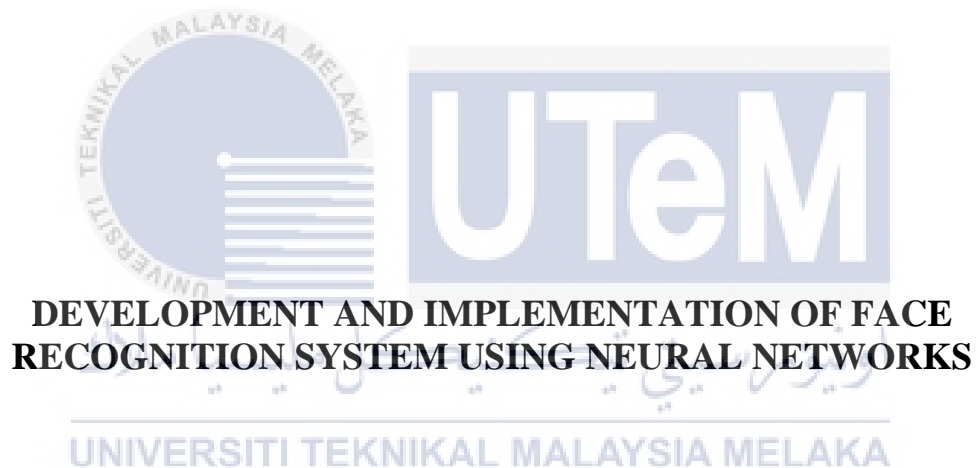




**Faculty of Electrical and Electronic Engineering Technology**



**CHANG KAI XIN**

**Bachelor of Computer Engineering Technology (Computer Systems) with Honours**

**2021**

**DEVELOPMENT AND IMPLEMENTATION OF FACE RECOGNITION SYSTEM  
USING NEURAL NETWORKS**

**CHANG KAI XIN**

**A project report submitted  
in partial fulfillment of the requirements for the degree of  
Bachelor of Computer Engineering Technology (Computer Systems) with Honours**



**Faculty of Electrical and Electronic Engineering Technology**

**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

**2021**

## DECLARATION

I declare that this project report entitled “Development and Implementation of Face Recognition System using Neural Networks” is the result of my own research except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature

:



Student Name

:

CHANG KAI XIN

Date

:

10/01/2022

اونيورسيتي تیکنیکل ملیسيا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## APPROVAL

I hereby declare that I have checked this project report and in my opinion, this project report is adequate in terms of scope and quality for the award of the degree of Bachelor of Computer Engineering Technology (Computer Systems) with Honours.

Signature

:

DR. JAMIL ABEDALRAHIM JAMIL ALSAYAYDEH  
Pensyarah Kanan  
Jabatan Teknologi Kejuruteraan Elektronik dan Komputer  
Fakulti Teknologi Kejuruteraan Elektrik Dan Elektronik  
Universiti Teknikal Malaysia Melaka (UTeM)



Supervisor Name

:

DR. JAMIL ABEDALRAHIM JAMIL ALSAYAYDEH

Date

:

10/01/2022

Signature

:

اونيورسيتي تيكنيكل مليسيا ملاك

Co-Supervisor

:

Name (if any)

Date

:

## DEDICATION

*To my beloved mother, Kho Yeok Peng, and father, Chang Choo Khean,  
and  
To my sister, Chang Kai Shan and  
My brother, Chang Kai Yuan.*



## ABSTRACT

One of the most widely used technologies in the world today is the facial recognition. It is used in biometric security systems to identify a person digitally before granting the access to the system or the data in it. The news of a female being kidnapped and murdered by a police officer in London has become a global topic and many are concerned about the safety of women in daily life. The researcher's motive is to help females and children under threat so that they can be rescued before it is too late. The objectives of this project are, to develop a device that will capture the image of a kidnapper as evidence for future reference and send the captured image to the family of the victim through email, to design a face recognition system to be used in searching kidnap suspects and to determine the best training parameters for the convolution neural network layers used by the proposed face recognition system. The system is divided into two parts, the hardware, and the software, where the hardware part consists of the ESP32-CAM programmed by Arduino IDE, which can capture image of the kidnapper and send it to the email of the victim's family through SMTP server; and the software which is a face recognition system built in MATLAB to match the image captured by the hardware with the faces stored in a database that resembles the database of the authorities. The system is tested with different images captured by the hardware and the software is able to recognise and compare the face with the image database and lastly provide the name or identity of the person. The hardware device of the system is small and compact to be carried around with ease, it can be attached to random items so that the user can easily trigger the button on the device to capture the image of the other person when they are in danger. The best training parameters for the proposed CNN model are kernel size of 5x5, 32 and 64 filters for first and second convolutional layers and learning rate of 0.001. The proposed system is robust as its overall face recognition accuracy is 98.48%.

## ***ABSTRAK***

Sistem pengesanan wajah merupakan salah satu teknologi yang banyak diguna pada masa ini. Sistem tersebut digunakan dalam sistem keselamatan biometrik untuk mengenalpasti identiti seseorang secara digital sebelum memberi kebenaran untuk mengguna sistem atau data yang dilindungi oleh sistem. Berita mengenai seorang wanita diculik dan dibunuh oleh seorang pegawai polis di London telah menjadi topik global dan terdapat ramai yang risau tentang keselamatan wanita dalam kehidupan seharian. Tujuan projek ini adalah untuk membantu wanita dan kanak-kanak di bawah ancaman supaya mereka dapat diselamatkan sebelum terlambat. Objektif projek ini adalah untuk membina sistem yang dapat mengambil gambar penculik dan menghantar gambar tersebut kepada keluarga mangsa untuk diserahkan kepada pihak berkuasa dan disimpan sebagai rujukan masa depan, mengaturcarakan sistem pengesanan wajah untuk mencari suspek penculik, serta untuk mencari parameter yang terbaik untuk lapisan rangkaian saraf konvolusi yang digunakan dalam sistem cadangan projek ini. Sistem ini dibahagikan kepada dua bahagian, iaitu hardware yang terdiri daripada ESP32-CAM diaturcarakan dengan Arduino IDE, komponen ini digunakan untuk mengambil gambar penculik dan hantar gambar tersebut ke akaun emel keluarga mangsa melalui SMTP dan software yang mengandungi sistem pengesanan wajah yang dibina dengan penggunaan MATLAB untuk memadamkan gambar yang diambil oleh hardware dengan gambar yang disimpan dalam pangkalan data. Sistem pengesanan wajah yang diimplementasikan dalam bahagian software telah diuji dengan pelbagai gambar yang diambil oleh hardware. Sistem telah berjaya mengenalpasti identiti seseorang selepas perbandingan antara gambar yang diambil dengan gambar yang disimpan dalam pangkalan data. Bahagian hardware untuk sistem ini adalah kecil dan senang dibawa, ia boleh dipasangkan atas pelbagai barang supaya pengguna boleh tekan butang yang akan menghidupkan sistem untuk mengambil gambar pengancam apabila mereka berada dalam situasi yang bahaya. Parameter yang terbaik untuk lapisan rangkaian saraf konvolusi adalah saiz penapis 5x5, 32 dan 64 penapis untuk lapisan konvolusi pertama dan kedua serta kadar pembelajaran 0.001. Sistem yang dicadangkan dalam projek ini adalah teguh kerana kadar pengesanan wajahnya mencapai 98.48% secara keseluruhan.

## ACKNOWLEDGEMENTS

My highest appreciation goes to my parents and family members for their love and support during the period of my study. Next, I would like to express my gratitude to my supervisor, Dr Jamil Abedalrahim Jamil Alsayaydeh for his precious guidance and patience throughout this project.





## TABLE OF CONTENTS

	PAGE
<b>DECLARATION</b>	
<b>APPROVAL</b>	
<b>DEDICATIONS</b>	
<b>ABSTRACT</b>	i
<b>ABSTRAK</b>	ii
<b>ACKNOWLEDGEMENTS</b>	iii
<b>TABLE OF CONTENTS</b>	i
<b>LIST OF TABLES</b>	iii
<b>LIST OF FIGURES</b>	iv
<b>LIST OF SYMBOLS</b>	vi
<b>LIST OF ABBREVIATIONS</b>	vii
<b>LIST OF APPENDICES</b>	viii
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Background	1
1.2 Problem Statement	2
1.3 Project Objective	2
1.4 Scope of Project	3
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>4</b>
2.1 Introduction	4
2.2 Factors that Affect the Accuracy of Face Recognition Systems	5
2.2.1 Aging	5
2.2.2 Facial Expressions	5
2.2.3 Partial Occlusion	6
2.2.4 Pose Variance	7
2.2.5 Illumination	7
2.3 Structure of Face Recognition Systems	8
2.4 Face Recognition Techniques	10
2.4.1 Eigenface	10
2.4.2 Neural Networks	12
2.4.2.1 Convolutional Neural Networks (CNN)	15
2.4.3 Hidden Markov Model	19
2.4.4 Support Vector Machine (SVM)	21
2.5 Summary	22

<b>CHAPTER 3</b>	<b>METHODOLOGY</b>	<b>26</b>
3.1	Introduction	26
3.2	Methodology	26
3.2.1.1	Equipment	31
3.3	Circuit Design	35
3.4	Expected Outcomes	35
3.5	Summary	37
<b>CHAPTER 4</b>	<b>RESULTS AND DISCUSSIONS</b>	<b>38</b>
4.1	Introduction	38
4.2	Hardware Testing	38
4.3	Image Preprocessing	41
4.4	Custom Face Dataset	43
4.5	Training the CNN Model	44
4.6	Software Testing	51
4.7	Results and Analysis	53
4.8	Conclusion	59
<b>CHAPTER 5</b>	<b>CONCLUSION AND RECOMMENDATIONS</b>	<b>60</b>
5.1	Conclusion	60
5.2	Limitation of proposed system	61
5.3	Project Potential of Commercialization	61
5.4	Future Research	62
<b>REFERENCES</b>		<b>63</b>
<b>APPENDICES</b>		<b>68</b>

## LIST OF TABLES

<b>TABLE</b>	<b>TITLE</b>	<b>PAGE</b>
Table 2.1	Results of Face Recognition Techniques Based on Eigenface	22
Table 2.2	Results of Face Recognition Techniques Based on NN	23
Table 2.3	Results of Face Recognition Techniques Based on CNN.	24
Table 2.4	Results of Face Recognition Techniques Based on HMM.	25
Table 2.5	Results of Face Recognition Techniques Based on SVM.	25
Table 4.1	The effect of initial learning rate on CNN of kernel size 5x5 and number of filters = 64 on both CONV layers.	46
Table 4.2	The effect of initial learning rate on CNN of kernel size 5x5 and different number of filters on the CONV layers.	47
Table 4.3	The effect of initial learning rate on CNN of kernel size 5x5 and number of filters = 64 on both CONV layers.	48
Table 4.4	The effect of initial learning rate on CNN of kernel size 5x5 and different number of filters on the CONV layers.	49
Table 4.5	Layers of the proposed CNN model.	52
Table 4.6	Face recognition result of Custom Face Dataset.	55
Table 4.7	Face recognition result of Dataset from reference [31].	55
Table 4.8	Face recognition result of AT&T database.	56
Table 4.9	Comparison between the face recognition accuracy of different face database.	57
Table 4.10	Effect of distance between the face and the ESP32-CAM on face detection rate and face recognition accuracy.	59

## LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 2.1	Happy expression shown by three people [5].	6
Figure 2.2	Development of a robust face recognition system [6].	8
Figure 2.3	Face recognition system with pre-processing step [4].	9
Figure 2.4	Dimensionality reduction by PCA [6].	11
Figure 2.5	Structure of Neural Network [6].	12
Figure 2.6	Facial features in frontal view and 5-state HMM [3].	20
Figure 2.7	Hidden Markov Model based face recognition process [4].	20
Figure 3.1	Flowchart of creating a custom face dataset.	27
Figure 3.2	Block Diagram for ESP32-CAM.	28
Figure 3.3	ESP32-CAM programming flowchart.	29
Figure 3.4	Face recognition system programming flowchart.	30
Figure 3.5	Hardware needed for this project.	31
Figure 3.6	ESP32-CAM	32
Figure 3.7	FTDI Adapter.	33
Figure 3.8	MATLAB.	34
Figure 3.9	Arduino IDE.	34
Figure 3.10	Circuit design of this project.	35
Figure 3.11	Expected outcome of image processing.	36
Figure 3.12	Third expected outcome, for face detection.	36
Figure 3.13	Known and unknown faces to the system.	37
Figure 4.1	Hardware setup.	39
Figure 4.2	Prototype of the proposed system.	39
Figure 4.3	The serial monitor of Arduino after the button is pressed.	40

Figure 4.4 Image taken by the ESP-32 CAM is sent to an assigned email address.	40
Figure 4.5 Image taken by the ESP-32 CAM.	41
Figure 4.6 The image is converted into grayscale and the face is detected.	42
Figure 4.7 The detected face is cropped and resized into 112x92 pixels.	42
Figure 4.8 Image properties after image pre-processing.	42
Figure 4.9 Custom dataset created by collecting images of celebrity.	43
Figure 4.10 Custom dataset created by collecting images of the researcher.	44
Figure 4.11 Validation accuracy and average elapsed training time of the CNN model with different training parameters.	50
Figure 4.12 Training progress of the chosen CNN structure.	52
Figure 4.13 Face recognition results for images from custom face dataset.	53
Figure 4.14 Face recognition results for images from dataset by reference [31].	53
Figure 4.15 Face recognition results for images from AT&T database.	54
Figure 4.16 Face recognition results for image taken by ESP-32 CAM.	54
Figure 4.17 Cropped faces after image processing.	58

## LIST OF SYMBOLS



## LIST OF ABBREVIATIONS

2D	-	2-Dimensional
3D	-	3-Dimensional
ANN	-	Artificial Neural Networks
BP-ANN	-	Backpropagation Artificial Neural Networks
CNN	-	Convolutional Neural Networks
CONV	-	Convolution Layers
DGWT	-	Discrete Gabor Wavelet Transform
DT	-	Decision Tree
EM	-	Expectation-Maximalization
ENN	-	Elman Neural Networks
FC	-	Fully Connected Layers
FFBPL	-	Feed Forward Back Propagation Learning
HMM	-	Hidden Markov Models
HOG	-	Histogram of Oriented Gradient
HSI	-	Hue, Saturation, Intensity
ICA	-	Independent Component Analysis
ICP	-	Iterative Closest Point
KNN	-	K Nearest Neighbour
LBPH	-	Local Binary Pattern Histograms
MLP	-	Multilayer Perceptron
NN	-	Neural Networks
PCA	-	Principal Component Analysis
POOL	-	Pooling Layers
ReLu	-	Rectified Linear Unit
RGB	-	Red Green Blue
ROI	-	Region of Interest
RPN	-	Region Proposal Networks
SMTP	-	Simple Mail Transfer Protocol
SSID	-	Service Set Identifier
SVM	-	Support Vector Machine

## LIST OF APPENDICES

APPENDIX	TITLE	PAGE
Appendix A	Gantt Chart for Final Year Project I	68
Appendix B	Gantt Chart for Final Year Project II	69
Appendix C	Program for Training the Proposed CNN model	70
Appendix D	Program for Testing the Proposed Face Recognition System	71
Appendix E	Program for ESP32-CAM	72





# CHAPTER 1

## INTRODUCTION

### 1.1 Background

One of the most widely used technologies in the world today is the facial recognition technology. With the use of facial recognition, a biometric system can identify a person digitally before granting the access to it or the data in it. Complicated and unrealistic biometric security systems are often portrayed by computer graphics in many futuristic movies; however, Apple had taken one step forward and release a face unlock feature for its iPhone X in 2017. This breakthrough uses a sensor to scan the face of the user and save it as the face ID. The phone can be unlocked when the face of the person unlocking the phone matched with the face ID. The release of this new authentication method has made a big impact in the smartphone industry and all the latest smartphones have started to implement the same face unlock feature in their systems. Facial recognition technology is also used in other systems, for instance, it is used in airport security, law enforcement, attendance systems and to search for a person.

In this project, neural network is used in the development and implementation the face recognition system. Neural network can be trained to process and analyse data, recognise patterns, and make prediction about specific operations. Generally, the programmer needs to provide numerous examples to train neural network, in order for it to learn the patterns [1]. This project proposed a face recognition system to be used for searching a person who is being suspected of committing crime. The proposed system works with a hardware device that captures the face of the suspect.

## 1.2 Problem Statement

There are many kidnappings or abduction cases happen around us. In early March 2021, there was a news that caused global anger, Sarah Everard, a 33-year-old female, was kidnapped and murdered by a police officer in London. Females and children are often the target of kidnappers, the fate of the victim remains unknown until the suspect is found and arrested. In the cases of victims being rescued in time by the authorities, the victims who suffer from post-traumatic stress disorder (PTSD), it is likely that they could not recognise or remember the face of the kidnapper during their testimony in court. Therefore, the purpose of this project is to develop a device to capture the image of the suspect as evidence and alert the victim's family through email when the victim triggered the device and use the image as an input to the face recognition system to identify the suspect.

## 1.3 Project Objective

The main aim of this project is to propose an idea which combines hardware with image capturing features and software with face recognition system, the aim is to capture the image of a kidnapper and reduce the time taken for the authorities to find the suspect. The objectives of this project are listed below:

- i) To develop a device that will capture the image of kidnappers as evidence for future reference and send the captured image to the family of the victim through email.
- ii) To design a face recognition system to be used in searching kidnap suspects.
- iii) To determine the best training parameters for the convolution neural network layers used by the proposed face recognition system.

