IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK

By

Ronald Koh Lee Xiang

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2023

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: Identity Prediction with Uncovered Facial Features while Wearing Mask

Academic Session: January 2023

I Ronald Koh Lee Xiang

declare that I allow this Final Year Project Report to be kept in Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:
A18, Pekan Gurney,
32010, Sitiawan,
Perak

Dr. Ashvaany a/p Egambaram
Supervisor's name

Date: 25/4/2023

Date: 26/4/2023

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 25/4/2023

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that ***Ronald Koh Lee Xiang*** (ID No: ***19ACB01567***) has completed this final year project/ dissertation/ thesis* entitled "*Identity Prediction with Uncovered Facial Features while Wearing Mask*" under the supervision of Dr. Ashvaany a/p Egambaram (Supervisor) from the Department of Computer Science, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

(*Ronald Koh Lee Xiang*)

DECLARATION OF ORIGINALITY

I declare that this report entitled "**IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____

Name : Ronald Koh Lee Xiang

Date : 25/4/2023

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr. Ashvaany a/p Egambaram who has given me this bright opportunity to engage in a machine learning project. I also would like to express my appreciation to my family, who has helped me in the project's experiments.

ABSTRACT

Since the Covid-19 pandemic broke out in 2019, our lives had been greatly impacted and problems had arisen from different angles. People must follow a standard operating procedure to control the spread of disease. One of the noticeable changes in behaviour was most people wear a face mask to decrease the infection of the disease. However, the action of wearing a mask had disrupted the usual face recognition process. In this project, a masked face recognition system was developed to tackle the problem mentioned. The task of building a masked face recognition had been broken down into steps, which include face detection, face embedding, face classification, and face verification. Each step was dealt with individually with a specific solution. Dataset acquired for this project includes self-collected data, LFW dataset, CelebA dataset, and GMF dataset. After trial of error though experiments, the final system was developed using OpenCV HaarCascade, FaceNet, SVM, and Euclidean distance. The developed system was able to achieve a great performance of 100.00 training accuracy and 99.787 testing accuracy on known identities. While maintaining a high accuracy for known identities, the system had also achieved a low FAR of 0.0152%, 0.0006%, and 0.0038% from CelebA, LFW, and GMF dataset respectively. The time taken for the system to inference a face image was 109.8 millisecond. When implementing the masked face recognition system in webcam, it was able to recognise the known identities while the presented face was unmasked or masked. Moreover, it was also capable of robustly distinguishing known identities with unknown identities. However, the developed system was not completely perfect, it was unable to recognise multiple identities at once in one capture, does not support integration on other devices, and unable to tell whether the face presented is in its physical form or not. Overall, the system had achieved a decent performance at recognising both unmasked and masked face, but further improvements can be implemented onto the system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

λ	Lambda
$\sum$	Summation

LIST OF ABBREVIATIONS

k-NN	k-Nearest Neighbours
LBP	Local Binary Pattern
LCD	Locality Constraint Dictionary
CNN	Convolutional Neural Network
RMFRD	Real-world face recognition dataset
SMFRD	Simulated face recognition dataset
ReLU	Rectified linear unit
SGD	Stochastic gradient descent
BoF	Bag of Features
MLP	Multiplayer perceptron classifier
YOLO	You only look once
EDA	Exploratory Data Analysis
HOG	Histogram of Oriented Gradients
SVM	Support Vector Machine
MTCNN	Multi-Task Cascaded Convolutional Neural Networks
P-Net	Proposal Network
R-Net	Refine Network
O-Net	Output Network
LFW	Labeled Faces in the Wild Home
CelebA	Large-scale CelebFaces Attributes
GMF	Gender Classified Dataset with Masked Face
RGB	Red Green Blue
SNN	Siamese Neural Network
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
FAR	False Acceptance Rate
Ms	Millisecond
AUC	Area Under Curve

Chapter 1
Introduction

1.1 Problem Statement and Motivation

Nowadays, it is inevitable that everyone is affected by the Covid-19 pandemic. Therefore, it is recommended for us to wear a mask most of the time to protect ourselves and others. However, there is an issue that arises from this pandemic: the effectiveness of face recognition is significantly degraded due to the mask obstructing some of the facial features in face recognition. This issue may affect a lot of people, including smart device users who use face recognition for unlocking their devices; facilities that use face recognition in cameras for detecting suspicious individuals; companies or events that use face recognition for allowing only authorised people. This is why this issue is important because if face recognition cannot be carried out correctly due to an obstructed face when wearing a mask, it would defeat the purpose of using the face recognition system in this pandemic. So, this project was carried out to study the case of face recognition while wearing a mask using artificial intelligence techniques. [1]

The aim of the thesis is to improve the ability of face recognition to recognise both masked and unmasked face. To achieve this aim, a model was trained and built to recognise a person based on their uncovered facial features such as eyes, iris, eyebrows, and forehead. Moreover, to provide convenience to smartphone users so they can unlock their smartphone while wearing a mask. Without the ability to recognise faces on masked faces, an individual must either remove their mask, which exposes them to the dangers of Covid-19 or requires them to use another unlock method to access their device. Therefore, masked face recognition can allow smartphone users to unlock their device without taking off their mask. This can provide safety to them, especially in this pandemic. Furthermore, to provide security measures to facilities that use face recognition to detect suspicious individuals. For example, if a known shoplifter enters facilities, the face recognition system is unable to detect this shoplifter and gives an alert because the shoplifter is wearing a mask. Therefore, this project is motivated to build a model that can carry out masked face recognition to provide convenience and security.

1.2 Objectives

The objective of the project includes using artificial intelligence techniques to develop a masked face recognition model. The face recognition model is aimed at providing reliable performance while maintaining an acceptable recognition speed. This project is mainly focused on maintaining the performance of face recognition when a masked face is provided. Since a masked face contains fewer features compared to an unmasked face, the performance of the face recognition model may not perform as well as it does on recognising unmasked faces. Therefore, it is focused on developing a robust face recognition model that can recognise both unmasked and masked faces.

However, in this project, the final face recognition model developed is in the format of simple files that contain the configuration of the model. The face recognition model is only limited to recognising faces in suitable environments that have been previously integrated. This project does not support the implementation of different devices such as mobile phones and surveillance cameras. Further configuration is required if it is implemented on different systems.

1.3 Project Scope and Direction

The scope of this project is to develop a model in a face recognition system that can:

- Perform face detection on image or webcam
- Extract feature from the uncovered facial region of the person
- Identify the identity of the face detected whether it is unmasked or masked

At the end of the project, the developed face recognition model is capable of identifying the identity of a person from an image or webcam. The novelty of this project is that the system does not only recognise faces that are masked but can also recognise unmasked faces. To solve the problems stated earlier, the masked face recognition system developed can help to recognise faces in the Covid-19 pandemic, where people are wearing masks most of the time.

1.4 Contributions

The purpose of this project is to contribute a masked face recognition model that can predict identity on a masked face. The process of masked face recognition includes a

few artificial intelligence techniques, including image processing, computer vision, and deep learning. As for image processing, the methods and techniques may provide reference and insights for further research on processing the images of faces. Subsequently, the project will explore the vast field of deep learning, which may contribute as a foundation for other researchers or developers working on deep learning as well. Finally, this project can benefit anyone that is interested and passionate about the artificial intelligence field. Since this is a small-scale project, it can act as a beginner-friendly guide for anyone that has just started their journey in the artificial intelligence field.

1.5 Background Information

Face recognition is the process of scanning and confirming the identity of a person based on their unique facial features. Facial recognition systems can be implemented to identify people in photos, videos, or in real-time. There are a lot of handy functions that facial recognition can provide in our lives. Face recognition can be used for criminal detection, unlocking smart devices, finding missing people, securing transactions and much more. It can be understood that face recognition is a handy technology that can help in providing utilities ranging from security to conveniency. [2]

Machine learning is a type of learning that helps software perform decisions and tasks without explicit programming or rules. The main aspect of machine learning is data. The data is entered into the algorithm, and the model is trained and used in the program. Machine learning can be approached in 3 ways, which are supervised learning, unsupervised learning, and reinforcement learning. After the model is trained and tested using the chosen approach, it can proceed to deployment, where the model will be used to observe external input data and make predictions. During the process of prediction, the model will analyse the features of input data and compare them with the data previously learned. Then output will be generated to evaluate the result. [3]

Deep learning is a part of machine learning techniques domain that teaches computers how to think like we humans do, like how we learn from mistakes we make. Many industries, including autonomous driving, biomedical research, electronics, and many

more, can benefit from deep learning. Deep learning has produced remarkable result that outperform the previous techniques. Deep learning models can achieve state-of-the-art accuracy, sometimes surpassing capability of humans. Deep learning models are trained upon labelled data and neural network.[4]

Neural networks are the most crucial component for deep learning algorithms. Neural networks are inspired by imitating how a brain works. Neural networks are structed in form of layers, which are input layers, hidden layers and output layers. Each layer in the neural network has multiple nodes, also called artificial neurons. Each node has its own weight and threshold and is connected to other nodes to make up the whole neural network. If the output for any individual node surpasses its threshold value, that node will transfer data to its connected nodes. Final result from neural network is presented at the output layers. [5]

1.6 Report Organization

The details of this project are divided into five chapters. In Chapter 2, work related to masked face recognition is studied. This aids in finding how others have approached the same problem and helps to narrow down the project scope. In Chapter 3, the overall proposed methods are briefed, including the theory of the algorithms used, information about datasets, system specification, and procedure to implement the system. In Chapter 4, all the experiments done are discussed in detail and comparison is made to determine the optimal approach for a niche problem. Finally, Chapter 5 summarises the overall work and findings of this project.

Chapter 2
Literature Review

2.1 CNN with Local Binary Patterns [6]

(H. N. Vu et al.) had approached the problem of masked face recognition with K-Nearest Neighbours (k-NN) and Local Binary Pattern (LBP). The authors had used RetinaFace to achieve real-time face detection. To balance between inference time and accuracy, the authors had used MobileNet and Feature Pyramidal Networks (FPN) was utilised to produce rich feature images. The author did not only use RetinaFace for face detection, but also face alignment, pixel-wise parsing, and 3D dense correspondence regression. The combination of these 4 factors had an outstanding increment in the detection model training and evaluation. Moreover, RetinaFace also improved the process of face detection by minimising the multi-task loss.

Next, this study used high dimension feature vectors in its face embedding, the high dimensional features contained more detail for differentiating the face. InsightFace which implements ArcFace was used. To recognise between vast number of identities, the output vector of face embedding was 512-dimension. To optimise the process of face embedding, the detected face was resized to 112x112 before passing it to the face embedding algorithm.

By using the data acquired from face embedding in the format of 512-d vectors, K-nearest neighbours (k-NN) was used to train the model for face recognition. Unlike traditional k-NN, the authors computed the dot product and distance of the two vectors. Then, the classification of categories was performed by finding the smallest distance between classes and the largest cos value.

Next, LBP-based voting was implemented to pick the best element among the top 5 entries given. This process was done through attaching a ‘voting layer’ that will provide the final decision. In the masked face recognition, the authors focused particularly on the eyebrow area. For each pair of inputs, the eyebrows were extracted from the images,

and a median filter was applied for image processing. Then, histogram equalisation was performed on the eyebrows to make a comparison between the new and original histogram distribution. Furthermore, the Local Binary Pattern (LBP) was used as a voting model for texture classification and face recognition. Figure 2.1.1 shows that voting was done by applying an operation to every pixel of an image of size 3x3. The value of the centre pixel acts as a threshold for neighbouring pixels. Neighbouring pixels that is higher than the threshold is set to 1, otherwise it is set to 0. Finally, the calculated regional histograms were combined to create the feature vector of an image.

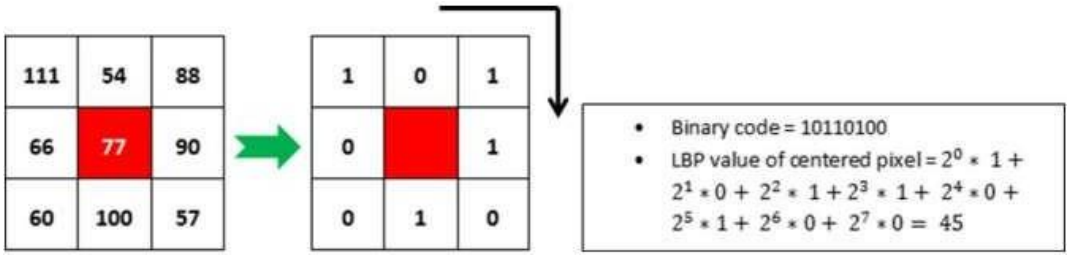


Figure 2.1.1: [6] LBP as voting layer

The datasets used in this study are Essex and COMASK20. In the training stage, the data was split into 85% for training and 15% for testing. The k-value parameters of k-NN used are k = 1,3,5,6,10. The best k-value is 1, performing with a f1-score of 0.95 on COMASK20 dataset and 0.98 on Essex dataset. Furthermore, the authors' proposed method had an AUC value of 0.9.

2.2 Attention Machine Neural Network [7]

(Guiling Wu) had used attention machine neural networks for masked face recognition. Before using the model for prediction, it went through the process of mask separation and feature extraction. In the process of mask separation, the Locality Constraint Dictionary (LCD) Learning Method was used to initially identify the occluded part of the face and separate the occluded part from the face. Next, feature extraction was carried out using dilated convolution and attention mechanisms. While maintaining a fixed view field, dilated convolution was used by the authors to improve the resolution. On the other hand, attention mechanism was used to enhance the region of interest (eyes, eyebrows, and face) the focus on background was reduced through the processing of the feature map with a specified weight. In figure 2.2.1, it can be seen that the attention mechanism is made out of a combination of 3 types of attention mechanisms, which are the spatial, channel, and pyramid attention mechanisms.

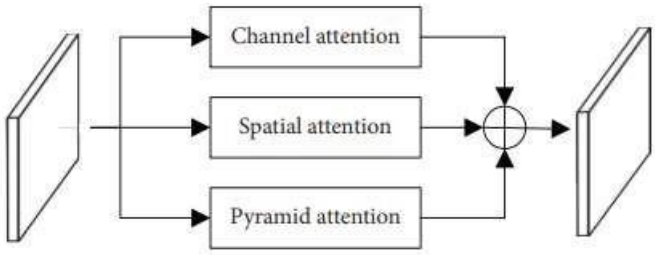


Figure 2.2.1: [7] Structure of attention mechanism

The spatial attention mechanism was derived from the human visual mechanism. For example, the human eye will pay more attention to key locations when seeing an image. Utilising this human behaviour, the spatial attention mechanism follows this principle by associating weight to specific part of the image.

Next, channel attention mechanism was targeted on important features only. The important features compromise of the eyes, eyebrows, and face around the face mask. This focusing mechanism was done by giving weight to the each of the targeted feature. Initially, the feature map was extracted using CNN, the feature map represents parts of the image with different textures and shapes. Then, weight was assigned to these features based on the level of attention.

The pyramid attention mechanism was also derived from human vision, where people tend to distinguish objects based on a variety of information, such as paying more attention to colour than shapes when distinguishing an object. The feature graphs were passed through 1x1, 3x3, and 5x5 convolution kernels and feature information under different perceptual fields was acquired.

In this study, the datasets used were RMFRD and SMFRD. In the SMFRD dataset, the dataset was split exactly in half for both the training and testing sets. The authors achieved remarkable performance on both datasets, with 95.31 and 95.22 accuracy on SMFRD dataset and RMFRD respectively.

2.3 DeepMaskNet for Mask Detection and Masked Facial Recognition [8]

(N. Ullah, A. Javed, M. Ali Ghazanfar et al.) were the founders of the DeepMaskNet, a type of deep learning-based model. The architecture of the model was made out of 6 convolutional layers and 4 fully connected layers. The convolutional layer was used to create feature maps from image. Subsequently, a leaky ReLU activation functions were performed on the feature map. In figure 2.3.1, the sequences of passing through different layers were shown, specifying the activation functions filters, size, stride, and padding.

