



***Dissertation on***

**“PERIOULAR RECOGNITION IN THE VISIBLE  
SPECTRUM”**

*Submitted in partial fulfilment of the requirements for the award of degree of*

**Bachelor of Technology  
in  
Computer Science & Engineering**

**UE17CS490B – Capstone Project Phase - 2**

*Submitted by:*

<b>Rohit Motamarry</b>	<b>PES1201700167</b>
<b>Yamini Cherukuri</b>	<b>PES1201701029</b>
<b>Avi Bansal</b>	<b>PES1201701116</b>
<b>Ganesha K S</b>	<b>PES1201701731</b>

*Under the guidance of*

**Prof. Preet Kanwal**  
Associate Professor  
PES University

**January - May 2021**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
FACULTY OF ENGINEERING  
PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)  
100ft Ring Road, Bengaluru – 560 085, Karnataka, India



## PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)  
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

### FACULTY OF ENGINEERING

## CERTIFICATE

*This is to certify that the dissertation entitled*

**‘Periocular recognition in the Visible Spectrum’**

*is a bonafide work carried out by*

**Rohit Motamarry  
Yamini Cherukuri  
Avi Bansal  
Ganesha K S**

**PES1201700167  
PES1201701029  
PES1201701116  
PES1201701731**

in partial fulfilment for the completion of eighth semester Capstone Project Phase - 2 (UE17CS490B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8<sup>th</sup> semester academic requirements in respect of project work.

Signature  
**Prof. Preet Kanwal**  
Associate Professor

Signature  
Dr. Shylaja S S  
Chairperson

Signature  
Dr. B K Keshavan  
Dean of Faculty

### External Viva

**Name of the Examiners**

**Signature with Date**

1. \_\_\_\_\_

\_\_\_\_\_

2. \_\_\_\_\_

\_\_\_\_\_

## DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled “**Periocular Recognition in the Visible Spectrum**” has been carried out by us under the guidance of Prof. Preet Kanwal, Associate Professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology in Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

**PES1201700167**

**Rohit Motamarri**



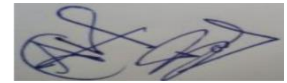
**PES1201701029**

**Yamini Cherukuri**



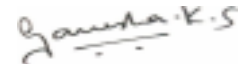
**PES1201701116**

**Avi Bansal**



**PES1201701731**

**Ganesha K S**



## **ACKNOWLEDGEMENT**

I would like to express my gratitude to Prof. Preet Kanwal, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE17CS490B - Capstone Project Phase – 2.

I am grateful to the project coordinators, Prof. Silviya Nancy J and Prof. Sunitha R, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

## **ABSTRACT**

With the growing applications of facial recognition systems in various fields, rising challenges have become prominent. The traditional facial recognition systems and technologies have become error prone in cases of occlusions, facial expression, and aging which is aimed to be addressed. The aim of this project is to develop a recognition model that makes use of the periocular region.

Periocular region refers to the region around the eye inclusive of various features including sclera, eyelids, lashes, brows, and skin. The model intends to make use of a unique contrast enhanced region of interest consisting of the region around both the eyes for feature extraction, using a custom dataset. The dataset comprises of various subjects shot in the visible spectrum. A convolutional neural network is used to perform the feature extraction that outputs a feature map for each eye. This intermediate result, when retrieved from gallery images, is fed to classification model for training. In case of performing the recognition process, this intermediate result is retrieved from input image selected by the user and fed to the classifier for recognition results

# TABLE OF CONTENTS

Chapter No.	Title	Page No.
1.	<b>INTRODUCTION</b>	1
2.	<b>PROBLEM STATEMENT</b>	5
3.	<b>LITERATURE REVIEW</b>	6
	3.1 Convolutional Neural Network- Based Periocular Recognition in Surveillance Environments:	6
	3.1.1 Background, emergence and overview of periocular recognition	6
	3.1.2 Pre-processing and Feature Extraction	7
	3.1.3 Feature Map Comparison and Recognition Score Generation	7
	3.1.4 Results and Conclusions	8
	3.2 Periocular Biometrics in the Visible Spectrum: A Feasibility Study	9
	3.2.1 Introduction	9
	3.2.2 Database Used	10
	3.2.3 Proposed Method	10
	3.2.4 Pros and Cons of Proposed Method	12
	3.2.5 Conclusion	12
	3.3 Periocular Recognition using CNN Features Off-The Shelf	13
	3.3.1 Introduction	13
	3.3.2 Periocular Recognition using CNN Architectures	14
	3.3.3 Periocular Recognition using Traditional Baseline Systems	15
	3.3.4 Dataset Used	15

3.3.5 Results	15
3.3.6 Conclusion	16
3.4 Periocular Biometric: Databases, Algorithms and Directions	17
3.4.1 Introduction	17
3.4.2 Datasets Used	17
3.4.3 Pros and Cons	18
3.4.4 Conclusion	18
3.5 A model periocular biometrics solution for authentication during Covid-19 pandemic situation	19
3.5.1 Introduction	19
3.5.2 Databases Used	19
3.5.3 Pre-Processing	21
3.5.4 Feature Extraction and Recognition	22
3.5.5 Results and Conclusions	23
<b>4. PROJECT REQUIREMENTS SPECIFICATION</b>	<b>24</b>
4.1 Project Scope	24
4.2 Constraints	24
4.2.1 Design Constraints	24
4.2.2 System Constraints	25
4.3 Product Perspective	25
4.3.1 Product Features	26
4.3.2 User Classes and Characteristics	26
4.3.3 General Constraints, Assumptions, and Dependencies	27
4.3.3.1 Constraints	27
4.3.3.2 Assumptions	27
4.3.3.3 Dependencies	27
4.4 Risks	28
<b>5. HIGH LEVEL DESIGN</b>	<b>29</b>

5.1 Introduction	29
5.2 Current System	29
5.3 Design Considerations	29
5.3.1 Design Goals	29
5.3.2 Architecture choices	30
5.3.3 Constraints, Assumptions, Dependencies	30
5.4 High-Level Design	31
5.5 Design Description	33
5.5.1 Master Class Diagram	33
5.5.1.1 Class 1: User	33
5.5.1.2 Class 2: Augmentor	34
5.5.1.3 Class 3: Face Detector	34
5.5.1.4 Class 4: Facial Landmarks Detector	34
5.5.1.5 Class 5: Preprocessor	34
5.5.1.6 Class 6: Feature Extractor	35
5.5.1.7 Class 7: Trainer	35
5.5.1.8 Class 8: Tester	35
5.5.2 Reusability Considerations	36
5.5.2.1 Face Detector	36
5.5.2.2 Facial Landmarks Detector	36
5.5.2.3 Preprocessor	36
5.6 State Diagram	36
5.6.1 State Diagram for Training Phase	37
5.6.2 State Diagram for recognition phase	39
5.7 User Interface Diagram	43
5.8 External Interfaces	49
5.8.1 Software Requirements	50
5.8.2 Hardware Interface	51



5.9 Help	52
<b>6. LOW LEVEL DESIGN</b>	<b>53</b>
6.1 Introduction	53
6.1.1 Overview	53
6.1.2 Purpose	53
6.1.3 Scope	54
6.2 Design Constraints, Assumptions and Dependencies	54
6.3 Design Description	55
6.3.1 Use Case Diagram	55
6.3.2 Class Diagram	57
6.3.2.1 Class: User	57
6.3.2.2 Class: Face Detector	60
6.3.2.3 Class: Facial Landmarks Detector	61
6.3.2.4 Class: Augmentor	62
6.3.2.5 Class: Preprocessor	63
6.3.2.6 Class: Feature Extractor	64
6.3.2.7 Class: Trainer	64
6.3.2.8 Class: Tester	65
6.3.3 Sequence Diagram	67
6.3.4 Deployment Diagram	69
6.3.5 Package Diagram	70
<b>7. SYSTEM DESIGN</b>	<b>72</b>
7.1 Modules	73
7.1.1 Module 1: Pre-processing	73
7.1.2 Module 2: Feature extraction	74
7.1.3 Module 3: Training	74
7.1.4 Module 4: Testing	74
7.1.5 Module 5: User Interface	75

<b>8.</b>	<b>PROPOSED METHODOLOGY</b>	76
	8.1 Augmentation	77
	8.2 Face and facial landmarks detection	77
	8.3 Region of Interest Extraction	78
	8.3.1 ROI extraction algorithm	78
	8.4 Contrast Limited Histogram Equalisation (CLAHE)	79
	8.5 Feature extraction	80
	8.6 Classifier training/testing	80
<b>9.</b>	<b>IMPLEMENTATION AND PSEUDOCODE</b>	81
	9.1 Dataset Creation	81
	9.2 Augmentation	84
	9.3 Face Detection	87
	9.4 Facial Landmarks Detection and region of interest extraction	89
	9.5 Contrast Limited Histogram Equalisation (CLAHE)	92
	9.6 Feature Extraction and Training	93
	9.7 Recognition	94
	9.8 Model Accuracy	96
<b>10.</b>	<b>RESULTS AND DISCUSSION</b>	98
<b>11.</b>	<b>CONCLUSION AND FUTURE WORK</b>	100
	<b>REFERENCES</b>	102
	<b>APPENDIX A DEFINITIONS, ACRONYMS AND ABBREVIATIONS</b>	107

## LIST OF FIGURES

Figure No.	Title	Page No.
1.1	Feature Extraction using traditional facial recognition	1
1.2	Occlusion and shortcomings for facial recognition systems	2
1.3	Feature extraction in periocular recognition model	3
5.4.1	High Level system design	31
5.5.1	Masterclass Diagram	33
5.6.1.1	State Diagram for the training phase	37
5.6.2.1	State Diagram for user interactions and recognition phase	39
5.7.1	Home page of the web app	43
5.7.2	About, Technologies and Requirements, team page displayed on User clicks	44
5.7.3	The registration page on the left and the page after successful registration on the right	44
5.7.4	User Login Page	45
5.7.5	Options displayed after logging in	46
5.7.6	Database View/Edit Page	46
5.7.7	Output for “Train Model” and “Get Model Accuracy”	47
5.7.8	Upload Image page	48
5.7.9	After uploading the file and After the user selects recognize	48
5.7.10	Output page	49
6.3.1.1	Use case diagram for the system	55

6.3.2.1	Masterclass Diagram	57
6.3.3.1	Sequence Diagram for the system	67
6.3.4.1	Deployment Diagram	69
6.3.5.1	Package Diagram	70
7.1	Modular System Design	72
8.1	Overall Procedure of the Model	76
9.1.1	Unmasked Images in the Database	82
9.1.2	Masked Images in the Database	83
9.2.1	Input for Augmentation	86
9.2.2	Output for Augmentation	86
9.3.1	Input for face detection	88
9.3.2	Output for face detection	88
9.4.1	Output for facial landmarks detection	91
9.4.2	Intermediate output for ROI	91
9.4.3	ROI Extraction Output	92
9.5.1	CLAHE Output	93
9.7.1	User Input Image	96

## LIST OF TABLES

<b>Table No.</b>	<b>Title</b>	<b>Page No.</b>
6.3.1.1	Use case Item Description	56
6.3.2.1.1	Class User data members	58
6.3.2.2.1	Class Face Detector data members	60
6.3.2.3.1	Class Facial Landmarks detector data members	61
6.3.2.4.1	Class Augmentor data members	62
6.3.2.5.1	Class Pre-processor data members	63
6.3.2.6.1	Class Feature Extractor data members	64
6.3.2.8.1	Class Feature Extractor data members	65

# CHAPTER-1

# INTRODUCTION

Automated human identification under surveillance environments has become an area of focus in the past few years. Thus, is used widely for various purposes that include security, privacy, crime, accidents, access control and authorization mainly.

In existent facial recognition systems, the subject being a human face, is identified and detected in the first step from an image, frame of a video or live stream. Further, desired features from the face are extracted to form a 128-dimensional feature vector. The images from the databases are processed in the same fashion to obtain several feature vectors corresponding to the images in the database. Each feature vector is then compared with the feature vector of the input image, retrieved in a similar fashion, with the intention to obtain an optimal recognition score. The resultant scores determine the output of the system.



