berparadigma 'berorientasi objek'." (Nugroho, 2010:6)

Dua definisi yang di atas dapat disimpulkan *UML* adalah bahasa pemrograman yang menyediakan mekanisme standar untuk memodelkan perangkat lunak.

UML menyediakan seperangkat alat standar untuk mendokumentasikan sistem perangkat lunak. *Toolset UML* berisi diagram yang memungkinkan orang untuk menggambarkan pembangunan sistem yang berorientasi objek. *UML* menyediakan sarana yang efektif untuk komunikasi atau menggambarkan suatu program. *UML* terdiri dari *things, relationships*, dan diagram.

Hal pertama yang merupakan elemen terpenting dalam *UML*, yaitu *things* terdiri dari *structural things*, *behavioral things*, *group things* dan *annotational things*. *Structural things* hal yang paling umum terdiri dari kelas, antarmuka, *collaborations, use cases, active classes, components,* dan *nodes*. *Behavioral things* menggambarkan hal perilaku cara kerja sesuatu. *Group things* digunakan untuk mendefinisikan batasan. *Annotational things* untuk menambahkan catatan ke diagram.

Relationships merupakan penghubung antara satu dengan yang lain, terdiri dari hubungan terstruktur dan hubungan perilaku. Hubungan terstruktur terdapat *dependencies*, *aggregations*, *associations*, dan *generalizations*. Hubungan perilaku terdiri dari *communicates*, *includes*, *extends*, dan *generalizes*.

Diagram terbagi dua kelompok diagram yaitu diagram terstruktur dan diagram yang berdasarkan perilaku.

Diagram terstruktur menggambarkan hubungan antar kelas. Pada buku *Systems Analysis and Design* terdapat diagram didalamnya sebagai berikut:

"They include class diagrams, object diagrams, component diagrams, and deployment diagrams."(Kendall, 2014:287)

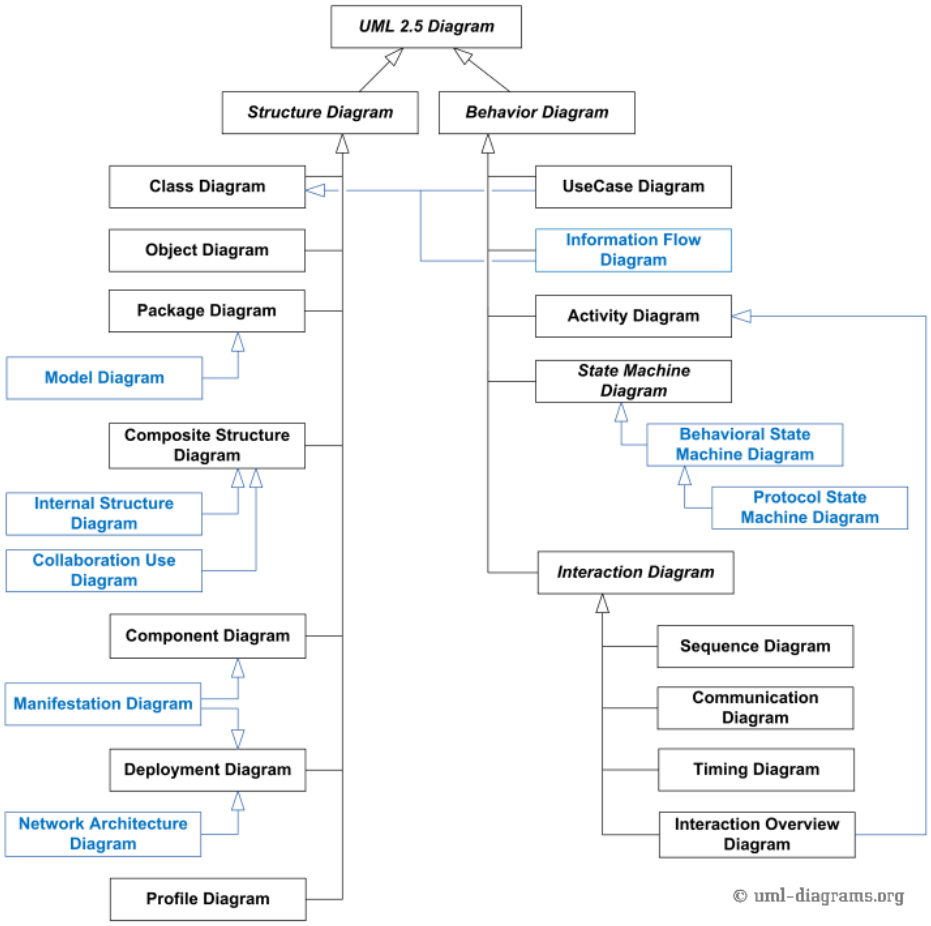
Diagram struktur menampilkan struktur yang statis pada suatu sistem. Mewakili konsep memiliki makna dari suatu sistem, dapat mencakup abstrak dan pengimplementasian konsep.

Diagram perilaku (*Behavioral*) menggambarkan interaksi aktor (diluar sistem) dengan sistem. Diagram perilaku pada buku *System Analysis and Design* terdapat diagram didalamnya sebagi berikut:

"Behavioral diagrams include use case diagrams, sequence diagrams, communication diagrams, statechart diagrams, and activity diagrams." (Kendall, 2014:287)

Diagram perilaku menampilkan perilaku yang dinamis pada suatu sistem, yang digambarkan sebagai rangkaian perubahan pada sistem waktu ke waktu.

Pada UML 2.5 Diagram yang ditunjukkan oleh Gambar 2.10 di bawah ini, diagram terbagi menjadi 2 kelompok utama, yaitu diagram struktur dan diagram perilaku (behavior), dengan total sebanyak 25 jenis diagram.



Gambar 2.10
Bagan *UML* 2.5
(Sumber: https://www.uml-diagrams.org/uml-25-diagrams.html)

1. *Class diagram*

   Diagram Kelas (*class diagram)* merupakan diagram struktur yang menampilkan struktur sistem yang di rancang di tingkat *classes* dan antarmuka (*interface*). Jenis diagram kelas yang umum antara lain adalah *domain model diagram* dan *diagram of implemation classes.*

2. *Object diagram*

Diagram objek dalam UML 1.4.2 merupakan turunan dari kelas diagram, dapat disebut diagram kelas dengan objek yang tidak menggunakan kelas. Pada UML 2.5, *class diagram* dan *object diagram* ditegaskan keduanya berbeda dan tidak saling berkaitan. *Component diagram* dan *deployment diagram* yang hanya berisi tuntutan spesifikasi yang termasuk ke dalam *Object diagram.*

3. *Package diagram*

   Package diagram merupakan diagram struktur yang menunjukan struktural sistem dan dirancang pada *package*. Emelen yang biasa digunakan pada *package diagram*, antara lain *package, packageable element, dependency, element import, package import, package merge.*

4. *Model diagram*

   Diagram model merupakan diagram struktur yang bersifat tambahan. Diagram ini menunjukan abstraksi dan padangan spsesifik pada sistem untuk menampilkan beberapa askpek arsitektur, logis dan perilaku sistem. Model berisi *package* dan memilki hubungan (*relationships*).

5. *Composite structure diagram*

   Composite structure diagram dapat digunakan untuk menunjukan *internal structure diagram* dan *collaboration use diagram*. Struktur dalam *composite structure diagram* diartikan elemen yang saling terhubung untuk mencapai tujuan bersama. *Internal structure diagram* menampilkan struktur pengklasifikasian menjadi *Properties, part* dan *relationships*. *Collaboration use diagram* dapat digunakan untuk memberikan deskripso tentang perilaku pnegklasifikasian pada tingkat abstrak yang berbeda ditawarkan struktur internal.

6. *Deployment diagram*

   Deployment diagram merupakan diagram struktur, menampilkan arsitektur sistem sebagai distribusi artefak peragkat lunak menuju target penyebaran. Dalam UML 2.X artefak digunakan untuk node dan dapat mengimplementasikan komponen. *Deployment diagram* menggambarkan arsitektur pada level spesifik mirip dengan *class diagrams* dan *object diagrams.* Jenis yang *deployment diagram* meliputi

Implementation (manifestation) of components by artifacts, Specification level deployment diagram, Instance level deployment diagram, Network architecture of the system.

7. *Network Architecture Diagram*

   Deployment diagram tidak menyediakan elemen secara spesifik untuk jaringan. *Network architecture of the system* akan menunjukan node jaringan dan jalur komunikasi. Windows Server System Reference Architecture (WSSRA) untuk memperlihatkan keseluruhan arsitektur jaringan.

8. *Use case diagram*

   Menurut buku *Software Engineering: APractitioner's Approach, 7th Edition* dijelaskan sebagai berikut:

   "A UML use-case diagram is an overview of all the use cases and how they are related." (Pressman, 2010, p. 848)

   Use case diagram sebagai diagram perilaku (*behavior*) unutk menggambarkan *use case* di kolaborasi dengan satu atau lebih *actor.* Pada fersi UML 2.5 FTF – Beta memindahkan *use case* dari pemodelan perilaku menjadi konsep tambahan UML. Di dalam *use case diagram* terdapat dua diagram, yaitu *business use case diagram* dan *system use case diagram*.

9. *Information Flow Diagram*

   Information flow diagram merupakan diagram perilaku yang menampilkan pertukaran informasi antara entitas sistem pada abstrak. *Information flow* berguna untuk menggabarkan aliran informasi dan pertukaran melalui sebuah sistem.

10. *Activity Diagram*

    Activity diagram merupakan diagram perilaku yang menunjukan aliran control dengan penekanan pada urutan. *Activity Diagram* pada buku "*Software Engineering: A Practitioner's Approach*" didefinsikan sebagai berikut:

    "The UML activity diagram supplements the use case by providing a graphical representation of the flow of interaction within a specific scenario."(Pressman, 2010:161)

11. *State Machine Diagram*

    State machine diagram merupakan diagram perilaku, menunjukan perilaku diskrit. State machine diagram digunakan mengekspresikan protokol dari sebuah sistem. Di dalam *state machine learning* terdapat dua diagram yaitu *behavioral state machine* dan *protocol state machine. Behavioral state machine* merupakan spesialisasi perilaku dan digunakan menentukan perilaku diskrit dalam keadaan yang terbatas. *Protocol state machine* merupakan spesialisasi *behavioral state machine* dan digunakan untuk mengekspresikan protocol pengguna.

12. *Interaction diagram*

    Interaction diagram di dalamnya terdapat empat diagram yang berbeda yang pertama *sequence diagram, communication diagram, timing diagram,* dan *interaction overview diagram*. *Sequence diagram* merupakan diagram interkasi yang paling umum, berfokus pada pertukran pesan antara beberapa *lifeline*, berfokus pada urutan pesan ditukarkan. *Communication diagrams* merupakan diagram interaktsi antara objek, menggunakan pesan berurutan.Timing diagram diagram interaksi digunakan untuk menunjukkan interaksi yang beralasan waktu. *Interaction overview diagram* merupakan gabungan elemen dari *activity diagram* dan *interaction diagram.*

2.7 Bahasan Pemrograman

Bahasa pemrograman Python pertama dikeluarkan pada tahun 1991 oleh Guido van Rossum di negara Belanda. Nama *Python* terinspirasi dari grup komedi yang berasal dari Negara Inggris bernama *Monty Pyton*.

Python dikembangkan bersifat *open source* dengan sebagian besar versi lisensi *GFL-compatible*. Perkembangan *Python* dikembangkan dari satu perusahaan ke perusahaan lainnya. Pada awal tahun 1990, Python dikembangkan di *Scitchting Mathematisch Centrum* Belanda. Tahun 1995 Guido meluncurkan 2 versi Python di *Corporation for National Research Initiative* Virginia, Amerika Serikat. Tahun 2000 Guido beserta tim Python membentuk tim BeOpenPythonLabs yang berfokus mengembangkan Python. Tahun 2001 *Python Software Foundation* yang dibuat untuk menangani hak

intelektual Python dan sampai pada saat ini masih melakukan riset untuk pengembangan Python.