Sr. No	Layer	Filters	Size	Stride	Padding
1	Input				
2	Convolutional-1 (Batch Normalization +	128	3X3	4X4	0X0
3	LeakyRelu)		3X3	2X2	0X0
4	Max pooling	512	3X3	1X1	2X2
5	Convolutional-2 (Batch Normalization +		3X3	2X2	0X0
6	LeakyRelu)	384	3X3	1X1	1X1
7	Max pooling		3X3	2X2	0X0
8	Convolutional-3 (Batch Normalization +	256	3X3	1X1	1X1
9	LeakyRelu)	256	3X3	1X1	1X1
10	Max pooling	256	3X3	1X1	1X1
11	Convolutional-4 (Batch Normalization +		3X3	2X2	0X0
12	LeakyRelu)				
13	Convolutional-5 (Batch Normalization +				
14	LeakyRelu)				
15	Convolutional-6 (Batch Normalization +				
16	LeakyRelu)				
17	Max pooling				
	Fully Connected + LeakyRelu + Dropout				
	Fully Connected + LeakyRelu + Dropout				
	Fully Connected + LeakyRelu + Dropout				
	Fully Connected				
	Softmax classification				

Figure 2.3.1: [8] Architecture of DeepMaskNet

For the training parameters, the authors trained the DeepMaskNet model using SGD with 0.01 learning rate, size 10 minibatch, and 14 epochs. MDMFR and Kaggle datasets were used in this study. In terms of the result of masked facial recognition, DeepMaskNet had shown great accuracy of 93.33% on the MDMFR dataset.

2.4 BoF Paradigm on CNN [9]

(Walid Hariri) had approached the masked face recognition by implementing BoF paradigm on the last convolutional layer of CNN. The method that this author had proposed consist of four steps. First, it went through the process of preprocessing and cropping. Initially, Dlib-ml was used to detect 68 facial landmarks, and 2D rotation was applied to the images based on the eye locations, and horizontal images were obtained from the 2D rotation. Then, the images were normalised into pixels and divided into 100 fixed-size square blocks. After that, the upper half of the face was obtained by cropping out only the blocks from 1 to 50.

Secondly, it was the feature extraction layer. The author had used 3 pre-trained models for feature extraction, which were VGG-16, AlexNet, and ResNet-50. Different layer was chosen from the pre-trained model to apply the BoF paradigm. The last layer was chosen for VGG-16 and ResNet-50 model, but the fifth convolutional layers was chosen for AlexNet.

Thirdly, it was the deep bag of features layer. The similarity between the extracted feature vectors was measured using RBF kernel. There were 2 sub-layers within the RBF kernel, which were RBG layer and quantization layer. RBF layer was responsible for triggering the RBF neurons using all the extracted features throughout the dataset, the triggering mechanisms can either be manual or automatic. On the other hand, the quantization layer was responsible to generate a histogram with a specified bin number.

Lastly, the histogram computed was passed for classification. To achieve this process, Multiplayer perceptron classifier (MLP) was used for labelling each of the face. Then, back-propagation and gradient descent techniques were used to train the deep BoF network.

The datasets used for this study were the RMFRD and SMFRD datasets. The parameters were varied and included different codewords of 50, 60, 70, and 100. The authors had been able to achieve a good accuracy of 91.3% on RMFRD dataset with the VGG-16 model of 60 codewords, and also achieved a decent accuracy of 88.9% on SMFRD dataset with ResNet-50 of 70 codeword

2.5 YOLO Object Detection Model [10]

(Adarsh Desai) had developed a real-time deployable application, which can recognise identity that was masked or unmasked using You Only Look Once (YOLO) deep learning model. Initially, a dataset of six people was created, where 15 seconds video was taken for each of them. The video consists of the individual looking up, down, left, and right. Then, the video was processed and separated into multiple images. Each image was labelled using the LabelImg tool, where each image was associated with a generated '.txt' extension file as a label.

The author used Exploratory Data Analysis (EDA) to obtain information about the images. The information contained shape and dimension, which displayed the height, width, and dimension of images. Moreover, the values of the RGB (Red, Blue, and Green) layers were calculated and the layers were separated using the NumPy library.

In the training phase, Transfer Learning were used on a model that is pretrained with an unmasked person's face. After that, the author appended masked face dataset to the model and the model learnt how to recognise a person that was wearing a mask. The YOLO model classified an identity by drawing regions on the image, and the region with high score would be considered as detection. In this research, YOLOv3 and YOLOv4, and the mini version YOLOv3 Tiny and YOLOv4 Tiny were experimented.

For evaluating the model's performance, the metrics used were mean average precision (mAP), F1-score, and the detection time. As shown in figure 2.5.1, YOLOv3 and YOLOv4 models were outstanding in providing high F1-score and mAP, while the tiny version of them fell short in these performance metrics. Despite overwhelming high mAP recorded by YOLOv4 model, the author concluded that YOLOv3 was better because it was significantly faster than YOLOv4.

Models	F1-score	mAP (%)	Time taken for detection (sec)
YOLOv3	0.85	88.36	18
YOLOv4	0.86	97.91	14,465
YOLOv3 Tiny	0.74	85.55	11
YOLOv4 Tiny	0.75	75.02	11

Figure 2.5.1: [10] Evaluation of YOLO models

2.6 MobileNetV2 with SoftMax [11]

(Y. M. Saib et al.). In this research, the datasets used were obtained from Pinterest, and MaskTheFace was used to apply a virtual mask onto the images. The authors had used various models for feature extraction and classification. The authors had experimented with HOG, VGG16, and MobileNetV2 for feature extraction. For classification, it was performed by the SoftMax layer and SVM. In figure 2.6.1, all the possible combinations of models were shown.

For the evaluation purpose in this paper, 5 types of datasets were tested to find limitations and determine the best one among them. After training and testing on these different datasets, the authors had determined that the best dataset was the one that consists of only the upper half of the face. The model that had obtained the best result was the MobileNetV2 with its trainable layer set to true and classifier set to SoftMax. This model had achieved an accuracy of 0.9529 in recognising both masked and unmasked faces respectively.

Feature Extraction	Trainable Layer	Classifier
MobileNetV2	True	SoftMax
	False	SoftMax
VGG16	True	SoftMax
	False	SoftMax
VGG16	True	SVM
	False	SVM
HOG	-	SoftMax
	-	SVM

Figure 2.6.1: [11] Combination of models

Chapter 3
Methodology

3.1 Face Recognition Pipeline

Face recognition technique in the computer vision domain that is used to classify or verify an individual by analyzing their face features in images or videos. There are two main types of face recognition, which are face classification and face verification. Face classification perform face recognition in one-to-many relationship with the facial features database. This means that given an input face image of an identity, face classification is intended to determine the person out of all possible identities from the database. An example for this would be a surveillance system, where the system may try to classify a person from the crowd by matching their face against a list of known identities. On the other hand, face verification involves verifying a classified identity, where it asks the question whether the classified identity is really who he/she claims. Face verification is a one-to-one mapping relationship, where the system checks the information stored for this particular identity. For example, when using a facial recognition to unlock a smartphone, the system verifies whether the face presented matches the registered face for the user. [12]

Figure 3.1.1: Face recognition pipeline

In Figure 3.1.1, the overall face recognition pipeline of a face recognition system is illustrated from the beginning to the end. There are three major stages in face recognition pipeline, which are Face Detection, Face Embedding, and Face Recognition. Besides Face Recognition, the output of each stage is used as an input for the next stage. Initially, an input in the form of image or video is passed to the Face Detection stage to detect face, Face Embedding stage then use the detected face to extract features of the face, and finally the features are used in Face Recognition stage to recognise the identity.

In the first stage, Face Detection, the system is used to locate and identify any faces present in an image or video. The algorithm used in this stage identifies the face and crops it out to create a compressed file, which can be used for further feature extraction. There are various algorithms available to perform this task, some of the popular algorithm includes OpenCV Haar Cascade, Dlib HOG, Dlib CNN, and MTCNN. Each face detection algorithm comes with its own strengths and weaknesses, the choice depends on the specific use case of face recognition. [12]

The second stage, Face Embedding, is a crucial step in the face recognition pipeline. This step extracts the unique biological features of a person's face that differentiate them from others. These features are then combined to create a feature vector that represents the person's face. For a pair of different persons, the features extracted from them can never be identical, except for the case of twins. The well-known algorithm used for face recognition today are such as VGGFace, FaceNet, DeepFace, and DeepID. This is the most important step in face recognition system, good feature extraction technique is key to developing a robust face recognition model. [12]

Finally, in the third stage, Face Recognition, the system compares the feature vector of the new sample with those in a facial database to determine a match. This stage is separated into two types, which are face classification and face verification. The face classification is carried out first by determining one identity out of many identities. Then, the face verification step uses the predicted identity to further confirm the correctness of the predicted identity. Based on the type of face recognition system, the algorithm used for Face Recognition can be different. Certain models may perform better at face classification while some models may perform better at face identification. [12]

3.2 Dataset

Dataset is a collection of any format of information, which includes images, videos, numerical, and so on. For a machine learning model to be developed, the model is fed with the dataset and learn from the pattern or structure of the dataset. Different dataset is used for machine learning model in different context. In this project, to develop a robust face recognition system, the quality and quantity of data available in the dataset is critical. Dataset with more variation and comprehensive can greatly impact the performance of the system. [13]

To train a face recognition system, a large and diverse dataset of faces is required. This dataset should include faces from different ethnicities, genders, ages, and poses. The images should also vary in lighting conditions, facial expressions, and backgrounds. It is essential to use a diverse dataset as it enables the system to learn and generalise across a broad range of features and variations. A good quality dataset is critical for producing a reliable and robust face recognition system. A well-curated dataset that has been cleaned can help ensure that the system does not make incorrect identifications. A robust dataset should be free from noise, outliers, and errors, and contain a sufficient number of images to cover the face space effectively.

Since the aimed face recognition system for this project includes the recognition of both unmasked and masked faces, the dataset collected requires the presence of unmasked and masked faces for every identity. For the dataset used in this project, there are mainly two types: known dataset, and unknown dataset. Known dataset compromised of face images labelled with their identity, the face recognition model is fed with these face images to recognise their identities. Unknown dataset are random face images without identity associated to them, these face images are mainly used to evaluate the performance of the trained model.

In the preparation of the known dataset, there are two types of categories associated, which are public dataset and self-collected dataset. Public dataset refers to online dataset that are made available to the community, these datasets may be sourced from other face recognition projects or any system that is developed using that particular

dataset. Subsequently, self-collected dataset are the face images that are captured via a camera from any electronic devices.

On the other hand, the unknown dataset is acquired fully through freely available online dataset. To ensure the trained model evaluate on both unmasked and masked images, the unknown dataset acquired includes various unknown face images that consist of both unmasked and masked faces.

3.2.1 Known Dataset

Self-collected dataset and public dataset are used to represent known dataest. The self-collected dataset contains of three identities. The self-collected face images are captured via a video stream using a laptop's camera, then a face detection is performed on the frame and crop out the face before saving the image to a folder. During the image capturing process, the person in the camera moves their head in multiple direction to simulate different face images. The same process is done twice where the person wears mask for the first time and not wear a mask for the second time.

The public dataset used is the Labeled Faces in the Wild Home (LFW) dataset, which is obtained from an online website. [14] LFW dataset is a public benchmark for face recognition. There are 13,000 images in this dataset with 1680 identities. All images in the LFW dataset have a resolution of 250X250. The images in LFW are captured in various environments, such as different lighting conditions and different backgrounds. The varsity in background environments is useful because it can reflect different scenarios where face recognition systems are put into use. LFW dataset is widely used in most research as a benchmark to evaluate the performance of the face recognition system developed.

3.2.2 Unknown Dataset

For building a robust face recognition model, the model does not only need to be able to recognise identities correctly, but it is also a crucial point for the model not to recognise an unseen identity as any known identity. This can be achieved via splitting the known dataset into training and testing set, but the scope of face images is within

those few identities. For a well-performing face recognition model, it should not overfitting to differentiate between those few identities. The model should also be able to tell apart any face images from outside out the known identities scope. To achieve this, more online dataset is acquired, focusing on obtaining both unmasked and masked face images. The datasets used for this are LFW dataset [14], Large-scale CelebFaces Attributes (CelebA) dataset [15], and Gender Classified Dataset with Masked Face (GMF) dataset [16]. The LFW dataset are reused, where it excludes the identities that are used in known dataset. CelebA dataset is a large-scale face attributes dataset which include 10,177 identities and 202,599 face images. CelebA dataset is commonly used in computer vision and machine learning research for tasks such as face detection, face recognition, and attribute prediction. CelebA dataset includes various facial attributes annotations for each image, such as gender, age, facial expression, and presence of facial hair. GMF dataset is masked and unmasked face images categorized by male and female, this dataset is used in other studies for mask detection, and gender classification through mask. GMF dataset contains 110,157 face images without identity label.

3.2.3 Data Augmentation

Given the dataset mentioned above, the dataset if further extended by using a technique called data augmentation. Data augmentation is a technique that is used to create artificial data out of the existing dataset. The artificial data is created by modifying the original data with a specific data augmentation technique. After a dataset has undergoes data augmentation, the dataset is appended by new data that is similar to the original data. With the modified and expanded dataset, the machine learning model can learn more information from it. Some of the common data augmentation techniques includes of flipping, rotating, scaling, shifting, or adding noise to the original data. [17]

The purpose of data augmentation is to let the machine learning model to learn on a greater diversity of dataset, which can be effective in improving the performance of model at generalising to new and unseen data. This allows the model to be less prone to overfitting to a pre-defined dataset. When the model is trained on a more diverse dataset that has undergoes data augmentation, the model can learn some new representations of the same dataset. [17]

In the context of face recognition systems, image noises like lighting conditions, pose, expression, and occlusion can be an issue that can degrade the face recognition performance. To mitigate this, data augmentation can be used to artificially produce these specific environments. With the help of data augmentation, a face recognition model is capable of learning how to recognise faces that is placed under different conditions. For example, data augmentation can be used to create new face samples that is has different poses, angles, or lighting conditions, the model that has been fed on the augmented data will be able to take these factors into consideration when predicting an identity in the future.

3.3 Face Detection Algorithm

Face Detection is a crucial Computer Vision task that allows a computer program to locate the presence of human faces in an image or video stream. It is the initial stage in most computer vision applications that involve a face. There are various industries seek for a good face detection system, this includes security, marketing, healthcare, entertainment, law enforcement, surveillance, photography, gaming, and so much more. For instance, face detection is used in some security system to look for potential suspects, identity verification, or detecting unauthorized access. Another example of face detection usage can be found in the healthcare industry, where face detection can be used to diagnosing a patient or remote consultations. [18]

Despite its significance, Face Detection poses many difficulties and challenges. The ability of a Face Detection system to detect the face can be greatly disrupted any form of occlusion, lighting conditions, skin color, irregular pose or orientation, facial expressions, accessories, facial hair, and modifications done on faces. Occlusion can pose a challenge to face detection algorithm, as it hides out some important parts of the face that are critical for face detection algorithm to correctly detect a face. Changes in lighting conditions can be an issue as well because the algorithm might not be designed or trained to handle the variation in the lighting. Skin color is also a topic of discussion in facial detection, as some of the face detectors were found to be biased towards certain skin colors. The face pose or orientation may also disrupt the face detection algorithm because some algorithms are only capable of detecting face at a specific angle. For example, the face detection algorithm that is only capable of detecting frontal faces may fail to detect rotated or turned faces. Facial expressions need to be taken into account when designing the features of a face or training a deep learning model as the face is unlikely to always be neutral. Additionally, the face may not always be bare as it may be equipped with accessories, facial hair, or modifications. The robustness of face detection algorithm can be greatly affected with all these attachments on the face. Lastly, the distance of face from the camera can also affect the performance of facial detection system. If the face is far away from the camera, the small-sized face presented might pose a difficulty for certain face detection algorithm to successfully detect the face. [18]

To this day, countless face detection models had been developed and it is still ongoing research. Choosing a specific face detection algorithm out of the many is reliant on the use case. For each face detection model, it has its strengths and weaknesses, the suitability of a face detection algorithm depends on the need of its strength the negligibility of its weakness. For example, if the use case involves real-time face detection, a lightweight model might prove to fit the use case better, as it can run quickly on devices with limited processing power. However, if the use case requires accurate face detection, a more complex model would be a better candidate, despite being more computationally expensive. Due to the time limitation of this project, only three face detection models are experimented, which are OpenCV HaarCascade, MTCNN, and YuNet.

3.3.1 OpenCV HaarCascade

OpenCV Haar Cascade is a popular and efficient face detection algorithm that uses Haar features and Cascade classifiers. The algorithm is based on the Viola-Jones framework proposed by researchers Paul Viola and Michael Jones in 2001. [19] Even though this algorithm has been existed for a long time, it is still one of the popular face detectors today.