*Fig 1.1: Feature extraction using traditional facial recognition.*

However, this system is highly flawed and error prone with aging, change of expression, occlusions, etc. To overcome these flaws, a unique concept called Periocular Recognition was introduced. This concept works in a similar fashion as the existing facial recognition model with the same aim of recognizing the subject in the input image fed to the system.

The periocular recognition model uses only a subset of features from the face unlike the traditional facial recognition model, that used the entire face. This model is said to accurately address the issues of facial recognition models since the region of interest does not even consist of the occlusions that could be present.

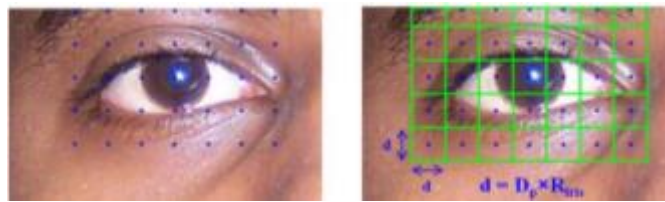


*Fig 1.2: Occlusion and shortcomings for facial recognition systems*

The region of interest is the key point to address and a very crucial step for effective feature extraction. The region of interest refers to the subset of image-wide characteristics that are defined for a specific reason. The Periocular area is the region of interest in the model.

The periocular region refers to the area around the eye that may include eyebrows, eyelashes, eyelids, shape of the eye, height, texture of the skin, eye width, etc. It is considered that only these features are accessible due to occlusions in the input image. The model aims to work with and provide optimum results with the minimal feature information available.

For the periocular recognition model, the input is an image. The input undergoes pre-processing techniques that include face and facial landmarks detection, cropping, for region of interest extraction and, Contrast Limited Adaptive Histogram Equalization (CLAHE) techniques for enhancing the contrast of the periocular region in order to highlight features in the image. OpenCV's DNN, Caffe Model, is used for face detection following which dib library aids in detecting the facial landmarks, consisting of the 4 canthus points, 2 for each eye.



*Fig 1.3: Feature extraction in the periocular recognition model*

Next the bounding rectangles enclosing the region of interest are calculated with respect to the canthus points, then, cropping and CLAHE techniques are performed with OpenCV. The resulting contrast enhanced cropped image for each eye is fed into a Convolutional Neural Network, VGG16, which performs feature extraction and outputs feature maps of size (1,7,7,512), one for each eye. These feature maps are processed to be compatible with weighted KNN classifiers by flattening it to (25088,). The resultant feature maps for each eye along with their respective true labels of the images are fed to two K Nearest Neighbours classification models, one for each eye, for training. An input image fed by the user is also processed in the same fashion to extract the two feature maps from the neural network. These feature maps are then fed into the respective K nearest neighbour trained models to perform recognition. Having K value set to 3, these models take the 3 closest neighbours or labels and assigns weights to them and outputs the label with the topmost score as the predicted person for the respected eye feature map, thus getting two outputs for the predicted person. On achieving the same person from both models, the predicted person is directly returned to the user as output. In unfortunate circumstances where the models predict two different people for each vector



map, the average of the sum of recognition probabilities from each model is calculated and the label with the highest corresponding average is returned to the user as the predicted person.

The primary goal of this model is to assist and recognize a decent number of individuals in surveillance environments. In diverse scenarios, such a model is found to be helpful. For example, today's COVID pandemic scenario places great emphasis on wearing masks and face shields, which would require a periocular model for recognition. The model may be of extreme benefit in the event of crime, such that even in the presence of occlusions such as masks, monkey caps and other headgear, perpetrators or offenders may be recognized.

## **CHAPTER-2**

### **PROBLEM STATEMENT**

The aim of the model is to design a periocular recognition system, given an image, which may only contain facial information in and around the periocular region.

There are existing models that assist in recognition but are based on feature extraction of entire facial information. These models extract features and construct a feature map from the entire face of the subject, following which a match is performed to obtain the recognition results. These models are currently deployed for preventing crime, accidents, access control, authorization, and so on. However, this existing model has various shortcomings such as failures in the case of occlusions, changes in facial expressions and aging.

The problem being addressed here is the limitation of facial information during occlusions such as masks, helmets, among a few. Thus, the proposed model focuses on recognition using the limited features available from the face.

# CHAPTER-3

## LITERATURE REVIEW

The related works lay focus on a periocular recognition model using various techniques. They all throw light on different methods of implementing periocular recognition. The current project uses a combination of several methods for attaining an effective and efficient model.

### **3.1 Convolutional Neural Network- Based Periocular Recognition in Surveillance Environments:**

#### **3.1.1 Background, emergence, and overview of periocular recognition**

The paper [1] presents a periocular recognition-based model for deployment in surveillance environments. It is highlighted that facial recognition systems are currently deployed on a large scale to prevent crime, accidents, privacy, security, etc and thus is one of the most active research areas [1]. It is said that facial recognition systems show excellent performance in most scenarios. However, aging, facial expression changes, occlusions have affected these systems widely cause them to fail [1]. To address this issue, various experiments were performed to emerge at a solution which led to the concept of periocular recognition [1]. It is found that these systems are accurate in comparison to the widely used existing facial recognition systems and addresses all its shortcomings [1].

A specific pre-processing technique called the focus assessment method is included to improve the performance of this model [1]. A focus assessment method is included so that all blurred images are excluded from the training process. This seems to help enhance accuracy of recognition [1]. Further feature normalisations are also done in the numerous layers in the convolutional neural network followed by comparison between feature vectors which provide information on the recognition accuracies [1]. This method implements a score fusion on loose regions of interest and tight regions

of interest. The datasets used here are a custom- made Dongguk periocular database and the ChokePoint database [2],[3].

### **3.1.2 Pre-processing and Feature Extraction**

The model involves an adaptive boosting algorithm on the input images as pre-processing to detect the face regions from the input images. Then dlib facial landmarks are used to detect various facial features required and various coordinates are calculated [4]. Next, a 5x5 kernel is used which helps in measuring the focus score [5]. A certain threshold is set which must be met by the focus score to proceed to the next step of performing feature extraction.

The feature extraction is done by using a Convolutional neural network. CNN is chosen due to its powerful ability to extract comprehensive features for various visual patterns in faces [1]. A pre-trained model of VGG face-16, which consists of 13 convolutional networks and 3 fully connected layers is used in the model [1]. The pre-processed pictures are passed as inputs to this CNN, resulting in a feature map as output [1]. This feature map comprises various values corresponding to various features in the region of interest. The extracted feature maps then undergo comparison for recognition. The comparison is performed using a Euclidean distance metric between the feature maps of input and gallery pictures [1]. The resulting similarity score is said to be the recognition score which aids in obtaining the final output [1].

### **3.1.3 Feature Map Comparison and Recognition Score Generation**

The model used Euclidean distance between the input and gallery images as a measure to find the similarity between two images. The computation is done between the feature maps, obtained after the feature extraction process, corresponding to the input and each gallery image [1]. Each comparison involves the computation of two Euclidean distances, one based on the loose ROI and the other based on tight ROI. These two distances are combined to obtain a fused final score which is used for determining whether a certain input image is within some known class in our database [1]. Further the recognition scores are computed to see which gallery image the input image matches to. The

recognition scores determine the result of recognition, and the details of the recognized person are fetched from the database and shown to the user.

### 3.1.4 Results and Conclusions

In the model [1], CNN-based periocular recognition was proposed for surveillance environments in the visible spectrum. The tests and experiments were done on the custom-made dataset and an open database.

The proposed method of performing a focus assessment in the pre-processing step, performing feature extraction using a convolutional neural network, comparison of feature vectors using Euclidean distances is found to be more effective and accurate compared to various other existing methods [1]. From the observed results it was concluded that this method of periocular recognition can be used as an alternative to facial recognition systems in all the fields [1].

A few unique aspects were considered in the model as stated. Color information was also used when extracting features from the images using a convolutional neural network. This helps enhance accuracy in the recognition process [1]. The focus assessment step for excluding the severely blurred images prevents recognition errors to a great extent [5]. The performance is also enhanced by score fusion on the loose regions of interest and tight regions of interest [1].

The only issue that could arise in the model is that CNNs have a tendency of having an overfitting problem in which the network becomes extremely dependent on the training data and could produce wrong results [1]. However, this issue can be taken care of in the model by inculcating dropout methods, [6,7]. The other drawback could be the enormous training time required by the convolutional neural network. This is inevitable for effective and efficient results in real time applications.

To conclude, the model is found to be better in performance and accuracy compared to other existing models. The accuracy is better even in the presence of occlusions than compared to face recognition and iris recognition combined [1].

## 3.2 Periocular Biometrics in the Visible Spectrum: A Feasibility Study

### 3.2.1 Introduction

Progress in Iris recognition research has enabled rapid strides in the development of ocular methods for facial recognition. Beside the iris two other features have been researched for their role as a biometric. The blood vessel pattern across the retina is believed to be unique across individuals. [8] This is called Retinal Vasculature. Another metric which is believed to be unique is the vasculature pattern in the sclera of the eye [9]. This is called Conjunctival Vasculature.

The paper puts forth an idea that there is significant potential in exploring these vasculature patterns along with the iris using a multispectral sensor for acquiring the image.[10] In Spite of the progress made in ocular biometrics there are some notable challenges. The iris is in a moving eyeball which is further located in an independently moving head. Therefore, reliably localising the iris in images obtained from a distance can be difficult [11]. Also, since the iris is usually captured in the near infrared spectrum there must be necessary invisible lighting (700 to 900 nm) [10] to acquire the image. To capture the Retinal vasculature the subject must be cooperative and near the camera. Even though Conjunctival vasculature can be imaged from a distance, extracting the features like curvature of the sclera, the specular reflections in the image and the fineness of the vascular patterns can be challenging to the biometric system [12]. The paper attempts to mitigate some of the concerns arising due to the classical methods of taking the ocular region for recognition. The small region around the eye is considered as an additional biometric. This is called the periocular region. The potential of this periocular region as a biometric is explored using colour images pertaining to the visible spectrum. [10]. The paper further goes on to list the benefits of using the periocular region as a biometric [10]. In situations where we are unable to reliably get the specifics of the iris or the image of the iris is not recorded at all, we can extract features of the area around the eye to validate the identification. The ROI used in this paper is a good balance between using the entire face and using only the iris information. When an image is captured from a distance, we may get good facial features, but the iris

features will be of poor quality. On the other hand, if we ignore the face and capture images at a close range then we must rely on the iris for facial recognition. The paper further explains that it has not used near-IR spectrum for capturing the image, but the eventual goal is to capture the periocular image in both the visible and the near-IR spectral bands.[13]. This combination would combine both the iris and the periocular details into one texture which would make for a secure biometric system.

### 3.2.2 Database Used

In two separate sessions 889 high resolution face images were captured. The camera used to capture the images was a Canon EOS 5D Mark II camera. The Camera settings when capturing was: Auto Focus, Maximum Resolution (21.1 Megapixels- 5616 x 3744), ISO Auto, Portrait Mode, Optical Vibration Reduction Image Stabilisation and JPEG format.[10]. The camera was placed approximately 4 feet away from the subject. [10]. The width and height without eyebrows were 419892 x 182400 and 265713 with eyebrows. The research assembled these images to two different databases: DB1 and DB2. DB1 consists of 120 images (60 for probe and 60 for gallery) with two periocular images (left and right eye) per subject (30 subjects). DB2 has 958 images (898 probes and 60 gallery).[10]. The first Database is used for parameter selection and the second one for evaluation of the matching performance.

### 3.2.3 Proposed Method

Two different approaches were adopted. The first one considers the global information and the second one Local information. Global features that are representative are colour, shape, and texture [14]. They are represented as a vector with fixed length and these fixed length vectors are simply compared in the matching process. While on the other hand the local feature approach detects a set of key points and then encodes them with the surrounding pixel values which makes a local key descriptor [15,16]. The matching key points are then compared between two images. Since one image does not have the same number of matching key points as the next image each individual key point of image 1 has to be compared to key point of image 2. This must be done to the entire

database which increases the time taken for matching. To achieve constant time using the local approach a bag of words representation has been proposed [9]. The paper makes use of both approaches since the global scheme provides faster matching with the fixed length vectors and the local feature approach offers the advantage of distinctiveness [10].