Python 2 merupakan pengembangan dari Python 1 dengan berbagai penambahan fitur. Dibandingkan dengan versi 1, Python 2 lebih tranparan dan inklusif untuk pengembangan aplikasi. Fitur baru pada Python 2 memiliki *Python Enchancement Proposal* yang memiliki kemampuan memberikan tuntunan informasi bagi para pengguna. Python 2 mendapatkan peningkatan dukungan untuk Unicode dan dilengkapi fitur *programmatical* yang diantaranya berfungsi untuk menajemen memori.

Python 3 merupakan pengembangan dari Python 2 yang berfokus untuk merapikan *codebase* dan menghilangkan *redundancy*. Fitur yang melambangkan perubahan besar membuat statemen print dalam fungsi yang *build-in*. Python 3 memiliki kelemahan dimana kurangnya kompatibilitas dengan Python 2, sehingga pemindahan versi harus dilakukan secara hati-hati.

Python merupakan bahasa pemrograman yang perkembanganya cukup pesat karena memiliki kelebihan, diantaranya aspek keterbacaan, efisiensi, dan multifungsi interoprabilitas. Aspek keterbacaan atau *readibility* Python mudah dibaca dan dipahami, memiliki sintaks yang sederhana dan mudah ditulis. Aspek efisien Python adalah memiliki library yang lengkap, Python Code lebih simple dibandingkan bahasa pemrograman seperti Java, Visual Basic, C yang dapat menghemat waktu serta meningkatkan produktivitas. Aspek multifungsi dengan menggunakan Python yaitu dapat membuat aplikasi jaringan, website, pengembangan aplikasi bidang robotika, pengembangan *deep learning, machine learning,* dan pengembangan *artificial intelligent.* Hal tersebut dikarenakan Phyton memiliki perpustakaan atau *library.* Aspek interoperabilitas Phyton memungkinkan dapat berintegrasi dengan bahasa pemograman lain, dimana program yang dikembangkan menggunakan Bahasa python dapat dioprasikan di semua sistem operasi. Python memiliki dukungan komunitas atau forum untuk berbagi dan mendapatkan informasi.

2.8 Library

2.8.1 *Tensorflow*

TensorFlow merupakan salah satu library untuk *machine learning* yang bersifat *open source.* Di dalam buku *TensorFlow in 1 Day Make your own Nural Network,* Tensorflow didefinisikan sebagai:

"TensorFlow is a library developed by the Google Brain Team to accelerate machine learning and deep neural network research." (Rungta, 2018, p. 29)

Dari kutipan tersebut, maka dapat digambarkan bahwa *TensorFlow* merupakan *Library* yang dikembangkan oleh Tim GoogleBrain untuk mempercepat pembelajran mesin dan penelitian jaringan saraf.

Tensorflow library memiliki keistimewaan dalam buku *Deep Learning with TensorFlow 2 and Keras* sebagai berikut (Gulli, Kapoor, dkk., 2019:1):

Most other deep learning libraries – like TensorFlow – have auto-differentiation (a useful mathematical tool used for optimization), many are open source platforms, most of them support the CPU/GPU option, have pretrained models, and support commonly used NN architectures like recurrent neural networks, convolutional neural networks, and deep belief networks.

TensorFlow memilki perubahan pada versi 2.0 karena perpustakaan ini sudah menjadi satu dengan perpustakaan atau *library* Keras. Membuat *tensorflow* sebagai *Back-end* dan Keras sebagai perpustakaan yang berada diatas *TensorFlow*.

2.8.2 *OpenCV*

OpenCV merupakan singkatan dari *open source computer vision* yang bersifat open source dalam buku yang berjudul *Learning Computer Vision Using OpenCV* sebagai berikut:

"The framework comprises tools, libraries, and modules that have built-in support for implementing computer vision applications." (Gollapudi, 2019:32)

OpenCV mendukung Bahasa pemrograman C/C++, Python, dan Java, untuk membangun aplikasi menggunakan sistem oprasi di desktop dan selurer seperti Windows, Linux, macOS, Android, dan iOS.

Library opencv dibagun untuk mempercepat pada proses visi komputer. Menurut situs resmi (https://opencv.org/about/), *opencv* memiliki lebih dari 2500 algoritme yang di dalamnya terdapat algoritme visi Komputer (*computer vision*) dan algoritme pembelajaran mesin (*machine learning*). (anonim, 2020)

Opencv diimplementasikan dengan beberapa contoh seperti pendeteksi wajah, pengenalan wajah, mengklasifikasi gerak manusia didalam video, dan deteksi objek. Opencv bersifat open source sehingga memiliki komunitas pengguna yang besar dan dapat digunakan oleh perusahaan maupun lembaga penelitian, baik swasta ataupun pemerintah.

BAB III

ANALISIS DAN PERANCANGAN PERANGKAT LUNAK

3.1 Gambaran Umum

Perangkat lunak ini dirancang berdasarkan pengimplentasian algoritme *Convolutional Neural Network* pada *Face Recognition*. Sistem akan mendapatkan citra yang akan menjadi dataset dari *capture webcam.* Kemudian, sebelum proses pengenalan dilakukan, maka *dataset* yang sudah tersimpan tersebut harus dilatih terlebih dahulu. *Library* yang digunakan untuk melatih data latih yang sudah tersedia adalah *Tensorflow* (*Keras*) dengan algoritme *CNN*, sedangkan pendeteksi wajah menggunakan *library opencv.* Pada sistem tersebut data dapat ditambah, kemudian dapat juga dilakukan pendeteksian dan pengenalan wajah.

Tahapan dimulai dengan proses pendaftaran, dimana dalam proses ini citra wajah akan disimpan pada web cam dan juga dilakukan pemberian label pada citra tersebut. Proses pendeteksian wajah berlangsung ketika pada citra yang tertangkap oleh kamera dapat dikenali objek yang menyerupai wajah manusia. Pada citra tersebut selanjutnya akan diberi tanda berupa border kotak sebagai penunjuk citra wajah.

Proses training dataset menggunakan algoritme *Convolutional Neural Network* menitik beratkan pada proses *preprocessing.* Terhadap citra wajah yang tersedia akan diakukan *augmentasi* data dengan tujuan untuk memperbanyak jumlah data citra. Hal tersebut dikarenakan pada algoritme *Convolutional Neural Network* yang digunakan ini harus mempunyai jumlah data *training* yang banyak. Proses training sendiri dapat menggunakan beberapa lapisan, di antaranya seperti *Conv2D, MaxPooling2d, Flatten,* dan *Dense*. Model training yang digunakan pada percobaan ini berupa model training berulang (*sequential*). Setelah model training ditentukan, maka proses training dilakukan. Kemudian setelah proses training selesai dilakukan, maka data hasil training (model) yang diperoleh akan disimpan. Jumlah dataset yang digunakan dalam pengujian ini adalah 36 orang yang 32 orang diantaranya berasal dari database bollywood_celeb_face_0 dan 4 orang selanjutnya berasal dari dataset yang dibuat sendiri.

Data hasil training tersebut diproses menggunakan algoritme *CNN* dan juga dilakukan pengecakan data terhadap citra yang diperoleh dari proses pendeteksian wajah, termasuk penentuan akurasi kemiripan dengan data yang sudah dilatih dengan algoritme tersebut.

3.2 Kebutuhan Spesifikasi

3.2.1 Kebutuhan Fungsional

Kebutuhan Funsional yang berisi proses yang nantinya harus disediakan oleh sistem. Aksi dan reaksi sitem pada suatu reaksi tertentu. Kebutuhan funsional pada *face recognition*:

1. Pengguna dapat memasukan data gambar wajah.
2. Pengguna dapat mencapture gambar wajah pada sistem.
3. Pengguna mampu menampilkan deteksi wajah.
4. Pengguna sistem mampu menampilkan pengenalan wajah.

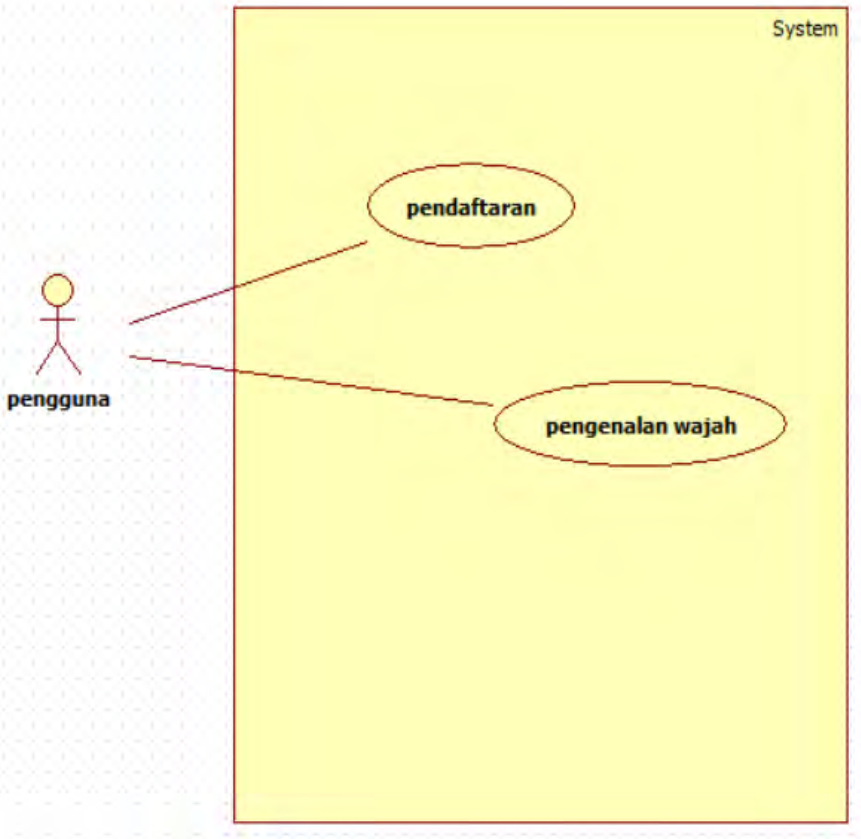
3.2.2 Kebutuhan Non-Fungsional

Kebutuhan non-funsional merupakan kebutuhan menitik beratkan pada perilaku yang dimiliki sistem.

1. Menggunakan oprasi sitem Windows.
2. Spesifikasi komputer intel i3.
3. Kebutuhan RAM 4GB.
4. Digunakan untuk menampilkan pendaftaran.
5. Digunakan untuk menapilkan deteksi wajah.
6. Digunakan untuk menampilkan pengenal wajah.

3.3 Pemodelan Sistem

3.3.1 *Usecase* Pengenalan Wajah



Gambar 3.1
Use Case diagram

3.3.2 Skenario

3.3.2.1 Skenario *Use Case* Pendaftaran

Use Case : Pendaftaran

Aktor : pengguna

Precondition : pengguan menekan tombol ‘Pendaftaran’ dan *frame* utama (*Home*) telah terbuka

Tabel 3.1
Skenario normal *use case* pendaftaran

Aktor	Sistem
1. Menekan tombol ‘pendaftaran’ pada *frame* home.	
	2. Menampilkan form pendaftran
3. Memberi nama gambar yang akan di *capture*	
	4. Menangkap gambar wajah
5. Tekan tombol simpan	
	6. Menyimpan gambar yang sudah diberi nama
7. Menutup form pendaftaran	

3.3.2.2 Skenario Use Case Pengenalan

Use Case : Pengenalan wajah

Aktor : pengguna

Precondition : pengguna menekan tombol "pengenalan" dan menutup frame pendaftaran.