Before jumping into the algorithm, it is important to understand the Haar feature. Haar feature is a rectangular region that occupy some area in the detection window, a Haar feature has some information that describe the full detection window. In figure 3.3.1.1, different type of Haar features is illustrated, each targeting different region in the rectangular region. To obtain features for each of these Haar feature, the sum of pixels under the white region is subtracted from the sum of pixels under the black region. The resulting difference of sums is used as the feature for that rectangular area. These features have actual importance in the context of face detection, as certain regions of the face tend to have characteristic brightness values. For example, the eye regions tend to be darker than the cheek regions, and the nose region tends to be brighter than the eye region. [19]

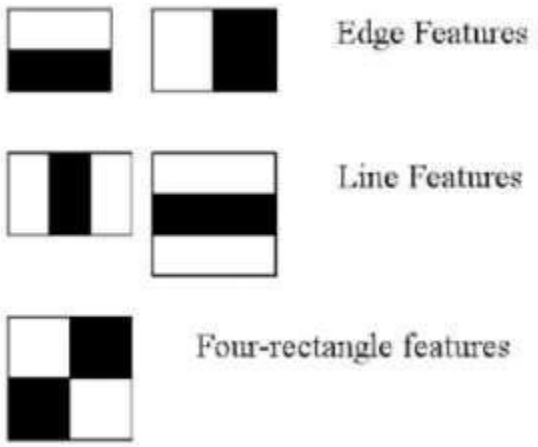


Figure 3.3.1.1: Five Haar features

These Haar features can be computed by sliding a fixed size window through the face image. However, sliding the window along every region of an image and perform complex feature extraction can be computationally expensive, which is not practical. The Viola-Jones algorithm resolve this by introducing the Attentional Cascade algorithm. Instead of executing complex computation on every sub-window, it divides this task into stages. Each stage will perform its specific computation on the sub-windows, reject negative sub-windows found, and pass the remaining sub-windows to the next stage. Throughout the stages, the complexity of each stage increases, so the processing time at the beginning of the stages is fast, and gradually increase along the path. At the beginning, a large number of negative sub-windows are rejected within short processing time. As sub-windows propagate down the line, it becomes harder to detect negative sub-windows, so more complex classifier will be needed to filter them out. By stacking classifier with different complexity in stages, the number of sub-windows need to be processed exponentially decrease, which introduce the quickness of this algorithm. This process of rejecting sub-windows by stages is illustrated in figure 3.3.1.2. [19]

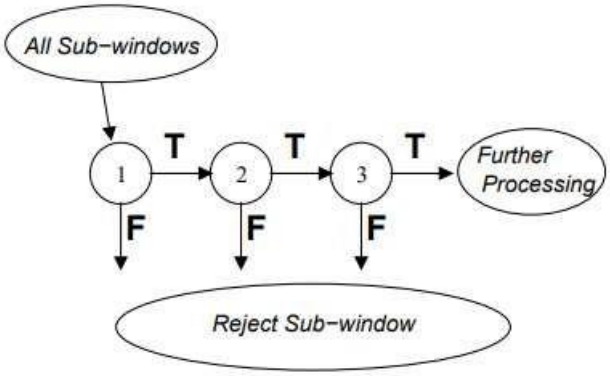


Figure 3.3.1.2: [19] Attentional Cascade algorithm

Finally, if the face image passes all the stages in the algorithm, all the remaining sub-windows are used for last computation. Once the features are computed for each of the rectangular regions, they are combined to form a single feature vector for the entire window. This feature vector is then used to classify the contents of the window as either a face or non-face. [19]

3.3.2 MTCNN

MTCNN (Multi-Task Cascaded Convolutional Networks) is a deep learning model that has achieve state-of-the-art at detecting face. MTCNN was developed by Zhang et al. in 2015. [20] Besides detecting the outer box of a face in image, MTCNN is also capable of detecting and pinpoint facial landmarks. This means that the algorithm can tell specifically where some facial landmark like nose, eyes, and mouth resides on the face detected. MTCNN consists of three stages of neural networks that work together to accurately detect faces, the summary is shown in figure 3.3.2.1 and the detail about the CNN of each stage is in the MTCNN is illustrated in figure 3.3.2.2.

The first stage is called the Proposal Network (P-Net), it resizes the input image to different scales, called image pyramid and proposes candidate bounding boxes for faces. These candidate boxes are then refined using estimated bounding box regression vectors and non-maximum suppression (NMS) to merge highly overlapped candidates. [20]

The second stage, called the Refine Network (R-Net), filters out false positives and further refines the candidate boxes proposed by the P-Net. It performs face classification by formulating a two-class classification problem and uses the cross-entropy loss to calculate the probability that a candidate window contains a face. The R-Net also performs bounding box regression by predicting the offset between each candidate box and its nearest ground truth. The learning objective is formulated as a loss is used to regression problem, and the Euclidean calculate the error between the predicted coordinates and the ground-truth coordinates. [20]

The third stage, called the Output Network (O-Net), performs final refinement of the bounding boxes and facial landmark localization. The O-Net uses the same approach as the R-Net for face classification and bounding box regression. Additionally, it also detects facial landmarks such as the eyes, nose, and mouth corners, using a similar regression approach. The learning objective for facial landmark localization is also formulated as a regression problem, and the Euclidean loss is used to calculate the error between the predicted landmark coordinates and the ground-truth coordinates. [20]

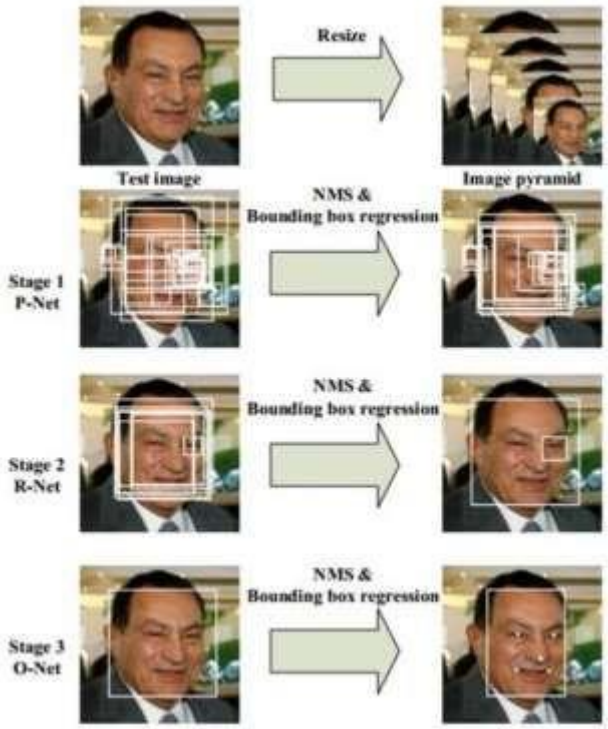


Figure 3.3.2.1: [20] MTCNN 3-stage pipeline

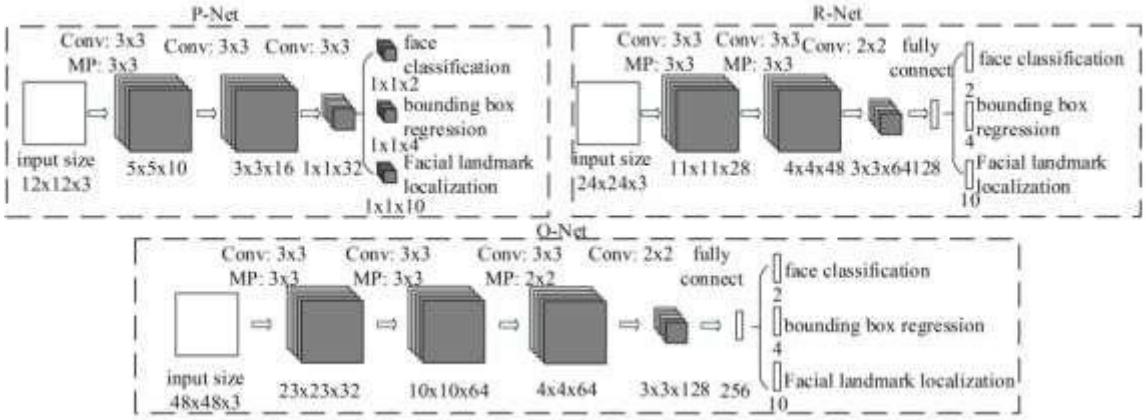


Figure 3.3.2.2: [20] Architecture of P-Net, R-Net, and O-Net

MTCNN is well known for its high accuracy and robustness and is able to detect faces even with different sizes, lighting conditions, and strong rotations. However, MTCNN is slightly slower than some other face detection algorithms, such as the Viola-Jones detector. Nonetheless, MTCNN can still perform efficiently with the help of higher processing power. MTCNN also takes advantage of color information, as the CNNs in the algorithm take RGB images as input.

3.3.3 YuNet

YuNet is a face detector developed by Shiqi Yu in 2018 that is utilising CNN as its underlying architecture. Since YuNet is a face detection model made available to the community recently, there is no research paper that explains the detail about the of the model. Based on [21], YuNet is a powerful lightweight model which can be loaded on many devices. It is said YuNet not only able to reach 1000 frames per second in efficiency but also has high accuracy in performance. YuNet is also famous for its ability to recognize difficult side faces and occluded faces. [21]

To get a picture of how powerful YuNet is in performance face detection, [21] has perform comparison of one of the face detection models mentioned earlier, OpenCV HaarCascade. The significance of the comparison experiment done by [21] is illustrated in figure 3.3.3.1. In detection rate wise, in the 320x320 image size, YuNet was able to make 37 detections, while HaarCascade only make 7 detections, including one false detection. In the 640x640 image size, YuNet was able to make 137 detections, while

HaarCascade only make 29 detections. In computation time wise, YuNet was faster than HaarCascade by 19.46ms and 88.94ms on 320x320 image size and 640x640 image size respectively. The percentage of improvement in computation time of YuNet is roughly 0.79%.

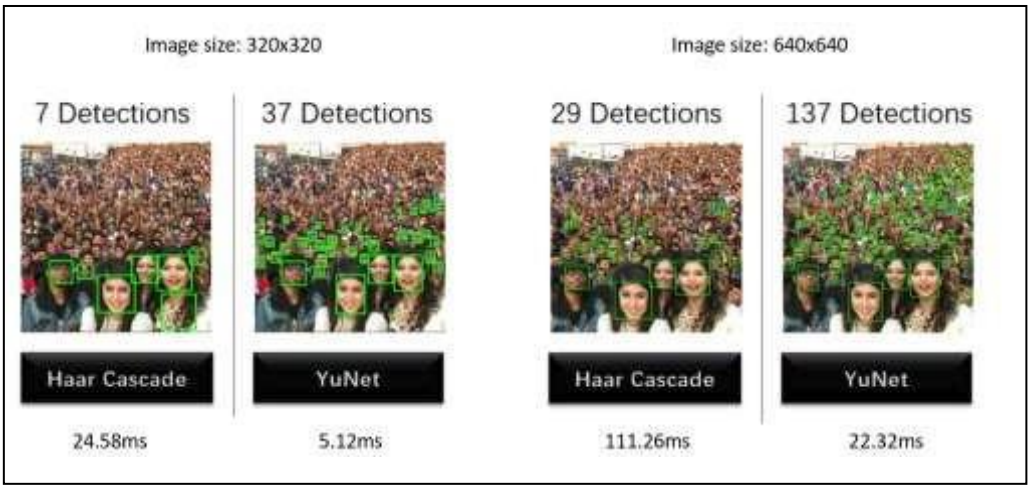


Figure 3.3.3.1: [21] Comparison between YuNet and Haar Cascade

3.4 Face Embedding Algorithm

Face embedding is a crucial step in face recognition, which involves the extraction of features from a face image and transforming it into a compact, numerical representation vector that can be used for matching and recognition. The challenges in face embedding are similar to those in face detection, such as pose variations, partial face recognition, and changing illumination conditions. Traditionally, the thought of using a classification architecture would be sufficient to recognise between different identities. However, the number of classes in the classification approach is fixes, so this approach is no longer feasible with the existence of new class to be detected. Hence, with embedding algorithm, the face image is projected into numerical representation that is dissimilar between each identity. [23]

A face embedding model consists of two main components, which are face feature extraction and loss function. The face feature extraction is the backbone on how features are extracted from face image, while loss function is responsible to penalize the model upon extracting bad features. The amount and quality of data is crucial in producing good face embedding. [24]

3.4.1 Comparison between Face Embedding Algorithms

Face embedding algorithm is the widely researched topic even today, so there are a lot of powerful models that can perform this task. The face embedding models that are explored in this project are narrowed down to VGGFace, DeepFace, FaceNet, and OpenFace. In table 3.4.1.1, the architecture and loss function of the mentioned face embedding models are tabulated. The details of all face embedding models are not discussed in this project. Instead, the comparison of performance of these models are research and the most suitable face embedding model is chosen for the project.

Table 3.4.1.1: Architecture and loss function of face embedding models

Model	Architecture	Loss Function
VGGFace	VGG-Very-Deep-16 CNN	Softmax loss
DeepFace	Deep CNN	Contrastive Loss
FaceNet	Deep CNN	Triplet Loss

OpenFace	Custom CNN	Multiple Loss

According to [25], a comparative study had been performed to evaluate VGGFace, FaceNet, OpenFace, and DeepFace face embedding models. The metrics used for comparing between the model are error rate, accuracy, time, and precision. In figure 3.4.1.1, the summary of comparison of face embedding models using these metrics is shown. The study shows that VGGFace is the best performing face embedding model, placing at the top for every metrics comparison.

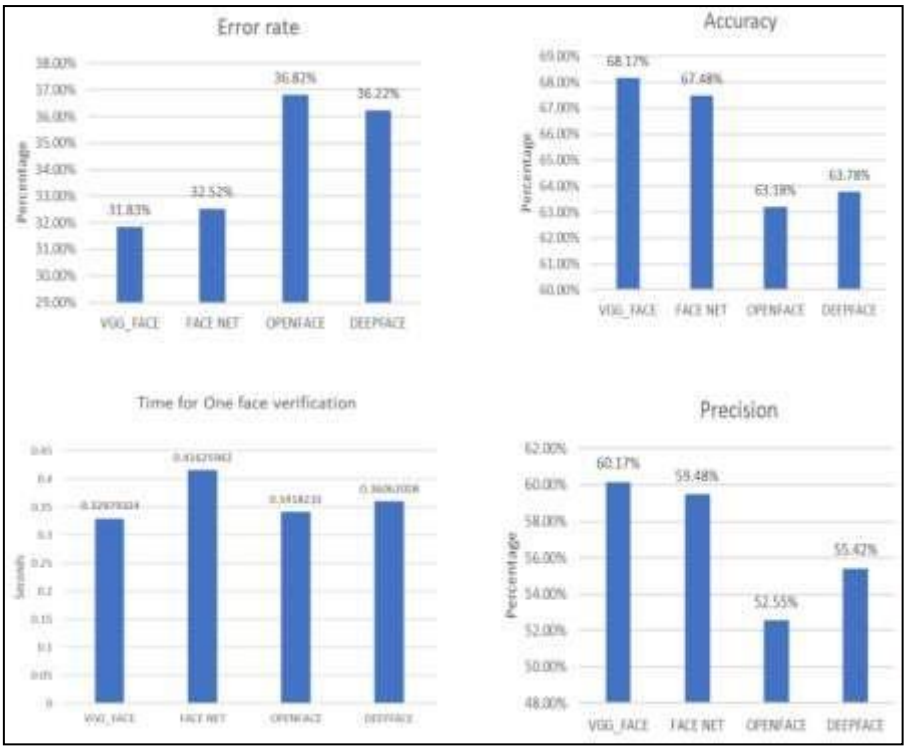


Figure 3.4.1.1: [25] Online study comparison of error rate, accuracy, time, and precision between face embedding models

A simple internal study was also performed to compare between these face embedding models. This process of comparison is eased with the utilisation of deepface, an open-sourced library that compromised of the state-of-the-art models for face recognition. To evaluate the performance comparison, a pair of images of the same identity, as shown in figure 3.4.1.2, is used to evaluate how well each model identity this pair of images as the same person. The metrics considered for this evaluation are distance and

time, where lower distance indicates that the model identity the pair of images to be similar. The result of evaluation is illustrated on the bar chart in figure 3.4.1.3. From the bar chart, it is found that VGGFace perform the best at convincing the pair of images are similar, but it was the slowest among the models. In time computation perspective, the OpenFace model had given the shortest computation time, but fall short on outputting a small distance between the similar identity. Based on these performance metrics on the face embedding models from both online study and personal experiment, the most well-balanced model, FaceNet was chosen to be used for the purpose of face embedding in this project.