The proposed periocular recognition process consists of a sequence of operations:

- *Image Alignment:*

The iris, sclera and the eyelid components of the periocular region can be represented in a coordinate system. Even though the iris can move, the data was collected in constrained conditions. Thus, we can ignore any variations caused due to the motion of the iris or the eyelids. The accurate detection of the eyelids is difficult. Additionally, we also cannot use the inner and the outer points of the eyelids because there can be multiple candidates [10]. Since frontal iris detection can be performed well due to the circular geometry of the iris and the clear contrast between the iris and the sclera, the iris is used for image alignment. The paper uses a public domain iris detector based on Hough transformation for localising the iris [10]. Only global matching needs iris-based image alignment since the key points in the local approach can be individually compared.

- *Feature extraction:*

The global features are extracted using all the pixel values in the detected region of interest that is defined with respect to the iris without considering the distinctiveness of the iris. Two distribution based [8] descriptors are used. Gradient orientation and Local Binary Pattern [11]. For local matching since a set of interest points does not take into account the rich detail that is present in the Region Of Interest and also proves difficult during occlusion, a publicly available SIFT Implementation was used as the local matcher [12].



- *Match Score Generation:*

Euclidean Distance is used to calculate match scores for Global Matching. The distance ratio-based matching scheme [13] is used for the local matcher (SIFT).

### 3.2.4 Pros and Cons of Proposed Method

Using a global and local matcher together is efficient since the global approach is faster and the local approach considers more details in the images of the subjects. Another advantage of the proposed sequence of steps is that matching the left and right side of the images individually and then combining them enhances accuracy since the model is better trained with every image. But the downside of using this approach is that since the model is trained with the same constraints as that of iris detection [10] it may not work with changes in expressions. Additionally, the model may not initially be accurate enough for all kinds of databases.

### 3.2.5 Conclusion

The paper investigated the use of the periocular region for biometric recognition and evaluated its matching performance using three different matchers based on global and local feature extractors, viz., GO, LBP, and SIFT. The time taken for feature extraction is about 6 sec and matching time was found to be negligible on 4GB RAM PC. A face recognition experiment was also performed with a commercial face recognition engine, FaceVACS [14]. The experiment yielded a 100% accuracy from which we can conclude 2 notable ideas. Periocular region contains ~80% identity details about the face which complies with an earlier study referenced in the paper [15]. Also, periocular recognition should be used in addition to the primary biometric system or in cases where the primary biometric system cannot be used. Future work includes using multispectral information during feature extraction and looking into the effects of cosmetics on the recognition model by using it around the periocular region [10].

## 3.3 Periocular Recognition using CNN Features Off-The Shelf

### 3.3.1 Introduction

The paper [17] states that periocular biometrics is a compromise between two entirely different parameters, the iris where long distance capture or surveillance may not be available, and the face that may be accidentally or deliberately obscured or occluded. It is also very suitable for non-cooperative biometrics since it can be captured largely relaxing the acquisition conditions [32]. In addition, it has shown to be more resistant to expression variation [18], aging [19], plastic surgery [20], or gender transformation [21], as compared with the entire face. Additional biometric systems such as fingerprint recognition along with iris recognition are incorporated, which would then further enhance security and performance. Over the last few years, Convolution Neural Networks (CNN) have advanced and are hugely popular for image processing or vision-based tasks. However, with recent works on face recognition [22] and detection [23], iris recognition [24], soft biometrics and image segmentation [25], their application is extremely limited. One reason for such limited research is due to the lack of large amounts of training data, as required by deep learning methods [32]. As a result, it is unnecessary to design, create a new CNN and train exclusively for periocular recognition due to insufficiency of large-scale data [32]. Therefore, ImageNet Large Scale Visual Recognition Challenge (ILSVRC) off-the-shelf network models trained on the ImageNet database[<http://www.image-net.org>] that can classify images into 1000 different object classes are used. Another reason to use these off-the-shelf networks is because of the success rate they provide not only in object detection and classification [26]. This paper will be looked into periocular recognition and also be compared with the traditional baseline Local Binary Patterns (LBP) [27], Histogram of Oriented Gradients (HOG) [28] and Scale-Invariant Feature Transform (SIFT) key-points models. The key experiment in this paper is to demonstrate that these off-the-shelf network models have a nearly 40 percent better performance over the traditional baseline using Equivalent Error Rate (EER) as the metric, and it also reported that when both off-the-shelf models and

traditional baseline models are combined, the overall performance increased [32]. Another point to note is the metric used here, i.e. Equal Error Rate (EER) which is an algorithm for the biometric security system which is used to determine the threshold value for the False Rejection Rate (FRR) and False Acceptance Rate (FAR) and, if the proportion of these FRR and FAR values is low, the accuracy rate for the biometric system is higher.

### 3.3.2 Periocular Recognition using CNN Architectures

The basic design details of all CNN architectures are explained along with how features are extracted and not only derived from convolution layers, but also from various techniques such as ReLu, fully linked layers, max pooling and dropout available in Matlab r2018a [32]. Below are the list wise architectures used in this experiment:

- **AlexNet** [29] is a network of 25 layers of which 5 are convolution layers. This has bagged 1st position in ILSVRC 2012 with just an error 15.3% and dominated the runner up by almost a factor 10% more by points.
- **GoogLeNet/Inception v1** [30] is a network of 144 layers of which 22 are convolution layers. This network has just an error rate of just 6.7% and has such a good design which allows a multi-level feature extraction in the same layer with the help of convolution filters.
- **ResNet** [31] this network is of 2 versions; ResNet50 and ResNet101 which has 177 and 347 total layers respectively (50 and 101 convolution layers respectively). It has won the ILSVRC 2015 with the top-5 error of 3.57% which by far is the best for periocular recognition. This architectural network also eases the training of CNNs by sending the input of lower layers to higher layers by hopping over the intermediate layers also known as residual connections.

- **VGG**[33] is a much more extensive network with a much higher complexity where the convolution filters are 3x3 which would help in producing larger layers.

### 3.3.3 Periocular Recognition using Traditional Baseline Systems

Here primarily addresses the old conventional methods of extracting features that have been commonly used for periocular recognition [17]: LBP, HOG and SIFT. HOG and LBP are quantized into 8 bins histogram per block [32]. Simple Euclidean distance measurements are then authenticated, but it has been observed that the  $\chi^2$  distance with normalized histograms provide better results, so it was used in this work [34]. Regarding the SIFT descriptor, key-points extraction is done first using the difference of Gaussians functions in the scale space.

### 3.3.4 Dataset Used

The UBIPr periocular database which was acquired with a CANON EOS 5D digital camera in 2 sessions, with distance, illumination, pose, and occlusion variability is also used here [35]. Image resizing takes place by interpolation to the same sclera radius which has been used for normalization as it is not affected by dilation of the pupil [32]. Further they have taken the square of 7.6 times the sclera radius for having sufficient marginal space and for alignment for the eye [32]. Later, these images would be grey-scaled and enhanced by Contrast-Limited Adaptive Histogram Equalization (CLAHE) [36]. All the images would be resized and then would be matched to the input size of the CNN and subtracting the pixel-wise mean image of the whole dataset for the recognition enhancement [32].

### 3.3.5 Results

The final normalized and pre-processed images are sent into each layer of the feature extraction layer of the CNN, the extracted features from the CNN are later used for calculating the accuracy [32]. These extracted features are then vectorized into feature vectors in vector space and compared for similarity by using either chi square distance, cosine similarity or simple Euclidean distance. The

observed result is that the EER values and chi square values for each CNN architecture was calculated and compared. The interesting point to note in the experiment is that the EER values are minimum in the random intermediate layers with either a chi square or cosine similarity scores for all the CNNs. Detection Error Trade-offs (DET) curves i.e., between the False Acceptance Rate (FAR) on X-axis and False Rejection Rate (FRR) on the Y-axis will be plotted for exclusively for all CNNs and as well as the combination of both CNNs and traditional baseline networks. These DET curves will help us understand the overall performance and mainly describe the Equal Error Rate (EER). It was seen that the ResNet architectures are giving the least shallow, very deep, and best performing network for periocular recognition. It was concluded that ResNet101 is the best for architecture CNN for periocular recognition with an equal error rate of just 5.6% and a FRR of 14% at FAR=1% [32]. These values show that the CNNs are far more efficient than the traditional baseline systems by almost over 40%. Another experimental observation made is that the equal error rate for the fusion of the best CNN and traditional baseline features, this came to a value of 5.1% for equal error rate and FRR of 11.3% which is almost an improvement of over 20% [32]

### 3.3.6 Conclusion

The best results are given by the ResNet architectures (ResNet 101), which are giving the least shallow network that are deep enough to give the maximum efficiency network with an EER of just 5.6% [32]. GoogleNet and VGG also provide good FRR performance. On the contrary, AlexNet is the least efficient and worst performing CNN. It is fascinating to see that VGG does not give a better performance over other CNNs when it has been exclusively developed for working on detecting faces and extracting information from them. This tells the effectiveness to the periocular recognition problem of those to generic object classifiers in neural networks.

## **3.4 Periocular Biometrics: Databases, Algorithms and Directions**

### **3.4.1 Introduction**

The paper [37] is providing a framework which covers different aspects of databases and algorithms for periocular facial recognition. It covered the classification of features for periocular recognition into global analysis and local analysis. Global analysis refers to extracting the properties of entire ROI. Local analysis refers to extracting properties of the neighbourhood key points. Modality has been generated due to concerns regarding the performance of facial system or periocular system-modular study only focus on the extraction of periocular region, but automatic detection and segmentation has been working in post-modular Periocular facial detection. Face detector such as VJ(Viola-Jones) [38] first detect the whole face then perform the periocular region extraction. Classification can be done on the basis of soft-biometric (it refers to classification based on height, colour, gender and so on). While soft-biometric can't be considered as main criteria but still it can help to reduce the search space and provide a boost to the performance. In recent study it is claim that by combining the soft-biometric with textual feature will boost the performance. An interesting study [37] claims the eyebrow region show considerable region in gender classification

### **3.4.2 Datasets Used**

This study provides a general view over the variety of databases generated such as Iris database, facial and periocular database [37]. Every single database has its own pros and cons. In case of iris [39] it provides a high degree of randomness and is also externally visible but has the limitation that eyebrow or other parts may not appear. Databases which captured the images have been made public FOCS (NIRportal), UBIPr (digital camera) or CSIP (smartphones) [37] and these databases are also the one which are widely used. The data in these databases have been acquired through personal devices such as smart-phones and tablets but in some scenarios, accuracy may be low due to low resolution. Similar problems have been encountered while processing the videos captured with surveillance cameras.

### 3.4.3 Pros and Cons

Periocular recognition tries to solve the problems of simple facial recognition.

Lots of data have been received using multiple sources (data may be received from personal devices or surveillance cameras) and after performing cross-modality, cross-spectral, hyper-spectral or cross-sensor led to mismatching of data. Data exchange often happens between the agencies working towards similar goals such as law enforcement agencies, or maybe different departments of the same agencies.

When data is received from multiple sources it will co-exist in most scenarios but leads to mismatching of data or format in some cases. One of the major advantages is in the field of safety and security. Agencies like Law enforcement use facial recognition to detect the criminal and find the missing children. Under the scenario when data came from heterogeneous sources, periocular modularity shown superior to the iris modality.

### 3.4.4 Conclusion

This paper [37] aims to provide insight in the issues and challenges of periocular biometric research. Due to increase in heterogeneous data led to the demand of modularity. The periocular region is more accurate and tolerant towards the change in expression. It is more reliable in the partial face scenario. Even in some cases it can help to reconstruct the face or segment of the face. Periocular have been rapidly involved against facial and iris recognition. Iris recognition is more suited towards the NIR spectrum. Periocular modularity work with higher potential against the cross-modality, cross-spectral, hyper-spectral or cross-sensor matching.