Tabel 3.2
Skenario normal *use case* pengenalan wajah

Aktor	Sistem
1. Menekan tombol ‘pengenalan wajah’ pada *frame* home.	
	2. Menampilkan *frame* pengenalan wajah
3. Menekan tombol ‘pengenalan’	
	4. sistem melakuka deteksi wajah
	5. Mendeteksi wajah
	6. Melakukan pencocokan data dengan *CNN*
	7. Menampilkan nama yang telah mengalami proses pembelajaran

3.3.2.3 Skenario Alternatif Pengenalan Wajah

Use Case : Deteksi wajah

Aktor : pengguna

Precondition : sudah menutup *frame* deteksi, tidak di temukan pada data

Tabel 3.3
Skenario Alternatif *use case* pengenalan wajah

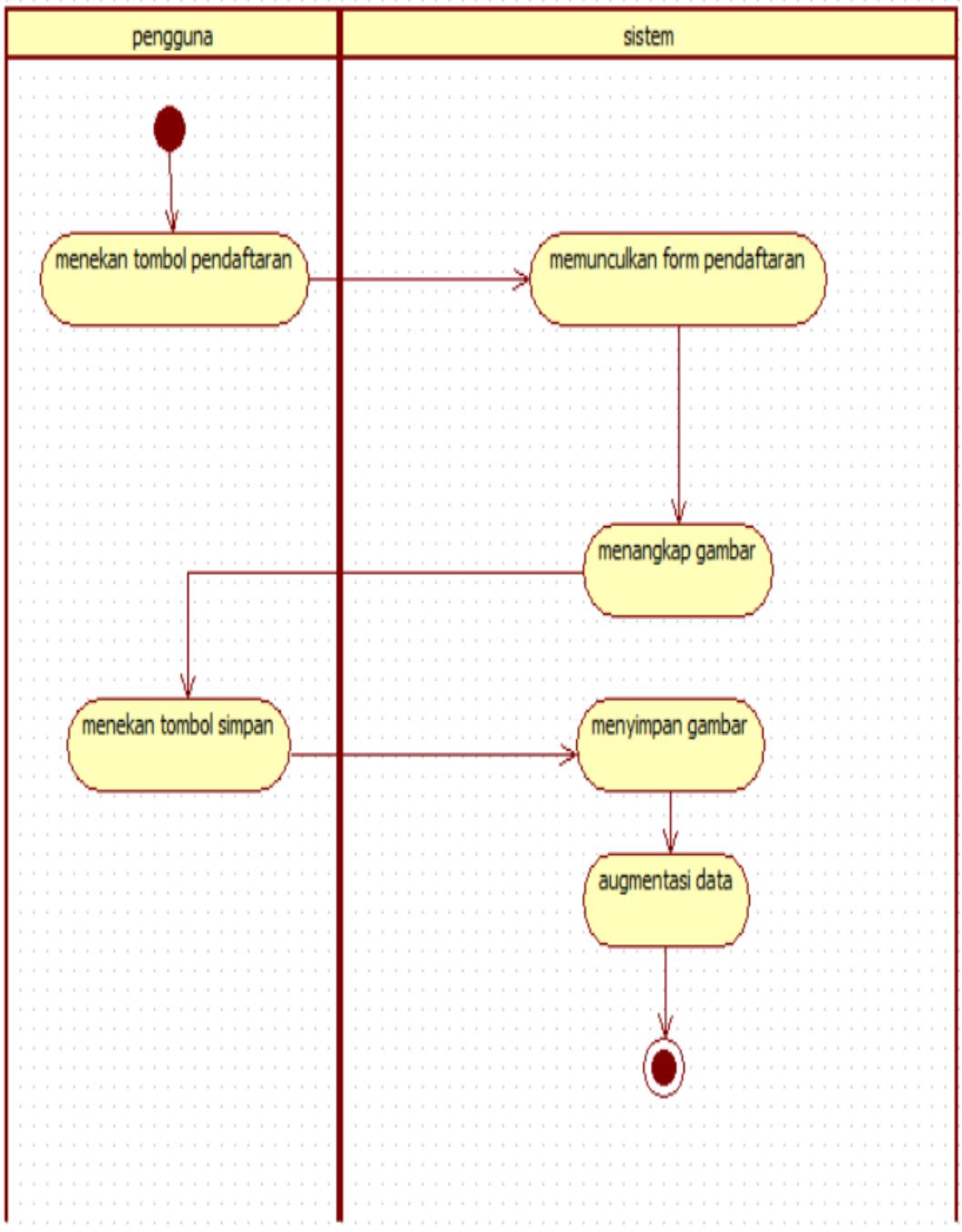
Aktor	Sistem
1. Menekan tombol 'pengenalan wajah' pada *frame* home.	
	2. Menampilkan *frame* pengenalan wajah
3. Menekan tombol 'pengenalan'	
	4. Melakukan capture gambar
	5. Mendeteksi wajah
	6. Melakukan pencocokan data dengan *CNN*
	7. Menampilkan pesan "Tidak dikenal"

3.3.3 *Activity Diagram*

Activity diagram merupakan pengambaran urutan aktivitas dalam sebuah proses yang dipakai untuk memperlihatkan urutan aktifitas proses. *Activity diagram* dibuat berdasarkan satu atau beberapa case pada use case diagram.

3.3.3.1 *Activity Diagram* Pendaftaran

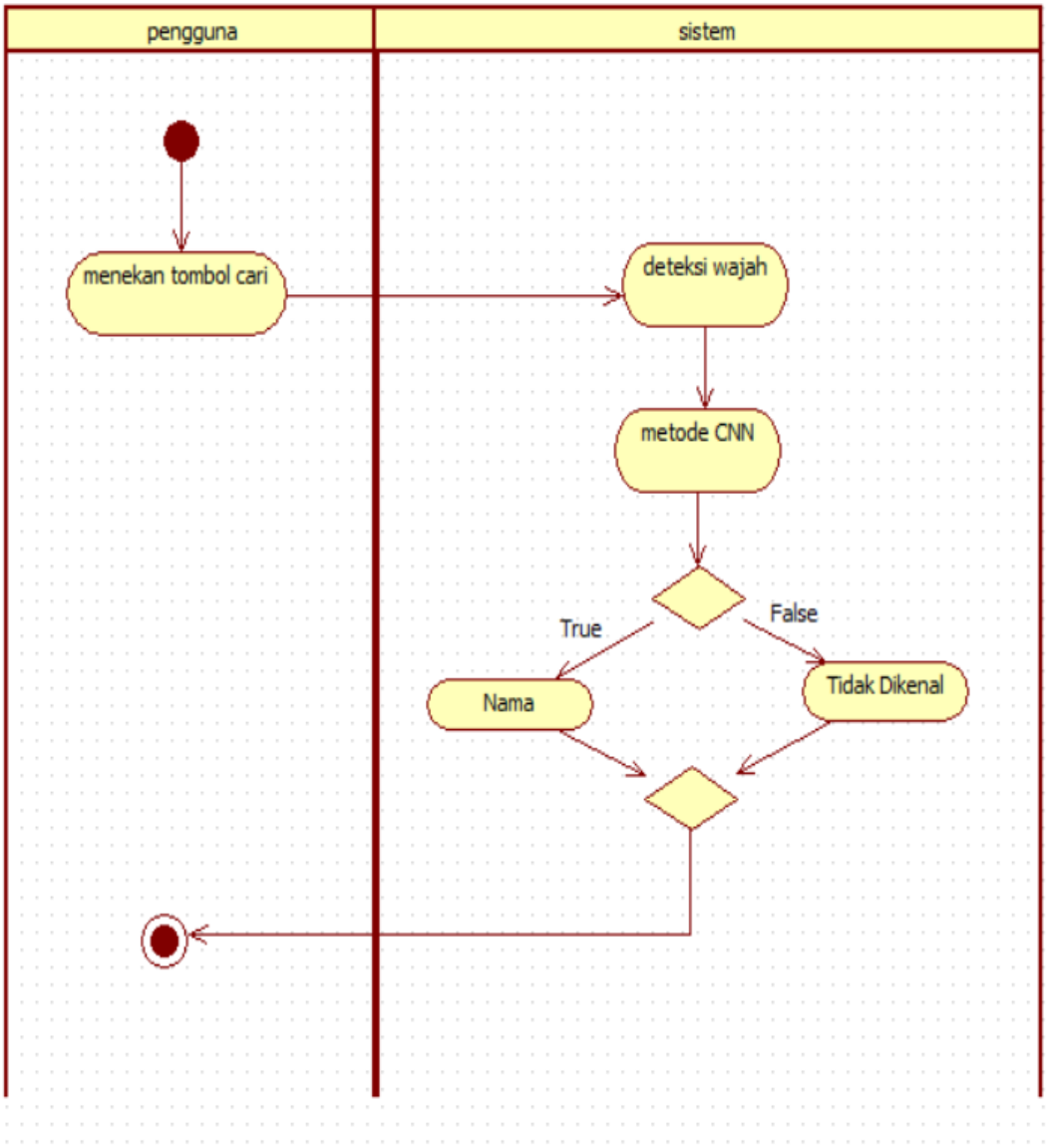
Activity diagram pendaftaran menjelaskan alur kerja aktor yaitu pengguna yang berinteraksi dengan sistem. *Activity diagram* pendaftaran dapat di lihat pada Gambar 3.2.



Gambar 3.2
Activity Diagram Pendaftaran

3.3.3.2 *Activity Diagram* Pengenalan

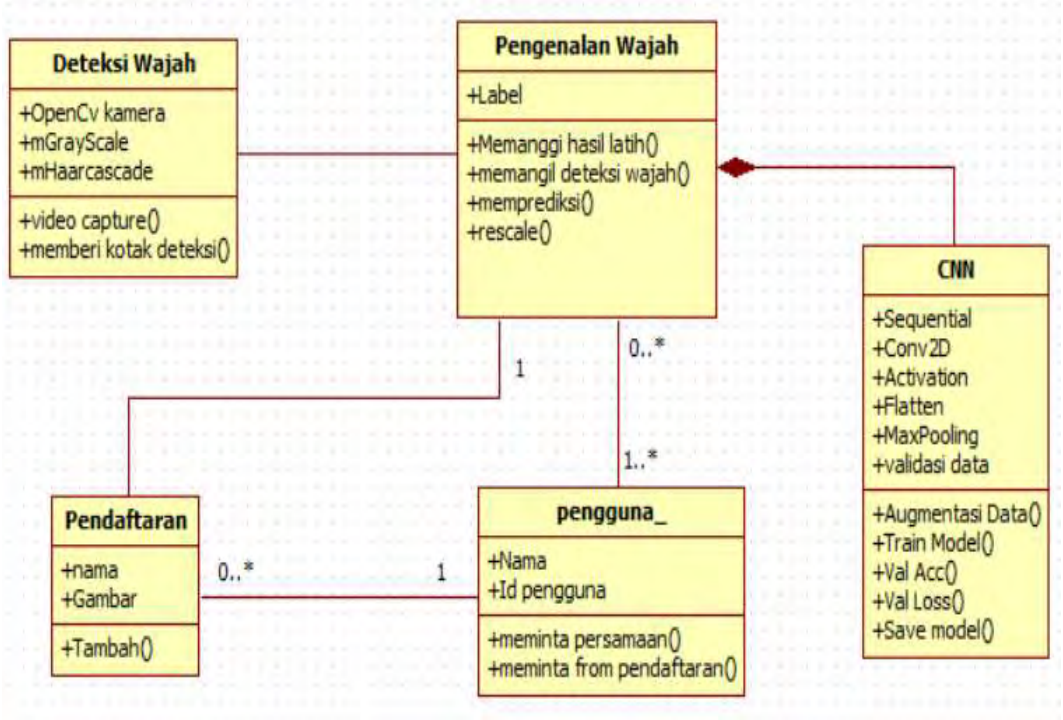
Activity diagram pengenalan wajah menjelaskan alur kerja aktor pengguna pada saat pengenalan wajah. Activity diagram pengenalan wajah dapat dilihat pada Gambar 3.3.



Gambar 3.3
Activity Diagram Pengenalan wajah

3.3.4 *Class Diagram*

Class diagram dipakai untuk menunjukan hubungan sebuah *class* dari sebuah sistem. *Class* diagram memberikan gambaran secara menyeluruh tentang *object* dan *class* serta relasinya pada arsitektur sistem yang dibangun. *Class diagram* yang dibangun dapat dilihat pada Gambar 3.4.