Figure 3.4.1.2: Pair of images of the same identity

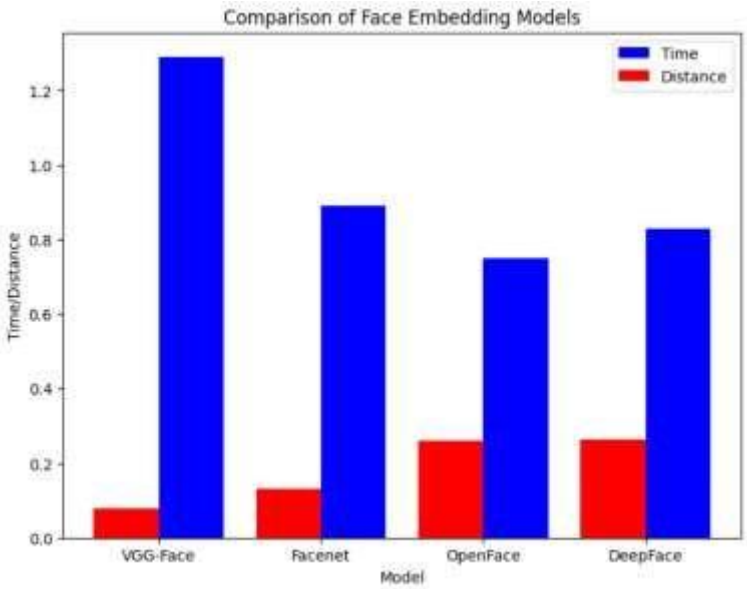


Figure 3.4.1.3: Personal experiment comparison of time and distance metrics between face embedding models

3.4.2 Triplet Loss

With the base understanding of the face embedding algorithm, FaceNet is explained in depth in this section. The magic behind FaceNet algorithm is its loss function, triplet loss. Triplet loss is a loss function used in machine learning algorithms for learning similarity or metric embeddings. The objective of triplet loss function is to tell the model how good or bad the prediction that it has performed and try to improve based on the feedback. [26]

There are three inputs required for forming a triplet in triplet loss function, which are anchor, positive, and negative inputs. Anchor input is used by positive and negative inputs for comparison, the position of anchor itself is immutable. Subsequently, positive input is the same as the anchor input and negative input is different from anchor. Using triplet loss function, the positive input is draw closer to the anchor input, while the negative input is separated further away from the anchor input. [26] The process of learning via the triplet loss function can be visualised in figure 3.4.2.1, where it can be observed that the positive dot moves close to the anchor dot while the negative dot move further from the anchor dot after learning.

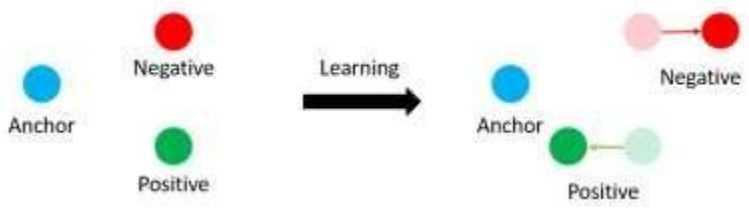


Figure 3.4.2.1: Triplet loss learning, move positive close to anchor, move negative further from anchor

The mathematical equation of triplet loss function:

$$Loss = \max(d(a,p) - d(a,n) + m)\,, 0) \quad (1)$$

where

d is distance between two points

a is anchor image

p is positive image

m is margin

Initially, the triplet loss equation calculates the distance between anchor image with positive image (*d(a,p)*) and distance anchor image and negative image (*d(a,n)*). The desired output would be low *d(a,p)*, and high *d(a,n)*. In the case where *d(a,p)* is lower than *d(a,n)*, the loss computed would be negative. The model does not learn anything important from a negative loss, so the computed value is wrapped around a max function with another value as 0. By doing so, the function will take the value 0 when computed value is negative. From this point, a margin is added to the loss computed to enforce fine tuning to the loss function. To understand the role of margin in this equation, take an example when the computed loss is 0, which may indicate the model is performing well. By adding a margin to the loss, it explicitly tells the loss function that simply computing a 0-loss value is not good enough. The margin value may play a role in hyperparameter fine tuning to ensure the embedding algorithm performs better at a specific use case. [26]

3.4.3 FaceNet

Now that the triplet loss function of FaceNet is covered, this section will explain in detail about how the FaceNet algorithm is trained. FaceNet is a deep learning model for face recognition that was developed by Google researchers in 2015. FaceNet is built upon the Siamese Network architecture that is effective at conveying the similarity between pair of images. FaceNet is used to generate a 128-dimension vector containing crucial face features from the input face image. This vector is known as an embedding. The term "embedding" refers to the fact that all the important information from the image is embedded into this vector. In figure 3.4.3.1, the overall procedure on how FaceNet extract embedding from face is shown. [27]

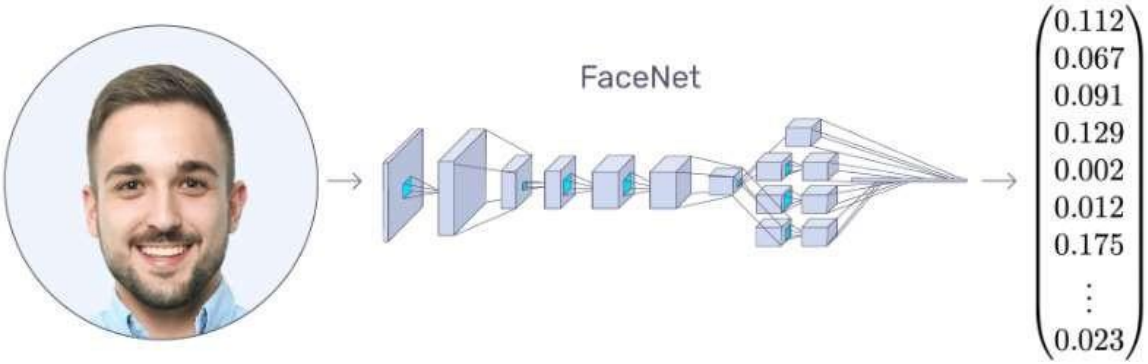


Figure 3.4.3.1: [28] FaceNet extract 128-dimension vector from a face image

The goal of FaceNet is to learn a mapping from the space of face images to the space of face embeddings, such that embeddings of similar faces are also similar. This is achieved through a deep neural network architecture that learns to extract features from the input image and compress them into a compact representation. The architecture of FaceNet includes multiple layers of (CNNs) and fully connected layers. The input image is first passed through several convolutional layers that learn to extract high-level features such as edges, corners, and textures. The resulting feature maps are then passed through several fully connected layers that learn to represent the features in a compact way. [27]

As mentioned earlier, the magic behind FaceNet is how the model learn using the triplet loss function. FaceNet make use of triplet loss function to learn from good and back face embedding, but there is a catch when selecting the triplet to train the model. If easy triplet is selected to train the model, the model does not learn much from its prediction. Easy triplet refers to selecting easy positive image, where the face is very similar to the anchor image, and selecting easy negative image, where the face is very dissimilar from the anchor image. For training the FaceNet efficiently, a harder triplet should be chosen so that the model can learn how wrongly it is prediction the positive and negative images. A hard triplet means that the positive image remains as the same identity, but it is not that easy to tell that it is the same as the anchor image, and the negative image remains as a different identity, but it is somewhat similar to the anchor image. However, simply selecting the hardest triplet is not feasible because it might cause the model to completely collapse by not being able to make a decision to differentiate between

positive and negative images. To mitigate this, [27] proposed to remain the positives images, but to select semi-hard negative images, where the *d(a,p)* is larger than *d(a,n)*.

Upon completion of training, the FaceNet model is not only limited to generate face embedding of face images that have been trained before, but it is a generalised approach and can generate embeddings for new faces. These embeddings can then be compared to embeddings of known faces to perform face recognition. FaceNet has achieved state-of-the-art performance on several face recognition benchmarks, demonstrating its effectiveness in generating high-quality face embeddings. [28]

3.5 Face Classification Algorithm

The last stage of the face recognition pipeline is Face Classification, which involves determining whether the facial features of a new sample match those of a face in a facial database or not. This process is carried out using various approaches, including Euclidean Distance, Cosine Similarity, SVM (Support Vector Machine), KNN (K-Nearest Neighbor), and ANN (Artificial Neural Network). Due to time limitation of this project, only SVM is used as face classifier due to its popularity in working vector input. On top of SVM, another model that is experimented is Siamese Network, which perform face embedding and face classification in one execution.

Moreover, face verification step is also performed along with face classification. Face verification is responsible for detecting an identity among a number of known classes. Once the identity is classified, face verification is done to add additional confidence on the detected identity by checking through the face images or embedding stored of that particular identity. By doing so, it decreases the chance of model incorrectly predict an identity.

3.5.1 SVM

Support Vector Machine (SVM) is a machine learning algorithm that is commonly used in face classification for face recognition. The basic idea behind SVM is to draw a separation line that differentiate between different class in n-dimensional space. The terminologies for SVM are hyperplane, support vectors, and margin, these terms can be visualised in figure 3.5.1.1. Hyperplane is the key in separating the classes, finding the right hyperplane is the key of a robust SVM. There are two types of hyperplanes, which are linear hyperplane and non-linear hyperplane, these hyperplanes are displayed in figure 3.5.1.2. Basically, linear hyperplane separates the data points using a straight line, while non-linear hyperplane separates data points using a curved or nonlinear boundary. Support vectors are the data points that are the closest to the hyperplane and it closely correlate in deciding the position of hyperplane. Margin is the separation distance of the hyperplane with the closest support vectors, an optimal hyperplane has a maximum margin from support vectors of all classes. [29]

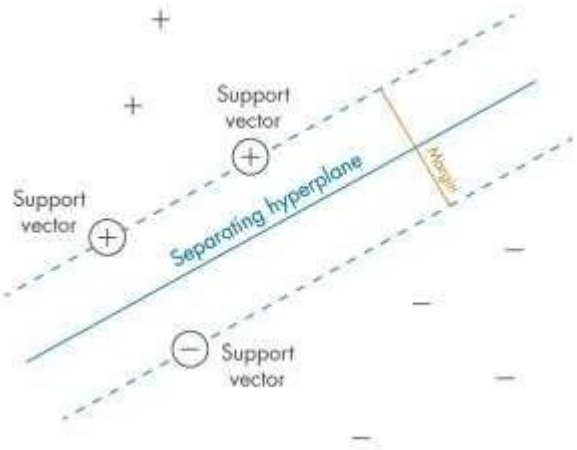


Figure 3.5.1.1: SVM terminologies

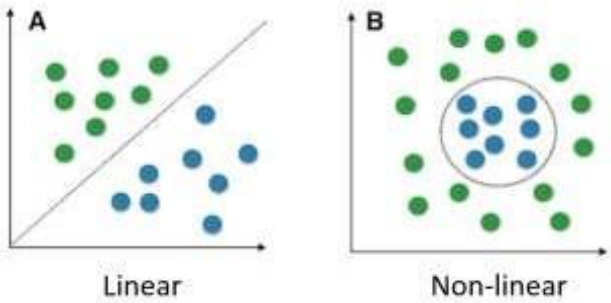


Figure 3.5.1.2: SVM linear and non-linear hyperplanes

There are 3 components that facilitate the training of SVM are loss function, cost function, and regularization [30]. The equations of the stated components are as below:

$$Loss\ function = \max(1 - (y \times f(x))\ , 0) \quad (2)$$

$$Cost\ function = \sum_{i=1}^{n} \max(1 - (y_i \times f(x_i, w)), 0) \quad (3)$$

$$Regularisation = \min_{w} \lambda \times || w ||^2 \quad (4)$$

where

y is the true class label (+1 or -1)

f(x) is the predicted output

w is the weight used for prediction

λ is the penalty parameter

The loss function used in SVM is called hinge loss, where it calculates the degree of error for each misclassified data point. The cost function takes the average of the losses across all the data points in the training set. The regularisation technique introduces a penalty term in the cost function to prevent overfitting of the model. controls the trade-off between the complexity of the model and the degree to which it overfits the training data. A higher value of λ increases the penalty for larger weights, reducing the complexity of the model and improving its generalization performance on unseen data. On the other hand, a lower value of λ reduces the penalty for larger weights, increasing the complexity of the model and its potential to overfit the training data. [31]

After training the SVM model, it can classify new face images into different classes by computing the distance between the input features and the decision boundary. In this project, the SVM is utilised to take 128-dimension vector of face embedding as input to classify the identity.

3.5.2 Siamese Network

Another face classifier used is Siamese network. Siamese network is the underlying architecture in the face embedding model discussed earlier, FaceNet. Instead of only extracting the face embedding from face image, Siamese network is used to perform face embedding and face classification in one run. The idea of Siamese network is to use two identical networks for extracting features from image and compare between their similarity. [32]

Instead of using triplet loss function from the FaceNet architecture, the loss function that is explored on Siamese network is contrastive loss. In triplet loss function, a set of three images are taken at once, which represent the anchor, positive, and negative images. On the other hand, contrastive loss only uses a pair of images, which are an anchor image paired with either a positive image or negative image. If an anchor-positive pair is chosen, the label for that pair would be true. Otherwise, the label for anchor-positive pair would be false. The equation for contrastive loss function is shown below [32]:

$$Loss = (Y)\left(\frac{1}{2}\right)(D_W)^2 + (1-Y)\left(\frac{1}{2}\right)(\max(0, m - D_W))^2 \quad (5)$$

where

Y is the true label

D_W is the distance calculated between two vectors with weight w

m is the margin

Based on the contractive loss function shown in equation (5), it can be noticed that it is somewhat similar to that of triplet loss function. Contractive loss is the predecessor of triplet loss, the learning process only include differentiating between two images. When it is a pair of similar images (*Y=1*), the first half part of the equation is considered, where the higher loss refers to higher distance between similar images. Subsequently, when it is a pair of dissimilar images (*Y=0*), the second half of the equation is considered, where higher loss comes from low distance of dissimilar images. Finally, the margin aims to set a higher bar for the loss of dissimilar images. [32]

In figure 3.5.2.1, the overall architecture of a Siamese network utilising contractive loss function is shown. In this project, contractive loss is covered in the experiment to make a comparison between the loss function used for Siamese network architecture.

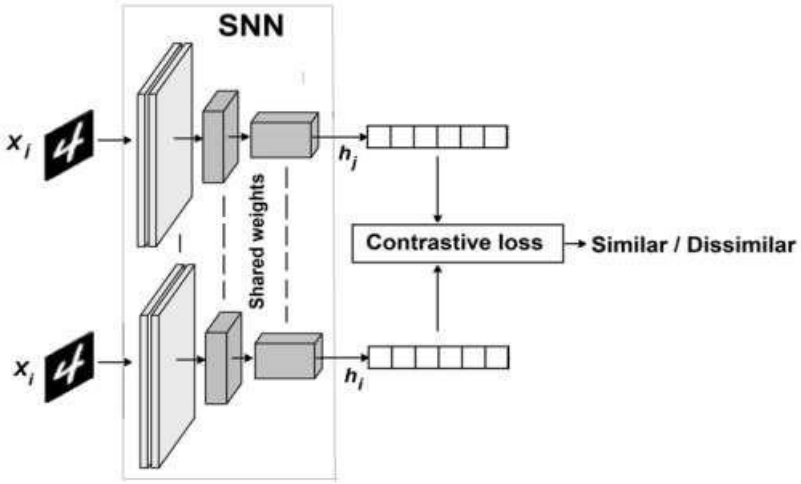


Figure 3.5.2.1: [33] Siamese network architecture with contractive loss function

3.5.3 Distance Metrics

Besides discussing about using a model to learn on the face embedding of a face image. There is another straightforward way to utilise these vectors computed from the face embedding algorithm, which is distance metrics. Distance metrics can also be understood as similarity measures, they are mathematical functions used to quantify the similarity or difference between two vectors. In the face recognition context, the function of distance metrics is to calculate the distance between two face embeddings of two different faces. [34]

The most commonly used distance metrics for face recognition are Euclidean distance, cosine similarity, and Mahalanobis distance. Euclidean distance is the straight-line distance between two points in n-dimensional space and is defined as the square root of the sum of the squared differences between corresponding elements of the two vectors. [35] Cosine similarity measures the cosine of the angle between two vectors and is often used to compute the similarity between high-dimensional sparse vectors. [35] Mahalanobis distance takes into account the covariance between the dimensions of the vectors and is useful when the dimensions are highly correlated. [36] In figure 3.5.3.1, the equation and the visualisation of the different distance metrics are illustrated.