## **3.5 A novel periocular biometrics solution for authentication during Covid-19 pandemic situation**

### **3.5.1 Introduction**

The ongoing pandemic has challenged the world in all dimensions, inclusive of the biometric systems that were considered to be the spinal bone of the country's security system [40]. With the extensive use of face masks, the existing face biometric authentication systems are failing to aid in effective recognition. Thus, under this situation the use of periocular region as a biometric trait is found to be a better and effective solution in comparison with the existing systems [40].

Periocular region refers to the periphery of eyes which contains eye, eyebrow and pre-eye orbital region [40]. Here various procedures are performed, and it was found that periocular region based recognition works well even when the face is occluded and non-cooperative. In the paper [40], researchers extracted handcrafted features or non-hand-crafted features for matching purposes and equal weightage is given to handcrafted, non-handcrafted and the limited semantic information available. A feature fusion method was used to combine all three features and result in a feature vector which was fed to a multiclass support vector machine for classification. HOG was used in order to extract handcrafted features, pre-trained Convolutional Neural Network models were used to extract non-handcrafted features and a 5 layered CNN was used to extract gender information [40].

The emphasis of this paper was laid on understanding the effect of eyeglasses, effect of eye occlusions and pose variations specifically which was achieved by using three benchmark databases, which are UBIPr, Color FERET and Ethnic Ocular.

### **3.5.2 Databases Used**

In this study, three benchmark databases were chosen for assessing the effectiveness of the proposed feature fusion approach.



UBIPr database created by Padole and Proenca (2012), Ethnic Ocular database created by Tiong et al. (2019) and Color FERET database created by Philips et al. (2000) has been selected to analyse the effect of glasses, masked eye portion and pose variation [40].

- *UBIPr database:*

This database contains total 10,252 RGB periocular images from 344 subjects in .bmp format with 501×401 pixels of image resolution [40]. Images in this dataset were captured from 4 m to 8 m distance to include distance variability in a controlled environment.

This dataset also contains metadata files for each image which include coordinates of canthus points, centre of the iris, end points and mid points of eyebrow as well as information about gender, level of pigmentation and angle of gaze etc [40]. Some images in the dataset suffer with hair and eyeglasses occlusions and pose variation arising from tilted heads.

- *Ethnic ocular database:*

This database consists of two folders namely Train\_data and Test\_data. Train\_data folder contains 70,142 RGB shot on 623 subjects in different pose, illumination and angles [40]. The Test\_data folder contains 15,262 RGB periocular images from 411 subjects. All the images are stored in .JPG format with 80×80 pixels image resolution [40]. This database also contains metadata files including information on name, nationality, ethnicity, gender, profession, and number of pair images available [40].

- *Color FERET database:*

This database contains 11,338 RGB periocular images from 994 subjects in .ppm format with 256×384 pixels of image resolution [40]. Dataset includes the metadata files for each image which comprises of the iris centre coordinates, coordinates on

nose, mouth and information on gender, race, expressions, etc. This dataset includes information about occlusions from hair, eye wear, pose variation etc as well [40].

### 3.5.3 Pre-Processing

In the study [40], a unique and effective region of interest algorithm is detailed. In general scenarios, the centre of iris is used to obtain the region of interest, but this study throws emphasis on how that method is faulty with occlusion, closed eyes, pose variation etc. Thus, an algorithm suggested by Liu et al. (2017) is used for region of interest extraction. In this method, the two canthus points of each eye are considered for obtaining the coordinates of the bounding box enclosing the required region of interest.

The algorithm is as stated.

#### Algorithm 1

**Input:**  $(x_1, y_1)$  are the coordinate for medial canthus points and  $(x_2, y_2)$  are the coordinate for lateral canthus points.

**Output:** Coordinated of the rectangle enclosing ROI

**Step 1:** Calculate Euclidean distance between medial and lateral canthus points

$$D((x_1, y_1)(x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

**Step 2:** Compute  $L_p = (L_{px}, L_{py})$ , midpoint of the line which connects medial and lateral canthus points.

$$L_{px} = (x_1 + x_2)/2 \quad (2)$$

$$L_{py} = (y_1 + y_2)/2 \quad (3)$$

**Step 3:** Compute the top left coordinate  $(x_3, y_3)$  and bottom right coordinate  $(x_4, y_4)$  of rectangle enclosing the desire region of interest

$$(x_3, y_3) = (L_{px} - aD, L_{py} - bD) \quad (4)$$

$$(x_4, y_4) = (L_{px} + aD, L_{py} + bD) \quad (5)$$

Here, a and b are constants whose values are chosen to cover the required periocular region. [40]

Next, in order to deal with poor resolution images this extracted ROI undergoes another effective method called CLAHE (Contrasted Limited Adaptive Histogram Equalization). This technique is an advancement of the Adaptive Histogram Equalisation (AHE) technique [40]. The latter is not utilised as it is prone to noise over amplification and the contrast is enhanced globally [40]. Whereas CLAHE aids in highlighting the various edges of the periocular region that help in overall efficiency amplification for recognition. With this technique, each ROI extracted image is divided into  $N \times N$  sub images where N is the number of tiles selected by the user, followed by calculation of the histogram for each sub image and its high intensity value [40]. A threshold is set to use as clip limit and for each bin in the histogram, redistribution of clipped pixels is performed uniformly to all bins, thus obtaining the normalized clipped histogram [40].

### 3.5.4 Feature Extraction & Recognition

In the proposed Methodology, HOG is used to extract the handcrafted features, and pre-trained convolutional neural network (CNN) model is used for non-handcrafted feature extraction. In this study seven off the shelf CNN models are used as feature extractors [40].

- AlexNet (Krizhevsky et al. 2012): 25 layers deep and consists of 5 convolutional layers.
- GoogLeNet (Szegedy et al. 2015): 144 layers deep comprising of 9 inception modules and 22 convolutional layers.

- ResNet 50 and ResNet 101 (He et al. 2016): Residual-Net50 is 177 layers deep and consists of 50 convolutional layers whereas ResidualNet 101 is 347 layers deep and consists of 101 convolutional layers.
- VGG16 and VGG19 (Simonyan and Zisserman 2014): VGG16 is 41 layers deep and consists of 16 convolutional layers whereas VGG19 is 47 layers deep and consists of 19 convolutional layers.

For semantic information such as gender, a 5-layer CNN classifier model is used [40].

The three different features extracted from the various architectures are concatenated to create an ensemble feature vector, which is inputted into the classifier for recognition results [40].

A multiclass Support Vector Machine classifier is implemented for final classification.

### 3.5.5 Results and Conclusions

It is found that the approach followed has a low performance for ethnic ocular and Color FERET dataset due to the high degree of randomness and high pose variations, and varying age [40].

The feature fusion method neutralizes the effect of pose variation and is found to achieve remarkable accuracies for non-ideal scenarios as well [40]. Thus, can be developed and used in real time scenarios even in the current situation of extensive face masks usage.

# CHAPTER-4

## PROJECT REQUIREMENTS SPECIFICATION

### 4.1 Project Scope

A model for periocular recognition is focused on such that, given a set of images that may contain only details of the periocular region (a region in and around the eyes) as input, performs a match or recognition procedure and fetches corresponding known details of the recognized face from a database as output.

Facial recognition is one of the most widely used applications today and is considered the spine of all security systems, however, it fails to accurately handle cases like changes in expressions, aging, occlusions, etc.

One of the main objectives of this model is to overcome and address these shortcomings.

Thus, the aim is to develop an effective model that delivers remarkable accuracy in surveillance environments in non-ideal cases of occlusions and un-cooperative subjects.

### 4.2. Constraints

#### 4.2.1 Design Constraints

- The model is being designed to handle only image inputs.
- The model performs pre-processing for feature enhancements regardless of the quality of the input image to amplify overall accuracy.
- The model uses a pre-trained CNN models for feature extraction thus requiring a significant amount of training time in case of a large number of subjects

### 4.2.2 System Constraints

- Linux and Windows operating systems are supported.
- The model requires a CPU of 1GHz or equivalent and a GPU with high computational power for training.
- The model requires a VGG16 architecture running on a system with a minimum RAM of 8GB.
- A high quality or medium quality camera for dataset creation.
- Memory Space needed over 30GB

## 4.3 Product Perspective

Face recognition algorithms are widely used in various applications in our day-to-day lives. It is considered to be very important due to its applicability in privacy, security, crime prevention, accident prevention, suspect identification, authorized access, and control, and so on. However, face recognition systems are accompanied by various other shortcomings such as changes in facial expression, aging, occlusions, etc. These occlusions could include helmets, eyewear, headgear, burqa, monkey cap, etc.

This arises the concept of using the periocular region for recognition instead of the entire face. It is found to be more accurate than facial recognition in non-ideal scenarios. Periocular recognition-based models are applicable and can be used in place of all facial recognition systems due to its much greater applicability in today's scenario. Additionally, it can also be used in various crime scenes after further enhancements, where facial recognition fails due to the presence of headgear.

### 4.3.1 Product Features

This model product being developed contains various features for smooth functioning as detailed below:

*At the initial stage of training on database:*

- The database inputted to the model to be trained is considered as input.
- The model performs pre-processing such as face and facial landmarks detection, region of interest extraction, Contrast Limited Adaptive Histogram Equalisation technique-based feature enhancements, followed by feature extraction.
- The features extracted are inputted to two classifier models, one for each eye, and then undergo the training phase. The trained models are store in the database so as to prevent repetitive training, thus aiding in execution time reductions.

*At the time of Recognition:*

- The user can input a picture, to the model through the User Interface from his/her file system.
- The product does the entire internal processing consisting of pre-processing such as face and facial landmarks detection, region of interest extraction, Contrast Limited Adaptive Histogram Equalisation technique-based feature enhancements, followed by feature extraction all that similar to that in the training phase, then followed by feature map comparison and recognition. The resulting output is returned to the user.

### 4.3.2 User Classes and Characteristics

This project has the numerous use cases such as:

- Generic Surveillance environments

*User Class:*

- Retail Stores
  - Banks
  - Public Places, etc
- 
- Forensic Department cases  
*User Class:*
    - Forensic department professionals
  - Personal Privacy and Security  
*User Class:*
    - Individuals who have purchased such devices

### 4.3.3 General Constraints, Assumptions, and Dependencies

#### 4.3.3.1 Constraints:

- Constraints on response time due to Deep Learning-based convolutional neural network.
- Insufficiency in data collection for large scale implementation.
- Unsuitable Accuracy for large scale implementation.
- Memory insufficiency in the product with large number of subjects.

#### 4.3.3.2 Assumptions:

- The person to be recognized is in the visible spectrum (broad daylight) and not in a dark environment where IR would be necessary.

#### 4.3.3.3 Dependencies:

- Python



- Technologies: HTML, CSS, JavaScript, Flask
- Libraries: OpenCV, NumPy, TensorFlow, Keras, dlib, scikit
- Models:
  - Pre-trained VGG16 model on ImageNet dataset,
  - OpenCV's caffe DNN model for face detection
  - Dlib's 5 facial landmarks detector

## 4.4 Risks

- Issues with camera and resolution
- Failure in the enormous training process
- Run time issue due to extended time for training
- Dataset private information breach
- Recognition failure in case of extreme use of cosmetics

# **CHAPTER-5**

## **HIGH LEVEL DESIGN**

### **5.1 Introduction**

The purpose of the High-Level Design is to add the necessary detail to the current project, periorcular recognition, and describe the structure of the model to help aid in detecting various issues. The detailed structure such as the database, server and over design is detailed here.

### **5.2 Current System**

Current facial recognition models depend on unobscured facial features. These models struggle to address issues such as the presence of occlusions and variations of pose angles.

For example, in the current situation, wearing masks has become a necessity, and various facial recognition technologies are failing to work as intended. This is where the periorcular recognition initiative hopes to succeed. Aside from that, numerous other accessories such as burqas, helmets, and face shields are used, which the current system cannot facilitate.