Gambar 3.4
Class Diagram

3.4 Rancangan antar muka

3.4.1 Rancangan antarmuka *Home*

Gamabaran umum dari perancangan antarmuka atau UI f*rame* menu dapat dilihat pada Gambar 3.5. Penjelasan yang ada pada tampilan awal atau form menu terdapat 3 tombol pada form menu yaitu:

1. Tombol pendaftaran

   Berfungsi untuk membuka form pendaftaran.

2. Tombol deteksi wajah

   Berfungsi untuk membuka form deteksi wajah

3. Tombol pengenalan wajah

   Berfungsi untuk membuka form pengenalan wajah

Gambar 3.5
Antar muka awal

3.4.2 Rancangan antarmuka Pendaftaran

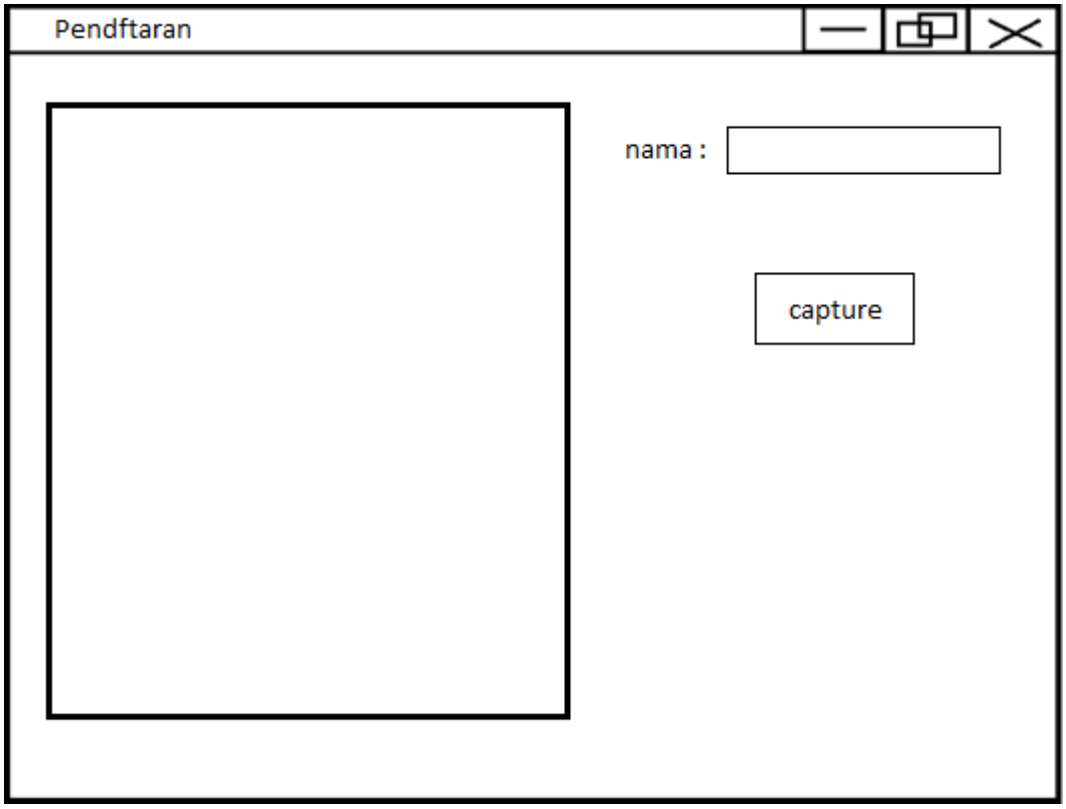
Gamabaran umum dari perancangan antarmuka atau UI form menu dapat di lihat pada Gambar 3.6. Penjelasan yang ada pada tampilan awal atau form menu terdapat 1 tombol pada form menu yaitu:

1. Text field Nama

   Berfungsi untuk memasukan nama untuk gambar yang akan diambil.

2. Tombol Capture

   Berfungsi untuk mengambil gamabar atau foto yang akan di daftarkan untuk pengenalan wajah.



Gambar 3.6
Antar muka Pendaftaran

3.4.3 Tancangan antarmuka Deteksi Wajah

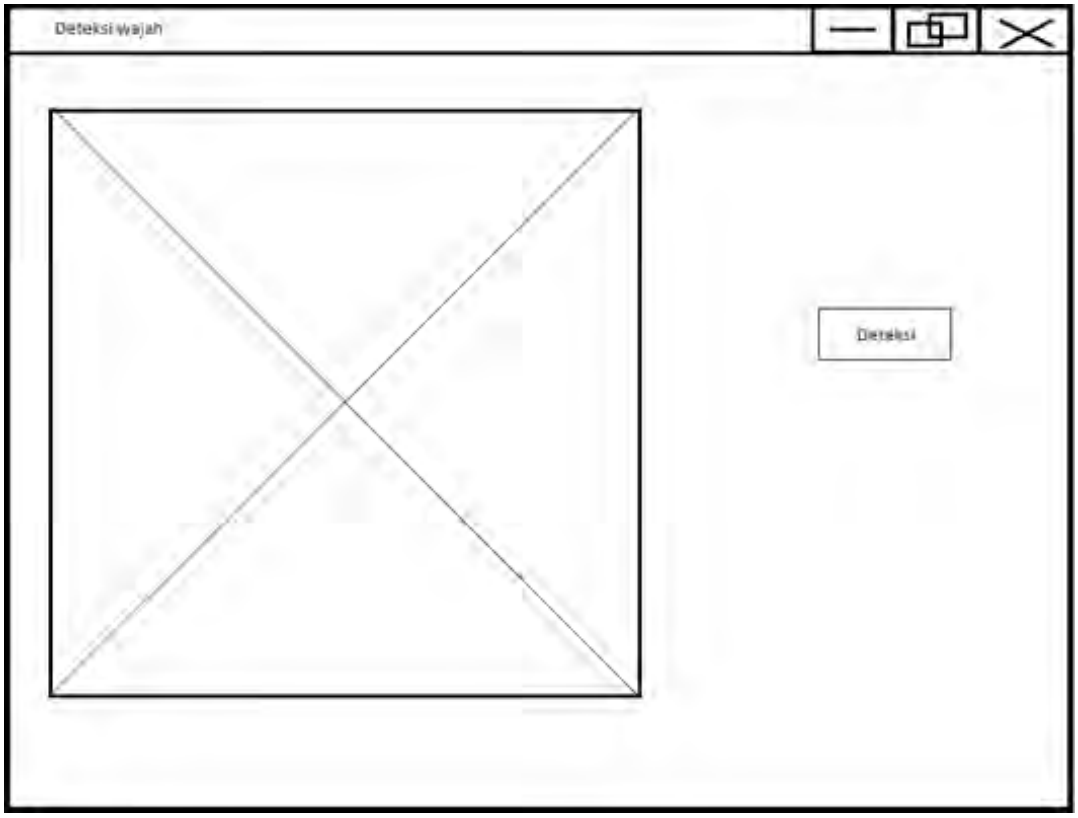
Gamabaran umum dari perancangan antarmuka atau UI form deteksi wajah dapat di lihat pada Gambar 3.7. Penjelasan yang ada pada form deteksi wajah terdapat 2 tombol pada form deteksi wajah yaitu:

1. Tombol deteksi

   Berfungsi untuk mendeksi secara langsung ketika di tekan akan muncul kotak yang menandakan bahwa itu adalah wajah manusia.

2. Kotak besar sebelah kiri

   Berfungsi untuk menampilkan wajah yang akan measuki proses deteksi wajah.



Gambar 3.7
Antar muka Deteksi wajah

3.4.4 Rancangan antar muka Pengenalan Wajah

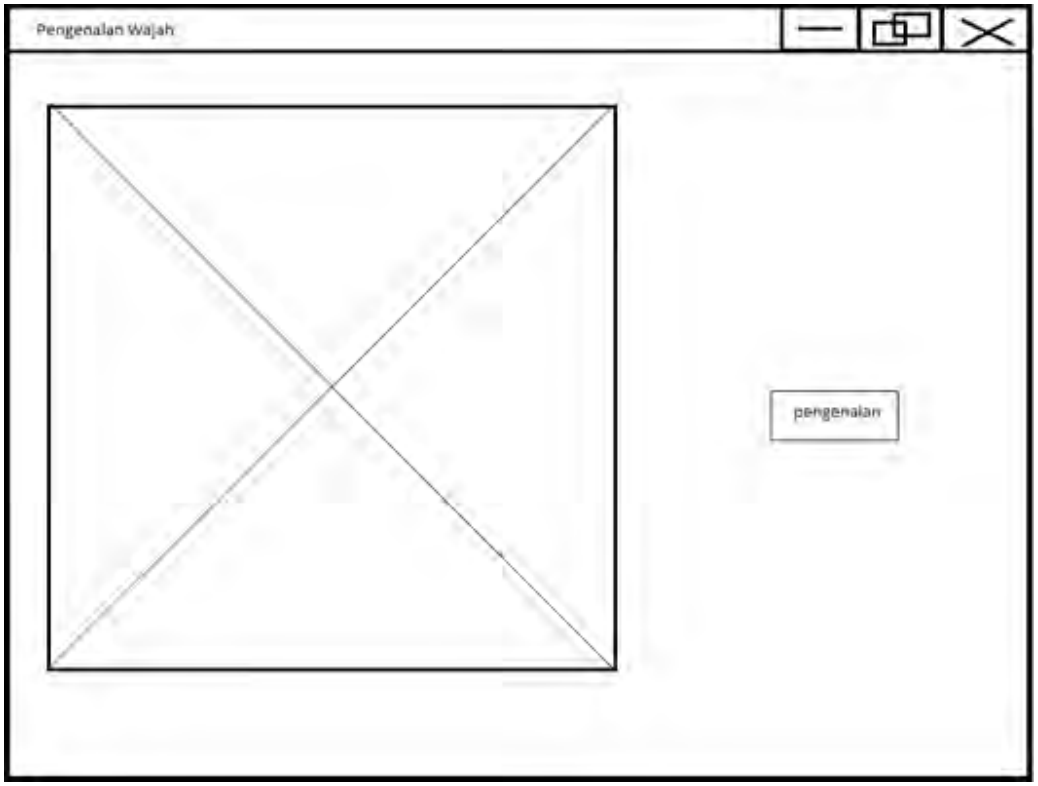
Gamabaran umum dari perancangan antarmuka *form* pengenalan wajah dapat di lihat pada Gambar 3.8. Penjelasan yang ada pada *form* pengenalan wajah yaitu:

1. Tombol pengenalan

   Berfungsi untuk pengenalan wajah.

2. Kotak besar sebelah kiri

   Berfungsi untuk menampilkan wajah akan memasuki proses pengenalan wajah.