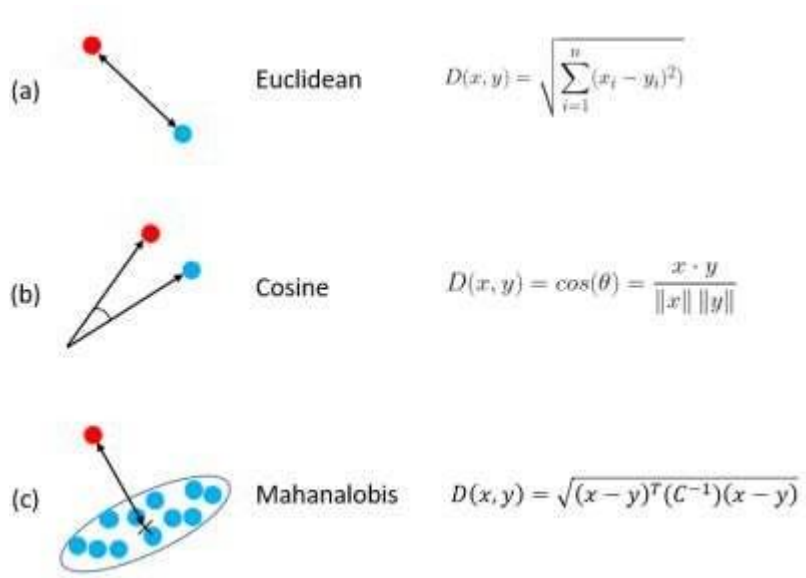


Figure 3.5.3.1: Distance metrics equation of (a) Euclidean distance (b) Cosine similarity (c) Mahanalobis

In face recognition context, these distance metrics are used to compute and determine how similar the two extracted features from face embedding algorithm. To classify the identity based on the calculated distance, a threshold is set. If the distance is within the threshold, it will deem that identity as the same, otherwise if the distance exceeds the threshold, it will claim that the two identities are unidentical. By playing around the threshold value for face classification, it is making a decision whether to give more priority to reduce false positives or false negatives.

3.6 System Implementation

In this section, overall procedure for the implementation of face recognition system in this project is discussed. This includes the hardware and software specification, setting up the environment, and the steps used to train the face recognition model.

3.6.1 System Specification

The hardware used in this project include a laptop and a mobile device. The details are shown in Tables 3.6.1.1 and Table 3.6.1.2. A laptop will be used for the overall process, which includes everything from obtaining datasets to deployment of the model. On the other hand, a mobile device will be used to take pictures that will be used as datasets for model training.

Table 3.6.1.1: Specifications of laptop

Description	Specifications
Model	Predator PT315-51
Processor	Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz, 2400 Mhz, 4 Core(s), 8 Logical Processor(s)
Operating System	Windows 11
Graphic	NVIDIA GeForce RTX 2060
Memory	8GB RAM
Storage	100GB HDD

Table 3.6.1.2: Specifications of mobile device

Description	Specifications
Model	Samsung Galaxy A32
Board	K6853v1_64_titan
Operating System	Android 11
Screen Resolution	720 x 1600 pixels (6.52 inches display)
Memory	8GB RAM
Storage	128GB

3.6.2 Setting Up

Before beginning the process of training the face recognition system, there are some software needed to be installed and downloaded in my laptop:

1. Git 2.37.1
2. CMake 3.24.0
3. Python 3.6.2
4. Anaconda 4.13.0

Table 3.6.2.1: Software description

Components	Description
Software	**Anaconda** Anaconda is an open-source distribution of the Python and R programming languages build for data science and machine learning that simplify package management and deployment. Anaconda is helpful when it comes to separating virtual environments, package installation is safe from conflicting with other project dependencies that have different package version installed. In this project, Anaconda will be mainly used to generate simulated mask and recognise faces using webcam. [37]
	Git Git is a free and open-source software that provide control of distributed system. In this project, Git will be used specifically for grabbing required resources from the open-source code and projects website, Github. [38]
	CMake CMake is an open-source software development, testing, and packaging tool. CMake is used to control the programme compilation process and uses simple, platform- and compiler-independent configuration files. Additionally, it generates native workspaces and make files that you can use with any compiler environment. [39]

Online tools	**Jupyter Notebook** Jupyter Notebook is an open-source web application provides an interactive computing environment that supports various programming languages such as Python, R, Julia, and others. This is the main platform that all the code implementation is performed. [40]
	MaskTheFace MaskTheFace is an open-source project that uses computer-vision based script to mask faces in images. In this project, the datasets used are entirely unmasked face images, so MaskTheFace will be used to help simulating the images of a person wearing a face mask. [41]
Language	**Python** Python is a popular programming language in the artificial intelligence industry. Python is utilised for this project since it has a robust library that may be employed. Python is utilised for a variety of reasons, including the fact that it is a universal language that can be used on iOS, Android, PCs, and mobile devices. [42]
Library	**Scikit-learn** A robust and well-known Python library for machine learning is called Scikit-learn. For data preprocessing, feature extraction, dimensionality reduction, model selection, and evaluation, it offers a variety of supervised and unsupervised learning techniques and tools. Built on top of NumPy, SciPy, and matplotlib, Scikit-learn is intended to be simple to use and connect with other Python frameworks and libraries. [43]
	Matplotlib Matplotlib is a data visualization library in Python. It is a popular choice for creating high-quality graphs, charts, and other types of visualizations from data. Matplotlib provides a range of functions and tools for creating static and interactive plots, histograms, bar charts, scatterplots, and more. It is also highly customizable,

	allowing users to fine-tune the appearance of their visualizations. [44]
	Deepface Deepface is an open-source Python package for face recognition and facial attribute analysis. It is built on top of Keras, TensorFlow, OpenCV, and other popular deep learning libraries, and provides a range of pre-trained models and tools for face detection, face recognition, face verification, and facial attribute analysis. [45]
	OpenCV OpenCV is an open-source computer vision library in Python. It provides a range of tools and functions for image and video processing, feature detection, object recognition, and more. OpenCV is widely used in a variety of fields, including robotics, augmented reality, and self-driving cars. [46]
	mtcnn The Multi-task Cascaded Convolutional Neural Networks (MTCNN) for Face Detection are implemented in the Python package mtcnn, which is based on TensorFlow. The MTCNN face detector can be used more easily by importing this Python module. [47]
	h5py H5py is a Python library for working with HDF5 files. HDF5 is a data storage format used for storing large and complex data sets. H5py provides a range of functions and tools for creating, reading, and manipulating HDF5 files in Python. It is widely used in scientific computing, machine learning, and other data-intensive applications. [48]
	Keras A high-level neural networks Python API is called Keras. Convolutional neural networks (CNNs), recurrent neural networks (RNNs), and other deep learning models can be built and trained

	using a variety of its tools and features. Built on top of TensorFlow, Theano, and other well-known deep learning libraries, Keras is made to be simple to use. It is widely employed in many different fields, including as robotics, natural language processing, and image and audio recognition. [49]

3.6.3 Code Implementation

In this section, the step-by-step implementation of building the face recognition system is discussed in detailed. To get an overview of the system design, figure 3.6.3.1 provide the visualization on the flow of the system. The main steps covered are data preparation, face embedding, face classification, and face verification. Each main step has its own sub-step to be completed before moving to the main step.

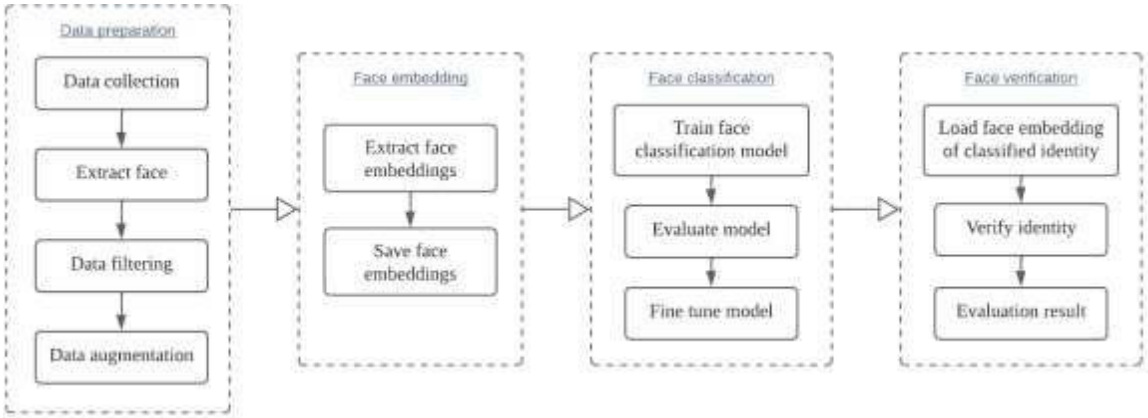


Figure 3.6.3.1: Face recognition block diagram

In data preparation step, it begins with data collection, where the known and unknown dataset mentioned in the previous section are obtained. The self-collected known dataset is captured through camera while the other datasets are acquired through online. Next, the face is cropped from the face image using a face detection algorithm. Next, the cropped faces are reviewed, and bad quality image or inappropriate image are filtered out. Some examples of bad images filtered can be seen in figure 3.6.3.2. Additionally, the LFW dataset is specifically filtered by removing identities that have lower than 100 images to ensure balance distribution of images among the identities.

The LFW dataset is left with 5 identities with more than 100 images upon filtering. Moreover, the LFW dataset is further process by generating simulated face mask for masked face recognition purpose. With the clean and filtered data, data augmentation is performed on the dataset, which includes blurring, rotating, flipping, darkening, and brightening. An example of a face image after data augmentation is shown in figure 3.6.3.3. Upon the completion of the data processing, some images from the final data are displayed in figure 3.6.3.4.

Figure 3.6.3.2: Bad images cropped (a) Self-collected dataset (b) LFW dataset

Figure 3.6.3.3: Data augmentation techniques

Figure 3.6.3.4: Known dataset (a) Self-collected (b) LFW dataset

The next main step is the face embedding, where the facial features are extracted from the prepared dataset. Using the pre-trained FaceNet model, the prepared dataset is transformed into the format of 128-dimension vectors, which contains represent important features for each identity. After these face embeddings are collected, there are save into the file system to be used in the later step.

Next, the subsequent step is the face classification, where model is trained to classify between the identities. Two models are chosen for this step, which are SVM and Siamese Network. These models are iteratively trained, evaluated, and fine-tuned. The result of the performance of the models at classifying among the identities are recorded. During the training process, different variations of dataset are fed to the models to make a comparison between how well the models learn from a specific dataset niche. The two variations of dataset are dataset with and without data augmentation. To enforce the quality of classification, a classification threshold is set to only classify the identity predict by the model that exceed the threshold. The classification threshold value is tweaked accordingly to establish a trade-off between false positives and false negatives.

Finally, the last main step is the face verification, which is a one-to-one verification on the classified identity. Initially, the face embedding of the classified identity previously stored is loaded. Then, the face embedding of the input image is compared with the face embeddings stored in the file system to compute the similarity. Verification threshold is used to determine how strictly the verification function should be in verifying an

identity. Completing all these steps, the result of the face recognition system is documented and discussed in the next section.

Chapter 4
Result and Discussion

In this chapter, the result of the performance of the trained face recognition system is discussed. The related performance metrics for evaluation are discussed. Then a decision is made to pick the optimal dataset variation, face detector, and face classifier. The performance metrics may differ to evaluate these dataset and algorithm.

4.1 Performance Metrics

Before discussing on the result of the experiment, there are some important terminologies about the performance metrics that are used to evaluate the performance of the face recognition system. The fundamental metrics of a machine learning model prediction are true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP means correct prediction of positive sample, and TN means correct prediction of negative sample. On the other hand, FP means prediction of negative sample as positive, and FN means prediction of positive example as negative.

The next level of performance metrics is derived from the fundamentals mentioned, these include precision, recall, accuracy, and f1-score. Precision is the total number of true positive among the positive predictions. Recall is the number of true positive among the total number of all positive samples. Accuracy the total number of true positive and true negative among all predictions. Finally, F1-score is the harmonic measurement of both precision and recall in a single metric.

Next, the other useful performance metrics are confusion matrix and classification report. Confusion matrix is a table used to evaluate the performance of a classification model by summarizing the number of true positive, true negative, false positive, and false negative predictions Confusion matrix provides an easy to see analysis about the prediction of each class. Especially when there is a wrong prediction, the actual and

predicted class can be observed easily. Subsequently, classification report in summary report of various performance metrics for a classification problem. It calculates and displays the precision, recall, F1-score, and support for each class of the target variable, along with the overall accuracy of the model. In short, the classification report provides a quick summary of the classification model's performance on the data.

Finally, the last performance metric is specific to evaluate the error rate of the face recognition model, which is false acceptance rate (FAR). FAR is a metric that represents the probability of a non-authorized user being mistakenly recognized as an authorized user. It is calculated as the ratio of the number of false positive predictions to the total number of negative predictions.

4.2 Evaluation of Different Dataset Variations

There were two types of dataset variations used in this project, which are dataset with and without data augmentation. The purpose of using these two different dataset variations is to determine whether the data augmentation techniques are useful in improving the performance of the trained face recognition model. To simplify the process of evaluating the performance of different dataset variations, the face detector used is OpenCV HaarCascade. After two different models were trained with dataset with and without augmentation, the results were tabulated in table 4.2.1. it can be seen that the performance of data without augmentation was slight better than data with augmentation.

Table 4.2.1: Training and testing accuracy between model trained with and without data augmentation

Data augmentation	**Accuracy**	
	Train	Test
Yes	0.9990	0.9974
No	0.9998	0.9984

Despite the performance of data without augmentation perform better than data with augmentation, providing only the accuracy from training and testing is not sufficient to tell whether data augmentation improves the model's performance. This is because compared to dataset without data augmentation, the dataset that undergoes data augmentation is five times larger. Thus, FAR metric was also used as a decision factor whether data augmentation was a good approach. In table 4.2.2, the result of both models predicting on the unknown dataset to calculate FAR is shown. It can be seen that the model that was trained with data augmentation provide lower FAR compared to model trained without data augmentation. Although the FAR was lower in model trained without data augmentation in the GMF dataset, it was only by a small margin of 0.0004. Overall, the FAR of model that was trained with data augmentation was significantly better.

Table 4.2.2: FAR of model trained with and without data augmentation

Data Augmentation	**FAR** (%)		
	GMF	LFW	CelebA
Yes	0.0103	0.0030	0.0817
No	0.0107	0.0024	0.0567

4.3 Evaluation of Face Detection Algorithm

Recalling back, the three main face detection algorithms used in this project are OpenCV HaarCascase, MTCNN, and YuNet. In this part, the performance of these face detectors is compared and evaluated. The most important metrics for evaluating these face detectors is time to inference, which is the measure of how long it takes for the face detector to detect the face in the image. In table 4.3.1, the computation time for all the face detector on one image is listed. It can be observed that OpenCV HaarCascade came out superior in term of speed, detecting face at only 0.0074 milliseconds. The second fastest face detector recorded was the YuNet at 0.0106 milliseconds and the slowest face detector was the MTCNN at 2.2864 milliseconds.

Table 4.3.1: Computation time of different face detector

Face Detector	**Computation Time (ms)**
OpenCV HaarCascade	0.0074
MTCNN	2.2864
YuNet	0.0106

Aside from evaluating the face detector in term of computation speed, another important factor that need to be taken into account is detection rate. Detection rate can also be understood as recall, it is the measure of detection count among all face images. As shown in table 4.3.2, the number of face detection out of 1000 face images are recorded. The best performing face detector of observed was MTCNN, where there was no miss detection. OpenCV HaarCascade came in second with 3 and 504 missed detections for unmasked face and masked face respectively. Lastly, YuNet under performed in face detection with only detection 210 unmasked faces and 128 masked faces out of 1000 face images in each category.