## 1.4 Scope of Project

The scope of this project are as follows:

- a) Hardware which includes the ESP32-CAM to capture image of a kidnapper and send the image to the victim's family.
- b) Software which implements the face recognition system.
- c) Image database with images separated into training set, validation set and testing set.
- d) Comparison between input image and database image to find the identity.



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

Being one of the most widely discussed topics in the research of computer-related field, face recognition technology has become a close to perfect solution to accommodate the demands of identification and verification of identity claims in systems or organizations that use biometric security systems. With the latest face recognition technology, many tasks which used to depend on manual internal control environments in the past such as transactions, building access control and security-related tasks have overcome some limitations since the machines that use face recognition technology are able to provide higher accuracy in identifying and verifying an identity claim [2], the implementation of face recognition technology in such systems and organizations has greatly increased productivity as the time taken for identification and verification is reduced. Besides, there are some flaws in traditional recognition systems. For instance, systems that use PINs and passwords for authentication are less user-friendly for people who are forgetful because there are chances that the PINs and passwords are forgotten by the users. This method is also less reliable because passwords that are not updated regularly are more likely to be hacked and leaked by attackers, whereas access cards, keys, tokens and the other physical devices can be misplaced, stolen or duplicated. As a result, the traditional recognition systems are more vulnerable and less secure. On the other hand, biological characteristics and traits of an individual are distinctive and unique in their own way, facial features cannot be stolen or misplaced like physical items. Furthermore, even if an individual forgets how he or she looks, the face recognition system would still recognise its user because the data is saved in

the database. Face recognition technology is more favourable than other biometric systems because, unlike fingerprint and iris recognition systems, face recognition systems do not require physical contact to activate the identification and verification processes [3].

## **2.2 Factors that Affect the Accuracy of Face Recognition Systems**

According to [3], the factors that might affect the accuracy of face recognition systems are classified into two main categories, intrinsic and extrinsic factors. Physical conditions of the human faces are considered as intrinsic factors, for example, aging and facial expressions. Extrinsic factors are made up of partial occlusion, pose variance and illumination. Reference [3] also mentioned that biological characteristics in a biometric system must be consistent and permanent because they act as the key to identify or authenticate a person.

### **2.2.1 Aging**

First of all, aging is an inevitable natural growth process in every human's life. It is uncontrollable and every individual experience it differently, depending on the genes and other external factors [4]. When people go through the aging process, the shape and texture of their faces might slightly change and thus the accuracy of face recognition will be lower. It is difficult to collect the data regularly and keep the system up to date because the speed of aging differs from person to person.

### **2.2.2 Facial Expressions**

Next, one of the most used non-verbal ways we use in daily communication is using facial expressions. It is easy for human brains to identify and recognise a person even when the person shows different facial expressions, but it is still a big challenge for computer to

perform the same task. More researchers have taken this factor into consideration and many solutions are proposed to improve the accuracy of the face recognition system, however, it only works in controlled environments. Research conducted by [5] states that it is difficult to separate the feature spaces of all possible expressions, to put it another way, when a person makes two different expressions, the facial feature may be very close in the feature space, while the facial features of two people making the same facial expression may be far from each other, in short, the feature extraction is very complicated when dealing with various facial expressions. Moreover, in some cases, the facial expressions for two different emotions, for instance, sad and fear, can be very similar [5].

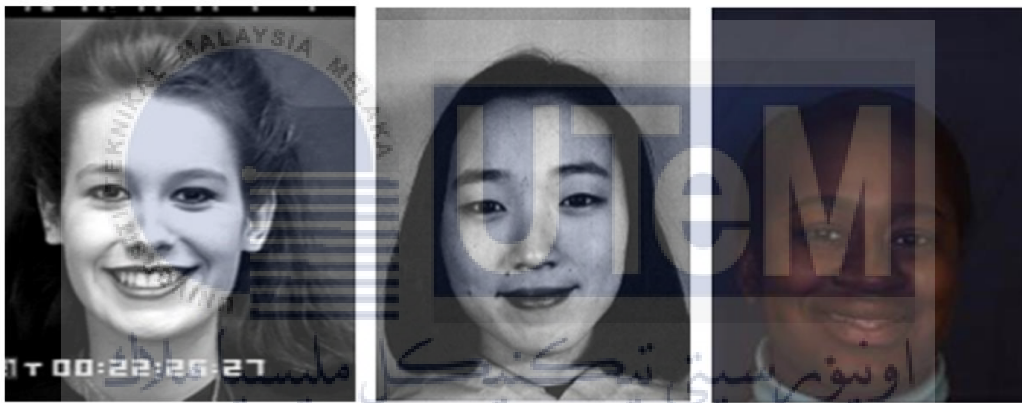


Figure 2.1 Happy expression shown by three people [5].

From Figure 2.1, without mentioning the differences in lighting and background, it is clear that the images vary from each other because the way that each person smiles is unique, despite the fact that all of these expressions convey the same emotional state, which is happiness [5].

### 2.2.3 Partial Occlusion

Extrinsic factors are external factors that cause changes on the facial features. Occlusion refers to natural or artificial obstacles in an image [3]. Face masks have become part of our daily lives since the COVID-19 pandemic began, people wear face masks in

public to protect themselves and others. The face recognition systems lack the ability to recognise an identity who has his or her nose and mouth covered, for instance, some iPhone users have experienced this inconvenience, they need to take off their face masks in order to unlock their iPhone with Face ID. Personal belongings such as sunglasses, caps, scarves and face masks are the common occlusion and sometimes shadows caused by lighting or reflection from glasses, earrings, necklace and other reflective objects are also degrading the performance of the facial recognition system. In the study conducted by [3], they have stated three approaches to face recognition with partial occlusion, namely Part Based Methods, Feature Based Methods and Fractal-Based Methods. They also stated another local approach which is dividing the faces into different parts to eliminate some of the features that caused an inaccurate result in recognition.

#### **2.2.4 Pose Variance**

Next, no one will have the exact same pose when taking a picture, some people like to cover one side of their face while some like to show their full profile. Therefore, it is difficult for the face recognition system to recognize the faces from images with different poses. An ideal face recognition system should have the capability to recognize face even when the poses of the person are changing. The current solutions to this problem are general algorithms, 2D methods for face recognition across pose and face recognition across pose with the assistance of 3D models, researchers are working on this problem but there is no system with 100% accuracy available in the market yet [3].

#### **2.2.5 Illumination**

The final extrinsic factor suggested by [3] is the illumination variation. The performance of the face recognition system is affected when the lighting is too much or too

less. When image is taken in a controlled environment, that is, with the appropriate amount of lighting and uniform background, the feature extraction by the face recognition system would be easier, however in practical, the face recognition system could be used at any places, therefore it is hard to control the background, weather condition or the lighting, this situation makes it harder for face recognition whether it is from still or video images. There are three methods that can be used to handle this issue including grey level transformation, which is an image processing technique that performs in-depth mapping with non-linear or linear function, gradient extraction technique which extracts the edges of image in grey level, and face reflection field estimation technique [3].

### 2.3 Structure of Face Recognition Systems

Reference [6] stated that a useful face recognition system must fulfil the following characteristics: firstly, it must be able to work well with both images and videos, secondly, it must be capable to process in real time, thirdly, it must be robust in illumination variation, fourthly it needs to perform its task without being affected by hair, ethnicity or gender of a person and lastly, it must be able to work with faces detected from all angles. They also stated that a robust face recognition system is made up of three basic steps: face detection, feature extraction and face recognition, the processes are shown in Figure 2.2.

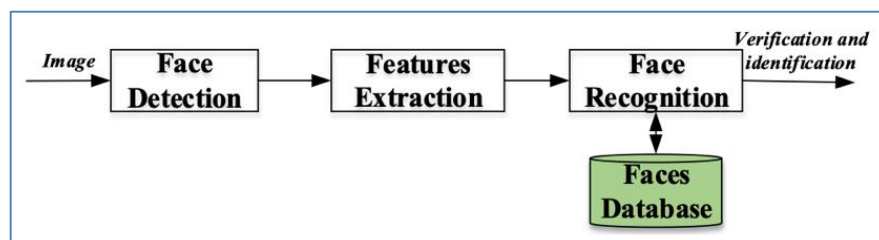


Figure 2.2 Development of a robust face recognition system [6].

Reference [6] stated that the face detection step is used to detect and locate the human face if it is present in a given input image, face detection can be done by many techniques, the most common ones are Histogram of Oriented Gradient (HOG), Viola-Jones detector and



Principal Component Analysis (PCA). As previously mentioned, there are some factors that will affect the accuracy of face recognition systems, therefore, to build a robust face recognition system, the input image must go through some pre-processing steps. Next, in the structure that [6] proposed, the function of the second step, feature extraction, is to extract the feature vectors called “signatures” which describe the prominent features of the human face detected in the first step with respective geometry distributions. In the final face recognition step, the system will make comparison between the features extracted from the second step with the known faces stored in database. Recognition is the generic term of identification and verification. Hence in this final step the system will first identify the identity of a person whom he or she claimed to be, by matching the input image with the face databases, before verifying whether the claim is true. Another structure of face recognition system is proposed by [4], the face recognition system also comprises three modules: pre-processing, feature selection and classification. This structure contains the pre-processing steps mentioned by [6] and the implementation is shown in Figure 2.3.

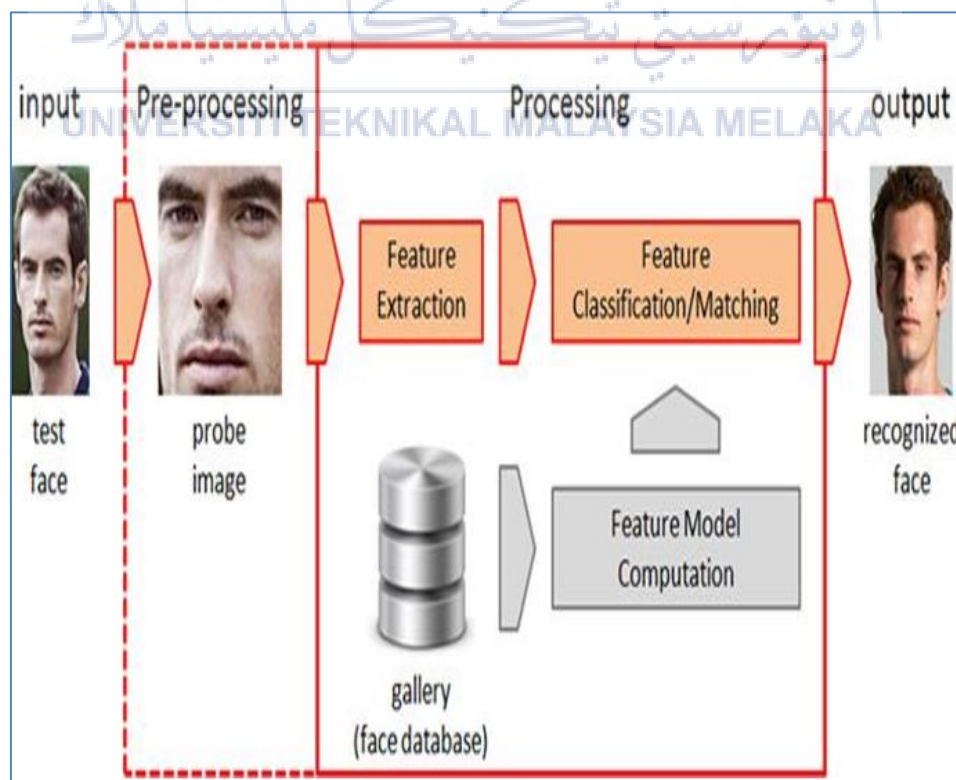


Figure 2.3 Face recognition system with pre-processing step [4].

Figure 2.3 shows the structure of face recognition system proposed by [4] that contains pre-processing step as mentioned by [6] in making a robust face recognition system. Typically, pre-processing is used to reduce the illumination variation [3], so that all input images have the similar condition in terms. In the system proposed by [7], the pre-processing step was included as well, according to them, the input image which has a human face will be passed to face detector module to detect the face and segment it as region of interest (ROI) and after that the ROI will be resized into preretinal size for alignment purpose.

## **2.4 Face Recognition Techniques**

There are various techniques that can be used for face recognition, the most widely used are Eigenface, Neural Network, Hidden Markov Model (HMM) and Support Vector Machine (SVM). This project focuses on Neural Network, but other techniques will also be reviewed. A detailed analysis will be carried out and a comparison of techniques that are mentioned will be made in the end of this chapter.

### **2.4.1 Eigenface**

In 1987, the Eigenface technique was developed by Sirovich and Kirby, based on the Principal Component Analysis (PCA). This technique was then improved by Turk and Pentland in 1991 as a solution to face recognition problems. The word eigen was originated from German, which means characteristic, therefore the Eigenface is the unique characteristic features on a person's face, and it is formed from the feature extraction process, which in this technique, is known as the PCA. The PCA is used to reduce the dimensionality of a huge data set. The eigenfaces can be generated by using the PCA. The first step is computing the covariance matrix, then from the covariance matrix, the eigenvectors and eigenvalues can be obtained. These two variables are later used to identify the principal

components or, the eigenfaces. Figure 2.4 shows the process of dimensionality reduction using the PCA.

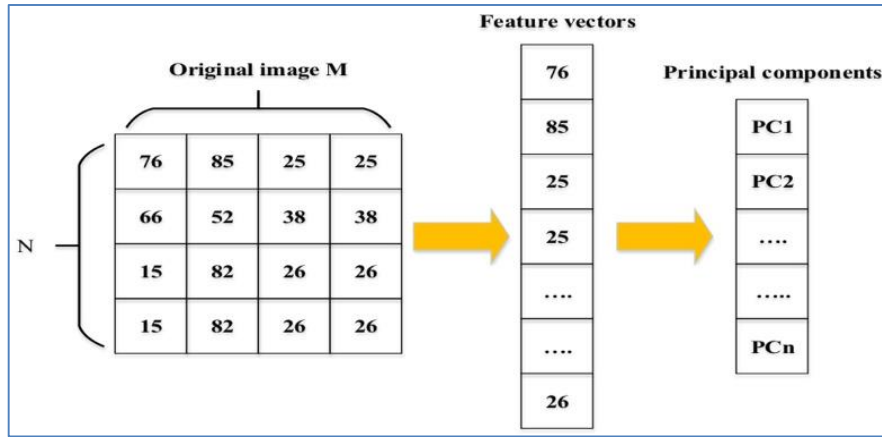


Figure 2.4 Dimensionality reduction by PCA [6].

After the eigenfaces are generated from the extraction process, they can be combined linearly to represent a face, whether it is a new or existing face. In other words, assume that there are 3 eigenfaces, a person's face might be the combination of 30% of the first eigenface, 5% of the second eigenface and 50% of the third eigenface. The recognition process is done by projecting a test image on the eigenfaces. The difference between the eigenface of the test image and the eigenfaces stored in the database are recorded to calculate the Euclidian distance. The smaller the Euclidian distance, the more the test image is identical to the image saved in the database, otherwise, the image is unrecognizable by the system [8].

Reference [9] used the eigenface technique in their research for building a face recognition software. The proposed system uses the laptop camera to capture the input image with an original size of 320 x 240 pixels. The size of the input image is then reduced to 100 x 100 pixels and saved as a master file of the face. They found out that the average accuracy for their face recognition software is 85%.

Reference [10] pointed out that the problem in using face recognition for biometric identification is its long process and the result accuracy, therefore they have proposed a

solution to make face recognition process faster and produces more accurate results. The system that they proposed is a hybrid approach which consists of the Haar Cascades and Eigenface methods. They stated that by using this hybrid approach in a single detection process, the system can detect multiple faces. The accuracy of this proposed solution was reported to be 91.67%.

### 2.4.2 Neural Networks

Artificial Neural Networks (ANN), also known as Neural Networks, is an artificial version of neural network which resembles the neural networks in human brain. They are widely used in Artificial Intelligence and are the basis of Deep Learning. Neural networks are made up of multiple layers, namely the input layer, hidden layers and output layer, each layer can have several nodes. Figure 2.5 illustrates a neural network which has a 3-node input layer, three 4-node hidden layers and a 1-node output layer.

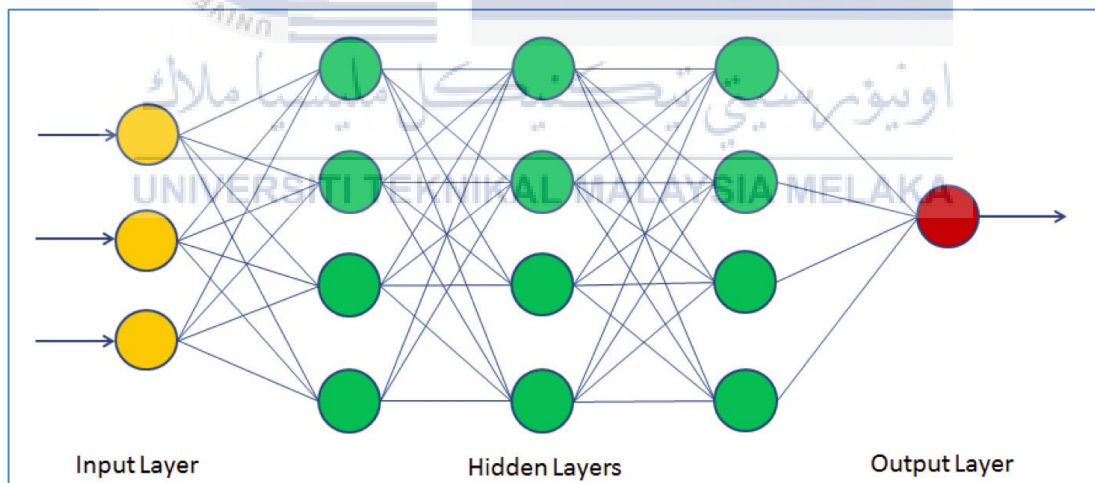


Figure 2.5 Structure of Neural Network [6].