Gambar 3.8
Antar muka Pengenal Wajah

BAB IV

IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK

4.1 Spesifikasi Kebutuhan

4.1.1 Spesifikasi Perangkat Keras

Spesifikasi minimum perangkat keras (*hardware*) digunakan untuk menjalankan program aplikasi sebagai berikut:

1. Intel® Core™ i5-7200U (3M Cache, 2.5GHz, up to 3.1GHz)
2. 8GB DDR4 2133 MHz RAM
3. NVIDIA® GeForce® GT940MX with 2GB VRAM
4. VGA Camera
5. 1TB 5400rpm HDD SATA
6. 15.6" HD (1920x1080)

4.1.2 Spesifikasi Perangkat Lunak

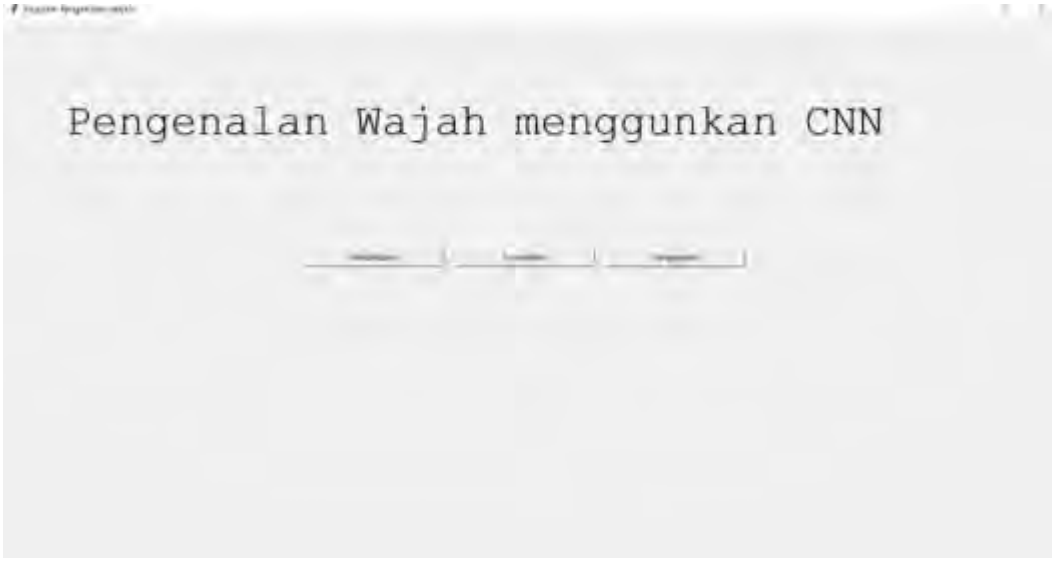
Spesifikasi perangkat lunak (*software*) dibutuhkan agar program aplikasi berjalan dengan baik:

1. Microsoft Windows 10
2. Pycharm

4.2 Pengujian Antarmuka

Pengujian antarmuka dilakukan pada semua tampilan pada program dibangun. Berikut merupakan pengujian pada antarmuka face recognition dibuat:

Pada gambar 4.1 dapat dilihat merupakan tampilan antarmuka pada awal aplikasi. Di dalam halaman menampilkan tiga *button* (tombol). Tombol pendaftaran ditekan akan menampilkan *frame* pendaftaran. Tombol Deteksi wajah menampilkan *frame* Deteksi Wajah. Tombol Pengenalan wajah memanggil *frame* Pengenalan.



Gambar 4.1
Tampilan pada halaman Utama

Gambar 4.2 menampilkan halaman pendaftaran. Halaman ini terdiri dari label nama, satu *textbox* untuk mengisi nama yang akan didaftarkan, dan terdapat tombol ddaftar menampilkan webcam untuk mendaftar.

Gambar 4.2
Tampilan pada Frame Pendaftaran

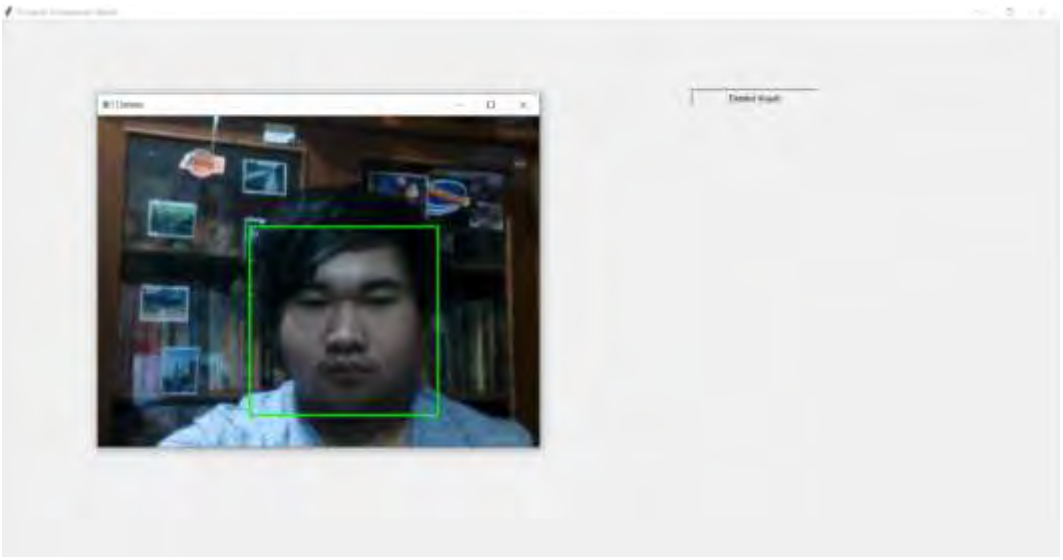
Pada gambar 4.3 merupakan uji ketika satu *texfield* untuk memberi nama gambar yang akan disimpan. Menekan tombol “c” pada keyboard untuk *capture* dalam proses ini dilakukan *capture, cropping,* dan *augmentasi.*

Gambar 4.3
Tampilan ketika menekan *button* Daftar

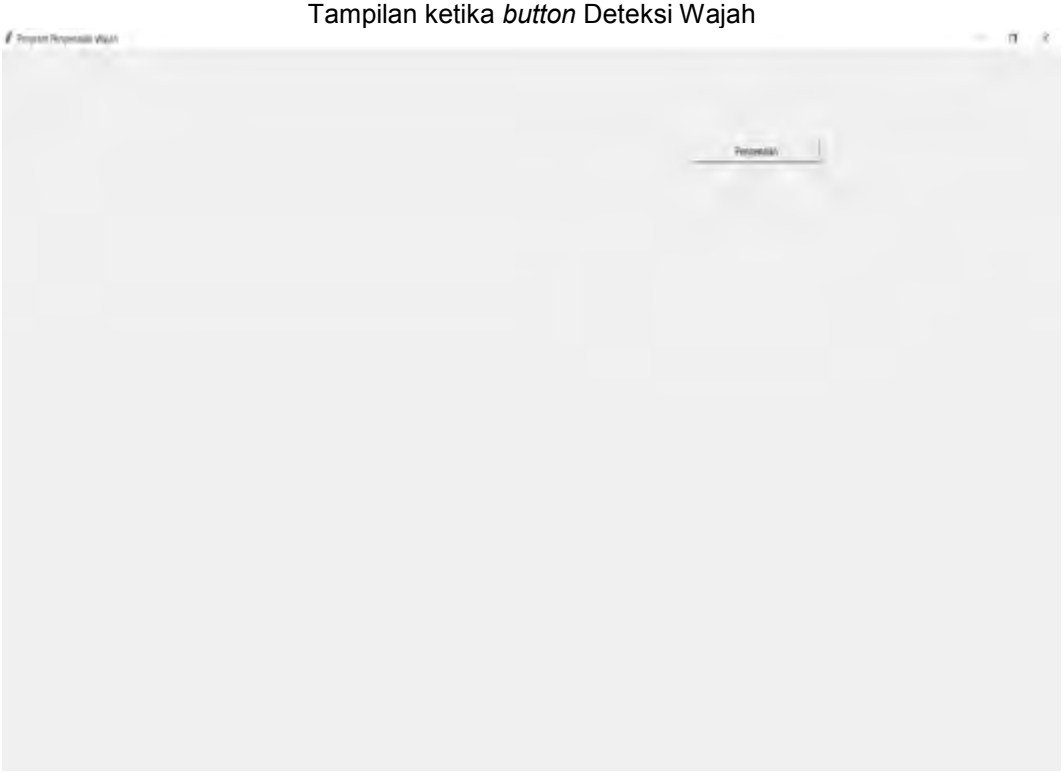
Pada gambar 4.4 merupakan tampilan pendeteksi wajah terdapat satu label untuk menampilkan citra webcam.

Gambar 4.4
Tampilan *frame* pendeteksi wajah

Tampilan pendeteksi wajah dapat berfungsi ketika tombol deteksi wajah ditekan dan akan menampilkan pendeteksi wajah seperti pada gambar 4.5.



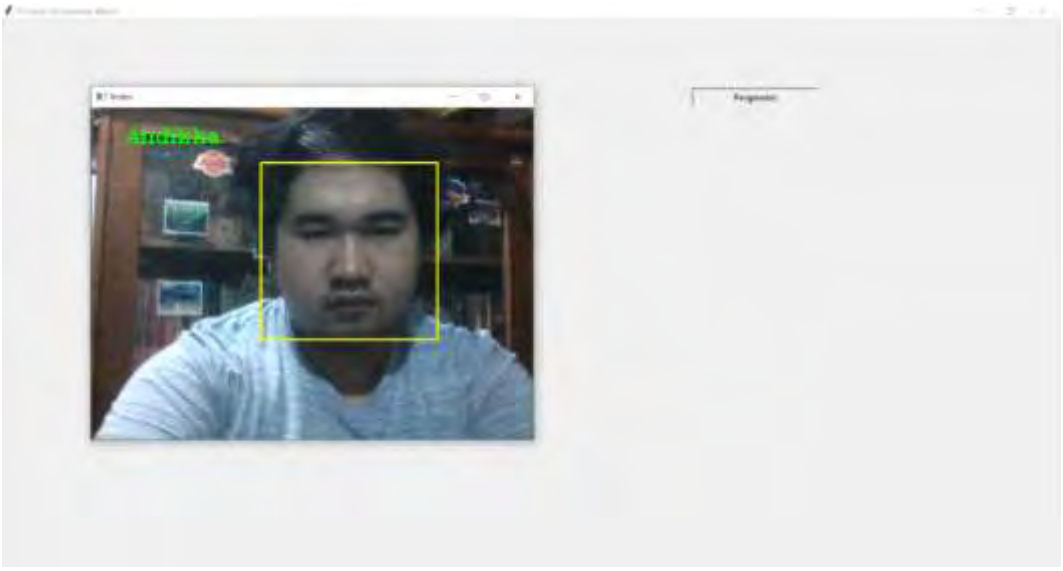
Gambar 4.5
Tampilan ketika *button* Deteksi Wajah



Gambar 4.6
Tampilan *frame* pengenalan wajah

Pada tampilan gambar 4.6 pengenalan wajah terdapat satu tombol pengenalan untuk menampilkan frame video untuk pengenalan.

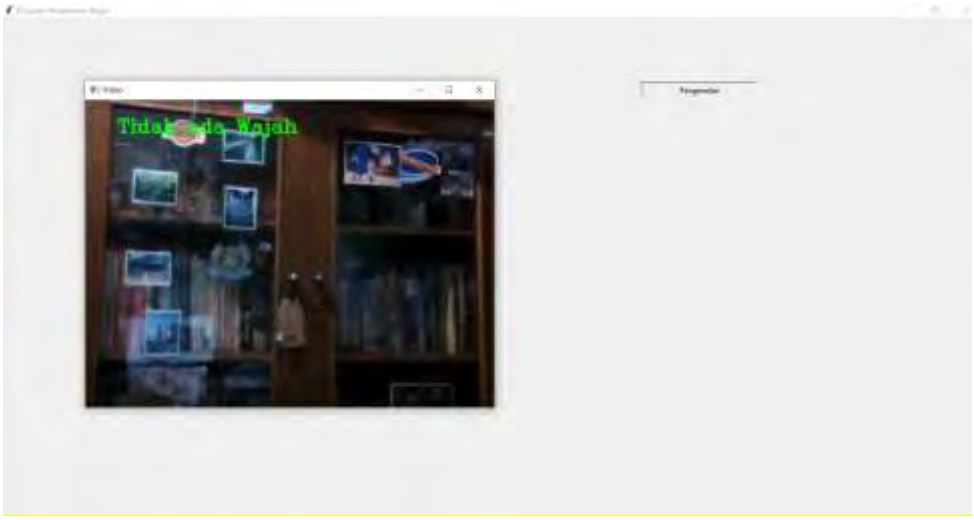
Menampilkan frame pengenalan wajah. Pada gambar 4.7 dengan menguji dengan menggunakan wajah dengan menghadap kedepan.