Table 4.3.2: Number of faces detected in unmasked and masked face images using different face detector

Face Detector	**Detection rate** (out of 1000)	
	Unmasked	Masked

OpenCV HaarCascade	997	496
MTCNN	1000	1000
YuNet	210	128

Although the detection rate of YuNet seems unreasonable and not viable for face detector, it is not entirely true. When all the face detectors were tested again but using real-time webcam instead, it can be observed that it is not the fault of YuNet for giving bad detection rate, but YuNet requires the face to be a certain distance away from the camera for it to be powerful. In figure 4.3.1, it is displayed that YuNet is bad at detecting face close to the camera, but it is very fast and accurate when detecting both unmasked and masked face when the face in specific distance away from the camera. Using webcam, the performance of OpenCV HaarCascade and MTCNN was tallied with the detection rate recorded. OpenCV HaarCascade was fast at detecting face, it was good at detecting unmasked face, but did not guarantee the masked face detection every time. On the other hand, MTCNN performed incredible at detecting face either unmasked or masked, but the detection from webcam was extremely laggy due to its slow detection speed.

Figure 4.3.1: YuNet face detection from webcam

With all the result from the above evaluation, it was decided for OpenCV HaarCascade to be the main face detector for the face recognition system. The reason of choosing OpenCV HaarCascade was due to its balance in both speed and detection rate. Although it was not effectively amazing at detecting masked face like MTCNN, it still provided sufficient detection rate for the face recognition system to work.

4.4 Evaluation of Face Classification Algorithm

Next, the performance of face classification algorithms is evaluated, these include SVM, Siamese Network, and distance metrics. SVM and Siamese Network are used as face classification, where the model classifies an identity out of all possible identities. With the classified identity, this identity is used for face verification via different distance metrics, the identity is compared with its facial embedding stored in database. In table 4.4.1, it can be observed that SVM perform better than Siamese network in term of accuracy from training and testing dataset.

Table 4.4.1: Accuracy comparison between SVM and Siamese network

Model	**Accuracy**	
	Train	Test
SVM	0.9990	0.9974
Siamese Network	0.9768	0.9792

Some drawbacks can be observed in Siamese network when it is tested on real-time webcam. It is observed that Siamese network had overfitted itself to recognise the identity in specific detail. Siamese network was unable to differentiate between the identities when accessories wore on the face is present, in the example shown in figure 4.4.1. The same identity was presented but one with spectacle and another without spectacle, the Siamese network that was only trained on the face with spectacle for this particular identity was unable to recognise the same identity when the spectacle was removed.

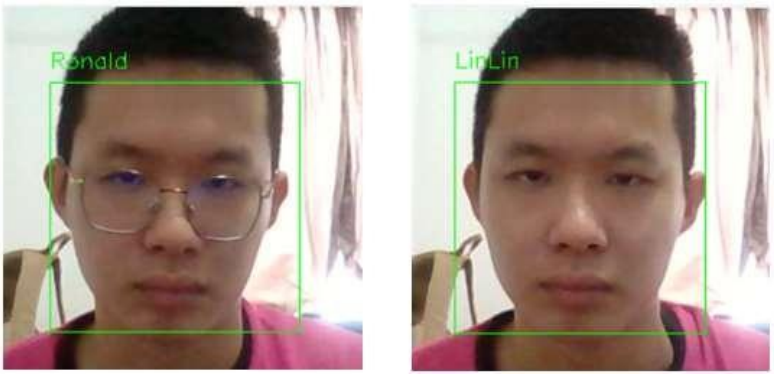


Figure 4.4.1: Siamese network fail to differentiate between face with spectacle and without spectacle

Provided the result of the classifier, SVM was used for further experimentation with the distance function. As discussed in previous section, there are three distance function covered in this project, which are Euclidean, Cosine, and Mahanalobis. To evaluate between the effectiveness of these distance functions at differentiating between identities, the distance was computed for pairs of same identity and different identity. The optimal distance function is the one that gives low distance between same identity and high distance between different identity. In table 4.4.2, the findings of different distance function were tabulated. It can be seen that Cosine distance function was the best at conveying same identity is similar, but it was unable to tell that two difference identities are dissimilar. On the other hand, Euclidean and Mahanalobis distance functions performed equally well at differentiating between identities. With this result, the Euclidean distance function was chosen for its ability to differentiate between identities and quick execution speed.

Table 4.4.2: Evaluation of different distance function

Type of identity pairs	**Distance**		
	Euclidean	Cosine	Mahanalobis
Same	5.4381	1.0491	5.4381
Different	7.2291	1.0144	7.2291

4.5 Evaluation of Final System

With the decision made for dataset variation, face detector, and distance function, this section explained how the final system was built and further improved. The final model was trained on dataset with data augmentation, use OpenCV HaarCascade for face detection, FaceNet for face embedding, SVM for identity classification, and Euclidean distance for identity verification. There are three main factors that can tweak the performance of final system, which are SVM hyperparameters, classification threshold, and verification threshold.

In SVM, there hyperparameters that could affect its performance include C for regularization, gamma for the kernel coefficient, coef0 for the independent term in kernel function, tol for the stopping criterion tolerance, degree for the polynomial kernel degree, and kernel for the kernel type. Their appropriate values depend on the specific problem being addressed. Usually, the hyperparameter C is the most important hyperparameter because it controls the trade-off between achieving a low training error and a low testing error. A small C value will create a wider margin hyperplane with more training errors, while a larger C value will create a smaller margin hyperplane with fewer training errors. In table 4.5.1, the list of values used for each SVM hyperparameters are listed. RandomSearch was used to efficiently find the optimal hyperparameters among the vast combination of them. After searching, the best combination of hyperparameters found was C=5, kernel=poly, degree=4, gamma=3, coef0=1, shrinking=True, tol=0.001. The score recorded for this combination of hyperparameters was 0.9989.

Table 4.5.1: List of values used for SVM hyperparameters

Hyperparameters	**Value used**
C	0.1, 1, 3, 5, 7
kernel	Linear, poly, rbf, sifmoid
degree	1, 2, 3, 4, 5
gamma	scale, auto, 0.1, 3, 6, 10
coef0	0.1, 0.3, 0.5, 0.7, 1

shrinking	True, False
tol	0.0001, 0.0003, 0.0005, 0.0007, 0.001

Next, the other two factors that can affect the performance of the system are classification threshold and verification threshold. Classification threshold is the minimum probability for a predicted identity to be classified. Verification threshold is the minimum similarity for a classified identity to be verified. Setting higher value for both thresholds increase the security of the system, where only strict confidence is required for an identity to be authenticated. On the other hand, setting lower threshold allow more identities to be recognised. The metric FAR is used to find the optimal threshold for classification and verification. As shown in table 4.5.2, the values used for classification threshold were 0.85 and 0.9, while the values used for verification threshold were 0.090, 0.095, 0.100, 0.105, and 0.110. It can be observed that classification threshold did not affect the FAR much, but tweaking the verification threshold can affect the result of FAR. While increasing the threshold can improve the FAR, it is a trade-off between accuracy and recall. To further evaluate the performance regarding of different threshold, the face recognition system is tested again on real-time webcam. It is found that at verification threshold of 0.100, the system is quite vulnerable to false positive. As shown in figure 4.5.1, when a similar face of known identity was presented in the webcam, the system had a quite high false positive rate at 0.100 verification threshold. However, at 0.110 verification threshold, the possibility of false positive appearing was rare to none. Despite using high value for both classification and verification thresholds, it does not significantly affect the performance of the system to recognise a known identity correctly. Therefore, it was concluded that the system performs the best at 0.9 classification threshold and 0.110 verification threshold. In summary, the configuration of the final system was OpenCV HaarCascade for face detection, FaceNet for face embedding, SVM for face classification, and Euclidean distance for face verification. For hyperparameters, SVM used C=5, kernel=poly, degree=4, gamma=3, coef0=1, shrinking=True, and tol=0.001. For threshold, the best result found was using 0.90 classification threshold and 0.110 verification threshold.

Table 4.5.2: FAR of final system using different classification and verification thresholds

Threshold		**FAR** (%)		
Classification	Verification	CelebA	LFW	GMF
0.85	0.090	0.1542	0.0112	0.0287
	0.095	0.1124	0.0053	0.0172
	0.100	0.0793	0.0026	0.0111
	0.105	0.0411	0.0013	0.0057
	0.110	0.0153	0.0006	0.0038
0.9	0.090	0.1536	0.0112	0.0287
	0.095	0.1119	0.0053	0.0172
	0.100	0.0790	0.0026	0.0111
	0.105	0.0409	0.0013	0.0057
	0.110	0.0152	0.0006	0.0038

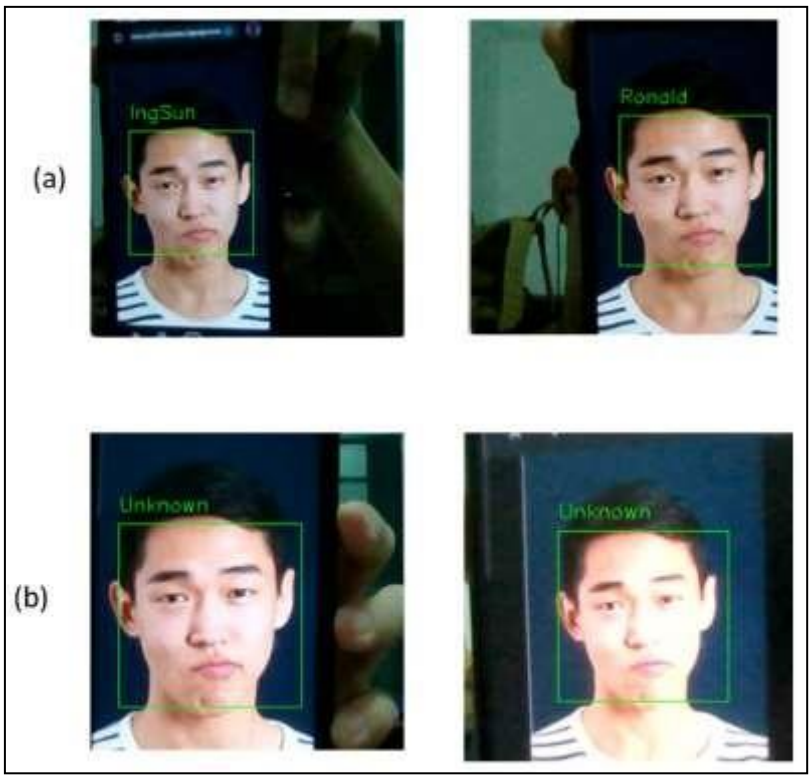


Figure 4.5.1: Detection of unknown identity similar to known identity (a) 0.100 verification threshold (b) 0.110 verification threshold

To evaluate the performance of the final face recognition system, a confusion matrix and a classification report were used. As seen in figure 4.5.2 and table 4.5.3, the final system was able to achieve a training accuracy of 100% with no misidentification of any identities. However, there were some misclassifications found in testing set. Four out of the total of eight identities had been identified wrongly for 13 times out of the total of 6097 images. The testing accuracy recorded is 99.787. Overall, the final face recognition system developed did a decent job at recognizing both unmasked and masked known identities.

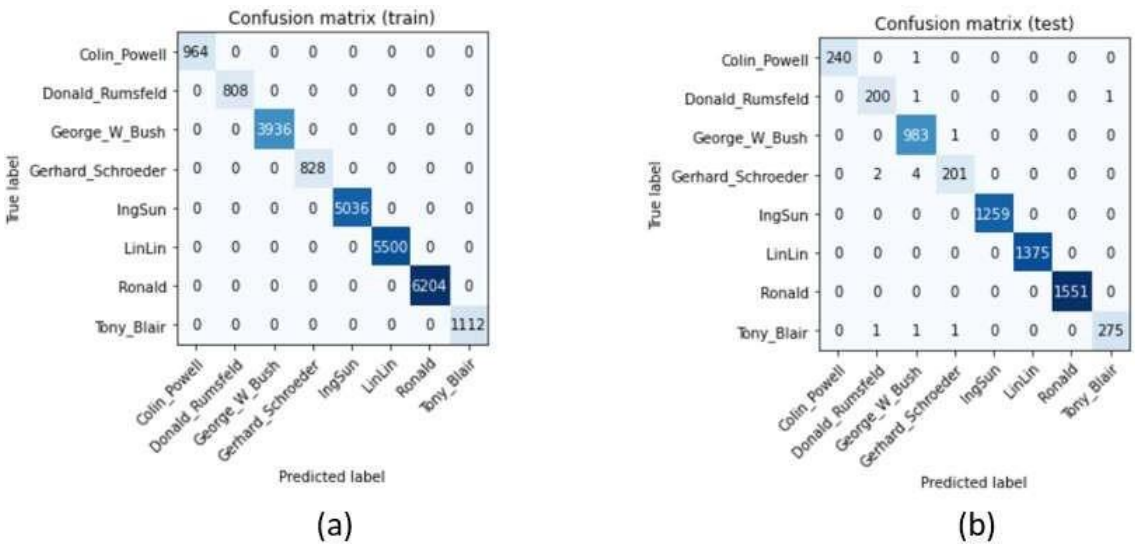


Figure 4.5.2: Confusion matrix of final system

Table 4.5.3: Classification report of final system

Type	**Metrics**				
	Identity	**Precision**	**Recall**	**F1-score**	**Support**
Train	Colin Powell	1.00	1.00	1.00	964
	Donald Rumsfled	1.00	1.00	1.00	808
	George W Bush	1.00	1.00	1.00	3936
	Gerhard Schroeder	1.00	1.00	1.00	828
	IngSun	1.00	1.00	1.00	5036
	LinLin	1.00	1.00	1.00	5500

	Ronald	1.00	1.00	1.00	6204
	Tony Blair	1.00	1.00	1.00	1112
	Accuracy				**Support**
	100.00				24388
Test	Colin Powell	1.00	1.00	1.00	241
	Donald Rumsfled	0.99	0.99	0.99	202
	George W Bush	0.99	1.00	1.00	984
	Gerhard Schroeder	0.99	0.97	0.98	207
	IngSun	1.00	1.00	1.00	1259
	LinLin	1.00	1.00	1.00	1375
	Ronald	1.00	1.00	1.00	1551
	Tony Blari	1.00	0.99	0.99	278
	Accuracy				**Support**
	99.787				6097

Finally, when the face recognition was tested on known identities, the system was able to recognise the identities whether the face was obstructed by a face mask or not. In figure 4.5.3, some snapshots of the recognition done on the real-time webcam is shown. It can be observed that the two identities presented to the real-time webcam were recognised successfully with or without face mask wore.

Figure 4.5.3: Snapshots of real-time face recognition on webcam

4.6 Comparison with YOLOv5

In the previous iteration of the project, the object detection model, YOLOv5 was used for performing the face recognition process. In this section, a discussion is carried out to compare between the YOLOv5 with the current face recognition system using FaceNet-SVM. Previously, using YOLOv5 for face recognition had several issues, which includes unable to detect the identity presented when the face was obstructed by spectacles or hair. Furthermore, YOLOv5 was also unable to differentiate between known dataset and unknown dataset, which means it was susceptible to false positive upon an unknown person. For the newly proposed system in this project, tasks in the face recognition pipeline were broken down and dealt with individually. This refers to using different algorithm for face detection, face embedding, face classification, and face verification. By doing so, the system was more robust at the specific job that it was given. Instead of taking the face of a person and let the model learn from it directly like it was in YOLOv5, the new system tried to extract unique feature from the face before feeding it to the model. The new system had mitigated the issue of performance degradation due to occlusion and high false positive rate. As displayed in table 4.6.1, it can be seen that the developed system had greatly improved in term of precision and recall but lost out in term of inference time compared to YOLOv5. The time of developed system may seem significantly slower than YOLOv5, but it did not prevent the system from being able to perform at real-time basis.

Table 4.6.1: Comparison between YOLOv5 with current system

Model	**Precision**	**Recall**	**Time** (ms)
YOLOv5	0.892	0.825	9.2
FaceNet-SVM	0.996	0.994	109.8

4.7 Limitation and Future Works

Even though the face recognition system developed had reached the scope of this project, there are some flaws exist within it. The face recognition was only capable of detecting one identity at a time, this means that when there were multiple faces displayed, only the first detected face was identified. With this limitation, the system is not suitable for face recognition system that needs to perform face recognition on multiple identities at once. For example, a security system in a large office building where the system needs to be able to recognise all identities within a frame captured from the camera. Moreover, the face recognition system developed was only a simple prototype and it was not ready for device integration, this means that if the face recognition system needs to be implemented for certain device, further configuration needs to be done to properly set up for proper usage. Furthermore, the face recognition system developed was incapable of checking whether the person presented in the camera was physically present or not. This means that someone can bypass the face recognition system by showing a face of a known identity through any form of media. This include showing the face image in mobile device, face printed in paper, or masking.