The neural networks learn from experience, therefore, in order to make the face recognition system to achieve high accuracy, the neural network must be trained with as many training samples as possible. For instance, a training set with only the faces of two different people cannot represent a general population but a training set with the faces of 200

people can achieve a more general representation of the population [11]. Neural networks can build a more robust face recognition system as it can work fine and maintain the accuracy even when the images have different lighting conditions [4]. There are many types of Neural Networks, such as Multilayer Perceptron, Radial Basis Function Neural Networks, Recurrent Neural Networks, Convolutional Neural Networks, Modular Neural Networks etc. In the study that [4] conducted, they stated that Radial Basis Function and Feed Forward Neural Networks can be implemented as classifiers for face recognition. They also stated that the downside of Neural Networks is it requires long time to train. However, many researchers have proposed different hybrid solutions to address this issue.

Reference [12] proposed a face recognition system that uses some basic neural network models. In the proposed system, there are three modules including (1) the face detection module which consists of a three-layer feedforward artificial neural network with an activation function called Tanh that combined with AdaBoost for higher face detecting rate, (2) the face alignment module which uses multilayer perceptron (MLP) with a three-layer linear function, (3) the feature extraction module that combines the geometric feature-based method and ICA method in facial feature extraction and (4) the face matching module that is made up of multi-artificial neural networks. The author used the 11000 faces from FERET database to train the face detector and MIT + CMU test set on face detection and alignment modules and the CalTech database for the feature extraction module. 441 images were obtained from the results of face detection and alignment modules and the author divided the database into two parts where 120 images were used for training and 321 images were used for testing. The average accuracy of this proposed system was 96.84%.

Reference [13] stated that the algorithms that have been developed in recent years are not robust enough to deal with complex images. They have proposed to improve the backpropagation artificial neural network (BP-ANN) for a better performance of the face

recognition system. The proposed system was designed to recognize 10 people, 10 images of each person are used, therefore there were 100 images in the dataset. 50 images were used in training while another 50 images were used for testing purposes. The proposed system was able to recognize 41 images, yielding a success ratio of 82%.

Reference [14] suggested an algorithm for face recognition system. They used a hybrid approach to develop a more accurate face recognition system which include Elman Neural Network, Curvelet transform and HSI colour space. The proposed algorithm is made up of several steps. The first step is to create dataset, the dataset that the authors used consists of 500 images of 20 students, where each of them provided 25 images. The images were resized, and any noise exist in the images were removed. Next, in the feature extraction step, the images were decomposed by using curvelet transform to get the coefficient of transformations, then PCA was used on the coefficient of transformations to obtain the features. After that, the Elman neural network was trained, from the total of 25 images of each student, 18 images were used as training set, three images were used for validation and four images were used for testing. The optimal weight for the neural network depends on the training results. Step four is to read image to recognition; the colour space was converted from RGB to HSI and saturation layer was selected. Then, step 2 was repeated but this time it was used for extracting features from the input image. The proposed system was tested, and the resulting accuracy obtained by the authors was 94%.

Reference [15] proposed an effective method for face recognition that uses Principal Component Analysis (PCA) and four models that were trained with Feed Forward Back Propagation Learning (FFBPL) and Elman Neural Network. The first two models were trained with 40 and 50 features respectively using FFBPL while the rest used the same features but are trained with Elman Neural Networks. A total of 420 images of 10 people (each with 42 images) were selected from the ORL and used as dataset. The training set used



24 images for each person and the remaining were for the testing set. The results of FFBPL for 40 features were 98.33% and that for 50 features were 98.80%. On the other hand, the results of Elman Neural Networks were 98.33% and 95.14% for 40 and 50 features respectively. The authors have proved that the proposed method was effective and accurate.

#### **2.4.2.1 Convolutional Neural Networks (CNN)**

Convolutional Neural Network (CNN) is one of the most popular variants of neural networks because it does not require much pre-processing steps and gives accurate results especially when the number of trainings given is increased [16]. According to [17], CNN is an algorithm under the study of deep learning, which is commonly used in image analysis or image processing as it can perform feature extraction and classification as a combined task. There are three layers in the structure of CNN, namely the convolutional layers, the pooling layers and the fully connected layer. The convolution and pooling layers are responsible for the feature extraction while the fully connected layer is responsible for classification.

Reference [17] proposed a real-time face recognition system using CNN. The authors designed the CNN architecture with Keras, which is an Open-Source Neural Network library that runs on top of Tensorflow. First, the camera will capture a real-time input image and then the Viola Jones algorithm will be used on the input image to detect face and pre-processing steps took place. Next, when a face is detected in the input image, it will be cropped, converted into grayscale and resized to 120 by 120 pixels. Then input image will go through a sequence of layers including convolution (CONV) + Rectified Linear Unit (RELU), pooling layer (POOL) and DROPOUT layers. After these processes, the output will be flattened before being sent to the DENSE/FC layer for the purpose of classification. They used 400 images in total from standard AT&T database, where 320 images were used as the training set and 80 images were used as the test set. Next, since the main goal of the proposed

system is to design a real-time face recognition system, the authors have also tested real-time inputs through a camera. For this condition, they have captured 200 images of themselves and their family members, 100 images for training the system and 100 images for testing purposes. It is reported that the maximum accuracies of the proposed system are 98.74% for standard datasets and 98.00% for real-time inputs.

Reference [18] proposed an algorithm for face detection and face recognition based on Convolutional Neural Networks (CNN). The authors have developed a face recognition-based attendance system to test the efficiency of the proposed algorithm. The proposed attendance system will get the input image either it is captured through the camera that is connected to the system or it can be uploaded from the directory. The proposed algorithm is divided into two parts, the first part is to detect faces in an input image and the second part is to recognize the detected faces. In the face detection part, the authors used Region Proposal Network (RPN) to draw anchors. Anchors are some boxes that possibly contain the targeted subject, which in face detection is, the face, therefore the output will return the anchors that have high probabilities of containing the faces. Then, the face recognition uses the CNN architecture proposed by the authors, which consist of five convolution (CONV) layers and three Fully Connected (FC) layers. LFW dataset was used to train the proposed system. The authors stated that the accuracy of the proposed system on the testing data was 97.9% and claimed that their algorithm works excellently in data latency and real-time response compared to other algorithms.

Reference [19] stated that training deep models such as CNN is not suitable for dataset with only a few samples. They proposed a method to address this issue which is using a modified deep learning neural network for face recognition with small dataset. The authors have applied Gaussian and Poisson noise to the samples of the training set, after this process, the training set is said to be augmented with samples that are synthetically generated, the



purpose of doing so was to double the size of the training set. The authors claimed that the generalization power of CNN was improved by the augmented training dataset. The standard AT&T database was used for training and the accuracy of the proposed system achieved 99.6%.

Reference [7] proposed a deep convolutional neural network-based face recognition system that uses transfer learning approach. The proposed system will first perform a pre-processing step to detect the face in a captured image and mark it as the region of interest (ROI). The image was resized for alignment purpose. Next, the pre-trained Alex Network was used to extract the features from the image. The last step is the face recognition step that uses Fog and Cloud computing. The authors compared their proposed algorithm with Decision Tree (DT), K Nearest Neighbour (KNN) and Support Vector Machine algorithms and the experimental results showed that the proposed algorithm was more accurate and more precise than the other algorithms. The accuracy of the proposed algorithm was 99.06%.

Reference [20] proposed an attendance system with face recognition based on deep learning technique. CNN cascade was used for face detection and the face embeddings was generated using FaceNet CNN. The proposed system went through a few stages, first, it must have a training dataset, the dataset used in the proposed system was small as it only took some images of five volunteers. The authors used augmentation to expand the dataset so that the proposed system can achieve a higher accuracy even with a limited dataset. The first image augmentation process was done by noising and blurring the images, using Python and OpenCV. The second augmentation process was done by using the Dlib to mark the position of eyes, nose, lips, and chin in the image before using Python to add random accessories to create new images for the training dataset. After the image augmentation processes, a face recognition model was developed, this model includes face detector with CNN cascade, face landmarks and image positioning, face embeddings with FaceNet CNN and SVM classifier.

The overall accuracy obtained by the proposed system in a real-time environment was 95.02%.

Reference [21] proposed an intelligent CNN-based face recognizing attendance system that can identify several people simultaneously. The algorithms used for the proposed system are Histogram of Oriented Gradients (HOG) face search algorithm, face landmark estimation-based face projection algorithm, deep CNN face encoding algorithm and an algorithm that uses SVM Classifier to find the name of the identity recognized by the system. The proposed system was tested with three different conditions, one with frontal view, another with a side view and the last with downwards view. The accuracies obtained for these three conditions were 81.25%, 75.00% and 43.75% respectively.

Reference [22] proposed a multi-face recognition system to detect the prisoners in jail. The authors used a combined method of CNN and Haar Cascade Classifier in their proposed system. The authors collected two different sets of videos and images, the first set was for videos and images that were taken in low light conditions and the second set was for otherwise. The images and videos were pre-processed to be changed to grayscale, binarized and cropped. The objectives are to eliminate noise and remove parts that are not needed for the system. After the pre-processing steps, Haar Cascade Classifier method was used to extract the features detected in the image. Then, the face recognition step uses CNN made up of 19 layers in total, where 13 are the convolutional layers and three layers for hidden layers and the last three layers for the fully connected layers. The proposed system was tested, and the authors reported that the accuracy was 87%.

Reference [23] proposed a face recognition using CNN. The CNN proposed by the authors consists of seven layers: INPUT-CONV-POOL-CONV-POOL-FC-FC, and ReLu was used as the activation function. The ORL database was used in the experiment and the highest accuracy achieved by the proposed system was 98.3%. The authors compared the

proposed system with other algorithms such as Principal Component Analysis (PCA), Local Binary Pattern Histograms (LBPH) and K-Nearest Neighbour (KNN), among these algorithms, the proposed system gave the best result in terms of accuracy.

Reference [24] proposed a face recognition system that can identify an individual in all possible conditions that might affect the accuracy of the face recognition system. They used deep convolutional neural network to extract the important features in an image then a filter to select, store the indices and compare the indices of the most significant features. The selected features will be subtracted from the original image and the system will search for the identity. They used two different databases, the LFW dataset and YouTube Faces DB to test the accuracy of the proposed system and the results were 99.7% and 94.02% respectively.

Reference [25] proposed a home security system that uses face recognition technology. The face recognition technology was developed based on CNN and was used in the door locking system. The authors divided the research into three stages, where the first stage was collecting data from the homeowner, the second stage was the data training process, and the last stage was face recognition process using Raspberry Pi with CNN installed in it. The Raspberry Pi was used as a microcontroller so that when the face of the homeowner was detected, the door will be unlocked automatically. The proposed system was able to achieve 97.5% accuracy.

### **2.4.3 Hidden Markov Model**

Hidden Markov Model (HMM) is a statistical model for characterizing the properties of signal [26]. A 5-state HMM is usually used for face recognition system because there are five facial features that can be seen in frontal view, including the forehead, eyes, nose, mouth and chin. Figure 2.6 shows the facial features that can be seen in a frontal view

image and a flowchart of a 5-state HMM. Figure 2.7 shows an example of face recognition system that uses HMM.

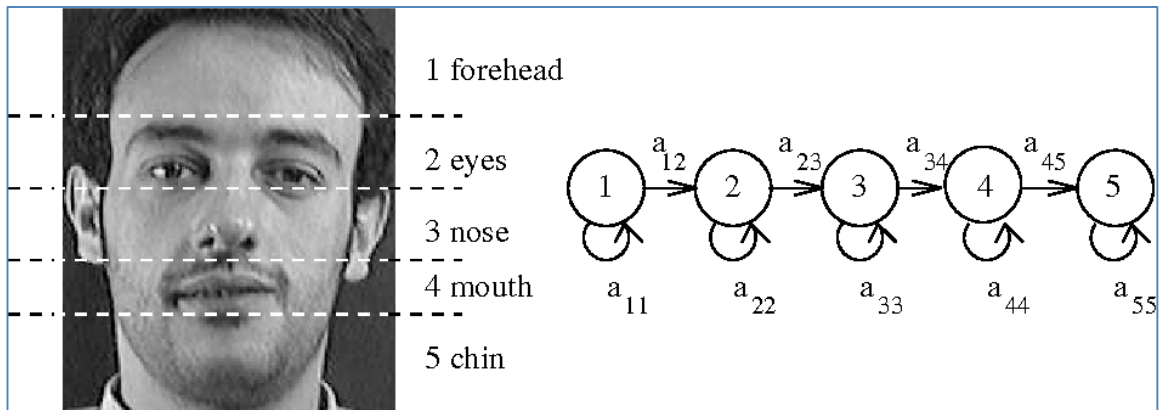


Figure 2.6 Facial features in frontal view and 5-state HMM [3].

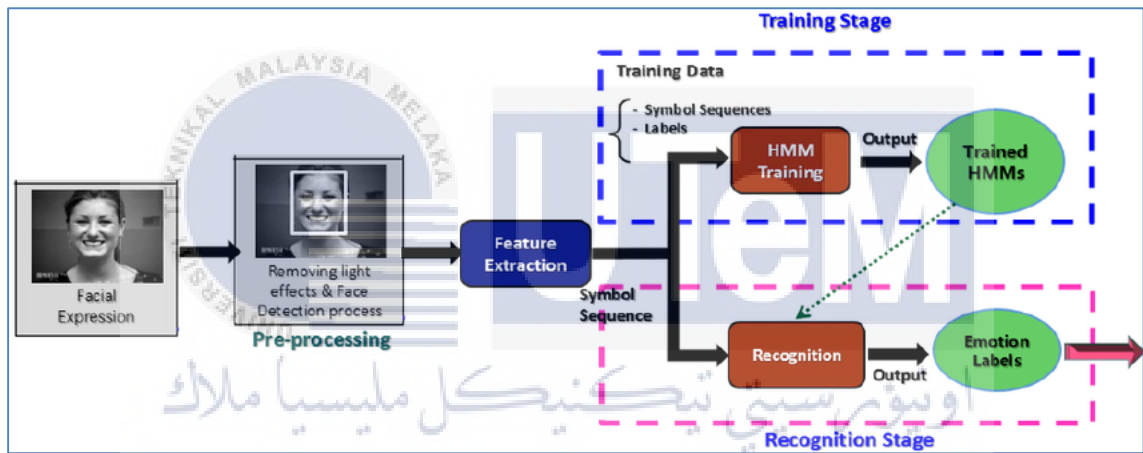


Figure 2.7 Hidden Markov Model based face recognition process [4].

Reference [27] proposed a face recognition that uses 2D Hidden Markov Model (2DHMM). Additionally, the 2DHMM was combined with an Expectation-Maximalization (EM) algorithm. Discrete Gabor Wavelet Transform (DGWT) in the feature extraction step to obtain the observation sequence vectors so that the accuracy of the proposed system can be maximized. The authors conducted experiments using the ORL database and the recognition rate obtained by the proposed system was 99%.

Reference [28] stated that the traditional HMM is not suitable for 2D and 3D image processing because part of information might be lost when the data are being converted to one-dimensional feature vectors. Thus, the author suggested a face recognition that uses two-

dimensional Hidden Markov Models. The author used FERET and UMB-DB databases for two-dimensional images; UMB-DB and FRGC databases for three-dimensional images. The recognition rates for 2D images were 93% for UMB-DB database and 95% for FERET database. Next, the recognition rates for 3D images achieved 94% for both UMB-DB and FRGC databases. Lastly, the recognition rate for 2D+3D images was 96% for both databases. The author claimed that the multimodal (2D+3D) data yield a better result, and the proposed method was faster than ICP method.

#### **2.4.4 Support Vector Machine (SVM)**

Support Vector Machine (SVM) is often used as a classifier because it works very well in classification by constructing a hyperplane in a high dimensional space. It is implemented after the feature extraction process to recognize faces [3]. SVM can be used independently or with other techniques as a hybrid approach.

Reference [29] proposed a 3D facial recognition system based on wavelet Gabor filtering and SVM. The database used by the authors was a database that contains 3D face models called BU-3DFE. The aim of the research conducted by the authors was to provide a more efficient way that can recognize people who cause harm to the society. The highest accuracy that the proposed system achieved was 97.3%. The authors observed that the quality of 3D images does not depend on lighting conditions anymore. They claimed that this observation was very important as illumination is one of the factors that will affect the accuracy of a face recognition system. They also stated that the limitations of the proposed system were the processing time of face projection and extraction time of the obtained characteristics.

Reference [30] proposed a face recognition that uses SVM with implementation of kernel as the classification method to identify lookalike faces. Two types of kernels were

used in the proposed system, namely the Radial Basis Function kernel and the polynomial kernel. The authors selected the images from the Look-Alike Faces Database (LAF), a total of 500 images were used. Both kernels had the same accuracy which is 94%. However, there was a difference between them, the running time for polynomial kernel was approximately 20 seconds shorter than the RBF kernel.