Gambar 4.7
Tampilan ketika menekan *button* pengenalan

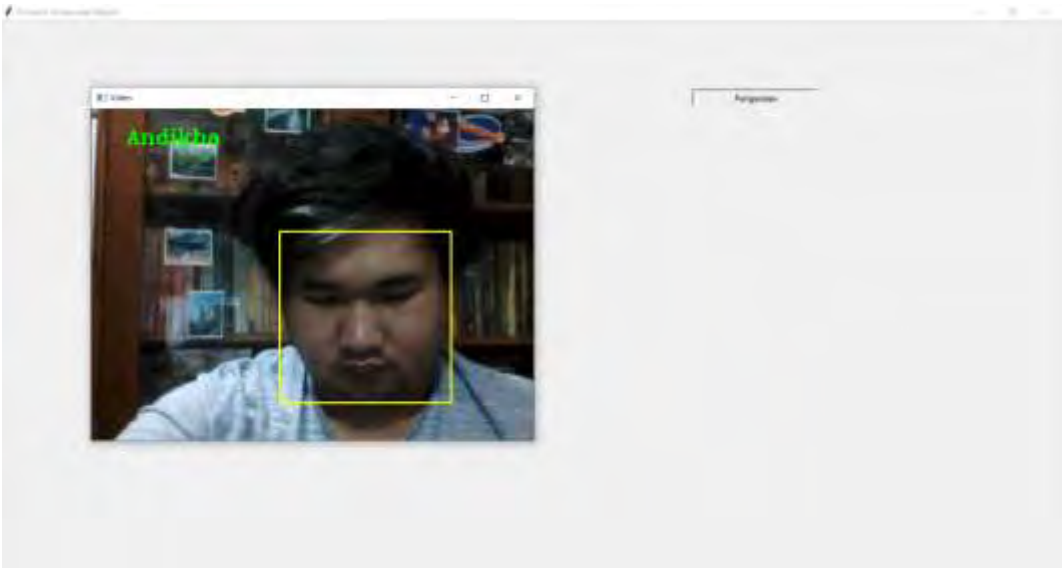
4.3 Pengujian *Face Recognition*

Pada gambar 4.8 merupakan tampilan pada pengenalan wajah ketika diuji dengan tidak ada wajah pada sorotan *webcam.*



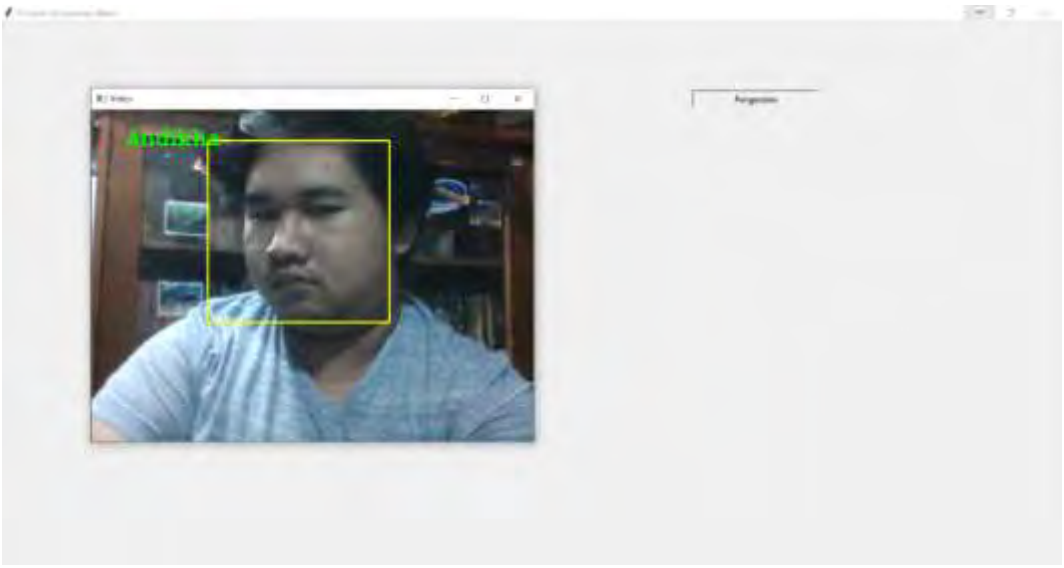
Gambar 4.8
Pengujian tidak ada wajah

Pada gambar 4.9 dengan menguji wajah dengan mengarah ke bawah (nunduk) dapat terlihat dapat dikenali.



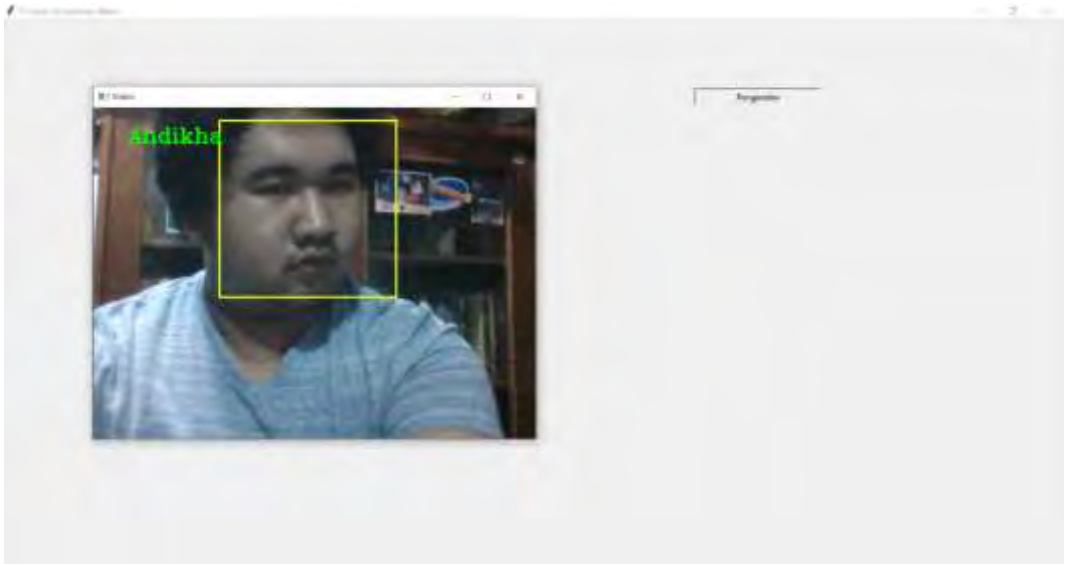
Gambar 4.9
Pengujian Wajah mengarah ke bawah

Pada tampilan gambar 4.10 pada pengujian dengan wajah menghadap kesebelah kanan. Citra wajah masih dapat dikenali.



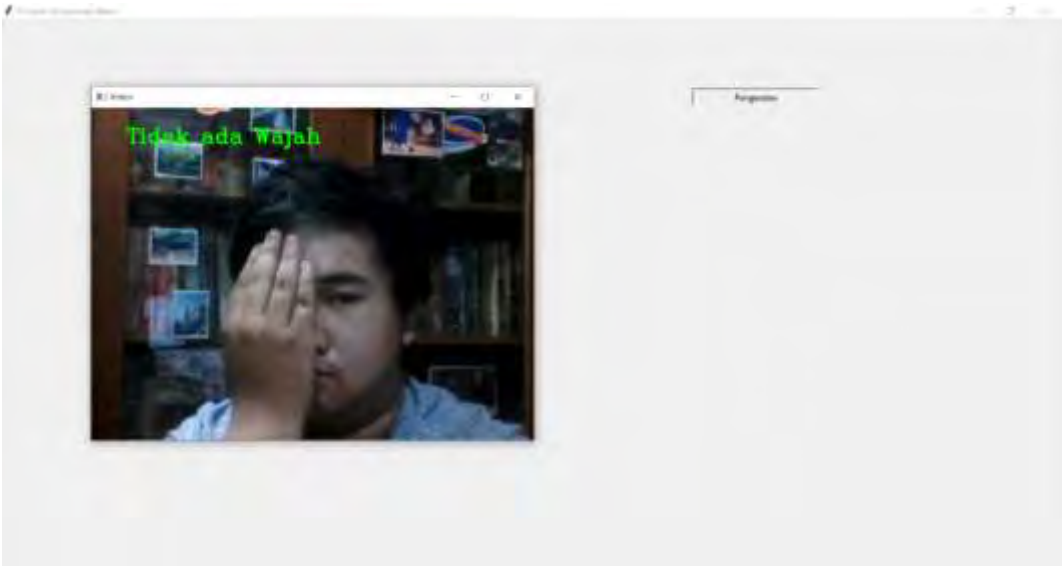
Gambar 4.10
Pengujian Wajah mengarah ke kanan

Pada tampilan gambar 4.11 menampilkan pengujian dengan megarahkan wajah ke sebelah kiri dan dapat dikenali.



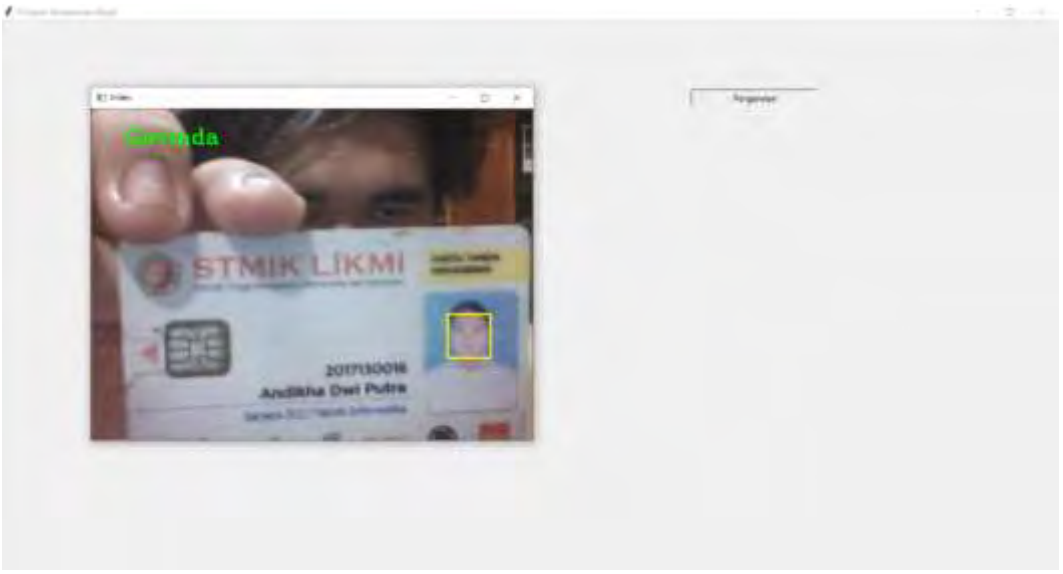
Gambar 4.11
Pengujian Wajah mengarah ke kiri

Pada tampilan gambar 4.12 menampilkan pengujian dengan sebelah wajah ditutup dengan tangan dengan hasil wajah tidak terdeteksi mengakibatkan tidak dapat dikenali.



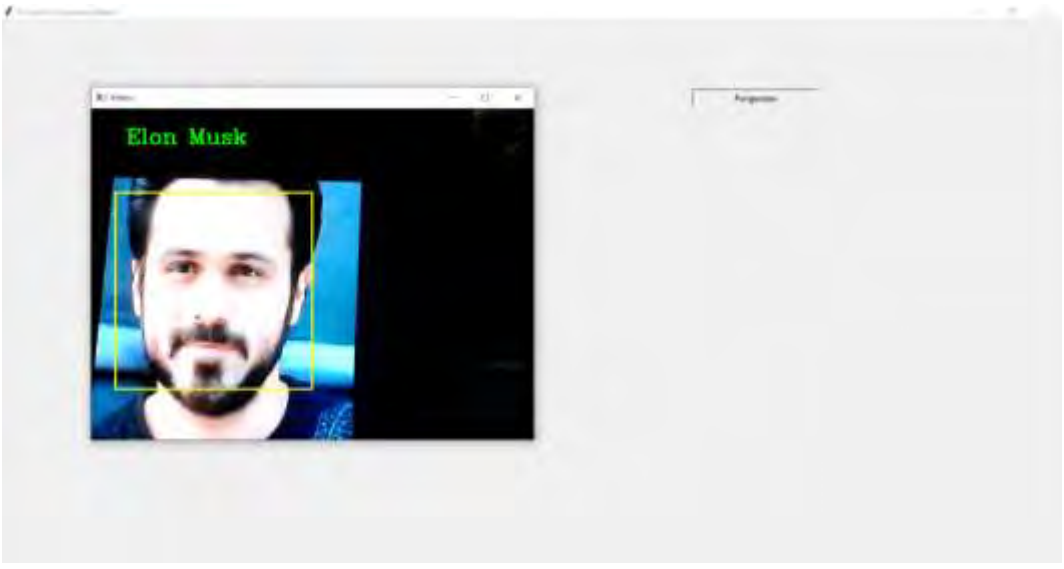
Gambar 4.12
Pengujian Wajah ditutup sebelah dengan tangan

Pada tampilan gambar 4.13 menampilkan pengujian terhadap foto gambar pada kartu mahasiswa dengan hasil dapat dikenali.