Some future works that can be done extending from this project includes implementing a better face detection algorithm that is more robust at detecting masked faces as well. In this project, the face detectors used were at the extreme point of being fast or accurate. OpenCV HaarCascade, the face detector of this project had acceptable performance, but better face detection model can be trained that can detect both unmasked and masked face efficiently. Additionally, more advanced techniques like 3D face modelling and face alignment can also be explored and integrated to the face recognition system.

Chapter 5
Conclusion

As conclusion, this project had worked on building a masked face recognition system to deal with the current pandemic that affects everyone, where wearing a face mask while outdoors is recommended. The system developed was built using several components, including face detection, face embedding, face classification, and face verification. Multiple experiments and research had been conducted to find the suitable algorithm for each of the component mentioned. The decision made was to use OpenCV HaarCascade for face detection, FaceNet for face embedding, SVM for face classification, and Euclidean distance for face verification. In the process of training the model, dataset of 7 identities was fed to the model. Data augmentation techniques were also applied on the dataset to improve the ability of the model to generalise to unknown dataset. After the model was trained, the performance of the model was evaluated using different performance metrics. The metrics include precision, recall, accuracy, f1-score, FAR, and time. Using these performance metrics as reference, the trained model was further improving by tweaking the hyperparameters of SVM and setting different threshold for classification and verification. After fine tuning the configurations of the system, the best settings decided for the system was SVM with hyperparameters of C=5, kernel=poly, degree=4, gamma=3, coef0=1, shrinking=True, and tol=0.001. Subsequently, the optimal threshold found is 0.90 and 0.110 for classification and verification respectively. With the settings mentioned, the final system was able to achieve a training and testing accuracies of 100.00 and 99.787 respectively. Also, the time taken to inference one face image was 109.8 millisecond. When the system was exposed to unknown dataset, it was able to keep the FAR of CelebA, LFW, and GMF dataset at 0.0152%, 0.0006%, and 0.0038% respectively. However, the system developed had its flaws. This includes its inability to recognise multiple identities presented in one frame at once, no integration on other devices, and unable to detect identity masquerading. Overall, the face recognition system developed had met the scope of the project. But further improvement can be done on the algorithm by using new and robust techniques.

REFERENCES

[1] D. Fitousi, N. Rotschild, C. Pnini, and O. Azizi, "Understanding the Impact of Face Masks on the Processing of Facial Identity, Emotion, Age, and Gender," *Frontiers in Psychology*, vol. 12, Nov. 2021, doi: https://doi.org/10.3389/fpsyg.2021.743793.

[2] Thales, "Biometrics in 2020 (A helpful illustrated overview)," *www.thalesgroup.com*, Jun. 02, 2021. [Online]. Available: https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/inspired/biometrics

[3] E. Burns, "What Is Machine Learning and Why Is It Important?," *SearchEnterpriseAI*, Mar. 2021. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

[4] Mathworks, "What Is Deep Learning? | How It Works, Techniques & Applications," *Mathworks.com*, 2019. [Online]. Available: https://www.mathworks.com/discovery/deep-learning.html

[5] IBM, "What are Neural Networks? | IBM," *www.ibm.com*. [Online]. Available: https://www.ibm.com/topics/neural-networks

[6] H. N. Vu, M. H. Nguyen, and C. Pham, "Masked face recognition with convolutional neural networks and local binary patterns," *Applied Intelligence*, Aug. 2021, doi: https://doi.org/10.1007/s10489-021-02728-1.

[7] G. Wu, "Masked Face Recognition Algorithm for a Contactless Distribution Cabinet," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–11, May 2021, doi: https://doi.org/10.1155/2021/5591020.

[8] N. Ullah, A. Javed, M. Ali Ghazanfar, A. Alsufyani, and S. Bourouis, "A novel DeepMaskNet model for face mask detection and masked facial recognition," *Journal*

of King Saud University - Computer and Information Sciences, Jan. 2022, doi: https://doi.org/10.1016/j.jksuci.2021.12.017.

[9] W. Hariri, "Efficient Masked Face Recognition Method during the COVID-19 Pandemic," *arXiv:2105.03026 [cs]*, May 2021 [Online]. Available: https://arxiv.org/abs/2105.03026

[10] A. Desai, "Applying Deep learning techniques - masked facial recognition in smartphone security systems using transfer learning," *esource.dbs.ie*, 2021. [Online]. Available: https://esource.dbs.ie/handle/10788/4273

[11] Y. M. Saib and S. Pudaruth, "Is Face Recognition with Masks Possible?," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 7, 2021, doi: https://doi.org/10.14569/ijacsa.2021.0120706.

[12] B. Lab, "Face Recognition Pipeline Clearly Explained," *Backprop Lab*, Dec. 07, 2020. [Online]. Available: https://medium.com/backprop-labs/face-recognition-pipeline-clearly-explained-f57fc0082750#:~:text=A%20Face%20Recognition%20pipeline%20can. [Accessed: Apr. 24, 2023]

[13] R. Khan, "Importance of Datasets in Machine Learning and AI Research," *datatobiz*, May 13, 2022. [Online]. Available: https://www.datatobiz.com/blog/datasets-in-machine-learning/

[14]"LFW Face Database : Main," *vis-www.cs.umass.edu*. [Online]. Available: http://vis-www.cs.umass.edu/lfw/

[15] "CelebA Dataset," *mmlab.ie.cuhk.edu.hk*. [Online]. Available: https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[16] "Gender Classified Dataset with Masked Face," *www.kaggle.com*. [Online]. Available: https://www.kaggle.com/datasets/itsshuvra/gender-classified-dataset-with-masked-face?select=GenderOcclusionData. [Accessed: Apr. 24, 2023]

[17] A. Takimoglu, “What is Data Augmentation? Techniques, Benefit and Examples,” *research.aimultiple.com*, Apr. 30, 2021. [Online]. Available: https://research.aimultiple.com/data-augmentation/

[18] Varun, “What is Face Detection? Ultimate Guide 2023 + Model Comparison,” *learnopencv.com*, Sep. 06, 2022. [Online]. Available: https://learnopencv.com/what-is-face-detection-the-ultimate-guide/. [Accessed: Apr. 24, 2023]

[19] P. Viola and M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features,” *ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*, 2001 [Online]. Available: https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

[20] N. Zhang, J. Luo, and W. Gao, “Research on Face Detection Technology Based on MTCNN,” *IEEE Xplore*, Sep. 01, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9239720

[21] P. Nelson, “OpenCV Face Detection: Cascade Classifier vs. YuNet,” *OpenCV*, Nov. 15, 2022. [Online]. Available: https://opencv.org/opencv-face-detection-cascade-classifier-vs-yunet/#:~:text=It%20is%20a%20powerful%20lightweight. [Accessed: Apr. 24, 2023]

[22] S. Yu, “libfacedetection,” *GitHub*, Apr. 24, 2023. [Online]. Available: https://github.com/ShiqiYu/libfacedetection. [Accessed: Apr. 24, 2023]

[23] W. Mumbi, “Unified Embedding for Face Recognition and Clustering using FaceNet,” *Engineering Education (EngEd) Program | Section*, Aug. 17, 2021. [Online]. Available: https://www.section.io/engineering-education/facenet-unified-embedding-face-recognition-clustering/. [Accessed: Apr. 24, 2023]

[24] G. Hoàng and H. Ngô, “Face embedding,” *HackMD*, 2020. [Online]. Available: https://hackmd.io/@gianghoangcotai/Sk05UiSFI. [Accessed: Apr. 24, 2023]

[25] Y. Chandra, K. Gouru, and Reddy, “A Comparative Analysis Of Face Recognition Models On Masked Faces,” Oct. 2020 [Online]. Available: https://www.ijstr.org/final-print/oct2020/A-Comparative-Analysis-Of-Face-Recognition-Models-On-Masked-Faces.pdf

[26] S. Reddy, “The intuition of Triplet Loss,” *Analytics Vidhya*, Nov. 02, 2021. [Online]. Available: https://medium.com/analytics-vidhya/triplet-loss-b9da35be21b8

[27] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering - IEEE Conference Publication,” *Ieee.org*, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7298682

[28] L. Dulčić, “Face Recognition with FaceNet and MTCNN,” *arsfutura.com*, 2020. [Online]. Available: https://arsfutura.com/magazine/face-recognition-with-facenet-and-mtcnn/

[29] R. Sunil, “Understanding Support Vector Machine algorithm from examples (along with code),” *Analytics Vidhya*, Mar. 11, 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[30] R. Gandhi, “Support Vector Machine — Introduction to Machine Learning Algorithms,” *Towards Data Science*, Jun. 07, 2018. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[31] aswathisasidharan, “Support Vector Machine Algorithm,” *GeeksforGeeks*, Jan. 20, 2021. [Online]. Available: https://www.geeksforgeeks.org/support-vector-machine-algorithm/

[32]S. Benhur, “A Friendly Introduction to Siamese Networks | Built In,” *builtin.com*, Jan. 25, 2022. [Online]. Available: https://builtin.com/machine-learning/siamese-network. [Accessed: Apr. 24, 2023]

[33] L. V. Utkin, M. S. Kovalev, and E. M. Kasimov, “An explanation method for Siamese neural networks,” *arXiv:1911.07702 [cs, stat]*, Nov. 2019 [Online]. Available: https://arxiv.org/abs/1911.07702. [Accessed: Apr. 24, 2023]

[34] P. Sharma, “Understanding Distance Metrics Used in Machine Learning,” *Analytics Vidhya*, Feb. 25, 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/02/4-types-of-distance-metrics-in-machine-learning/#:~:text=Distance%20metrics%20are%20used%20in%20supervised%20and%20unsupervised%20learning%20to. [Accessed: Apr. 24, 2023]

[35] M. Grootendorst, “9 Distance Measures in Data Science,” *Medium*, Feb. 03, 2021. [Online]. Available: https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

[36] Stephanie, “Mahalanobis Distance: Simple Definition, Examples,” *Statistics How To*, Nov. 21, 2017. [Online]. Available: https://www.statisticshowto.com/mahalanobis-distance/

[37] “Anaconda,” *Anaconda*, 2018. [Online]. Available: https://www.anaconda.com/

[38] Git, “Git,” *Git-scm.com*, 2019. [Online]. Available: https://git-scm.com/

[39] “CMake,” *Cmake.org*, 2018. [Online]. Available: https://cmake.org/

[40] Jupyter, “Project Jupyter,” *Jupyter.org*, 2019. [Online]. Available: https://jupyter.org/

[41] A. Anwar, “aqeelanwar/MaskTheFace,” *GitHub*, Feb. 07, 2021. [Online]. Available: https://github.com/aqeelanwar/MaskTheFace

[42] Python, “Welcome to Python.org,” *Python.org*, May 29, 2019. [Online].

Available: https://www.python.org/

[43] scikit-learn, “scikit-learn: machine learning in Python,” *Scikit-learn.org*, 2019. [Online]. Available: https://scikit-learn.org/stable/

[44] Matplotlib, “Matplotlib: Python plotting — Matplotlib 3.1.1 documentation,” *Matplotlib.org*, 2012. [Online]. Available: https://matplotlib.org/

[45] S. I. Serengil, “deepface: A Lightweight Face Recognition and Facial Attribute Analysis Framework (Age, Gender, Emotion, Race) for Python,” *PyPI*, Feb. 23, 2023. [Online]. Available: https://pypi.org/project/deepface/

[46] OpenCV, “OpenCV library,” *Opencv.org*, 2019. [Online]. Available: https://opencv.org/

[47] I. de P. Centeno, “mtcnn: Multi-task Cascaded Convolutional Neural Networks for Face Detection, based on TensorFlow,” *PyPI*, Jul. 09, 2021. [Online]. Available: https://pypi.org/project/mtcnn/

[48] “HDF5 for Python,” *www.h5py.org*. [Online]. Available: https://www.h5py.org/

[49] Keras, “Home - Keras Documentation,” *Keras.io*, 2019. [Online]. Available: https://keras.io/

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 3**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE

Research on the approach to be taken to start up the project. Which include the following:

1) Model

a) **Siamese network**:

- neural network architecture designed to learn similarity between 2 input data points
- this will be first model to be tried, due to its compatibility with smaller dataset

b) **FaceNet**:

- deep learning model that cluster similar faces together and dissimilar faces far apart
- this model can handler larger dataset and perform better with larger dataset

(Note: Previously in FYP1, YOLOv5 which is an object detection model is used to carry out the face recognition process. However, it is found that the object detection approach is not quite suitable for face recognition process which has deep feature. So, FYP2 will approach the problem with other models instead)

2) Dataset

The dataset used for FYP2 will almost be similar to that of FYP1, which consist of **Labelled Faces in the Wild (LFW)** dataset, and **self-collected dataset**. In addition, dataset that contains **mask wearer images** will be explored in this project.

3) Platform

In FYP1, the platform used to carry out the model training is **Google Colab** but there is an issue of GPU limit for free users which has limited the training time allowed. Therefore, in FYP2, **Jupyter Notebook** will be used instead.

4) Library

The library that will be used to train the Siamese network will include **Tensorflow**, which provides a set of tools and APIs for training the model.

2. WORK TO BE DONE

- Explore & prepare dataset
- Setup suitable environment to train models
- Research on how others have implemented Siamese network

3. PROBLEMS ENCOUNTERED

4. SELF EVALUATION OF THE PROGRESS

Supervisor's signature	Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 4**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE - Obtaining self-collected dataset of 250x250 by using opencv - Prepare the dataset for Siamese network (anchor, negative, and positive)
2. WORK TO BE DONE - Data preprocessing - Siamese network training
3. PROBLEMS ENCOUNTERED
4. SELF EVALUATION OF THE PROGRESS

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 5**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE - Use L1 dist to find the different between 2 face images - Train Siamese Network - Setting a threshold whether the image is determined as same - Verify the identity by iterating through a directory containing that identity, if the identified/total images exceed a threshold, determine as same
2. WORK TO BE DONE - Train the Siamese Network to recognize masked faces as well
3. PROBLEMS ENCOUNTERED
4. SELF EVALUATION OF THE PROGRESS The recognition currently is only the masked faces

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 6**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE - Capture masked and unmasked face images - Get an online masked face Real World Fasked Face Recognition Dataset (RMFRD) to represent negative masked dataset
2. WORK TO BE DONE - Train the Siamese Network to recognize masked faces as well
3. PROBLEMS ENCOUNTERED
4. SELF EVALUATION OF THE PROGRESS

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 7**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE

- Downloaded a masked face dataset, *Gender Classified Dataset with Masked Face*. This represents the **masked negative** images in the dataset.
- Get face images by capturing from webcam and crop them to the face using *opencv*
- Filter the dataset by removing bad quality images
- Train Siamese Network model with unmasked and masked model

2. WORK TO BE DONE

- Build or find a mask detection model
- Perform mask detection and refer to the right *verification_image* folder to face verification

3. PROBLEMS ENCOUNTERED

4. SELF EVALUATION OF THE PROGRESS

- Face verification is now done by manually setting the path to *verification_image* folder when wanting to verify for either unmasked or masked faces. This is because there is not mask detection algorithm being used currently, so need to manually determine which folder to refer to before face verification.

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 8**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE

- Include masked images in the dataset
- Re-capture the anchor and positive images into 2 categories (face & eyes)
- Re-process the negative images into 2 categories also (face & eyes)
- Re-label the image to be such input → (face_anc, eyes_anc, face_neg_or_pos, eyes_neg_or_pos)
- Modify the model's layers to take 2 input (face & eyes), and 2 validations (face & eyes)
- Train the model on the new dataset which include unmasked, and masked images
- Perform real-time face recognition on face using newly trained model

2. WORK TO BE DONE

- Improve the model's performance
- Include more identity to the model

3. PROBLEMS ENCOUNTERED

4. SELF EVALUATION OF THE PROGRESS

- The model now only recognize face of one identity, which is me.
- Also, the model does not consider the person is wearing a spectacle or not, which in my case will cause the model unable to identity me when I am not wearing my spectacles

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 9**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE

- Capture 2 more identities images beside myself
- Convert face input & validation images to grey scale to prevent the model having bias over a specific colour mask
- Pre-process and arrange the multiple identities dataset to be train
- Modify the output layer of model to a one-hot encoding format
- Change the loss function from 'BinaryCrossEntropy' to 'CategoricalCrossEntropy'
- Trained the model on multiple identities
- Changed the verification pipeline, first recognize against one image for each identity and pick the most probable identity, then further authenticate this identity by verifying it on multiple images of the identity

2. WORK TO BE DONE

- Improve the model's performance to detect identity correctly

3. PROBLEMS ENCOUNTERED

- Model is overfit to the image background. When presented identity A in webcam under background B, the model will recognize as identity B.