## 2.5 Summary

Table 2.1 Results of Face Recognition Techniques Based on Eigenface

Year	Method	Accuracy	Reference
2018	Haar Cascades & Eigenface	91.76%	[10]
2020	Eigenface	85%	[9]

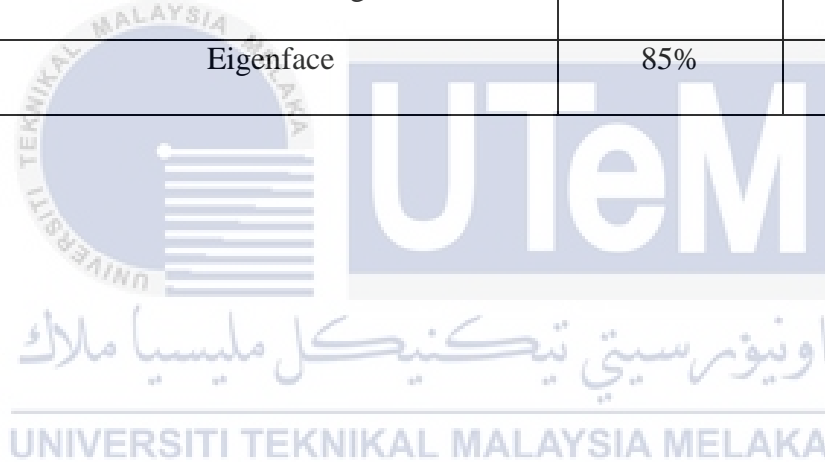


Table 2.2 Results of Face Recognition Techniques Based on NN

Year	Method	Database	Number of images in training set	Number of images in testing set	Accuracy	Reference
2011	ANN, Tanh, AdaBoost, MLP, ICA	FERET, MIT + CMU, CalTech	120	321	96.84%	[12]
2019	BP-ANN	-	50	50	82%	[13]
2019	ENN, Curvelet Transform, HSI Color Space	-	360	60 (Testing) 80 (Validation)	94%	[14]
2020	PCA, FFBPL, ENN	ORL	240	180	95.14% - 98.80%	[15]

Table 2.3 Results of Face Recognition Techniques Based on CNN.

Year	Methods	Database	Number of images in training set	Number of images in testing set	Accuracy	Reference
2017	CNN, Image Augmentation using Gaussian and Poisson Noise	AT&T	-	-	99.6%	[19]
2017	CNN cascade, FaceNet CNN, Image Augmentation with Dlib, Python & OpenCV, SVM	-	-	-	95.02%	[20]
2017	CNN	ORL	-	-	98.3%	[23]
2019	CNN, RPN	-	-	-	97.9%	[18]
2020	CNN with Keras, Viola Jones	AT&T	320 100 (Real-time environment)	80 100 (Real-time environment)	98.74% 98.00% (Real-time input)	[17]
2020	CNN with pre-trained Alex Network, Fog and Cloud Computing	-	-	-	99.06%	[7]
2020	CNN, HOG, SVM	-	-	-	43.75% - 81.25%	[21]
2020	CNN & Haar Cascade Classifier	-	-	-	87%	[22]
2020	Deep CNN, Viola Jones	CelebA, LFW, YouTube Faces	-	-	99.7%, 94.02%	[24]
2020	CNN, Raspberry Pi	-	1040	60	97.5%	[25]

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



Table 2.4 Results of Face Recognition Techniques Based on HMM.

Year	Methods	Database	Accuracy	Reference
2013	2DHMM, EM, DGWT	ORL	99%	[27]
2017	2DHMM	FERET, UMB-DB, FRGC	93% - 96%	[28]

Table 2.5 Results of Face Recognition Techniques Based on SVM.

Year	Methods	Database	Accuracy	Reference
2018	Wavelet Gabor Filtering & SVM	BU-3DFE	97.3%	[29]
2020	SVM, RBF kernel, Polynomial kernel	LAF	94%	[30]



## CHAPTER 3

### METHODOLOGY

#### 3.1 Introduction

A good face recognition system must fulfil several criteria, the most important is the accuracy of the result. Face recognition systems are widely used in law enforcement, a slightly inaccurate result might cause an innocent person being wanted and wrongly accused of crime that he or she did not commit. To test the functionality of a face recognition system, face datasets are used. There are many different face datasets available publicly such as LFW, Yale and AT&T databases. This project uses AT&T databases and a custom face dataset. This chapter explains the development of a face recognition system and the combination of the system with some hardware that will be discussed in the following section.

#### 3.2 Methodology

The proposed methodology will be implemented according to the flowcharts shown below. Figure 3.1 shows the flowchart of creating a face dataset. A custom face dataset is created because in this project, an ESP32-cam will be used to capture images of people. It is certain that during the testing process of this experiment, the face captured by the device is not in the pre-curated face databases, such as the AT&T Database that contains a total of 400 images of 40 individuals. Hence, in order to yield higher accuracy, other than using the publicly available face dataset, the researcher decided to create a custom face dataset in addition for this project.

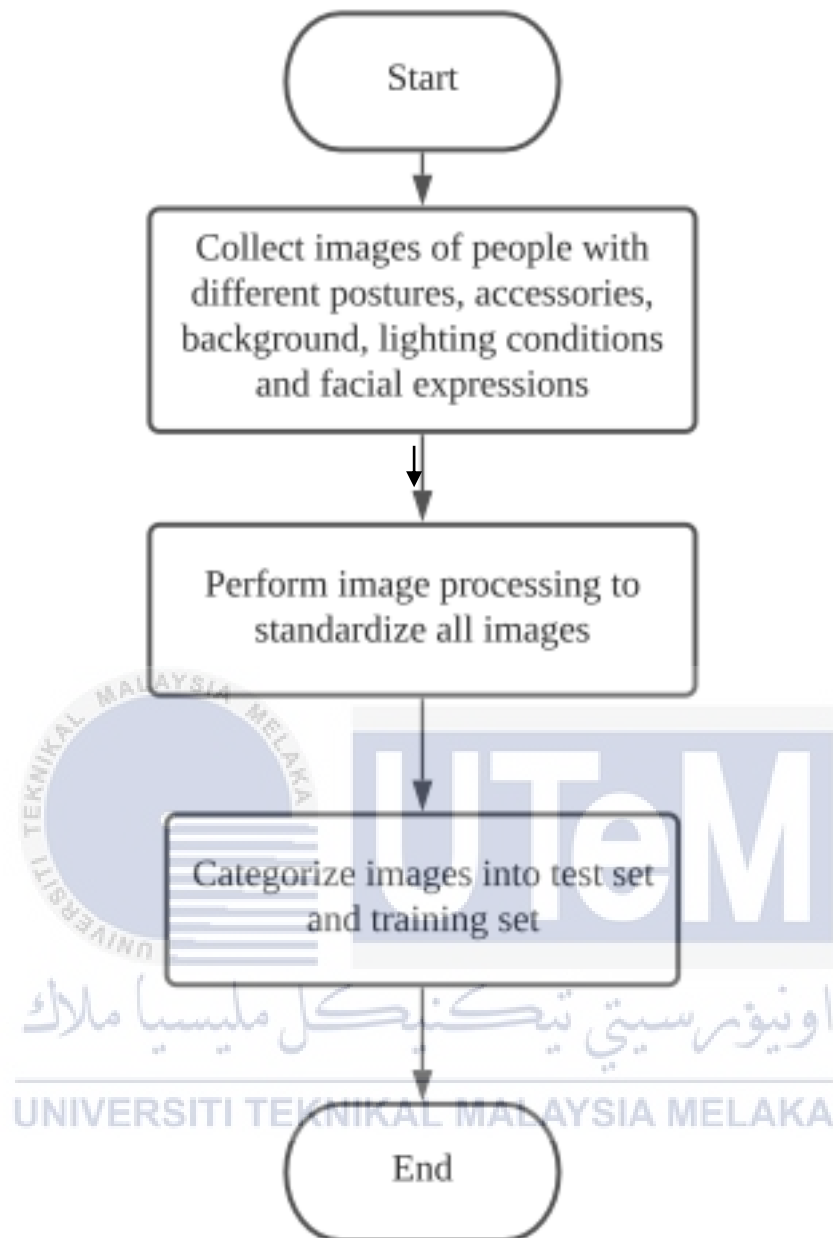


Figure 3.1 Flowchart of creating a custom face dataset.

After creating a face dataset, the face recognition system can be trained. The images in the face dataset created manually will be separated into two folders, one for testing purpose and another for training purpose. The same image cannot exist in both folders, in other words, all images of each individual in both folders must be different in terms of postures, lighting condition, facial expressions. The proposed system will be trained and

tested with the images available before testing with the image captured directly from the ESP32-CAM. The hardware implementation is shown in Figure 3.3.

Figure 3.2 shows the block diagram for ESP32-CAM while Figure 3.3 shows the programming of the hardware used in this project, which is the ESP32-CAM, using Arduino IDE. The initialization process includes assigning the SSID and password to the device so that it can connect to the WiFi or mobile hotspot and also setting up the email account that will be receiving the captured image by SMTP server. The device will be in deep sleep mode and when it is triggered by a button, it will wake from the deep sleep mode to perform its function, which is capturing images. After an image is captured, the image will be sent to the assigned email account through SMTP server. After sending the email successfully, the device will go back to deep sleep mode again.

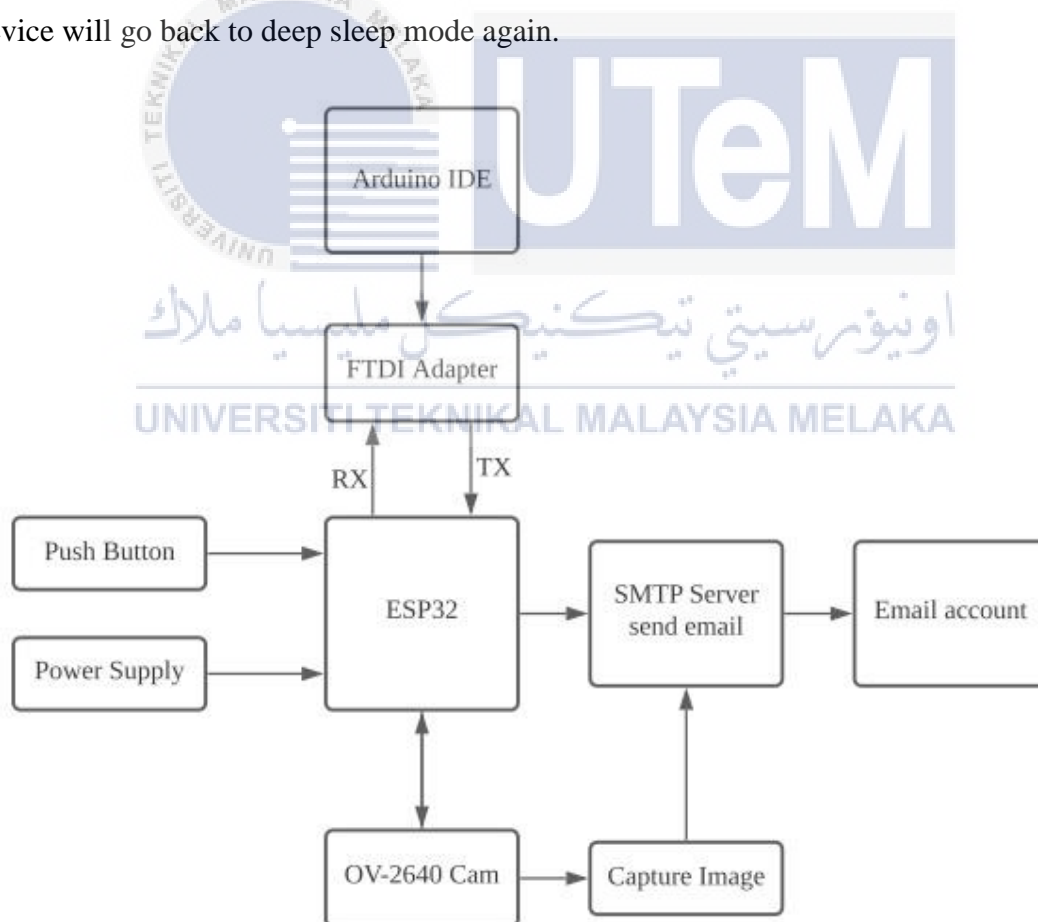


Figure 3.2 Block Diagram for ESP32-CAM.

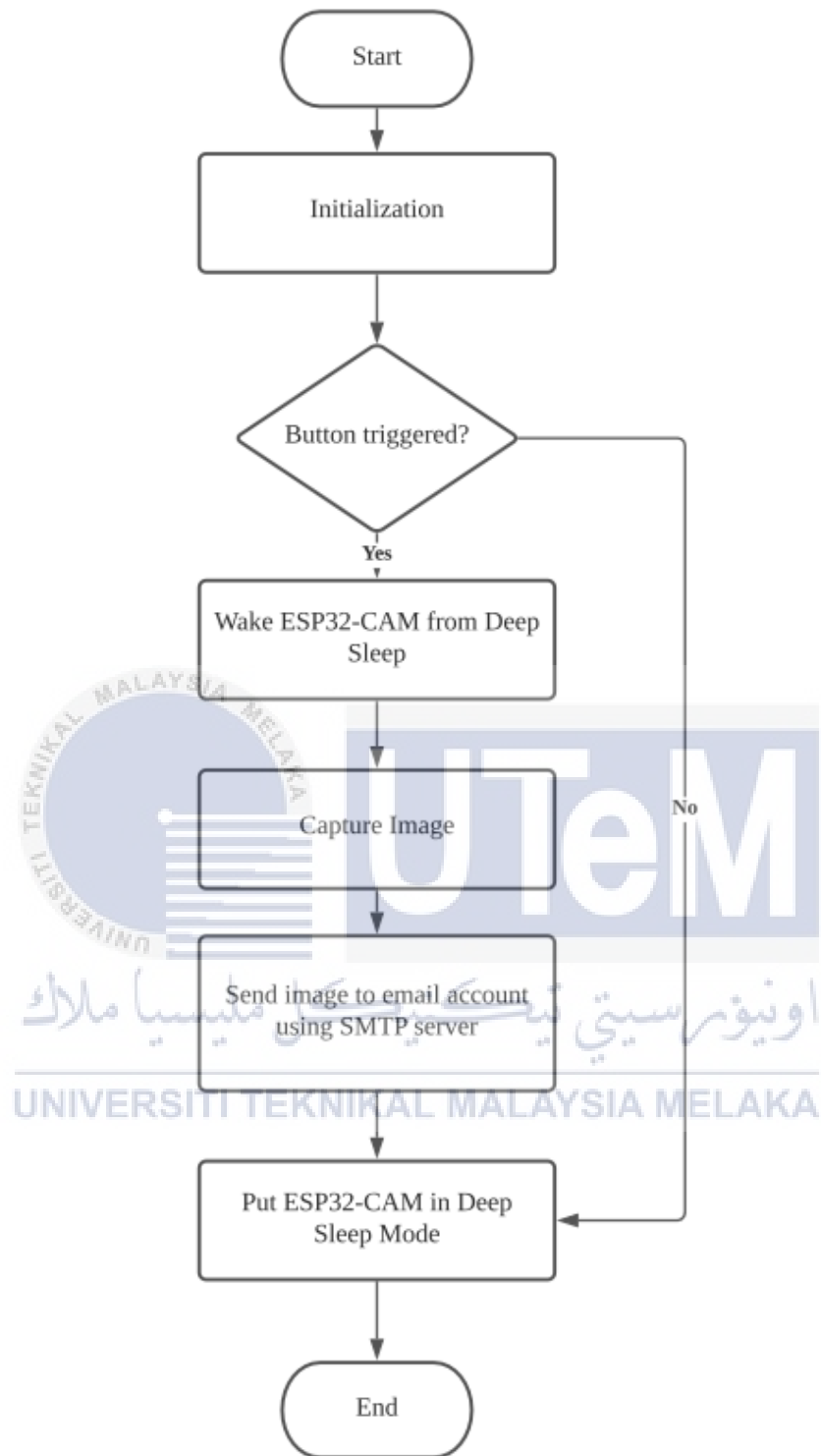


Figure 3.3 ESP32-CAM programming flowchart.

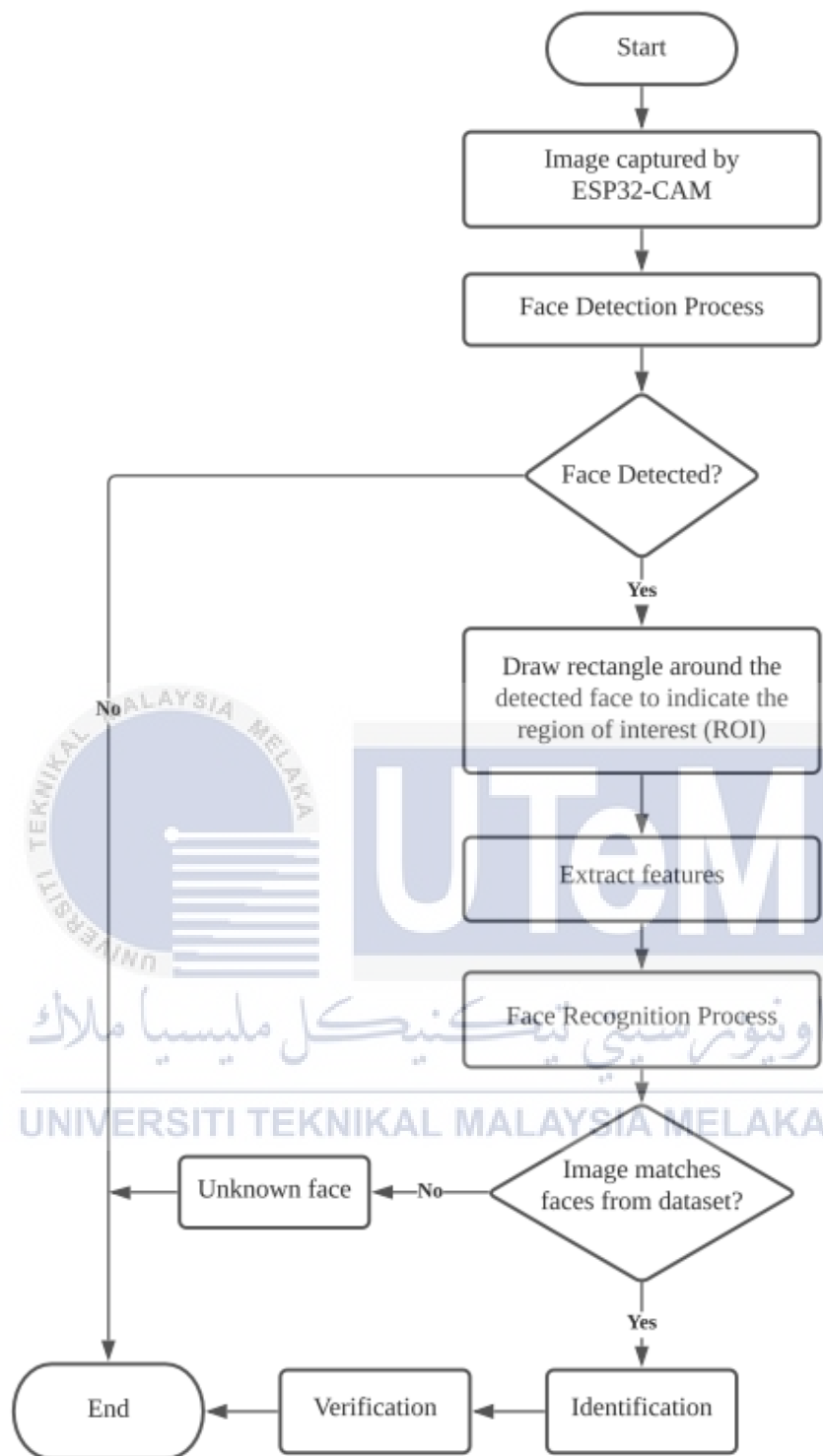


Figure 3.4 Face recognition system programming flowchart.