### **5.3 Design Considerations**

#### **5.3.1 Design Goals**

Some of design goals of the project are:

- Developing a model for periocular recognition, given a set of images which may contain only details of the periocular region (a region in and around the eyes) as input, from which we output recognition results to the user.
- To perform Augmentation on the images for dataset expansion, then followed by enhancing algorithms to increase the overall accuracy.
- To Develop a fully integrated user interface that would allow the user to input the image, train the model, obtain model accuracy, and perform recognition.

### 5.3.2 Architecture choices

The ResNet101 architecture was considered for this model. Since the Keras library ResNet101 architecture was incompatible with our working environment, the VGG16 architecture was chosen. VGG16 was found to be accurate in comparison with the results provided by the ResNet101 architecture.

In comparison to other architectures, VGG16 is simpler to use and alter.

Since there are fewer parameters in ResNet101, it has a lower time complexity than VGG16.

### 5.3.3 Constraints, Assumptions, Dependencies

#### Constraints:

- The model is being designed to handle only image inputs.
- The model performs pre-processing regardless of the quality of the input image. For example, even if the image has perfect lighting, no occlusions, and is unblurred, image resizing and grayscale conversion have to be done.
- Constraints on response time due to Deep Learning-based convolutional neural network.
- Unsuitable Accuracy for large scale implementation due to the lack of a large dataset.
- Memory insufficiency in case of extremely large dataset
- A fair prediction would not be possible in presence of extensive usage cosmetics

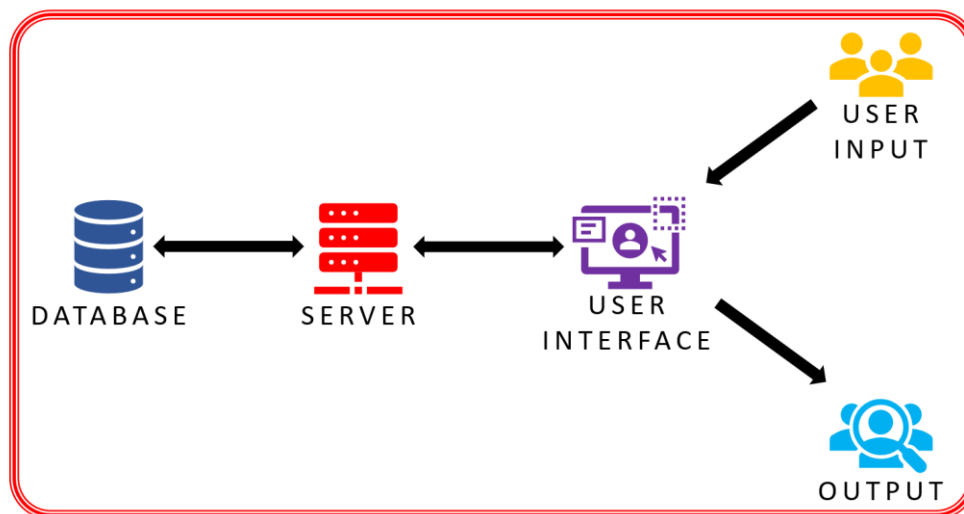
**Assumptions:**

- The person to be recognized is in the visible spectrum (broad daylight) and not in a dark environment where IR would be necessary.
- The subject's pose must be in a way that both eyes are clearly visible, and that their pose angle must be as little as possible.

**Dependencies:**

- Languages: Python, HTML, JavaScript, CSS
- Libraries: OpenCV, ImageDataGenerator, NumPy, dlib, Flask, TensorFlow, Keras, scikit or sklearn
- Models: VGG16 convolutional neural network, OpenCV's caffe model in the DNN module, dlib's 5 facial landmark detector
- CNN model with VGG16 architecture and OpenCV's caffe model DNN

## 5.4 High-Level Design



*Fig 5.4.1: High level system design*

In the model, a user interface is built using HTML, JavaScript, CSS along with Flask to provide the user with the entire functionality of periocular recognition. The model lets the user upload an image which is sent to the server. Ideally, the image should be captured within the visible spectrum, with sufficient lighting. It is taken that the periocular region is clearly visible. For each image in the dataset, augmentation is performed to attain 10 variants for each image of each test subject. Then, pre-processing is done that includes the region of interest extraction to obtain only the periocular region. Following that, Contrast Limited Adaptive Histogram Equalisation is performed to highlight the features and edges for successful recognition.

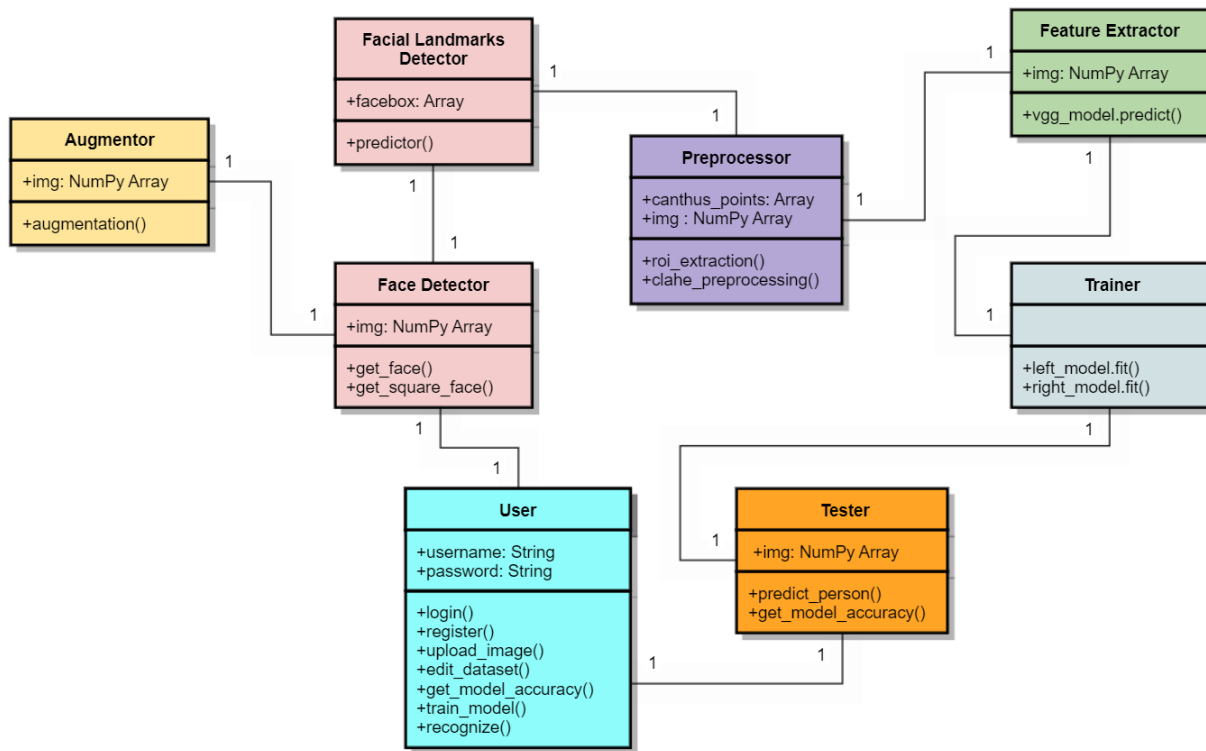
Then feature extraction is performed by using a convolutional neural network, VGG16 architecture that is pretrained on the ImageNet dataset.

Two weighted KNNs are used as the classifiers for each eye. The image from the database along with its true label is fed into the KNNs based on the left or right position of the eye. The classifier is trained on this input data.

When the user chooses to perform recognition, the entire pre-processing and feature extraction is performed on the user inputted image. The feature map obtained is fed to the trained KNNs and the predicted results are processed and returned to the user.

## 5.5 Design Description:

### 5.5.1 Master Class Diagram



*Fig 5.5.1: Masterclass Diagram*

The project design includes seven classes which have one to one interactions with each other. Figure 5.5.1 depicts the entire structure of the classes that have been developed for the project. All these classes work together to give the user the recognition results.

#### 5.5.1.1 Class 1: User

The project design includes a user with a secure authentication, on which he/she is given access to the database and the recognition process as well. The user can either log in to the account or register

in case he/she does not have an account yet. Then after successfully logging in, the user can retrieve the model accuracy, edit, or change subjects or images in the database and can proceed to recognize an uploaded image selected.

#### **5.5.1.2 Class 2: Augmentor**

This class is used for the sole purpose of training data expansion from already existing/collected data, in order to amplify accuracy. It receives each image from each test subject from the database and creates ten variants for this input, and stores it back into the training data directory in the database.

#### **5.5.1.3 Class 3: Face Detector**

This class consists of a face detection model, namely OpenCV's Caffe Model of the DNN module, that assists in detecting the face region in an image. The face region is obtained using the `get_face()` function which returns an array consisting the coordinates of the bounding box enclosing the face region. The bounding box is later converted to a perfect square for consistency using the `get_square_face()` function, which returns the square coordinates of the bounding box enclosing the face region.

#### **5.5.1.4 Class 4: Facial Landmarks Detector**

This class consists of a facial landmarks detector, namely dlib's 5 facial landmarks detector. This aids in locating and retrieving the 5 coordinates of 5 facial landmarks corresponding to the 4 canthus points and one point for the nose. This 5<sup>th</sup> point is ignored in the project since that point is not enclosed in the periocular region. These four canthus points coordinates are prime for extracting the Region of Interest.

#### **5.5.1.5 Class 5: Preprocessor**

From the previous class, the 4 canthus points are taken which consist of the medial and lateral canthus points coordinates for each eye. The region of interest bounding box coordinates are calculated based on these canthus points of each eye, resulting in 4 points, corresponding to the top

left corner and bottom right corner points for each eye that enclose only the periocular region. Thus, region of interest extraction is performed. The whole process is performed by the `roi_extraction()` function.

Following that, the next function in the pre-processing class is the `clahe_preprocessing()`. This function takes in the periocular images extracted in the previous step and enhances the contrast locally highlighting the features and edges for a crisp feature extraction resulting in successful recognition.

#### **5.5.1.6 Class 6: Feature Extractor**

The convolution neural network, VGG16 model architecture, that is pre-trained on the ImageNet dataset is used in this model to function as a feature extractor. The pre-processed input images, i.e., the contrast enhanced periocular images, are fed to the network. VGG16 then extracts the features using the `vgg_model.predict()` function that results in a (1,7,7,512) feature map that is flattened later to (25088,), thus completing the feature extraction.

#### **5.5.1.7 Class 7: Trainer**

This class performs the training procedure on two weighted KNN classifiers having k value set to 3. The flattened features extracted from the enhanced periocular images are fed to the respective KNN along with their true labels and are trained on. It is made sure that all left eye images are sent to the left model and all right eye images are sent to the right model for training, since we have two classifications, one from each model contributing to the final recognition.

#### **5.5.1.8 Class 8: Tester**

This class performs the final validation and testing, i.e., fetching the model accuracy of the trained model and performing recognition based on user input. When the user tries to retrieve the model accuracy, this function runs the `get_model_accuracy()` function that validates the model with the entire test set that has the true labels. The resulting accuracy is returned to the user.



When the user tries to recognize a certain test subject, the class calls the `predict_person()` function that performs the necessary pre-processing on the user inputted image, then performs the recognition and returns the result to the user.

## 5.5.2 Reusability Considerations

### 5.5.2.1 Face Detector

The Face Detector class is reused. It is used first during the training phase and later when the user performs recognition. Its input and output remain the same for both phases and it just aids to provide the bounding box coordinates enclosing the face region in the image.

### 5.5.2.2 Facial Landmarks Detector

The Facial Landmarks Detector class is also reused. It is also used first during the training phase and later when the user performs recognition. It aids in finding the canthus points in the face region which is of prime use in the region of interest extraction step that is necessary for both training and recognition.