4. SELF EVALUATION OF THE PROGRESS

- Not sure if the inference speed of the model is acceptable or not. It takes around 3-4 seconds to recognize a detected face.

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 10**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE - Explore face embedding extraction using pre-trained FaceNet model - Use SVM to recognize identity from face embedding
2. WORK TO BE DONE - Explore using Siamese Network on face embedding feature
3. PROBLEMS ENCOUNTERED
4. SELF EVALUATION OF THE PROGRESS The FaceNet model is much better at extracting feature from a face images rather than passing the whole image to the Siamese Network previously.

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 11**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE

- Pre-process the dataset so that each identity uses the other identities as negative images
- Train the Siamese Network to use feature extracted from FaceNet model
- Perform real-time verification, draw bounding box and detected identity

2. WORK TO BE DONE

- Compile the results and methods done and document in report

3. PROBLEMS ENCOUNTERED

- The model will still have false positive when I use very similar image in the web as input. Since all my identities used are Asian-face and there are very less Asian-face for my negative images, the model predicts wrongly when I find a Asian-face image online as input

4. SELF EVALUATION OF THE PROGRESS

- Using FaceNet model for feature extraction greatly shorten the training time of the model

Supervisor's signature

Student signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

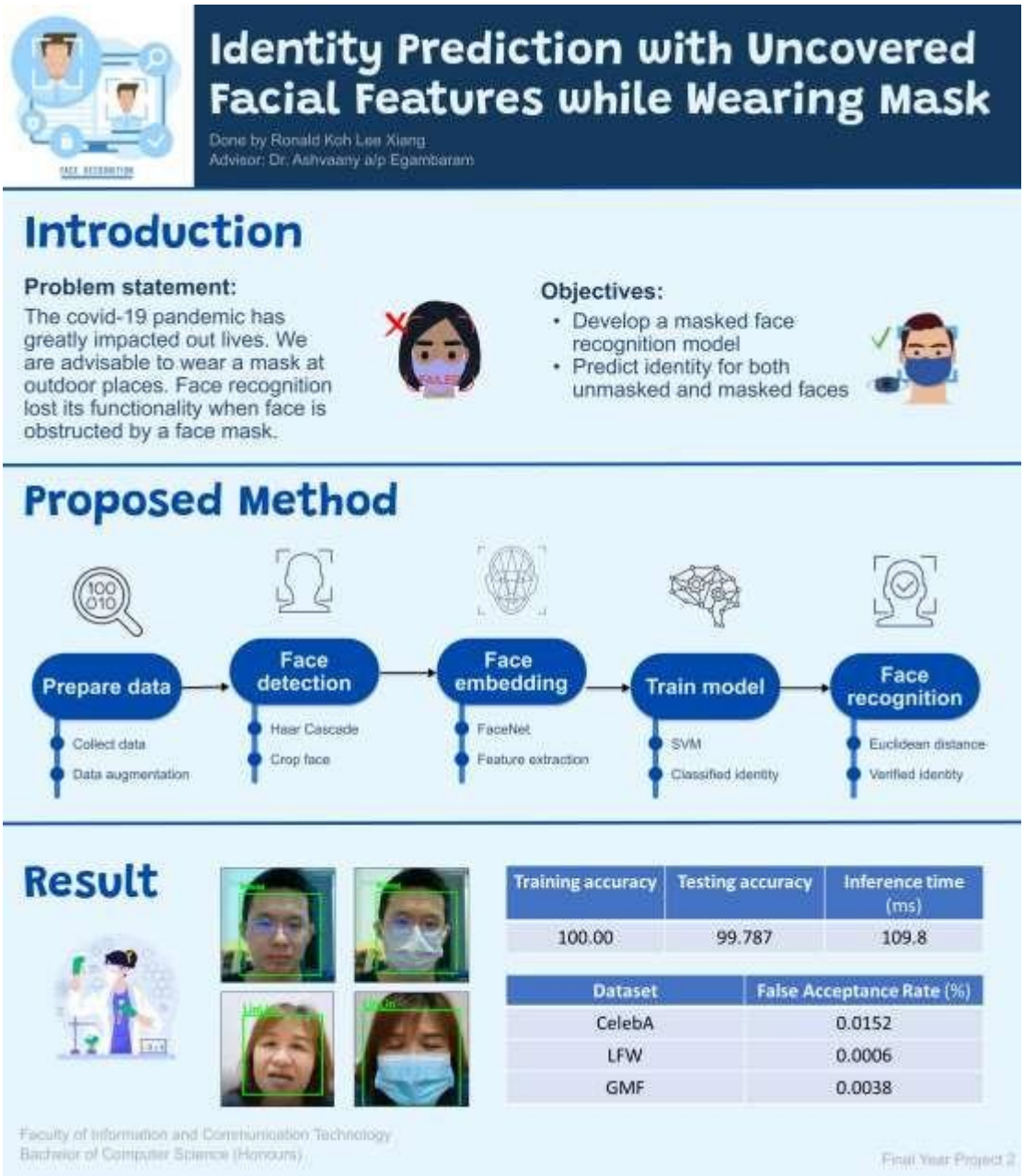
Trimester, Year: Y3S3	**Study week no.: 12**
Student Name & ID: Ronald Koh Lee Xiang	
Supervisor: Dr. Ashvaany a/p Egambaram	
Project Title: IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK	

1. WORK DONE - Record result of all experiments done - Complete draft for report
2. WORK TO BE DONE - Check in report
3. PROBLEMS ENCOUNTERED
4. SELF EVALUATION OF THE PROGRESS

Supervisor's signature

Student signature

POSTER


Identity Prediction with Uncovered Facial Features while Wearing Mask
Done by Ronald Koh Lee Xiang
Advisor: Dr. Ashvaany a/p Egambaram
Introduction
Problem statement:
The covid-19 pandemic has greatly impacted out lives. We are advisable to wear a mask at outdoor places. Face recognition lost its functionality when face is obstructed by a face mask.
Objectives:
• Develop a masked face recognition model
• Predict identity for both unmasked and masked faces
Proposed Method
Prepare data
Collect data
Data augmentation
Face detection
Haar Cascade
Crop face
Face embedding
FaceNet
Feature extraction
Train model
SVM
Classified identity
Face recognition
Euclidean distance
Verified identity
Result
Training accuracy | Testing accuracy | Inference time (ms)
100.00 | 99.787 | 109.8
Dataset | False Acceptance Rate (%)
CelebA | 0.0152
LFW | 0.0006
GMF | 0.0038
Faculty of Information and Communication Technology
Bachelor of Computer Science (Honours)
Final Year Project 2

PLAGIARISM CHECK RESULT

IDENTITY PREDICTION WITH UNCOVERED FACIAL FEATURES WHILE WEARING MASK

ORIGINALITY REPORT

8%	4%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	learnopencv.com Internet Source	1%
2	Hoai Nam Vu, Mai Huong Nguyen, Cuong Pham. "Masked face recognition with convolutional neural networks and local binary patterns", Applied Intelligence, 2021 Publication	<1%
3	opencv.org Internet Source	<1%
4	"Computer Vision – ECCV 2018", Springer Nature America, Inc, 2018 Publication	<1%
5	Submitted to Middle East College of Information Technology Student Paper	<1%
6	www.mdpi.com Internet Source	<1%
7	"Neural Information Processing", Springer Science and Business Media LLC, 2017 Publication	<1%

8	apessay.elementfx.com Internet Source	<1 %
9	Submitted to University of Portsmouth Student Paper	<1 %
10	medium.com Internet Source	<1 %
11	Wei Zhou, XiaoWei Yuan, Wenjun Chai, Hui Ma. "Deep Learning Based Attack On Social Authentication System", 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2019 Publication	<1 %
12	www.ijirset.com Internet Source	<1 %
13	Lv, Jiang-Jing, Xiao-Hu Shao, Jia-Shui Huang, Xiang-Dong Zhou, and Xi Zhou. "Data augmentation for face recognition", Neurocomputing, 2016. Publication	<1 %
14	Submitted to Cerritos College Student Paper	<1 %
15	Submitted to Cranfield University Student Paper	<1 %
16	www.ijraset.com Internet Source	<1 %

17	Submitted to Heriot-Watt University Student Paper	<1 %
18	Submitted to Khalifa University of Science Technology and Research Student Paper	<1 %
19	Mohamed Amine Mahjoubi, Soufiane Hamida, Oussama El Gannour, Bouchaib Cherradi, Ahmed El Abbassi, Abdelhadi Raihani. "Improved Multiclass Brain Tumor Detection using Convolutional Neural Networks and Magnetic Resonance Imaging", International Journal of Advanced Computer Science and Applications, 2023 Publication	<1 %
20	Submitted to University of Westminster Student Paper	<1 %
21	pt.scribd.com Internet Source	<1 %
22	Hossein Nejati. "Enhancement of Template-Based Face Detection by Belief Propagation in Ordered Component Search", Lecture Notes in Electrical Engineering, 2012 Publication	<1 %
23	Submitted to Monash University Student Paper	<1 %
24	ipfs.io	

	Internet Source	<1 %
25	repository.smuc.edu.et Internet Source	<1 %
26	vdoc.pub Internet Source	<1 %
27	"Communications, Signal Processing, and Systems", Springer Science and Business Media LLC, 2019 Publication	<1 %
28	Lecture Notes in Computer Science, 2012. Publication	<1 %
29	Submitted to Nanyang Technological University Student Paper	<1 %
30	Viet-Duy Nguyen, Minh Tran, Jiebo Luo. "Exploring Facial Differences in European Countries Boundary by Fine-Tuned Neural Networks", 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018 Publication	<1 %
31	Submitted to Bahcesehir University Student Paper	<1 %
32	Submitted to University of Lincoln Student Paper	<1 %

33	Submitted to University of Southampton Student Paper	<1 %
34	amachinelearningjourney.wordpress.com Internet Source	<1 %
35	Submitted to RMIT University Student Paper	<1 %
36	arnabsaha1.medium.com Internet Source	<1 %
37	github.com Internet Source	<1 %
38	Submitted to Gisma University of Applied Sciences GmbH Student Paper	<1 %
39	Submitted to Indian Institute of Technology Jodhpur Student Paper	<1 %
40	Lu Zhang, Xin Li, Ying Tang, Jingjie Xin, Shanguo Huang. "A survey on QoT prediction using machine learning in optical networks", Optical Fiber Technology, 2022 Publication	<1 %
41	Submitted to Universiti Tenaga Nasional Student Paper	<1 %
42	Submitted to Associatie K.U.Leuven Student Paper	<1 %

43	Submitted to City University of Hong Kong Student Paper	<1 %
44	Manohar Kuse, Sunil Prasad Jaiswal, Shaojie Shen. "Deep-mapnets : A residual network for 3D environment representation", 2017 IEEE International Conference on Image Processing (ICIP), 2017 Publication	<1 %
45	Meng Zhang, Rujie Liu, Daisuke Deguchi, Hiroshi Murase. "Masked Face Recognition With Mask Transfer and Self-Attention Under the COVID-19 Pandemic", IEEE Access, 2022 Publication	<1 %
46	Qifeng Shen, Linfeng Jiang, Huilin Xiong. "Person Tracking and Frontal Face Capture with UAV", 2018 IEEE 18th International Conference on Communication Technology (ICCT), 2018 Publication	<1 %
47	Ximin Cai, Fangyu Hu, Lizhi Ding. "Detecting Abnormal Behavior in Examination Surveillance Video with 3D Convolutional Neural Networks", 2016 6th International Conference on Digital Home (ICDH), 2016 Publication	<1 %
48	www.gurobi.com Internet Source	<1 %

49	Ivan William, De Rosal Ignatius Moses Setiadi, Eko Hari Rachmawanto, Heru Agus Santoso, Christy Atika Sari. "Face Recognition using FaceNet (Survey, Performance Test, and Comparison)", 2019 Fourth International Conference on Informatics and Computing (ICIC), 2019 Publication	<1 %
50	"Biometric Recognition", Springer Science and Business Media LLC, 2017 Publication	<1 %
51	Kin Wai Cheuk, Yin-Jyun Luo, B T Balamurali, Gemma Roig, Dorien Herremans. "Regression-based Music Emotion Prediction using Triplet Neural Networks", 2020 International Joint Conference on Neural Networks (IJCNN), 2020 Publication	<1 %
52	Savath Saypadith, Supavadee Aramvith. "Real-Time Multiple Face Recognition using Deep Learning on Embedded GPU System", 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2018 Publication	<1 %
53	Yanfei Liu, Junhua Chen. "Unsupervised face Frontalization for pose-invariant face recognition", Image and Vision Computing, 2020	<1 %

	Publication	
54	ijariie.com Internet Source	<1 %
55	morioh.com Internet Source	<1 %
56	pu.edu.pk Internet Source	<1 %
57	upcommons.upc.edu Internet Source	<1 %
58	Geetika Singh, Indu Chhabra. "Chapter 3 Genetic Algorithm Implementation to Optimize the Hybridization of Feature Extraction and Metaheuristic Classifiers", Springer Science and Business Media LLC, 2018 Publication	<1 %
59	Haochen Zou, Kun Xiang. "A Novel Rigorous Measurement Model for Big Data Quality Characteristics", 2022 IEEE International Conference on Big Data (Big Data), 2022 Publication	<1 %
60	Muhammad Pajar Kharisma Putra, Wahyono -. "A Novel Method for Handling Partial Occlusion on Person Re-identification using Partial Siamese Network", International Journal of Advanced Computer Science and Applications, 2021	<1 %

	Publication	
61	Natalia Nikoloulopoulou, Isidoros Perikos, Ioannis Daramouskas, Christos Makris, Povilas Treigys, Ioannis Hatzilygeroudis. "A Convolutional Autoencoder Approach for Boosting the Specificity of Retinal Blood Vessels Segmentation", Applied Sciences, 2023 Publication	<1 %
62	Suting Chen, Xin Li, Yanyan Zhang, Rui Feng, Chuang Zhang. "Local Deep Hashing Matching of Aerial Images Based on Relative Distance and Absolute Distance Constraints", Remote Sensing, 2017 Publication	<1 %
63	Uricar, Michal, Vojtech Franc, Diego Thomas, Akihiro Sugimoto, and Vaclav Hlavac. "Real-time multi-view facial landmark detector learned by the structured output SVM", 2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), 2015. Publication	<1 %
64	academic.oup.com Internet Source	<1 %
65	duoduokou.com Internet Source	<1 %

66	wandb.ai Internet Source	<1 %
67	www.diva-portal.se Internet Source	<1 %
68	www.science.gov Internet Source	<1 %
69	Walid Hariri. "Efficient Masked Face Recognition Method during the COVID-19 Pandemic", Research Square, 2020 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Ronald Koh Lee Xiang
ID Number(s)	19ACB01567
Programme / Course	Computer Science
Title of Final Year Project	Identity Prediction with Uncovered Facial Features while Wearing Mask

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 8 % **Similarity by source** Internet Sources: 4 % Publications: 5 % Student Papers: 3 %	I support submission of the report as the similarity index is <20%.
Number of individual sources listed of more than 3% similarity: 0	

Parameters of originality required and limits approved by UTAR are as Follows:
- **(i) Overall similarity index is 20% and below, and**
- **(ii) Matching of individual sources listed must be less than 3% each, and**
- **(iii) Matching texts in continuous block must not exceed 8 words**

Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Ashvaany Egambaram

Date: 26/04/2023

Signature of Co-Supervisor

Name: _____

Date: _____

Bachelor of Computer Science (Honours)
Faculty of Information and Communication Technology (Kampar Campus), UTAR

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	19ACB01567
Student Name	Ronald Koh Lee Xiang
Supervisor Name	Dr. Ashvaany a/p Egambaram

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
-	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)
Date: 25/4/2023