Figure 3.4 shows the programming flowchart of the proposed face recognition system. Firstly, the image captured by the ESP32-CAM must be downloaded from the email account before it can go through the face detection process. In face detection process, if the

program detected a face, it would draw a rectangle around the face to mark it as a region of interest (ROI) so that the program can omit other unwanted parts to focus on features extraction. After extracting the facial features, the program will perform the face recognition process which uses the Neural Network as foundation. When the captured image matches the faces from dataset, the system will identify and verify the identity of the person being captured in the image. If the image does not match any of the faces from the dataset, it is an unknown face to the system and the program will terminate.

### 3.2.1.1 Equipment

This project consists of two parts, the first part is the software part which is the face recognition system and the second part is the hardware part which made up of few components. The hardware required including jumper wires, power supply, ESP32-CAM and since there is no USB port on the ESP32-CAM, an adapter called the FTDI adapter is needed to program the ESP32-CAM. The software required including Arduino IDE and MATLAB. Figure 3.5 shows all the hardware needed for this project.

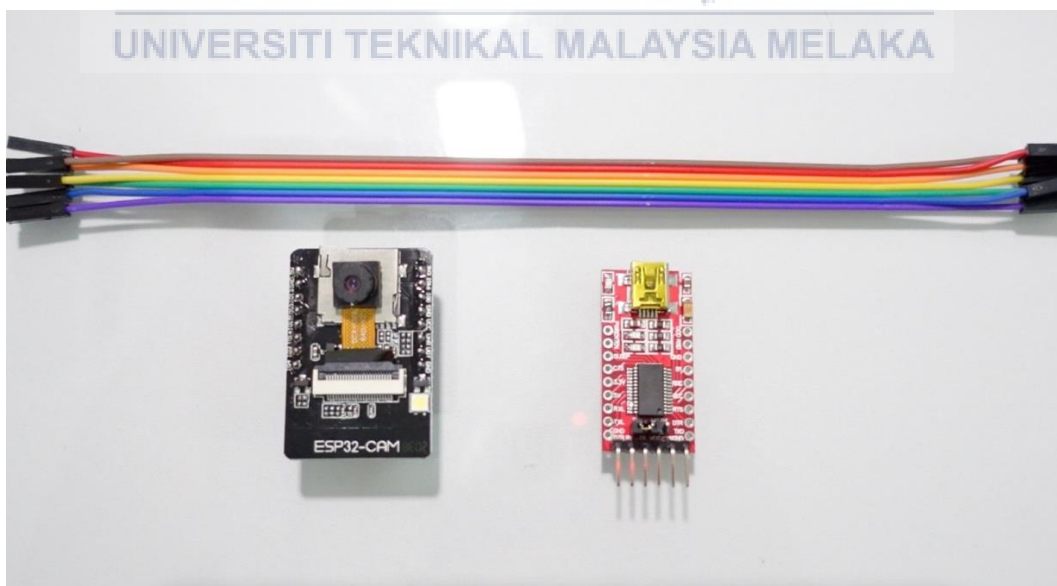


Figure 3.5 Hardware needed for this project.



Figure 3.6 ESP32-CAM.

Figure 3.6 shows the main component in the hardware part, which is the ESP32-CAM. The ESP32-CAM has two power pins, 3.3V and 5V respectively, and three ground (GND) pins. There are a few General-Purpose Input/Output (GPIO) pins, on the left there are GPIO 2, 4, 12, 13, 14 and 15 pins while on the right GPIO 0, 1, 3 and 16. GPIO 1 (U0T) and 3 (U0R) are transmit (TX) and receive (RX) serial pins. As mentioned before, the ESP32-CAM does not have USB port, therefore these two pins are connected to FTDI adapter to upload the Arduino code to the ESP32-CAM. Next, a 2MP OV-2640 camera is included in ESP32-CAM and a slot for microSD card is also built on the ESP32-CAM. On the back of the ESP32-CAM, there is an antenna connector so that an external antenna can be attached to it and increase the Wi-Fi signal strength.



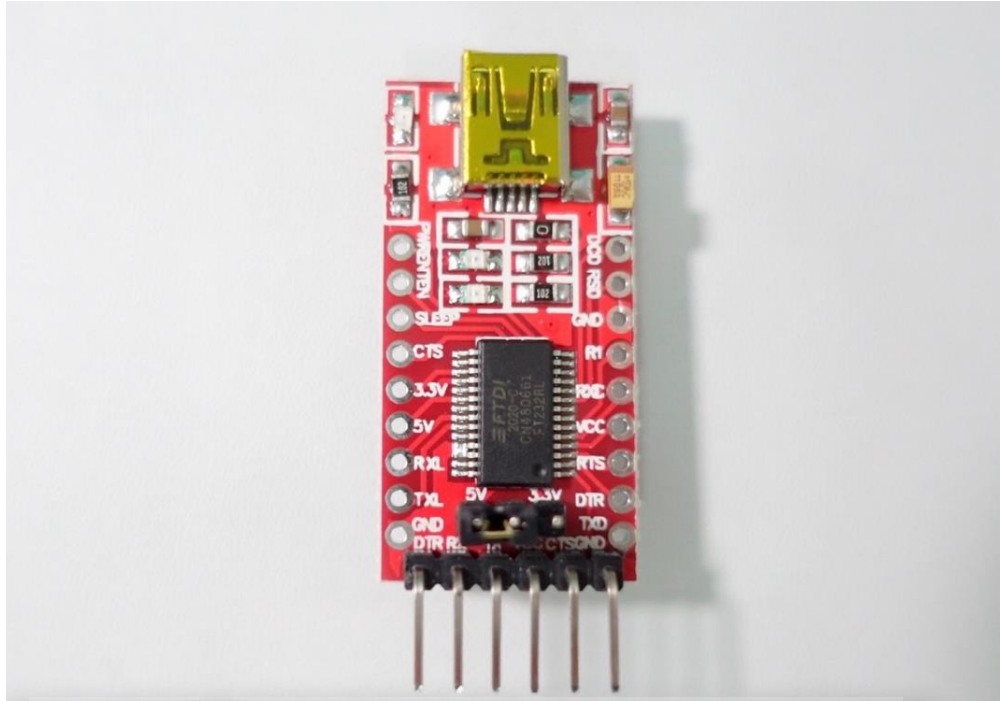


Figure 3.7 FTDI Adapter.

Figure 3.7 shows the FTDI Adapter which is a USB to TTL serial converter. There are six pins on it, namely DTR, RX, TX, VCC, CTS and GND. The RX and TX pins are responsible for receiving and transmitting the serial data from/to the ESP32-CAM respectively. The GND pin will be connected to ESP32-CAM's GND pin, and VCC pin will be connected to ESP32-CAM's 3.3V pin. The function of the DTR pin is to reset the ESP32-CAM automatically, while the CTS pin is used to control the programming mode of the device. Next, the software of this project will be developed by using MATLAB and the hardware will be configured by the code written in Arduino IDE shown in Figure 3.8 and Figure 3.9 respectively.

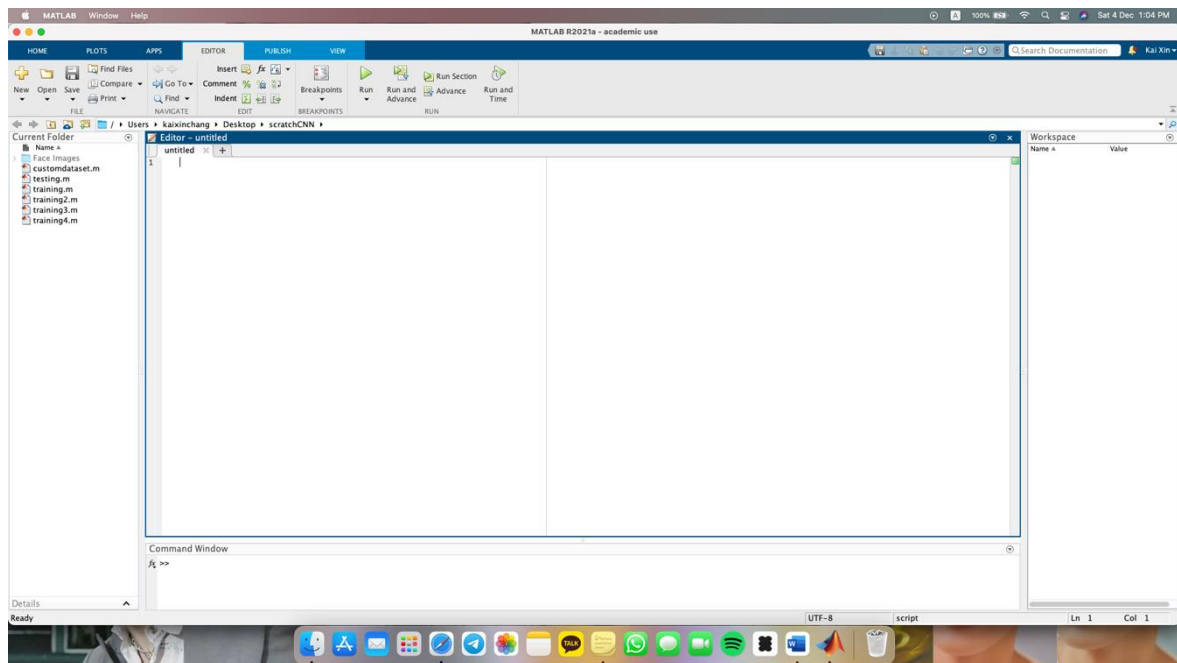


Figure 3.8 MATLAB.



Figure 3.9 Arduino IDE.

### 3.3 Circuit Design

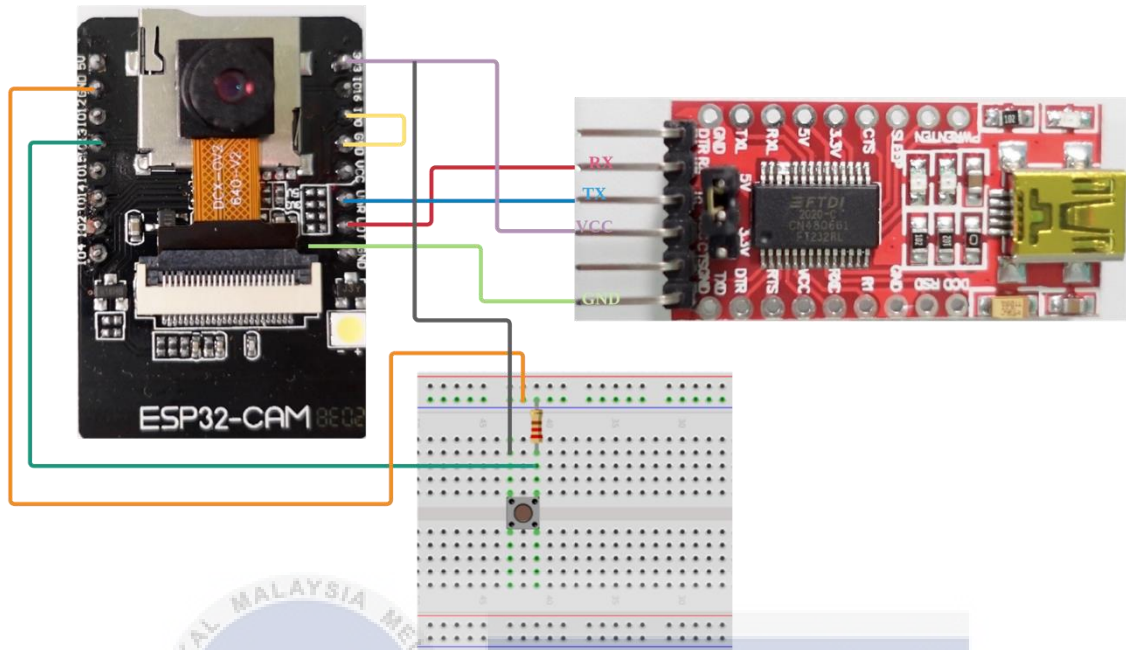


Figure 3.10 Circuit design of this project.

Figure 3.10 shows the circuit connection between ESP32-CAM, FTDI Adapter and an external push button. A jumper wire is connected from GPIO 0 to GND for programming purpose and can be removed once the programming is finished. The push button is used to wake the ESP32-CAM from deep sleep mode and it is connected to the GPIO 13 pin. The TX pin of ESP32-CAM is connected to the RX pin of the FTDI Programmer and the RX pin of ESP32-CAM is connected to the TX pin of the FTDI Programmer so that data can be exchanged between these two devices in serial communication.

### 3.4 Expected Outcomes

Firstly, the ESP32-CAM is expected to capture an image and send it to an assigned email account. After that, the image will be used as an input image to the proposed face recognition system. At this point the image has not gone through any image processing, therefore the second expected outcome is the input image will be processed and become gray

in colour to normalize the image. All the images in training and testing set of the proposed system are also needed to be processed.



Figure 3.11 Expected outcome of image processing.

Figure 3.11 shows the before and after of image processing that turns RGB image into grayscale image. Next expected outcome is the face detection and a rectangle labelling the ROI.

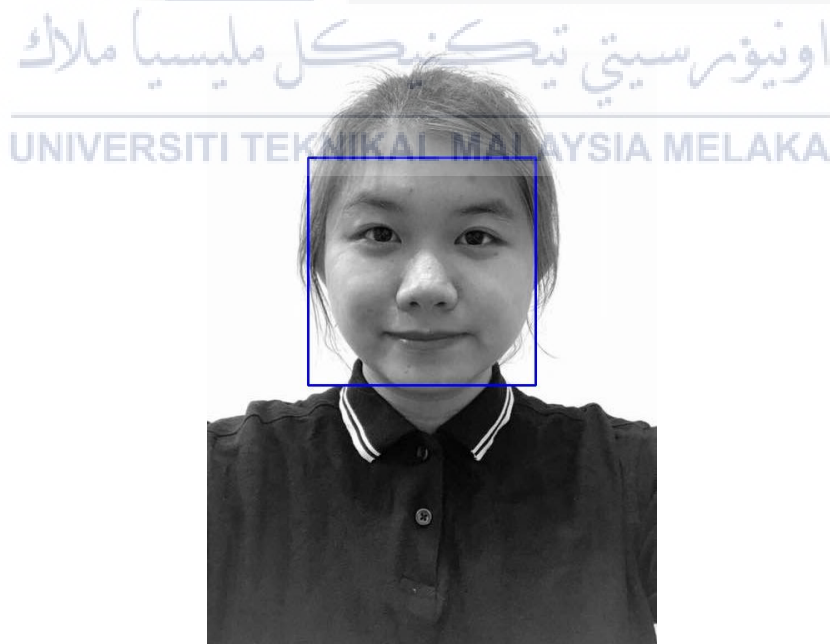


Figure 3.12 Third expected outcome, for face detection.

Figure 3.12 shows the ROI surrounded by a rectangle. After the face is successfully detected, the system is expected to recognise and identify the person in the picture. When the identity is known, the system will show the name of the person, however, if the face is unknown to the system, the system will display “Unknown”, above the rectangle drawn around the ROI as shown in Figure 3.13.