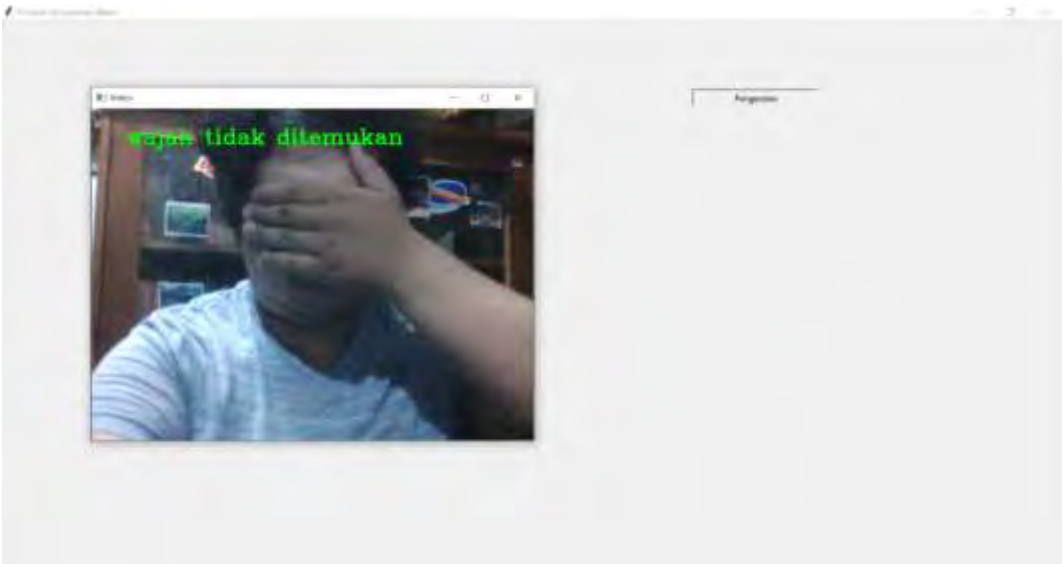
Gambar 4.13
Pengujian dengan Kartu Pengenal Mahasiswa

Pengujian pada gambar 4.14 menampilkan pengujian dengan pengenalan wajah dengan menampilkan wajah yang berada pada *Hand Phone* dengan hasil prediksi tidak cocok.



Gambar 4.14
Pengujian terhadap gambar wajah terdapat di *Handphone*

Pengujian pada gambar 4.15 menampilkan pengujian dengan pengenalan wajah dengan menutup mata dengan tangan dengan hasil wajah tidak terdeteksi.



Gambar 4.15
Pengujian dengan kedua mata ditutup dengan tangan

4.4 Pengujian Fungsi

Pengujian yang digunakan untuk menguji sistem menggunakan pengujian *black box*. *Black box* menganalisis kebutuhan Perangkat lunak. Menentukan hasil yang diharapkan (seharusnya).

4.4.1 Pengujian Halaman Utama

NO	Data Uji	Hasil diharapkan	Output	Kesimpulan
1	Klik tombol pendaftaran	Menampilkan frame pendaftaran	Dapat menampilkan frame pendaftaran	diterima
2	Klik tombol pendeteksi	Menampilkan frame pendeteksi wajah	Menampilkan frame pendeteksi wajah	diterima
3	Klik tombol pengenalan	Menampilkan frame pengenalan wajah	Dapat menampilkan frame pengenalan wajah	diterima

Tabel 4.1
Tabel Pengujian *Black Box* halaman utama

4.4.2 Pengujian Halaman Pendaftaran

NO	Data Uji	Hasil diharapkan	Output	Kesimpulan
1	Klik tombol daftar	Menampilkan webcam	Tampil webcam	diterima
2	Menekan tombol “c” ketika sudah memasukan nama	Data capture webcam disimpan	Capture dengan nama tersimpan	diterima

Tabel 4.2
Tabel Pengujian *Black Box* Halaman Pendaftaran

4.4.3 Pengujian Halaman Deteksi

NO	Data Uji	Hasil diharapkan	Output	Kesimpulan
1	Klik tombol deteksi	Menampilkan webcam display dengan deteksi wajah	Menampilkan webcam display dengan deteksi wajah	diterima
2	Memasukan wajah dengan mengarah ke kiri 90°	Menampilkan nama kemiripan dengan data yang sudah di proses	Tidak terdeteksi wajah	ditolak
3	Memasukan wajah dengan mengarah ke kanan 90°	Menampilkan nama kemiripan dengan data yang sudah di proses	Tidak terdeteksi wajah	ditolak
4	Menekan tombol “c”	Menutup tampilan deteksi	Menutup tampilan deteksi	diterima

Tabel 4.3
Tabel Pengujian *Black Box* Halaman Pendeteksi

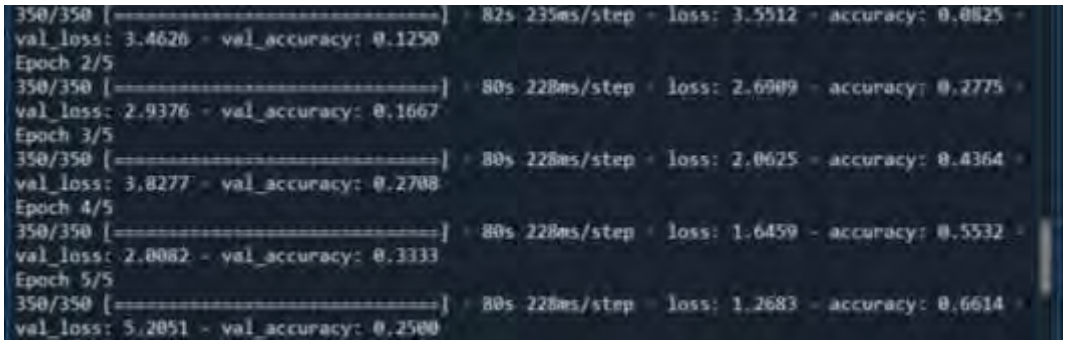
4.4.4 Pengujian Halaman Pengenalan

NO	Data Uji	Hasil diharapkan	Output	Kesimpulan
1	Klik tombol Pengenalan	Menampilkan webcam display dengan deteksi wajah & pengenalan	Menampilkan webcam display dengan deteksi wajah & pengenalan	diterima
2	Memasuka wajah dengan mengarah ke kanan 45°	Menampilkan nama kemiripan dengan data yang sudah di proses	Menampilkan nama kemiripan dengan data yang sudah di proses	diterima
3	Memasuka wajah dengan mengarah ke kiri 45°	Menampilkan nama kemiripan dengan data yang sudah di proses	Menampilkan nama kemiripan dengan data yang sudah di proses	diterima

Tabel 4.4
Tabel Pengujian *Black Box* Halaman Pengenalan

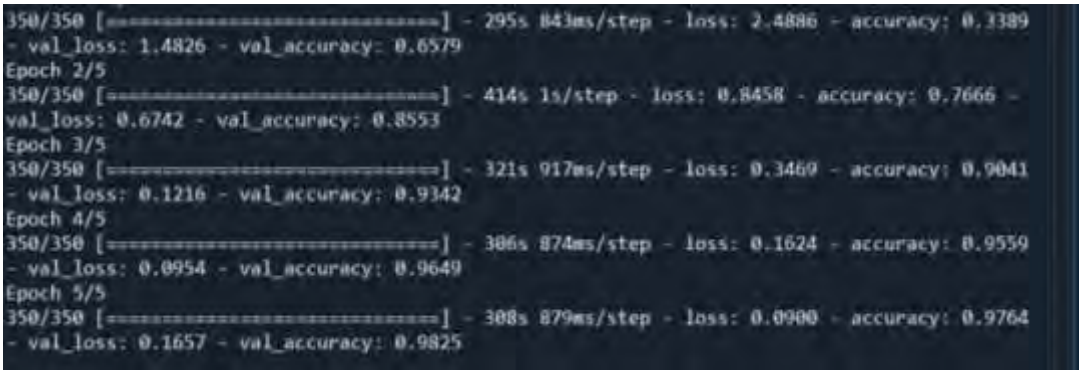
4.5 Performa *Testing*

Pada proses *testing Performa* pertama dengan menggunakan 36 orang dengan jumlah 16.704 mendapatkan hasil 66%. Pengujian performa dengan menggunakan GPU. Dengan *epoch* 5 dan *step per epoch* 350.



Gambar 4.16
Proses testing performa 36 orang total 16.704 data

Pada proses testing dilanjutkan masih dengan menggunakan 36 orang dengan jumlah diperbanyak dengan teknik augmentasi. Dengan data total 90.000 wajah. Dengan tidak merubah *epoch* dan *step per epoch*. Dapat dilihat pada gambar 4.18 dengan hasil 97%.



Gambar 4.17
proses testing performa 36 orang dengan total 90.000 data

BAB V

KESIMPULAN DAN HASIL

5.1 Kesimpulan

Berdasarkan dari hasil implementasi dan pengujian yang sudah dilakukan, maka dapat menarik kesimpulan dari tugas akhir yang berjudul "*Implenentasi Convolutional Neural Network Untuk Face Recognition*" sebagai berikut:

1. Pada tugas akhir ini dengan mengunakan metode *Convolutional Neural Network* dan dengan melakukan augmentasi untuk memperbanyak data training, maka dapat tingkat akurasi sebesar 97%.
2. Pengujian terhadap wajah yang terdapat dalam kartu pengenal dan wajah yang ditampilkan dilayar *Handphone* tidak benar memprediksi wajah yang dikarenakan pengenalan wajah merupakan sistem *biometric* yang berarti sistem bersifat unik dan tidak bisa di palsukan.

5.2 Saran

Dalam pembangunan ini, adapun saran diajukan untuk menjadi masukan dalam Implenetasi *CNN* untuk pengenalan wajah sebagai berikut:

1. Gunakan data set atau data yang diuji dengan menggunkan jumlah yang banyak karena akan berpengaruh terhadap hasil *training* dengan minimal per kelas 5000 data gambar.
2. Untuk pengembangan selanjutnya, perlu dilakukan penelitian lanjut dengan sistem pendeteksi wajah yang mampu meng*capture* wajah dengan lebih baik seperti menerapkan dengan pendeteksi *MT-CNN*.
3. Penelitian selanjutnya dapat dikembangkan ke dalam pengaplikasian seperti mengenali penjahat atau pelaku kriminal dan absensi kehadiran.

DAFTAR PUSTAKA

Alskeini, N. H., Thanh, K. N., & dkk. (2018). Face recognition: Sparse Representation vs. Deep Learning. *ICGSP'18: Proceedings of the 2nd International Conference on Graphics and Signal Provessing*, 31-37.

anonim. (2020, April 28). *About*. Retrieved from OpenCV: https://opencv.org/about/

Arifin, M., Bejo, A., & Najib, W. (2017). Integrasi Login Tanpa Mengetik Passwordpada WordPress. *JNTET*, 162-167.

Datta, A. K., Datta, M., & dkk. (2015). *Face Detection and Recognition.* CRC Press.

Ferraz, C. T., Saito, J. H., & dkk. (2018). A comprehensive analysis of Local Binary Convolutional Neural Network for face recognition in surveillance video. *webMedia '18: Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*, 265-268.

Gan, Y. (2018). Facial Expression Recognition Using Convolutional Neural Network. *ICVISP 2018: Proceedings of the 2nd International Conference on Vision, Image and Signal Processing*, 1-6.