### 5.5.2.3 Preprocessor

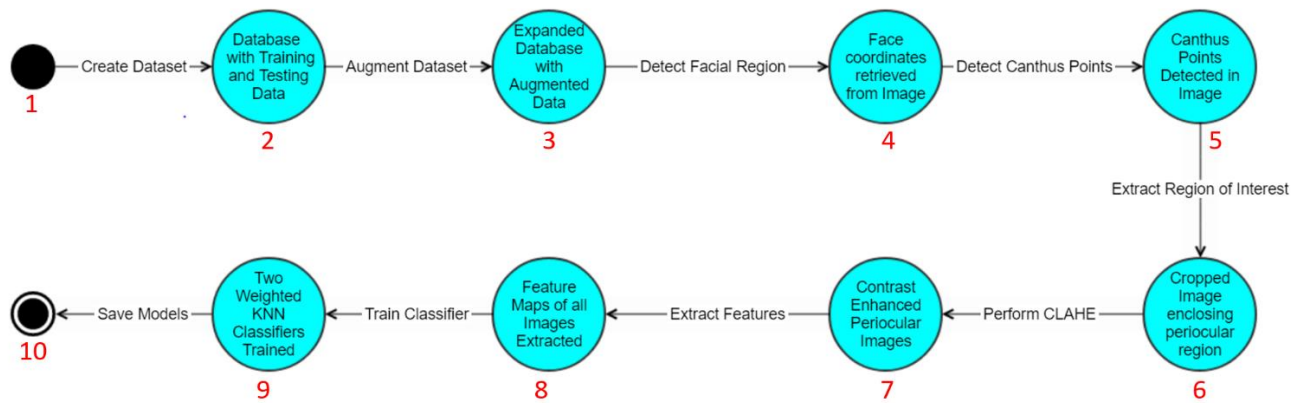
This class pre-processor is reused in the model. Both its functions `roi_extraction()` and `clahe_preprocessing()` are used during the initial classifiers training phase when the database is first made and also later during the testing phase, when the user tries to recognize a certain test subject.

## 5.6 State Diagram

The project is carried out in two phases, one being the training phase and the other being the testing phase. Though they are very closely related, they are still carried out in entirely separate phases since the user does not come into the picture in the training phase. In the recognition phase, we have various user inputs and functions as well which are calling for various functions internally defined.

Thus, we have two state diagrams, Fig 5.6.1.1 depicts the state diagram during the training phase while Fig 5.6.2.1 depicts the state diagram during the recognition phase with user input.

### 5.6.1 State Diagram for training Phase



*Fig 5.6.1.1: State Diagram for the training phase*

#### ➤ State 1: Empty Database (Start State)

At this state, the database is entirely empty that the client must create in order to recognize any test subject.

#### ➤ State 2: Database with Training and Testing Data

This state is entered when the client creates the database where he/she adds the training data for the recognition process.

#### ➤ State 3: Expanded Database with Augmented Data

This state is entered on successfully augmenting each image in each test subject to produce 10 fold data. The database in this state composes of the expanded training data saved after the augmentation function has completed.

➤ **State 4: Face coordinates retrieved from image**

The system is in this stage after the coordinated of the bounding box enclosing the face region in the image is detected and converted to a perfect square.

➤ **State 5: Canthus Points Detected in image**

After the face region bounding box coordinates are found, the canthus points are detected with the help of the dlib 5 facial landmarks detector. On successful detection the model enters into this state.

➤ **State 6: Cropped Image enclosing Periocular Region**

Following the canthus point detection, the coordinates of the bounding box enclosing the periocular region are calculated with the ROI extraction function which sends the system into this state.

➤ **State 7: Contrast Enhanced Periocular Images**

The extracted periocular images from the previous step are taken as input and CLAHE (contrast limited adaptive histogram equalisation) techniques are performed to enhance the features in the region of interest. On saving these contrast enhanced images, this state is entered by the system. This completed the pre-processing phase.

➤ **State 8: Feature Maps of all images extracted**

After completion of the pre-processing phase, a VGG16 pretrained network is used to retrieve the feature maps of each periocular image. The extracted feature maps of shape (1,7,7,512) are flattened to a shape of (25088,). After completing this step for each periocular image, the system enters into this state.

➤ **State 9: Two weighted KNN classifiers trained**

Following feature extraction, two KNN classifiers are instantiated with k value set to 3. The feature maps of all left periocular images along with their true labels are fed to the left KNN classifier model and vice versa. Post training the system enters this state.

➤ **State 10: Saved Models (Final State)**

The system then reaches the final state or the finish state on saving both the trained classifiers into the database which can next be used for the recognition process.

### 5.6.2 State Diagram for recognition phase

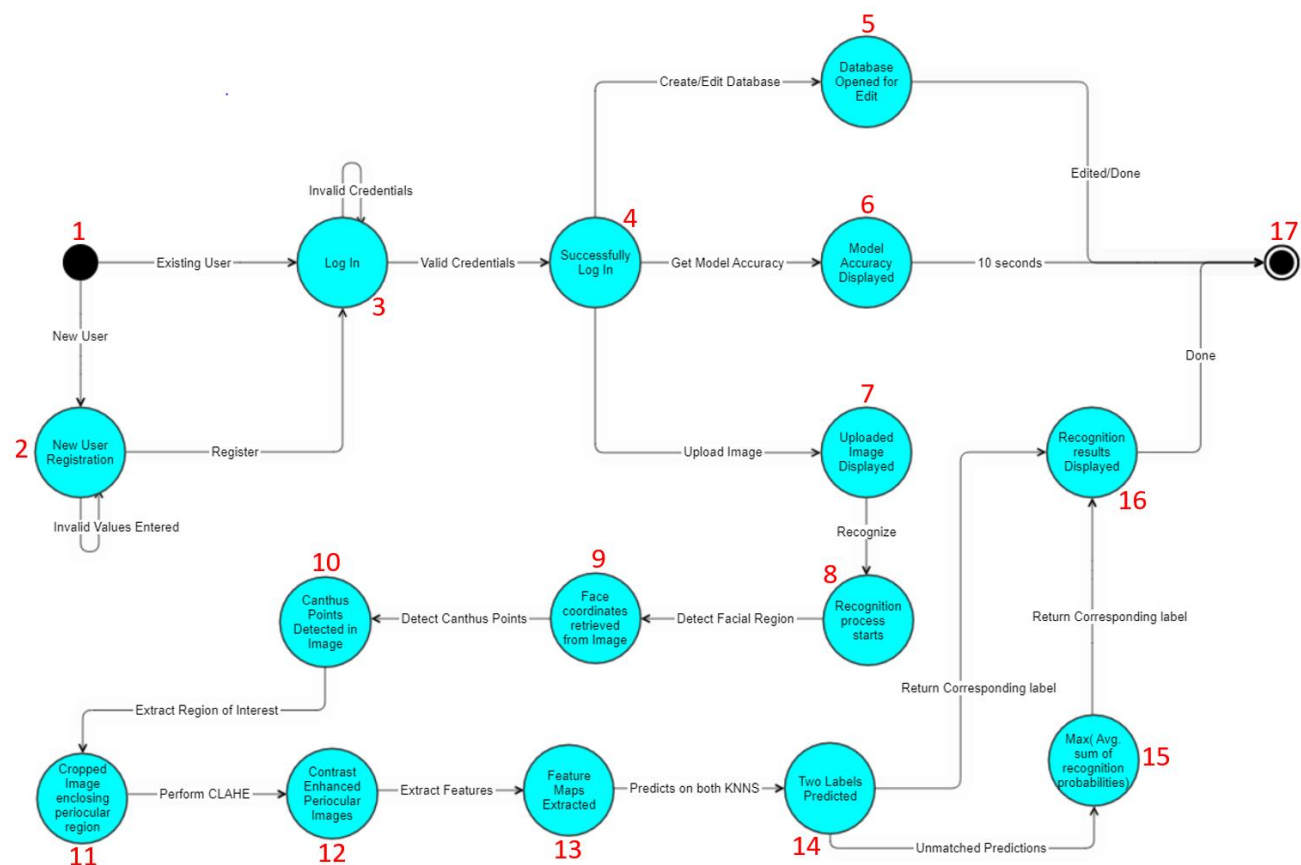


Fig 5.6.2.1: State Diagram for user interactions and recognition phase

➤ **State 1: Home Page (Start State)**

At this state, the user opens the web app, and the home page is displayed. Various options like log in or register are present where the web page responds to the user based on his/her activity.

➤ **State 2: New User Registration**

In case the user does not have an account to proceed and wished to register a new account with the product, the system enters into this state and opens the registration form for the user, after which the user is prompted to log in into to the account.

➤ **State 3: Login**

This state is entered when a new user or existing user choose to log in into the product. A form opens where username and password are requested.

➤ **State 4: Successfully Logged In**

The user on entering his/her username and password, the credentials entered are validation. On successful validation, the system enters into this state and conveys to the user that he/she had successfully logged and can start using the various functionalities provided by the product.

➤ **State 5: Database Opened for Edit**

The system enters this state if the user after login chooses to open the database to view/edit it. The database is opened in a new window if this state is entered.

➤ **State 6: Model Accuracy Displayed**

The system enters this state if the user after login chooses to retrieve the model accuracy. The model accuracy is just displayed to the user in the web app.

➤ **State 7: Uploaded Image Displayed**

The system is in this state when the user decided to perform recognition by uploading an image. The uploaded image by the user is displayed to the user.

➤ **State 8: Recognition process starts**

On the user clicking the recognize button in the web app, the process of recognition starts, and the system enter into this state for a brief span of time.

➤ **State 9: Face coordinates retrieved from image**

This state is entered after the recognition process starts and the image uploaded is successfully loaded into the backend for processing. The system is in this stage after the coordinated of the bounding box enclosing the face region in the image is detected and converted to a perfect square.

➤ **State 10: Canthus Points Detected in image**

After the face region bounding box coordinates are found and the canthus points are detected with the help of the dlib 5 facial landmarks detector, the system automatically flow into this state.

➤ **State 11: Cropped Image enclosing Periocular Region**

Following the canthus point detection, the coordinates of the bounding box enclosing the periocular region are calculated with the ROI extraction function which sends the system into this state.

➤ **State 12: Contrast Enhanced Periocular Images**

The extracted periocular images from the previous step are taken as input and CLAHE (contrast limited adaptive histogram equalisation) techniques are performed to enhance the features in the region of interest. On saving these contrast enhanced images, this state is entered by the system. This completed the pre-processing phase.

➤ **State 13: Feature Maps Extracted**

After completion of the pre-processing phase, a VGG16 pretrained network is used to retrieve the feature maps of the two periocular regions in the image. The extracted feature maps are of shape (1,7,7,512) and then

flattened to a shape of (25088,). After completing this step for each periocular image, the system enters into this state.

➤ **State 14: Two labels predicted**

The two feature maps extracted in the image are then fed to the KNN models, such that the right periocular image goes to the right KNN classifier and vice versa. Then each classifier predicts the label resulting in two predicted labels. This makes the system enter this state.

➤ **State 15: Max (Avg. Sum of Recognition Probabilities)**

If the two predictions made by each KNN model do not match, then the system enters into this state and calculates the average sum of the recognition probabilities and the highest value is noted. The system halts at this stage until the corresponding label of this recognition probability is returned.

➤ **State 16: Recognition Results Displayed**

On completing the recognition phase, if the match was perfect in state 14, the model directly enters this state, and the recognition results are calculated. Otherwise, the highest value of the average sum of recognition probabilities is calculate and the corresponding label is displayed to the user, thus competing the entire recognition phase.