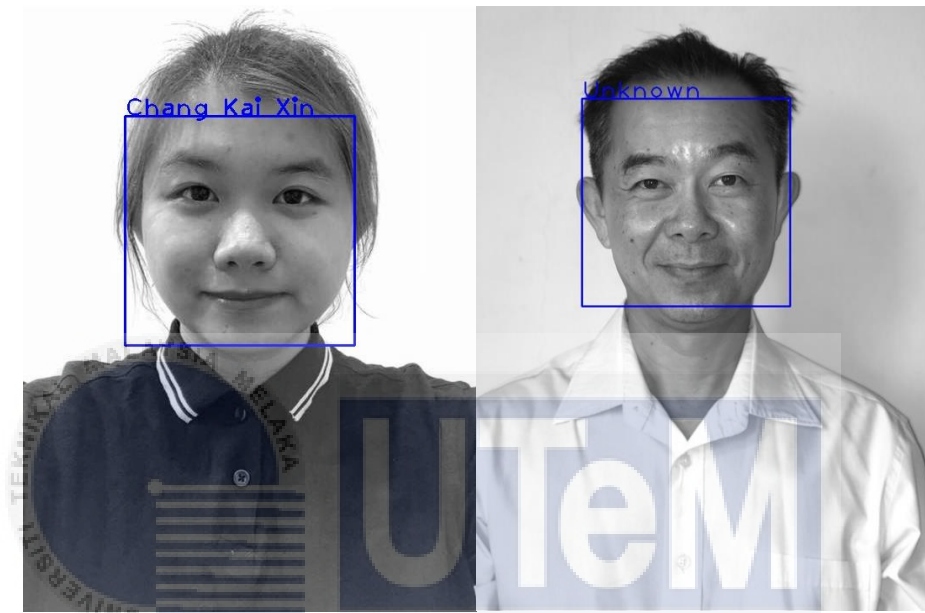


Figure 3.13 Known and unknown faces to the system.

### 3.5 Summary

The proposed methodology presented in this chapter aimed to develop a face recognition system with the combination of software and hardware. Other than using a face database named AT&T and a face database from [31] that are publicly available online, the proposed system will also be using a custom face dataset to have a more reliable, realistic and accurate result as the ESP32-CAM will be used to take images of people in real life and these images will be unknown to the system if the public face databases are used.

## CHAPTER 4

### RESULTS AND DISCUSSIONS

#### 4.1 Introduction

The hardware part of the proposed system is made up of ESP-32 CAM and an external button. The hardware components are configured by using Arduino IDE. The face recognition system of this project is developed with CNN in MATLAB. The system is trained and tested by 2228 images from three different face databases, where two face databases are obtained online, namely the AT&T database and a face database provided by reference [31], and a custom face dataset created by the researcher. This chapter presents the implementation and testing process of the proposed system, with the results and a detailed analysis.

#### 4.2 Hardware Testing

The hardware is configured so that when the external button is pressed, the ESP32-CAM will take a picture, connect to the internet and send the picture to an assigned email address. Then, the image can be downloaded and be used as an input image to the face recognition system. Figure 4.1 shows the hardware setup, Figure 4.2 shows the prototype of the proposed system, Figure 4.3 shows the serial monitor of Arduino after the button is pressed, Figure 4.4 shows the attachment received by the assigned email address and Figure 4.5 shows the image taken by the ESP32-CAM.



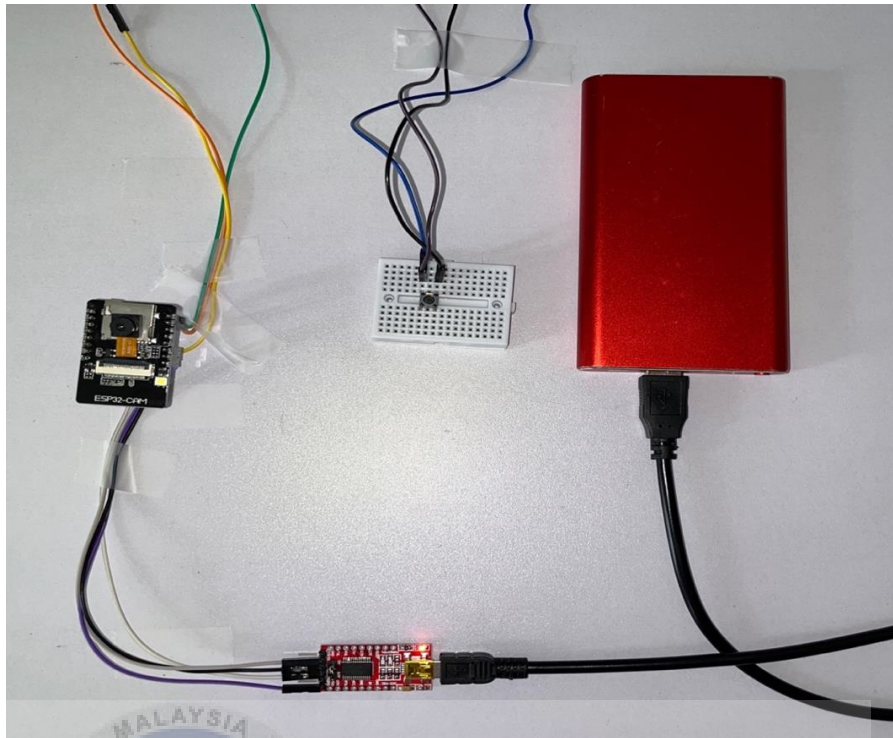



Figure 4.1 Hardware setup.



Figure 4.2 Prototype of the proposed system.



```
COM6
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:10944
load:0x40080400,len:6388
entry 0x400806b4
E (164) psram: PSRAM ID read error: 0xffffffff
?
Connecting to WiFi.....
SPIFFS mounted successfully
IP Address: http://192.168.0.124
Taking a photo...
Picture file name: /photo.jpg
The picture has been saved in /photo.jpg - Size: 10112 bytes
Sending email...
Connecting to SMTP server...
SMTP server connected, wait for response...
Identification...
Authentication...
Sign in...
Sending Email header...
Sending Email body...
Sending attachments...
/photo.jpg
Finalize...
Finished
Email sent successfully
```

Figure 4.3 The serial monitor of Arduino after the button is pressed.



Figure 4.4 Image taken by the ESP-32 CAM is sent to an assigned email address.





Figure 4.5 Image taken by the ESP-32 CAM.

### 4.3 Image Preprocessing

As the size of all images in AT&T database is  $112 \times 92$  and they are all in grayscale, the proposed system is trained with images with the same properties. The image properties of AT&T database are used as guideline and all the images from other databases used in this project are pre-processed to meet the requirement. For the custom face dataset, the images are converted into grayscale, then the face area of all images is cropped from the full image before resizing to  $112 \times 92$  pixels. The same goes to the images from [31], the only difference is, the images in this dataset do not have to be cropped as only the face area is shown in every image. Figure 4.6 and Figure 4.7 show the pre-processing and face detection process of the image taken by the ESP-32 CAM. Figure 4.8 shows the image properties after the pre-processing.

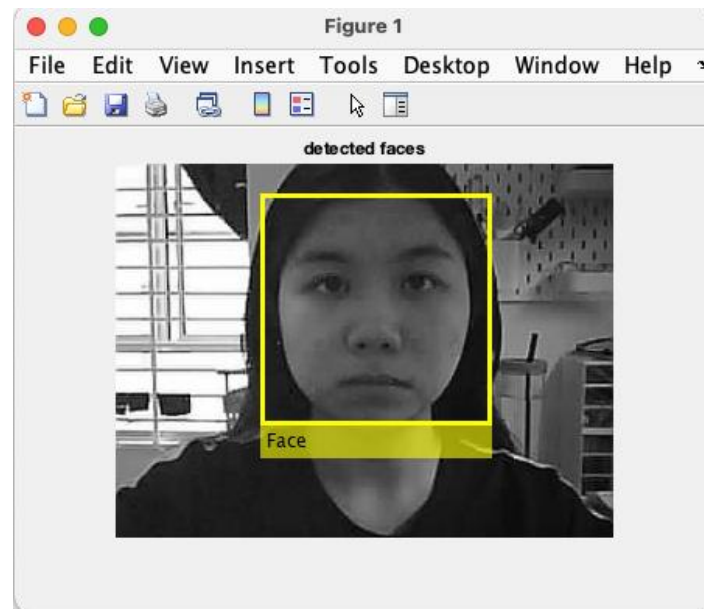


Figure 4.6 The image is converted into grayscale and the face is detected.

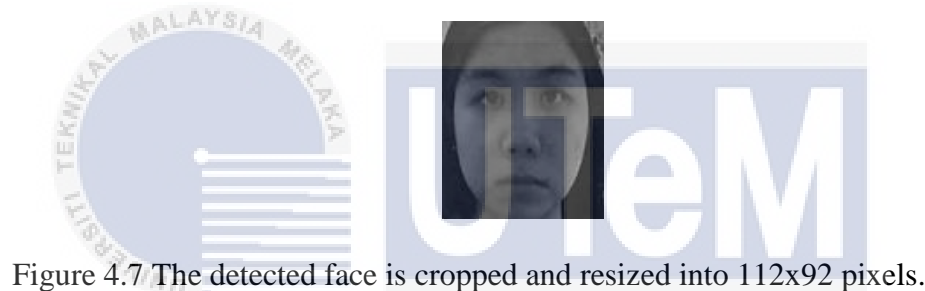


Figure 4.7 The detected face is cropped and resized into 112x92 pixels.



Figure 4.8 Image properties after image pre-processing.

#### 4.4 Custom Face Dataset

A custom face dataset is created in the project for two purposes, firstly, the custom face dataset is used to increase the reliability of the proposed system as the subjects in the custom face dataset are the people that the researcher knows in real life; secondly, the proposed system is to be tested with the picture taken by the hardware, therefore it needs a custom dataset to test the accuracy of the whole proposed system. The custom face dataset is made up of two classes, or in other word, two people. The first class consists of the images of a celebrity which are available and can be easily obtained on the internet, and the second class consists of the images of the researcher of this project. A total of 110 images are collected for the first class and 1410 images are collected for the second class. The images are then split into training set and testing set with a ratio of 80:20. Figure 4.9 and Figure 4.10 show some of the images from the custom face dataset.



Figure 4.9 Custom dataset created by collecting images of celebrity.



Figure 4.10 Custom dataset created by collecting images of the researcher.

#### 4.5 Training the CNN Model

The proposed face recognition system is developed using Convolutional Neural Networks. The layers of CNN used in the proposed system including convolutional layers, pooling layers and fully connected layers will be explained in this section. The proposed system is made up of 12 layers, which are the input layer, two convolutional layers, two batch normalization layers, two ReLU layers, two pooling layers, a fully connected layer, a Softmax layer and an output layer. The researcher carried out an experiment to find out the values of the parameters that will give the CNN the best accuracy in face recognition. The first parameter is the kernel size. The most widely used kernel size is the 3x3 filter. In the experiment that the researcher carried out, the researcher used two different kernel sizes which are the 3x3 and the 5x5 filters. The 3x3 filter is considered as a small kernel size while the 5x5 filter is considered as a big kernel size. Although there are other smaller kernel sizes

such as the 1x1, 2x2 and 4x4 filters, there are reasons why they are not chosen to be used in the project. Firstly, the 1x1 kernel size is usually used to reduce the number of channels or the dimensionality. It only captures one pixel of the feature map; it has no neighbourhood pixels thus no information can be obtained and the feature extracted will be finely grained. Next, the 2x2 and 4x4 kernel sizes are not commonly used because they cannot divide the previous layer symmetrically like the odd-sized filters and will cause distortions across the layers.

The second parameter is the number of filters. The researcher used 32 and 64 filters in convolutional layers. Similar to the kernel size, there is no certain answer or rule to what the number of filters should be. 32 filters are typically used in the first layer, and the number of filters of the next layer can be increased or remained the same, the increasing power of 2 is also commonly used in most face recognition systems. The researchers created two options for testing this parameter, where the first option is using 32 filters in the first convolution layer and 64 filters in the second; and the second option is using 64 filters in both convolution layers.

The third parameter is the initial learning rate. This parameter is one of the training options for training the CNN. The solver used in this project is the stochastic gradient with momentum (sgdm) where the default value of the initial learning rate is 0.01. The researcher tested the system with six different values of initial learning rate, including 0.01, 0.001, 0.0001, 0.03, 0.003 and 0.0003. The researcher aims to find the value that will give the highest accuracy without compromising the training speed. The training will take a long time if the rate is too low, but if it is too high, the training might diverge and reach a suboptimal result. Hence, an optimal value of initial learning rate must be obtained through trial and error. Table 4.1 to Table 4.4 show the effect of these three parameters on training the CNN structure.

Table 4.1 The effect of initial learning rate on CNN of kernel size 5x5 and number of filters = 64 on both CONV layers.

Kernel Size	Number of filters in first CONV layer	Number of filters in second CONV layer	Initial Learning Rate	Total Time Elapsed	Mini batch loss	Validation loss	Validation accuracy
5x5	64	64	0.01	8 m 26 s	0.0014	0.8767	0.8307
			0.001	8 m 35 s	0.0012	0.9511	0.8466
			0.0001	8 m 49 s	0.0030	0.4337	0.8810
			0.03	8 m 50 s	0.0001	1.7347	0.7989
			0.003	7 m 17 s	7.6908e-05	0.9474	0.8730
			0.0003	9 m 01 s	0.0015	0.3997	0.8995

Table 4.2 The effect of initial learning rate on CNN of kernel size 5x5 and different number of filters on the CONV layers.

Kernel Size	Number of filters in first CONV layer	Number of filters in second CONV layer	Initial Learning Rate	Total Time Elapsed	Mini batch loss	Validation loss	Validation accuracy
5x5	32	64	0.01	5 m 36 s	0.0004	0.9443	0.8492
			0.001	5 m 18 s	0.0004	0.6719	0.9048
			0.0001	5 m 07 s	0.0075	0.4315	0.8995
			0.03	5 m 05 s	0.0005	2.3800	0.7302
			0.003	5 m 05 s	0.0009	0.9963	0.8651
			0.0003	5 m 07 s	0.0020	0.6897	0.8386



Table 4.3 The effect of initial learning rate on CNN of kernel size 5x5 and number of filters = 64 on both CONV layers.

Kernel Size	Number of filters in first CONV layer	Number of filters in second CONV layer	Initial Learning Rate	Total Time Elapsed	Mini batch loss	Validation loss	Validation accuracy
3x3	64	64	0.01	6 m 13 s	8.6913e-05	1.7427	0.7751
			0.001	5 m 58 s	0.5553	1.8533	0.7566
			0.0001	6 m 17 s	0.0047	0.4535	0.8651
			0.03	6 m 17 s	1.6689e-07	1.6613	0.8122
			0.003	6 m 07 s	0.0014	1.9096	0.8042
			0.0003	6 m 23 s	0.0063	0.4414	0.8915



Table 4.4 The effect of initial learning rate on CNN of kernel size 5x5 and different number of filters on the CONV layers.

Kernel Size	Number of filters in first CONV layer	Number of filters in second CONV layer	Initial Learning Rate	Total Time Elapsed	Mini batch loss	Validation loss	Validation accuracy
3x3	32	64	0.01	4 m 06 s	0.0001	0.9505	0.8386
			0.001	4 m 04 s	0.0005	0.8976	0.8677
			0.0001	4 m 15 s	0.0114	0.4537	0.8836
			0.03	4 m 07 s	0.2555	1.4597	0.7460
			0.003	4 m 05 s	0.0010	0.6773	0.8836
			0.0003	4 m 01 s	3.6273e-05	0.5490	0.8492

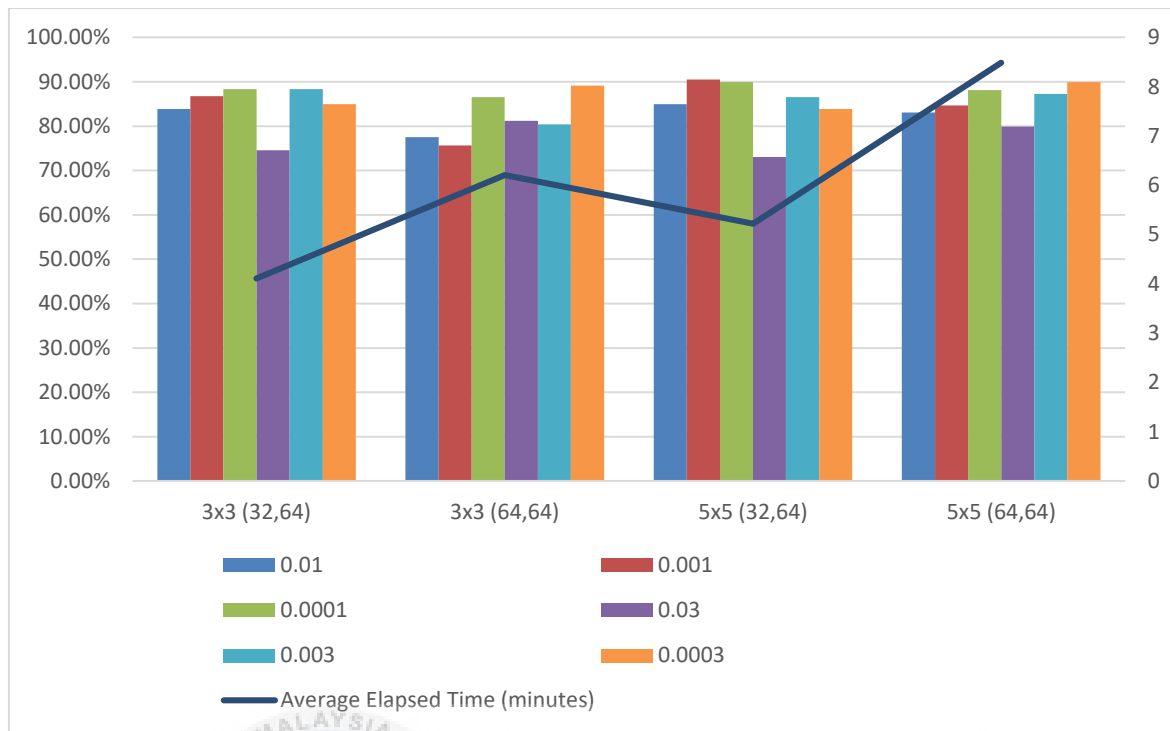


Figure 4.11 Validation accuracy and average elapsed training time of the CNN model with different training parameters.