Gollapudi, S. (2019). *Learn Computer Vision Using OpenCv: With Deep Learning CNNs and RNNs.* Apress.

Graupe, D. (2019). *Priciples of Artificial Neural Networks.* World Scientific.

Gulli, A., Kapoor, A., & dkk. (2019). *Deep Learning With TensorFlow 2 and Keras.* Packt.

Iliadis, L., Maglogiannis, i., & dkk. (2012). *Artificial Intelligence Applicaitions and Innovations.* Springer.

Kendall, K. E., & Kendall, J. E. (2014). *Systems Analysis and Design.* Pearson.

Labolo, I. (2019). Implementasi QRCode Untuk Absensi Perkuliahan Mahasiswa Berbasis Paperless Office. *Jurnal Informatika UPGRIS*, 99-102.

Lantz, B. (2013). *Machine Learning with R.* Packt.

Marcel, S., Nixon, M., & dkk. (2019). *Handbook of Biometric Anti-Spoofing: Presentation Attack Detection.* Springer.

Mueller, J. P., & Massaron, L. (2016). *Machine Learning For Dummies.* John Wiley & Sons.

Nugroho, A. (2010). *Rekayasa Perangkat Lunak Berorientasi Objek dengan Metode USDP.* Andi.

Pressman, R. S. (2010). *Software Engineering A Practioner's Approach.* Higher Education.

Rungta, K. (2018). *Tensorflow Learn in 1 Day.* Krishna Rungta.

Shanmuganathan, S., & Samarasinghe, S. (2016). *Artificial Neural Network Modelling.* Springer.

Suyanto. (2018). *Machine Learning Tingkat Dasar dan Lanjut.* Informatika.

Unhelkar, B. (2017). *Software Engineering with UML.* CRC Press.

Wahyono, T. (2018). *Fundramental of Python for Machine Learning.* Gava Media.

LAMPIRAN

Main.py

```
import tkinter as tk
from Pendaftaran import Pendaftaran
from Deteksi import Deteksi
from Pengenalan import Pengenalan
class Program:
    def __init__(self, master):
        self.master = master
        self.master.geometry("1400x720")
        self.master.title("Program Pengenalan Wajah")
        self.label1 = tk.Label(self.master, text="Pengenalan Wajah menggunkan CNN",
font=("Courier", 44))
        self.label1.place(x=80,y=100)
        self.button1 = tk.Button(self.master, width = 25, text = 'Pendaftaran', command =
self.bukaDaftar)
        self.button1.place(x=400, y=300)
        self.button2 = tk.Button(self.master, width = 25, text='Pendeteksi',
command=self.bukaDetek)
        self.button2.place(x=600, y=300)
        self.button3 = tk.Button(self.master, width = 25, text='Pengenalan',
command=self.bukaPengenalan)
        self.button3.place(x=800, y=300)

    def bukaDaftar(self):
        self.bukaDaftar = tk.Toplevel(self.master)
        self.app = Pendaftaran(self.bukaDaftar)
    def bukaDetek(self):
        self.bukaDetek = tk.Toplevel(self.master)
        self.app = Deteksi(self.bukaDetek)
    def bukaPengenalan(self):
        self.bukaPengenalan = tk.Toplevel(self.master)
        self.app = Pengenalan(self.bukaPengenalan)
def main():
    root = tk.Tk()
    app = Program(root)
    root.mainloop()
if __name__ == '__main__':
    main()
```

Training.py

```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
#import save model
import tensorflow as tf
from keras.models import load_model

model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape = (224, 224, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Conv2D(32, (3, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Flatten())
model.add(Dense(units = 128, activation = 'relu'))

model.add(Dense(units = 36, activation = 'softmax'))
model.summary()

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


from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255,
                                  shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

training_set = train_datagen.flow_from_directory('Datasets/Train', target_size = (224, 224),
                                                 batch_size = 10, class_mode = 'categorical')

test_set = test_datagen.flow_from_directory('Datasets/Test', target_size = (224, 224),
                                            batch_size = 10, class_mode = 'categorical')


model.fit_generator(training_set, steps_per_epoch =350, epochs = 5,
                    validation_data = test_set, validation_steps =200)


model.save('./Model/coba17d.h5
```

Pendaftaran.py

```
import tkinter as tk
import os
from keras.preprocessing.image import ImageDataGenerator
import cv2
from tkinter import messagebox

class Pendaftaran:
    def __init__(self, master):
        self.master = master
        self.master.geometry("1400x720")
        self.label1 = tk.Label(self.master, text="Nama :")
        self.label1.place(x=750,y=50)
        self.quitButton = tk.Button(self.master, text = 'Daftar', width = 25, command=self.make)
        self.quitButton.place(x=800,y=100)
        self.label = tk.Label(self.master, text="Pendaftaran Wajah")
        self.label.pack()
        self.entry = tk.Entry(self.master, width = 25)
        self.entry.place(x=800,y=50)
        self.entry.get()


    def make(self):
# directory dan capture
        self.entry.get()
        #print(self.entry.get())
        path = './Capture/' + self.entry.get() + '/'
        print(path)
        os.makedirs(path)
        cap = cv2.VideoCapture(0)
        count = 0

        while (True):
            # mengambil frame-by-frame
            _, frame = cap.read()
            # menampilkan display
            cv2.imshow('Capture', frame)
            if cv2.waitKey(1) & 0xFF == ord('c'):
                count += 1
                print(count)
                path1 = path + str(count) + '.jpg'
                cv2.imwrite(path1, frame)
            elif count == 1:
                break
        cv2.destroyAllWindows()
        messagebox.showinfo("Perhatian", "Capture Berhasil")
        self.Cropping()
        self.augmentasi()
#Cropping
    def Cropping(self):
        face_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        #mengambil gamabar untuk dicropping
        image = cv2.imread('./Capture/'+ self.entry.get() + '/1.jpg')
        faces = face_classifier.detectMultiScale(image, 1.3, 5)
        for (x, y, w, h) in faces:
            x = x - 10
            y = y - 10
```

```
            cropped_face = image[y:y + h + 50, x:x + w + 50]
            face_Crop = cv2.resize(cropped_face, (400, 400))
        #save crop
        save = './Cropping/Crop/1.jpg'
        cv2.imwrite(save, face_Crop)
        messagebox.showinfo("Perhatian", "Cropping Berhasil")


#Augmentasi
    def augmentasi(self):
        data = './Cropping/'
        dirr = './Datasets/Train/'+ self.entry.get() + '/'
        dirr1 = './Datasets/Test/'+ self.entry.get() + '/'
        os.makedirs(dirr)
        os.makedirs(dirr1)


        target_size = (400,400)
        batch_size = 32

        train_datagen = ImageDataGenerator( #featurewise_center = True,
                                    rotation_range = 10,
                                    #width_shift_range = 0.1,
                                    #height_shift_range = 0.1,
                                    #zca_whitening = True,
                                    brightness_range = (0.4,0.99),
                                    rescale = 1./255,
                                    shear_range = 0.1,
                                    #zoom_range = 0.1,
                                    horizontal_flip = True)

        train_generator = train_datagen.flow_from_directory(data,target_size = target_size,
                                              batch_size = batch_size,
                                              class_mode = 'categorical',
                                              subset = 'training',
                                              save_to_dir = dirr)

        test_generator = train_datagen.flow_from_directory(data,
                                             target_size = target_size,
                                              batch_size = batch_size,
                                              class_mode = 'categorical',
                                              subset = 'training',
                                              save_to_dir = dirr1)

        for i in range(0,2000):
          next(train_generator)

        for i in range(0,500):
          next(test_generator)

        messagebox.showinfo("Perhatian", "Data wajah sudah tersimpan")
```

Deteksi_wajah.py

```
import tkinter as tk
import cv2

class Deteksi:
    def __init__(self, master):
        self.master = master
        self.master.geometry("1400x720")
        self.Button1 = tk.Button(self.master, text = 'Deteksi Wajah', width = 25, command =
self.Deteksi)
        self.Button1.place(x=1000,y=100)

    def Deteksi(self):
        face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
        cap = cv2.VideoCapture(0)
        while (True):
            _, frame = cap.read()
            faces = face_cascade.detectMultiScale(frame, scaleFactor=1.5, minNeighbors=5)
            for (x, y, w, h) in faces:
                color = (0, 255, 0)
                stroke = 2
                end_x = x + w
                end_y = y + h
                cv2.rectangle(frame, (x, y), (end_x, end_y), color, stroke)

            cv2.imshow('Deteksi', frame)
            if cv2.waitKey(20) & 0xFF == ord('q'):
                break

        cap.release()
        cv2.destroyAllWindows()
```

Pengenalan.py

```
import tkinter as tk
import cv2
from keras.models import load_model
from PIL import Image
import numpy as np

class Pengenalan:
    def __init__(self, master):
        self.master = master
        self.master.geometry("1400x720")
        self.quitButton = tk.Button(self.master, text = 'Pengenalan', width = 25, command = self.Pengenalan)
        self.quitButton.place(x=1000,y=100)

    def Pengenalan(self):
        self.model = load_model('Model/coba17b.h5')
        face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

        def face_extractor(img):

            faces = face_cascade.detectMultiScale(img, 1.3, 5)

            if faces is ():
                return None
            # Crop semua wajah
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 255), 2)
                cropped_face = img[y:y + h, x:x + w]

            return cropped_face

        video_capture = cv2.VideoCapture(0)
        while True:
            _, frame = video_capture.read()


            face = face_extractor(frame)
            if type(face) is np.ndarray:
                face = cv2.resize(face, (224, 224))
                im = Image.fromarray(face, 'RGB')

                img_array = np.array(im)

                img_array = np.expand_dims(img_array, axis=0)
                pred = self.model.predict(img_array)
                print(pred)

                name = ""
                    if (pred[0][0] > 0.8):
                    name = 'Aamir'
                cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)

                if (pred[0][1] > 0.8):
                    name = 'Abhay'
                cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)

                if (pred[0][2] > 0.8):
                    name = 'Abhishek'
```

```
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][3] > 0.8):
                name = 'Aftab'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][4] > 0.8):
                name = 'Aishwarya'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][5] > 0.8):
                name = 'Ajay'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][6] > 0.8):
                name = 'Akshay'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][7] > 0.8):
                name = 'Akshaye'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][8] > 0.8):
                name = 'Alia'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][9] > 0.8):
                name = 'Ameesha'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][10] > 0.8):
                name = 'Amitabh'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][11] > 0.8):
                name = 'Amrita'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][12] > 0.8):
                name = 'Amy'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][13] > 0.8):
                name = 'Anil'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)
```

```
            if (pred[0][14] > 0.8):
                name = 'Anushka_sharma'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][15] > 0.8):
                name = 'Anushka_Setty'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][16] > 0.8):
                name = 'Arjun_Kapoor'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][17] > 0.8):
                name = 'Arjun_Rampal'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][18] > 0.8):
                name = 'Arshad'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][19] > 0.8):
                name = 'Asin'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][20] > 0.8):
                name = 'Ayushmann'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][21] > 0.8):
                name = 'Bhumi'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][22] > 0.8):
                name = 'Bipasha'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][23] > 0.8):
                name = 'Bobby'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][24] > 0.8):
                name = 'Deepika'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][25] > 0.8):
                name = 'Disha'
```

```
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][26] > 0.8):
                name = 'Emraan'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][27] > 0.8):
                name = 'Esha'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][28] > 0.8):
                name = 'Farhan'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][29] > 0.8):
                name = 'Govinda'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][30] > 0.8):
                name = 'Hrithik'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][31] > 0.8):
                name = 'Huma'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][32] > 0.8):
                name = 'Ileana'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][33] > 0.8):
                name = 'Andikha'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][34] > 0.8):
                name = 'Bill Gates'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)

            if (pred[0][35] > 0.8):
                name = 'Elon Musk'
            cv2.putText(frame, name, (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,
255, 0), 2)
        cv2.imshow('Video', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    video_capture.release()
    cv2.destroyAllWindows()
```