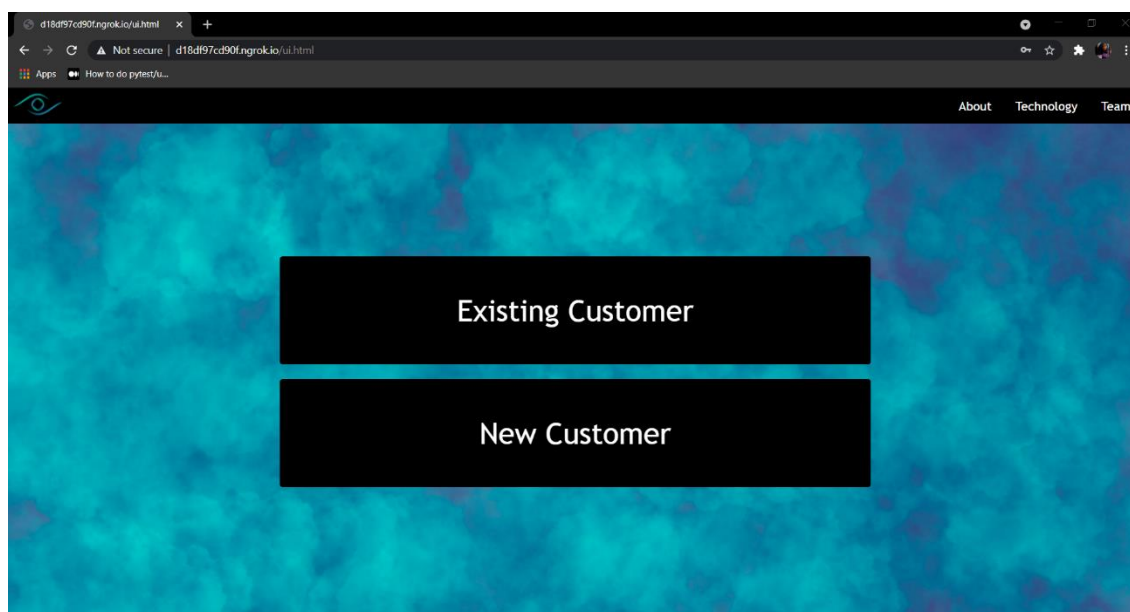
➤ **State 17: Web App Closed (Final State)**

On completion of the recognition phase and closure of the web app by the user, this final/finish state is reached. The system stays at this state until the web app is opened again.

## 5.7 User Interface Diagram

The user interface, a web-based UI, is built on HTML, CSS, JavaScript, and Flask. The application consists of user authentication which gives them access to their databases. This ensures data privacy of various users. Access is given in order to edit the database for adding or removing subjects as per user needs. The model accuracy can be retrieved and can also be further trained on with a single click. For recognition activities, the users can upload or drag their images into the provided space to be taken as input for the model. The necessary functioning takes place when the users select to recognize the uploaded image and the results are displayed back the user.

Various displays of each functionality are shown below:



*Fig 5.7.1.: Home page of the web app*

The figure 5.7.1 displays the home page that the user observes on opening the web app.



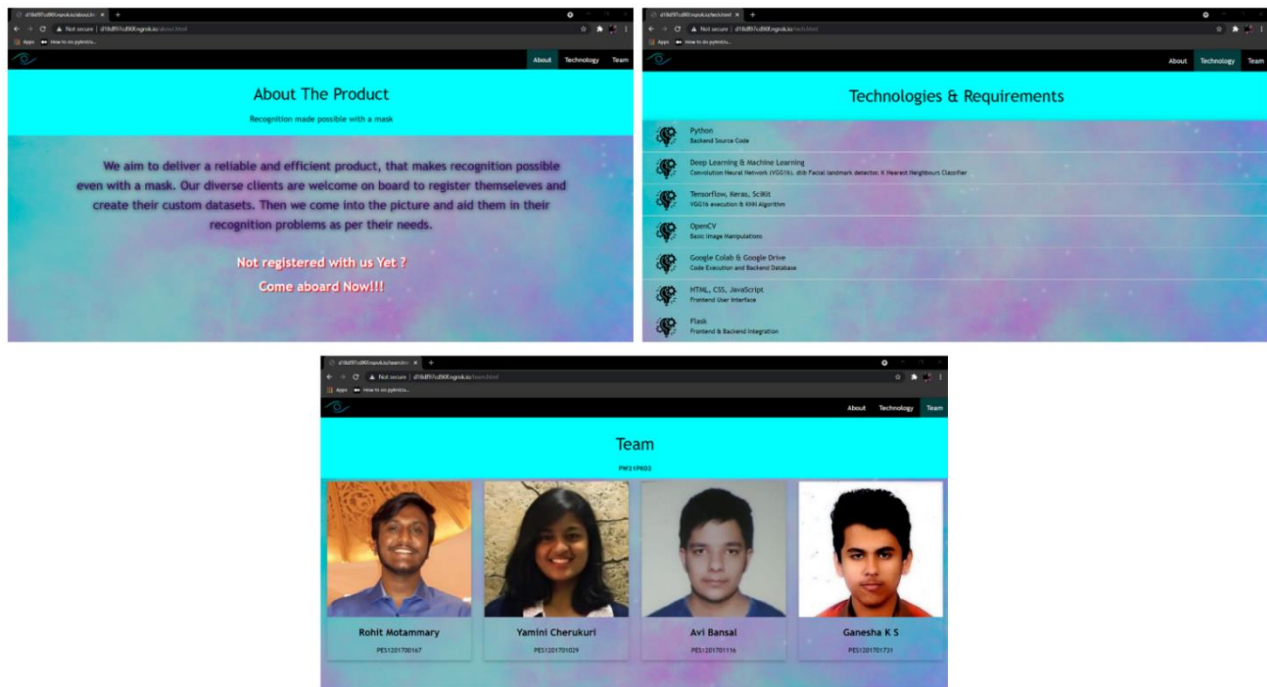


Fig 5.7.2: About, Technologies and Requirements, team page displayed on User clicks

The figure 5.7.2 depicts the “about the product” page, the “technologies and requirements” page, and the “team” page when the user clicks on the “About”, “Technology”, and “Team” navigation buttons.

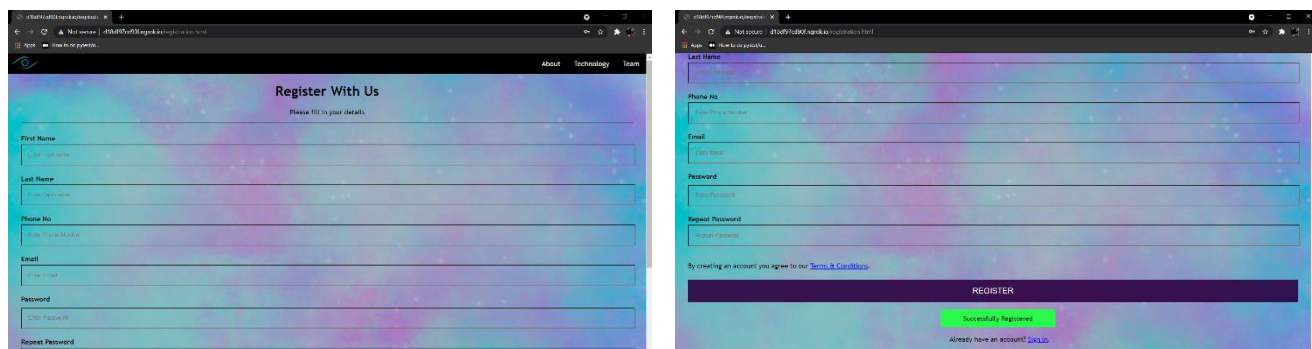
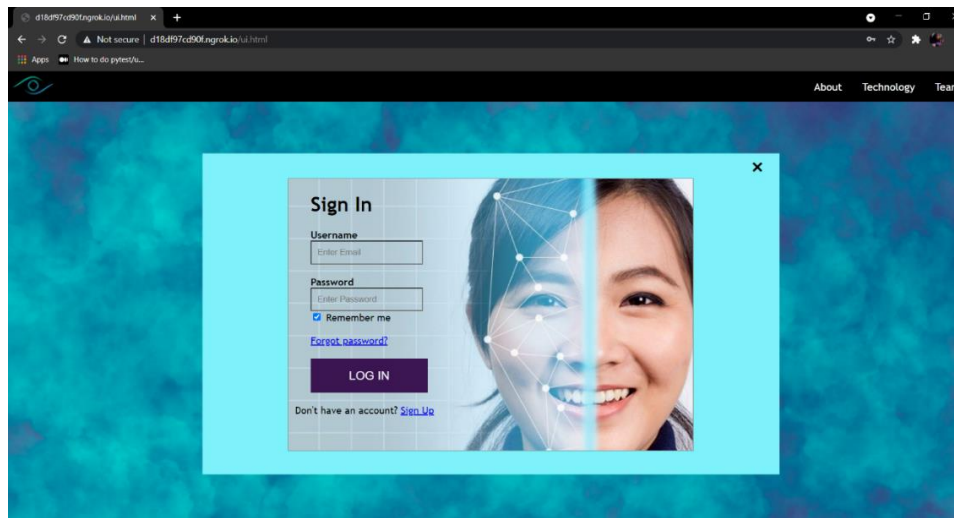


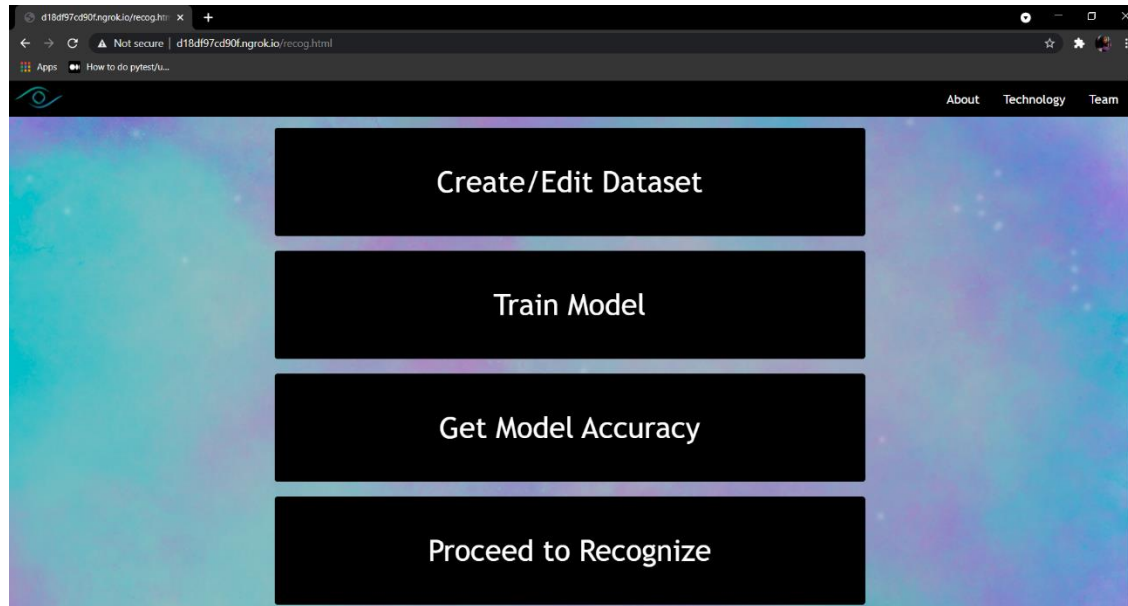
Fig 5.7.3: The registration page on the left and the page after successful registration on the right

In case the user is new to the product, he/she will have to register an account to use all the functionalities provided. On clicking on the “New Customer” button the registration page is opened and is as shown in figure 5.7.3 on the left. The user on entering his/her details and registering successfully, a toast is shown stating “successful registration” as shown in figure 5.7.3 on the right.