Figure 4.11 shows the validation accuracy and average elapsed training time of the CNN model as a summary of Table 4.1 to Table 4.4. For easier understanding, the four values of the x-axis from the graph will be referred as set A, B, C and D in the following discussion. From the graph, it can be concluded that in general, the validation accuracy for learning rate of 0.03 is the lowest compared to other learning rates, except for set B because the learning rate that gives the lowest validation accuracy for the set is 0.001. Thus, learning rate of 0.03 is the first to be eliminated.

Next, the total time elapsed for training range between 4 minutes and 9 minutes. The CNN model that uses the shortest average training time is set A while the longest is set D. Set C has the second shortest average training time, and compared to set A, the validation accuracy of set C is higher in overall. Recall that the aim is to find the value that will give the highest accuracy without compromising the training speed, set C meets the requirements, and from all the learning rates in set C, 0.001 gives the highest accuracy, 90.48%. Therefore,

the parameters that will be used for training the CNN model are, kernel size of 5x5, 32 filters on first convolutional layer and 64 filters on second convolutional layer and learning rate of 0.001.

#### **4.6 Software Testing**

The face recognition system needs to be trained with the face dataset before it can recognise faces. The images in the dataset will be split randomly by MATLAB into two sets, namely the training set and the validation set with the ratio of 80:20, which means 80% of the whole dataset will be used as training set while the remaining 20% as validation set. The CNN structure is trained for 10 epochs, where one epoch means the full pass of the training algorithm over the entire training set. Based on the analysis made in the previous section, a CNN model with kernel size of 5x5, number of filters of 32 for first convolution layer, number of filters of 64 for second convolution layer and initial learning rate of 0.001 is chosen to be used for this project. Table 4.5 shows the layers of the proposed CNN structure and Figure 4.12 shows the graph of the CNN training progress where the solid line indicates training and dotted line indicates validation.

Table 4.5 Layers of the proposed CNN model.

Layer	Type	Activations	Learnable
1	Input	112 x 92 x 1	-
2	Convolution	112 x 92 x 32	Weights 5 x 5 x 1 x 32 Bias 1 x 1 x 32
3	Batch Normalization	112 x 92 x 32	Offset 1 x 1 x 32 Scale 1 x 1 x 32
4	ReLU	112 x 92 x 32	-
5	Max Pooling	56 x 46 x 32	-
6	Convolution	56 x 46 x 64	Weights 5 x 5 x 32 x 64 Bias 1 x 1 x 64
7	Batch Normalization	56 x 46 x 64	Offset 1 x 1 x 64 Scale 1 x 1 x 64
8	ReLU	56 x 46 x 64	-
9	Max Pooling	28 x 23 x 64	-
10	Fully Connected	1 x 1 x 57	Weights 57 x 41216 Bias 57 x 1
11	Softmax	1 x 1 x 57	-
12	Classification Output	1 x 1 x 57	-

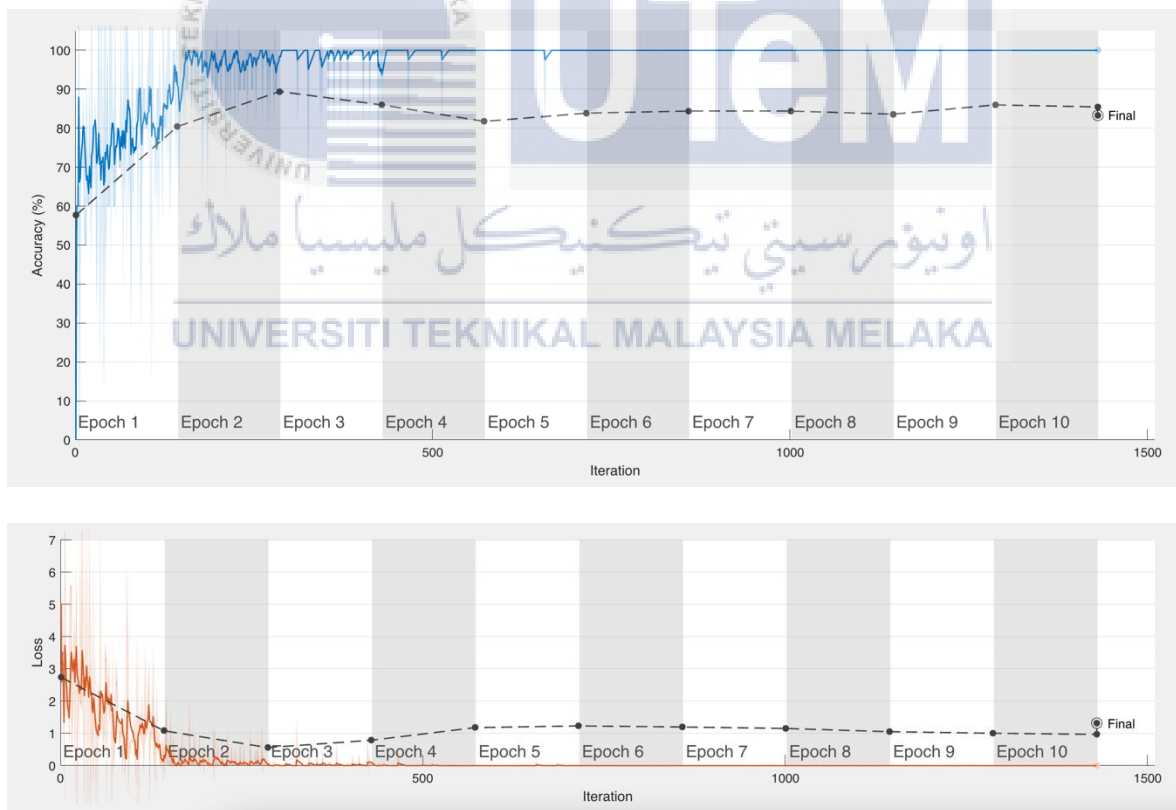


Figure 4.12 Training progress of the chosen CNN structure.

## 4.7 Results and Analysis

To test the accuracy of the proposed face recognition system, a folder of testing images which are not seen by the trained model are created. The last task of the proposed system is to perform face recognition on the testing images. Figure 4.13 to Figure 4.15 shows some of the face recognition results of testing the system with images from custom face dataset, dataset from reference [31] and AT&T database respectively, while Figure 4.16 shows the results for image taken by ESP-32 CAM. Table 4.6 to Table 4.8 show the face recognition results for the three databases used in this project and Table 4.9 shows the comparison between the face recognition accuracy of the databases.

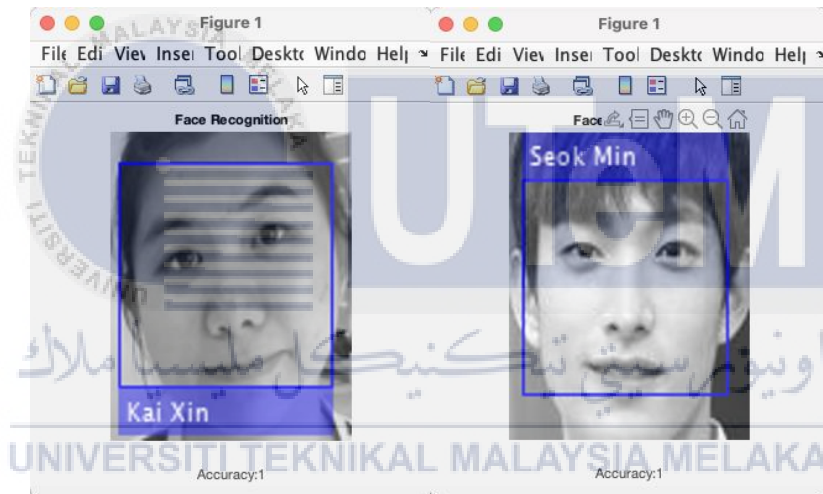


Figure 4.13 Face recognition results for images from custom face dataset.



Figure 4.14 Face recognition results for images from dataset by reference [31].

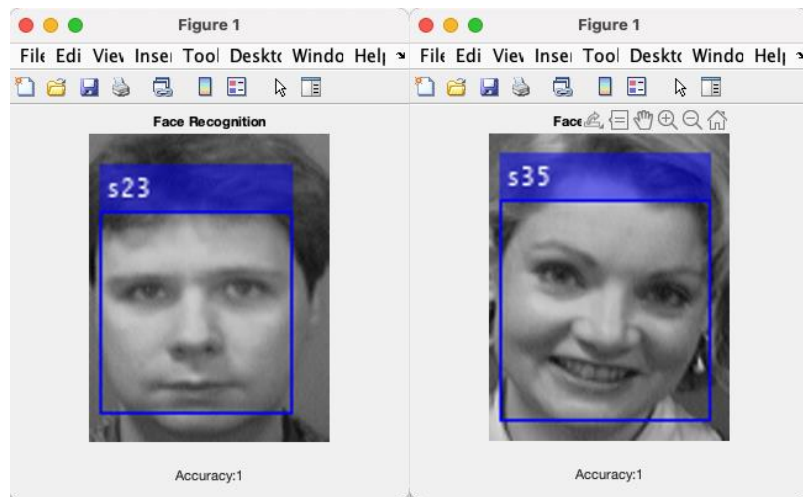


Figure 4.15 Face recognition results for images from AT&T database.



Figure 4.16 Face recognition results for image taken by ESP-32 CAM.

Table 4.6 Face recognition result of Custom Face Dataset.

Class	Testing Images	TP	TN	FN	FP	Accuracy
1	20	20	0	0	0	100.00%
2	250	239	0	4	7	95.60%
Total		259	0	4	7	-
Average Accuracy						95.93%

Table 4.7 Face recognition result of Dataset from reference [31].

Class	Testing Images	TP	TN	FN	FP	Accuracy
1	4	3	0	1	0	75.00%
2	4	4	0	0	0	100.00%
3	4	4	0	0	0	100.00%
4	4	4	0	0	0	100.00%
5	4	4	0	0	0	100.00%
6	4	4	0	0	0	100.00%
7	4	4	0	0	0	100.00%
8	4	4	0	0	0	100.00%
9	4	4	0	0	0	100.00%
10	4	4	0	0	0	100.00%
11	4	4	0	0	0	100.00%
12	4	4	0	0	0	100.00%
13	4	2	0	2	0	50.00%
14	4	2	0	2	0	50.00%
15	4	4	0	0	0	100.00%
16	4	4	0	0	0	100.00%
Total		59	0	5	0	-
Average Accuracy						92.19%

Table 4.8 Face recognition result of AT&amp;T database.

Class	Testing Images	TP	TN	FN	FP	Accuracy
1	2	2	0	0	0	100.00%
2	2	2	0	0	0	100.00%
3	2	1	0	1	0	50.00%
4	2	2	0	0	0	100.00%
5	2	2	0	0	0	100.00%
6	2	2	0	0	0	100.00%
7	2	2	0	0	0	100.00%
8	2	1	0	1	0	50.00%
9	2	2	0	0	0	100.00%
10	2	2	0	0	0	100.00%
11	2	2	0	0	0	100.00%
12	2	2	0	0	0	100.00%
13	2	2	0	0	0	100.00%
14	2	2	0	0	0	100.00%
15	2	2	0	0	0	100.00%
16	2	1	0	1	0	50.00%
17	2	2	0	0	0	100.00%
18	2	2	0	0	0	100.00%
19	2	1	0	1	0	50.00%
20	2	2	0	0	0	100.00%
21	2	2	0	0	0	100.00%
22	2	2	0	0	0	100.00%
23	2	2	0	0	0	100.00%
24	2	2	0	0	0	100.00%
25	2	2	0	0	0	100.00%
26	2	1	0	1	0	50.00%
27	2	2	0	0	0	100.00%
28	2	2	0	0	0	100.00%
29	2	2	0	0	0	100.00%
30	2	2	0	0	0	100.00%
31	2	1	0	1	0	50.00%
32	2	2	0	0	0	100.00%
33	2	2	0	0	0	100.00%
34	2	0	0	2	0	0.00%
35	2	1	0	1	0	50.00%
36	2	2	0	0	0	100.00%
37	2	1	0	1	0	50.00%
38	2	2	0	0	0	100.00%
39	2	2	0	0	0	100.00%
40	2	2	0	0	0	100.00%
Total		70	0	10	0	-
Average Accuracy						87.50%



Table 4.9 Comparison between the face recognition accuracy of different face database.

Face Database	Total Training Images	Total Testing Images	Accuracy
AT&T	320	80	87.50%
Dataset from reference [31]	244	64	92.19%
Custom face dataset	1250	250	95.93%

The results of the proposed system are classified into four groups, the true positive (TP), false positive (FP), true negative (TN) and false negative (FN). The result is called a true positive when the predicted identity matches the actual identity and there are two types of error that will be generated by face recognition system, which are the false positive and the false negative. False positive happens when the system predicted person A but the actual identity is person B or in other words, the input image is the face of person B but the output result returns person A, while false negative happens when system fails to predict the identity of a person which is known to the system and lastly, true negative is when the actual identity is other than person A and the predicted result is also other than person A. The face recognition accuracy is calculated by the following formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

From Table 4.9, the face recognition result for the images from the custom dataset is the highest, with the accuracy of 95.93%, while face recognition result using AT&T database gives 87.50%, which is also the lowest among the three databases. The accuracy of the face recognition for the custom dataset is the highest because the two other face databases are trained with less than 20 images for a subject while the custom face database created by

the researcher has 90 training images for the first class and 1160 for the second class. This is to ensure the result for image taken by ESP32-CAM to have a higher accuracy. Therefore, with the face recognition result, it can be concluded that the goal is achieved. In short, the overall accuracy of the proposed system is 98.48%.

Since the proposed system is supposed to take picture of the kidnap suspect using the ESP32-CAM, the distance between the ESP32-CAM and the kidnapper is another concern. An approach is taken to find out the maximum distance between the camera and the face so that the face can be detected and recognised by the proposed system. The ESP32-CAM is placed at a certain distance from the face and five images are taken for each distance.



Figure 4.17 Cropped faces after image processing.

The cropped faces after image processing are shown in Figure 4.17, where the distance between the ESP32-CAM and the face increases from top left to right, then bottom left to right. It can be seen that from the bottom right picture, the face can hardly to be identified even with human eyes.

Table 4.10 Effect of distance between the face and the ESP32-CAM on face detection rate and face recognition accuracy.

Distance (cm)	Testing images	Images with detected face	Face detection success rate	Face recognition accuracy
30	5	5	100%	100%
40	5	5	100%	100%
50	5	5	100%	100%
60	5	5	100%	100%
70	5	5	100%	100%
80	5	5	100%	100%
90	5	5	100%	100%
100	5	5	100%	100%
110	5	5	100%	100%
200	5	5	100%	100%
300	5	0	0%	0%

The effect of distance between face and the ESP32-CAM on face detection rate and face recognition accuracy is recorded in Table 4.10. The results show that when the distance is beyond 200 cm, the face cannot be detected by the proposed system and when there is no face detected, face recognition process cannot be carried out.

#### 4.8 Conclusion

This chapter presented the results and analysis of the proposed system. There are two main outcomes that are produced by the proposed system. The first outcome is provided by the ESP32-CAM and the second outcome is provided by the face recognition system. The analysis suggested that the best training parameters for the proposed CNN model are, 5x5 kernel size, 32 and 64 filters for the first and second convolutional layer respectively and learning rate of 0.001. The validation accuracy of the system that used these parameters is 90.48%, which is also the highest among all options. At a distance beyond 200 cm, the image captured by the ESP32-CAM is too blurry for the system to detect and recognise. The accuracy for the custom face dataset is the highest as many training images are provided to train the system. The overall accuracy of the proposed system is 98.48%.

## CHAPTER 5

### CONCLUSION AND RECOMMENDATIONS

#### 5.1 Conclusion

This thesis presents the development and implementation of face recognition system using neural networks. The proposed system is divided into two parts, the first part, which is the hardware, is used to capture a picture and send it to an assigned email; and the second part, which is the software, is used for face recognition process. The face recognition system is built with a 12-layer CNN model in MATLAB. The combination of both parts is expected to work as a device to assist in searching for a kidnap suspect with the image taken by the device being sent to the authorities to perform face recognition process to determine the identity of the suspect.

This research also aimed to identify the best training parameters for the proposed convolutional neural network. The training parameters chosen to test the validation accuracy of the proposed system are the kernel size, number of filters for convolutional layers and initial learning rate. Based on the results obtained and the analysis done in Chapter 4, it can be concluded that the best training parameters for the proposed system are kernel size of 5x5, number of filters of 32 for first convolutional layer, number of filters of 64 for second convolutional layer and initial learning rate of 0.001. The proposed system is robust as its overall face recognition accuracy is 98.48%, where it is trained and tested with three different face datasets and also images taken by the hardware.

## **5.2 Limitation of proposed system**

One of the limitations of the proposed methodology is that it is time consuming in creating the custom face dataset. It is recommended to have as many images as possible for a better result, therefore, more images are required for an individual. Next, the quality of the image captured by the ESP32-CAM is low compared to the images taken by any electronic devices such as smartphones or digital camera because the ESP32-CAM uses a 2MP OV-2640 camera module. Besides, the quality of the images captured by ESP32-CAM in low light or at night are too low for the proposed system to recognise, there is also a limitation in distance between the ESP32-CAM and the face as the image captured by the ESP32-CAM will be too blurry for the proposed system to detect and recognise when the face is too far away from the camera.

Furthermore, the hardware device will only take a picture when it is triggered by an external button, when the button is pressed for a few times, we might assume the device will capture multiples pictures and send to the assigned email address. However, that is not the case, the device will only take one picture at the last time the button is pressed. This is because when the button is pressed, the program will run and when it is pressed again without the program fully executed, the program will run again from the beginning. Since the program cannot be executed from the beginning to the end, the picture will not be taken and be sent to the assigned email address. Moreover, the time taken for the device to connect to the internet is inconsistent. Therefore, there must be a time delay between pressing the button to capture a photo.

## **5.3 Project Potential of Commercialization**

The potential customers of the proposed system could be women and children as they are often the target of abduction. The device can be attached to the users' belongings

and the users can press the button to activate the device when they are in danger and their family will be alerted in no time. Besides, since the face recognition technology of the proposed system can also be implemented in the law enforcement or any security systems.

#### **5.4 Future Research**

Based on the conclusion made in the previous section, future studies could consider the following recommendations:

- i) Use a smaller size camera module with higher resolution and night vision.
- ii) Use or create a larger face dataset.
- iii) Modify the CNN layers with different parameters for training options.

Use a wireless pushbutton that can be carried around with ease.



## REFERENCES

- [1] Sonali. B. Maind, Priyanka Wankar (2014). Research paper on Basic of Artificial Neural network. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(1), 96-100.
- [2] M. I. Zarkasyi, M. R. Hidayatullah, E. M. Zamzami (2020). Literature Review: Implementation of Facial Recognition in Society. *Journal of Physics: Conference Series*, 1566, 12069.
- [3] Muhammad Sharif, Farah Naz, Mussarat Yasmin, Muhammad Alyas Shahid, Amjad Rehman (2017). Face Recognition: A Survey. *Journal of Engineering Science and Technology Review*, 10(2), 166-177.
- [4] Madan Lal, Kamlesh Kumar, Rafaqat Hussain, Abudullah MAitlo, Sadaquat Ali Ruk, Hidayutullah Shaikh (2018). Study of Face Recognition Techniques: A Survey. *International Journal of Advanced computer Science and Applications*, 9(6).
- [5] André Teixeira Lopes, Edilson de Aguiar, Alberto F. De Souza, Thiago Oliveira-Santos (2017). Facial Expression Recognition with Convolutional Neural Networks: Coping with few data and the training sample order. *Pattern Recognition*, 61, 610-628.
- [6] Yassin Kortli, Maher Jridi, Ayman Al Falou, Mohamed Atri (2020). Face Recognition Systems: A Survey. *Sensors*, 20(2), 342.
- [7] Salama AbdELminaam D, Almansori AM, Taha M, Badr E (2020). A deep facial recognition system using computational intelligent algorithms. *PLoS ONE*, 15(12), e0242269.
- [8] M. Çarikçi, F. Özen (2012). A Face Recognition System Based on Eigenfaces Method. *Procedia Technology*, 1, 118-123.

- [9] R. Rosnelly, M. Simanjuntak, A. Clinton Sitepu et al. (2020). Face Recognition Using Eigenface Algorithm on Laptop Camera. *2020 8th International Conference on Cyber and IT Service Management (CITSM)*, 1-4.
- [10] T. Mantoro, M. Ayu, Suhendi (2018). Multi-Faces Recognition Process Using Haar Cascades and Eigenface Methods. *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)*, 1-5.
- [11] David Habrman (2016). Face Recognition with Preprocessing and Neural Networks. Master of Science Thesis in Electrical Engineering, Linköping University.
- [12] T. Le (2011). Applying Artificial Neural Networks for Face Recognition. *Advances in Artificial Neural Systems*, 673016.
- [13] M. M. Hussein, A. H. Mutlag, H. Shareef (2019). Developed artificial neural network based human face recognition. *Indonesian Journal of Electrical Engineering and Computer Science*, 16(3), 1279-1285.
- [14] A. S. Abdullah, M. A. Abed, I. Al\_Barazanchi (2019). Improving face recognition by elman neural network using curvelet transform and HSI color space. *Periodicals of Engineering and Natural Sciences*, 7(2), 430-437.
- [15] S. A. Baker, H. H. Mohammed, H.A. Aldabagh (2020). Improving face recognition by artificial neural network using principal component analysis. *TELKOMNIKA Telecommunication, Computing, Electronic and Control*, 18(6), 3357-3364.
- [16] R. Zhu, X. Tu, J. Xiangji Huang (2020). Chapter seven - Deep learning on information retrieval and its applications. *Deep Learning for Data Analytics*, 125-153.
- [17] K.B Pranav, J. Manikandan (2020). Design and Evaluation of a Real-Time Face Recognition System using Convolutional Neural Networks. *Procedia Computer Science*, 171, 1651-1659.



- [18] M. Khan, S. Harous, S. Hassan et al. (2019). Deep Unified Model For Face Recognition Based on Convolution Neural Network and Edge Computing. *IEEE Access*, 7, 72622-72633.
- [19] U. Aiman, V. Vishwakarma (2017). Face recognition using modified deep learning neural network. *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1-5.
- [20] M. Arsenovic, S. Sladojevic, A. Anderla, S. Darko (2017). FaceTime—Deep learning based face recognition attendance system.
- [21] Nurkhamid, P. Setialana, H. Jati, R. Wardani, Y. Indrihapsari, N. Norwawi (2021). Intelligent Attendance System with Face Recognition using the Deep Convolutional Neural Network Method. *Journal of Physics: Conference Series*, 1737, 012031.
- [22] I. Diyasa, A. Fauzi, A. Setiawan (2021). Multi-face Recognition for the Detection of Prisoners in Jail using a Modified Cascade Classifier and CNN. *2020 2<sup>nd</sup> International Conference on Science & Technology*, 1884, 012005.
- [23] P. Kamencay, M. Benco, T. Mizdos, R. Radil (2017). A New Method for Face Recognition using Convolutional Neural Network. *Advances in Electrical and Electronic Engineering*, 15(4).
- [24] M. Alghaili, Z. Li, H. Ali (2020). FaceFilter: Face Identification with Deep Learning and Filter Algorithm. *Hindawi Scientific Programming*, 2020, 7846264.
- [25] N. S. Irjanto, N. Surantha (2020). Home Security System with Face Recognition based on Convolutional Neural Network. *International Journal of Advanced Computer Science and Applications*, 11(11), 408-412.
- [26] K. Meethongjan, D. Mohamad (2007). A Summary of literature review: Face Recognition. *Postgraduate Annual Research Semminar 2007*.

- [27] M. Srivasan, N. Ravichandran (2013). A new technique for Face Recognition using 2D-Gabor Wavelet Transform with 2D-Hidden Markov Model approach. *International Conference on Signal Processing, Image Processing and Pattern Recognition 2013, ICSIPR 2013*, 1, 151-156.
- [28] J. Bobulski (2017). Multimodal face recognition method with two-dimensional hidden Markov model. *Bulletin of the Polish Academy of Sciences Technical Sciences*, 65(1), 121-128.
- [29] J. Moreano, N. Palomino (2020). Global Facial Recognition Using Gabor Wavelet, Support Vector Machines and 3D Face Models. *Journal of Advances in Information Technology*, 11(3), 143-148.
- [30] Z. Rustam, R. Faradina (2018). Face Recognition to Identify Look-Alike Faces using Support Vector Machine. *Journal of Physics: Conference Series*, 1108, 012071.
- [31] F. Hashmi (2021), "Face recognition using deep learning CNN in python," *Thinking Neuron* [Online]. Available: <https://thinkingneuron.com/face-recognition-using-deep-learning-cnn-in-python/>. [Accessed: 11-Nov-2021].



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## APPENDICES

### Appendix A Gantt Chart for Final Year Project I

Tasks/Week	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Title Selection & Registration														
Proposal														
Chapter 1: Introduction														
Chapter 2: Literature Review														
Progress Work 1 Evaluation														
Create face database														
Chapter 3: Methodology														
Assemble hardware														
Code														
Initial Results														
Progress Work 2 Evaluation														
Thesis Submission														
Presentation Slides														
Presentation Video														
Presentation Q&A														

## Appendix B Gantt Chart for Final Year Project II

Tasks/Week	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Code														
Hardware testing														
Progress Work 1 Evaluation														
Software testing														
Whole system testing														
Results Analysis														
Chapter 4: Results and Discussion														
Progress Work 2 Evaluation														
Chapter 5: Conclusion and Recommendations														
Thesis Submission														
Presentation Slides														
Presentation Video														
Presentation Q&A														

## Appendix C Program for Training the Proposed CNN model

```
Training_Images = imageDatastore('Face Images','IncludeSubfolders',...
                                true,'LabelSource','foldernames');
[Training_Set, Validation_Set] = splitEachLabel(Training_Images, 0.8);
Number_of_Classes = numel(categories(Training_Set.Labels));

Input_Layer_Size = [112 92 1];
layers = [...
imageInputLayer([112 92 1])
convolution2dLayer(5,32,'Padding','same')
batchNormalizationLayer
reluLayer
maxPooling2dLayer(2,'Stride',2)
convolution2dLayer(5,64,'Padding','same')
batchNormalizationLayer
reluLayer
maxPooling2dLayer(2,'Stride',2)
fullyConnectedLayer(Number_of_Classes)
softmaxLayer
classificationLayer];

Pixel_Range = [-30 30];
Scale_Range = [0.9 1.1];

Image_Augmenter = imageDataAugmenter('RandXReflection',true,...
    'RandXTranslation',Pixel_Range,'RandYTranslation',Pixel_Range,...
    'RandXScale',Scale_Range,'RandYScale',Scale_Range);

Augmented_Training_Image = augmentedImageDatastore(Input_Layer_Size(1:2),
Training_Set,...
    'DataAugmentation',Image_Augmenter);
Augmented_Validation_Image = augmentedImageDatastore(Input_Layer_Size(1:2),
Validation_Set);
Size_of_Minibatch = 10;
Validation_Frequency = floor(numel(Training_Set.Files)/Size_of_Minibatch);

options = trainingOptions('sgdm','InitialLearnRate', 1e-3,...
    'MaxEpochs', 10,'MiniBatchSize',Size_of_Minibatch,...
    'Shuffle','every-epoch','ValidationData', Validation_Set,...
    'ValidationFrequency',Validation_Frequency,...
    'Verbose',true,'Plots','training-progress');

net = trainNetwork(Training_Set, layers, options);
YPred = classify(net,Validation_Set);
YValidation = Validation_Set.Labels;
accuracy = sum(YPred == YValidation)/numel(YValidation)
```

## Appendix D Program for Testing the Proposed Face Recognition System

```
[file, path] = uigetfile('*.jpg','Select an image');
loc = strcat(path, file);
I = imread(loc);

if size(I, 3) == 3 %if image is rgb change it to grayscale first
    gray = rgb2gray(I);
    I = imresize(gray, [112, 92]);
end

[Label, Prob] = classify(net,I);
face_Detector = vision.CascadeObjectDetector();
faceDetector.MergeThreshold = 5;
roi = step(face_Detector, I);

if ~isempty(roi) %face detected

    label_str = char(Label);

    face_label = insertObjectAnnotation(I,'rectangle', roi,
label_str,'Color','blue','TextColor','white',...
    'TextBoxOpacity',0.5,'FontSize',9);

    figure;
    resized = imresize(face_label, [250 195]);
    imshow(resized);
    title('Face Recognition');
    xlabel(['Accuracy:', num2str(max(Prob),2)]);

else %no face detected
    figure;
    resized = imresize(I, [250 195]);
    imshow(resized);
    title('Face Recognition');
    xlabel('No face is detected');
end
```

## Appendix E Program for ESP32-CAM

```
#include "esp_camera.h"
#include "SPI.h"
#include "driver/rtc_io.h"
#include "ESP32_MailClient.h"
#include <FS.h>
#include <SPIFFS.h>
#include <WiFi.h>

const char* ssid = "changchookhean@unifi_plus";
const char* password = "kenken@88888888";

#define emailSenderAccount "fyp10426@gmail.com"
#define emailSenderPassword "utemfyp2021"
#define smtpServer "smtp.gmail.com"
#define smtpServerPort 465
#define emailSubject "ESP32-CAM Photo Captured"
#define emailRecipient "chloe.c026@gmail.com"

#define CAMERA_MODEL_AI_THINKER

#if defined(CAMERA_MODEL_AI_THINKER)
#define PWDN_GPIO_NUM 32
#define RESET_GPIO_NUM -1
#define XCLK_GPIO_NUM 0
#define SIOD_GPIO_NUM 26
#define SIOC_GPIO_NUM 27

#define Y9_GPIO_NUM 35
#define Y8_GPIO_NUM 34
#define Y7_GPIO_NUM 39
#define Y6_GPIO_NUM 36
#define Y5_GPIO_NUM 21
#define Y4_GPIO_NUM 19
#define Y3_GPIO_NUM 18
#define Y2_GPIO_NUM 5
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 23
#define PCLK_GPIO_NUM 22
#else
#error "Camera model not selected"
#endif

const int buttonPin = 16; //initialize external button
int buttonState = HIGH;
int lastButtonState = LOW;
unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;
```



```

SMTPData smtpData;

#define FILE_PHOTO "/photo.jpg"

void setup() {
  pinMode(buttonPin, INPUT);
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout
  detector

  Serial.begin(115200);
  Serial.println();

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  if (!SPIFFS.begin(true)) {
    Serial.println("An Error has occurred while mounting SPIFFS");
    ESP.restart();
  }
  else {
    delay(500);
    Serial.println("SPIFFS mounted successfully");
  }

  // Print ESP32 Local IP Address
  Serial.print("IP Address: http://");
  Serial.println(WiFi.localIP());

  camera_config_t config;

  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;

```

```

config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
} else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
}

// Initialize camera
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
}
sensor_t * s = esp_camera_sensor_get();
// initial
sensors are flipped vertically and colors are a bit saturated
if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1); // flip it back
    s->set_brightness(s, 1); // up the brightness just a bit
    s->set_saturation(s, -2); // lower the saturation
}
// drop down frame size for higher initial frame rate
s->set_framesize(s, FRAMESIZE_QVGA);

#ifdef CAMERA_MODEL_M5STACK_WIDE ||
defined(CAMERA_MODEL_M5STACK_ESP32CAM)
    s->set_vflip(s, 1);
    s->set_hmirror(s, 1);
#endif

loop();
capturePhotoSaveSpiffs();
sendPhoto();
}

void loop() { //whole loop is added
int reading = digitalRead(buttonPin);

if (reading != lastButtonState){

```

```

    lastDebounceTime = millis();
}

if((millis() - lastDebounceTime)>debounceDelay){
    if(reading!=buttonState){
        buttonState = reading;
        if(buttonState == LOW){
            Serial.println("Button Pressed");
        }
    }
}
}

// Check if photo capture was successful
bool checkPhoto( fs::FS &fs ) {
    File f_pic = fs.open( FILE_PHOTO );
    unsigned int pic_sz = f_pic.size();
    return ( pic_sz > 100 );
}

// Capture Photo and Save it to SPIFFS
void capturePhotoSaveSpiffs( void ) {
    camera_fb_t * fb = NULL; // pointer
    bool ok = 0; // Boolean indicating if the picture has been taken correctly

    do {
        // Take a photo with the camera
        Serial.println("Taking a photo...");

        fb = esp_camera_fb_get();
        if (!fb) {
            Serial.println("Camera capture failed");
            return;
        }

        // Photo file name
        Serial.printf("Picture file name: %s\n", FILE_PHOTO);
        File file = SPIFFS.open(FILE_PHOTO, FILE_WRITE);

        // Insert the data in the photo file
        if (!file) {
            Serial.println("Failed to open file in writing mode");
        }
        else {
            file.write(fb->buf, fb->len); // payload (image), payload length
            Serial.print("The picture has been saved in ");
            Serial.print(FILE_PHOTO);
            Serial.print(" - Size: ");
            Serial.print(file.size());
            Serial.println(" bytes");
        }
    }
}

```

```

}
// Close the file
file.close();
esp_camera_fb_return(fb);

// check if file has been correctly saved in SPIFFS
ok = checkPhoto(SPIFFS);
} while ( !ok );
}

void sendPhoto( void ) {
// Preparing email
Serial.println("Sending email...");
// Set the SMTP Server Email host, port, account and password
smtpData.setLogin(smtpServer, smtpServerPort, emailSenderAccount,
emailSenderPassword);

// Set the sender name and Email
smtpData.setSender("ESP32-CAM", emailSenderAccount);

// Set Email priority or importance High, Normal, Low or 1 to 5 (1 is highest)
smtpData.setPriority("High");

// Set the subject
smtpData.setSubject(emailSubject);

// Set the email message in HTML format
smtpData.setMessage("<h2>Photo captured with ESP32-CAM and attached in this
email.</h2>", true);
// Set the email message in text format
//smtpData.setMessage("Photo captured with ESP32-CAM and attached in this email.",
false);

// Add recipients, can add more than one recipient
smtpData.addRecipient(emailRecipient);
//smtpData.addRecipient(emailRecipient2);

// Add attach files from SPIFFS
smtpData.addAttachFile(FILE_PHOTO, "image/jpg");
// Set the storage type to attach files in your email (SPIFFS)
smtpData.setFileStorageType(MailClientStorageType::SPIFFS);

smtpData.setSendCallback(sendCallback);

// Start sending Email, can be set callback function to track the status
if (!MailClient.sendMail(smtpData))
Serial.println("Error sending Email, " + MailClient.smtpErrorReason());

// Clear all data from Email object to free memory
smtpData.empty();

```

```
}  
  
// Callback function to get the Email sending status  
void sendCallback(SendStatus msg) {  
    //Print the current status  
    Serial.println(msg.info());  
}
```