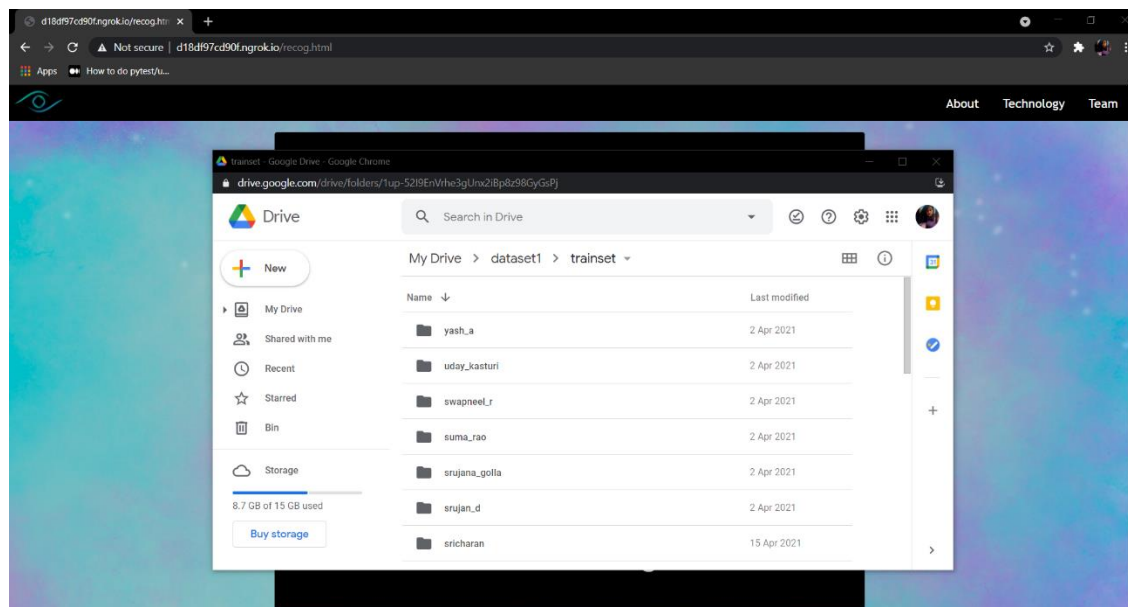
*Fig 5.7.4: User Login Page*

Whether a new customer or an existing customer, both must login into their accounts to use all the functionalities provided by the product. The existing user on clicking the “Existing Customer” button or a new user on clicking on the “Sign in” link after successful registration, open this page depicted in Figure 5.7.5. The username and password must be entered by the user which then is validated to login.



*Fig 5.7.5: Options displayed after logging in*

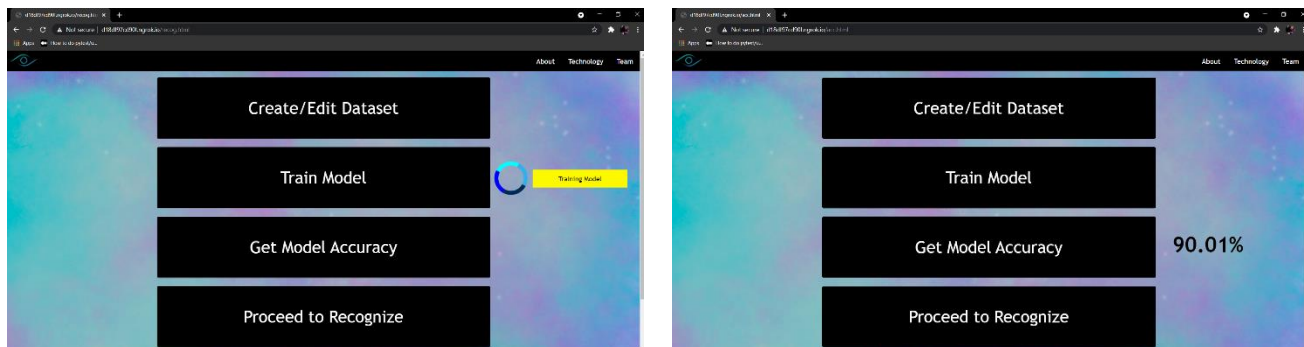
The figure 5.7.5 shows all the options that the user can click on to obtain some service after successfully logging in into his/her account.



*Fig 5.7.6: Database View/Edit Page*

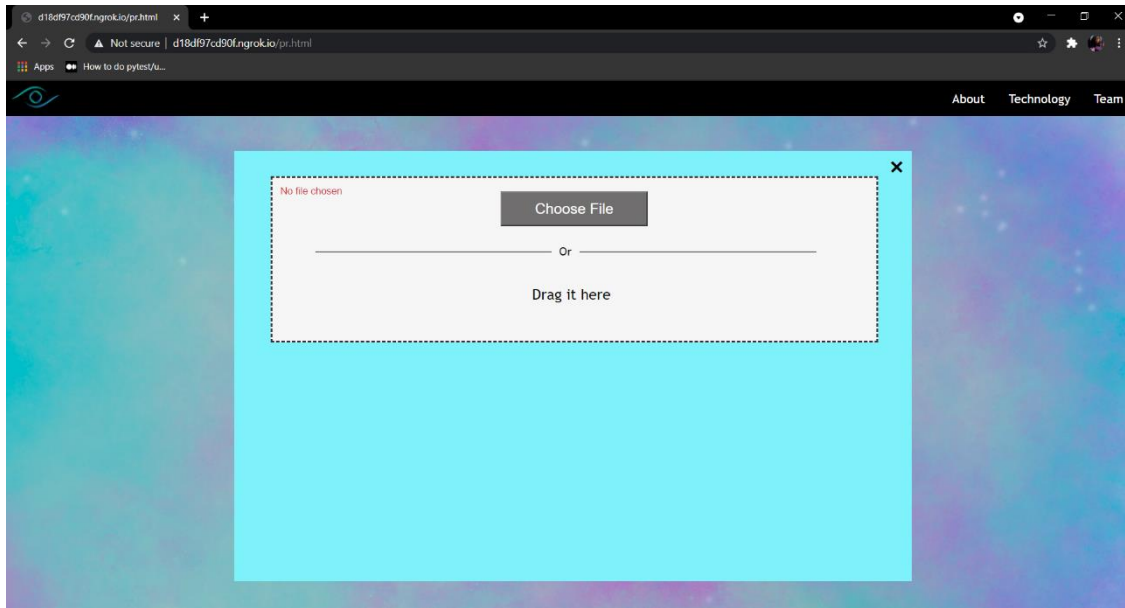
The figure 5.7.6 shows how the database is opened when the user clicks on the “Create/Edit Database” button.

In this new window opened, the user can view, edit or create an entirely new database as per their requirements.

*Fig 5.7.7: Output for “Train Model” and “Get Model Accuracy”*

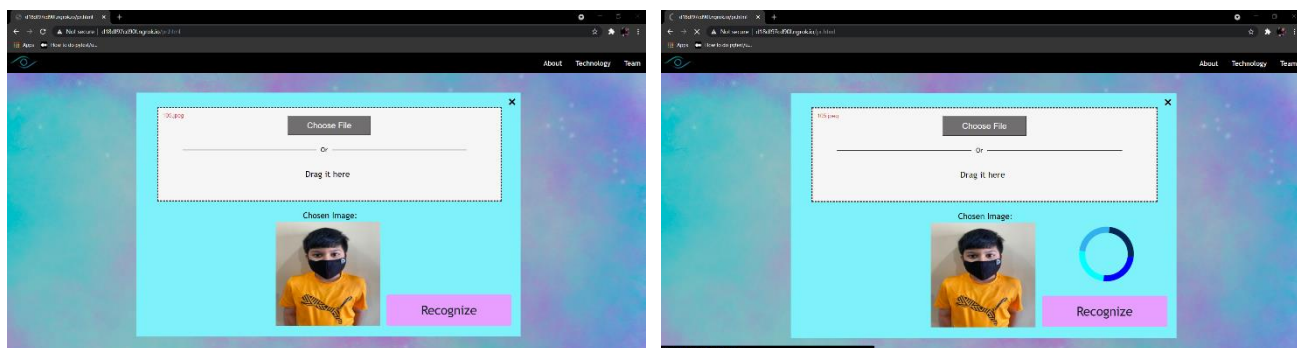
The figure 5.7.7 shows the output when the user clicks on the “Train Model” option on the left and when the user clicks on the “Get Model Accuracy” button on the right.

When the user clicks on the “Train Model” option, a loader is displayed while the model is undergoing training, which disappears after successful training. Next, when the user clicks on the “Get Model Accuracy” button, the model accuracy is computed using the validation data and the training data and the result is displayed to the user for around 10 seconds.



*Fig 5.7.8: Upload Image page*

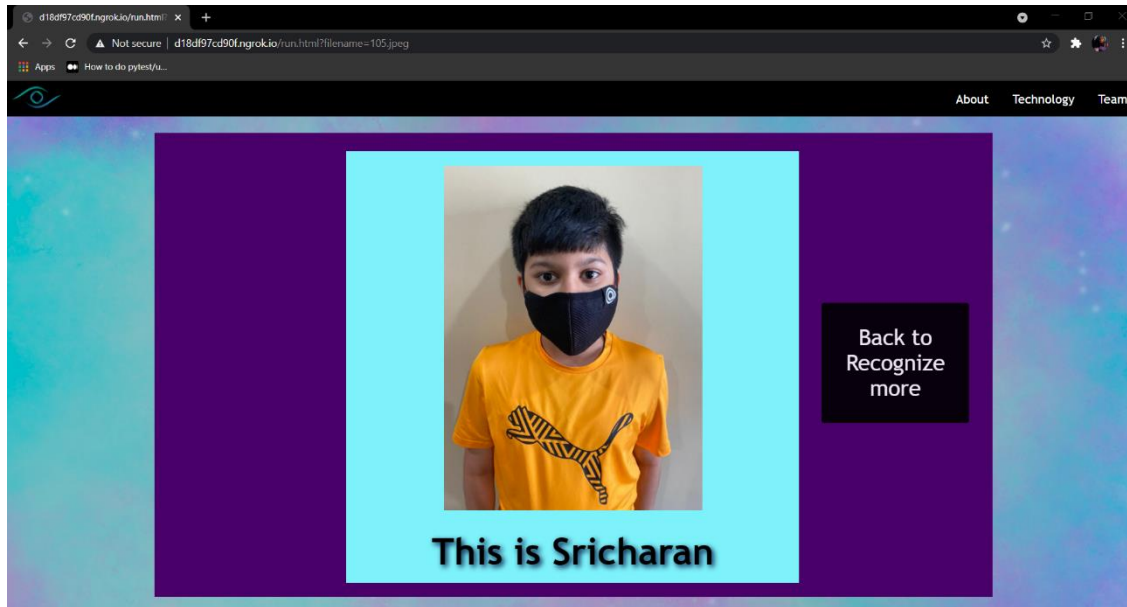
The figure 5.7.8 shows the page for the image upload when the user selects the “Proceed to Recognize” option. An image can be dragged and dropped in the white box or an image file can be chosen as well.



*Fig 5.7.9: After uploading the file and After the user selects recognize*



The figure 5.7.9 depicts the page view after the image uploads where the uploaded image is shown on the left and on the right it shows the page view when the user clicks on the “recognize” button while the process is still on going in the backend. The loader is displayed until the final output is fetched



*Fig 5.7.10: Output page*

The figure 5.7.10 shows the output page where the recognition results are fetched to the user. A “Back to Recognize more” button is provided in case the user wants to recognize another test image. Otherwise, the user can just close the web app.

## 5.8 External Interfaces

- The system must train on a collected dataset of images and undergo pre-processing such as region of interest extraction and CLAHE (Contrast Limited Adaptive Histogram Equalisation) technique for feature enhancements, followed by training for recognition.

- The system should take various images, extract features and perform comparisons for recognition even in the presence of occlusions.
- On successful recognition, the system should fetch the details of the recognized person from the database and display the same to the user as output.
- On failure, the system must inform the user that the detected person is unknown.

### 5.8.1 Software Requirements

- The user must not be given access to the recognition process to avoid tampering.
- The code must be portable from one OS to another to make the system compatible across numerous devices.
- Python
  - Version/Release Number - 3.6.9
- OpenCV
  - Description: Real-time optimized Computer Vision library used for image manipulations.
  - Version/ Release Number - 4.1.2
- Dlib
  - Description: A C++ toolkit used for facial landmarks detection.
  - Version/ Release Number - 19.18.0
- Caffe DNN
  - Description: OpenCV's deep neural network model for face detection.
- TensorFlow
  - Description: Open-source platform which has flexible tools for machine learning.
  - Version/ Release Number - 2.4.1

- Keras VGG16
  - Description: Open-source library that provides python interface for neural networks.
  - Version/ Release Number - 2.4.0
  
- Scikit KNN
  - Description: A machine learning library which supports python and features various algorithms for classification, regression, and clustering.
  - Version/ Release Number - 0.22.2.post1
  
- HTML, JavaScript, CSS
  - Description: Languages used for implementation for implementation of a web-based user interface.
  
- Flask
  - Description: A python written micro web framework used for development of the web-based user interface.
  - Version/ Release Number - 1.1.2

### 5.8.2 Hardware interface

The hardware environment in this system will use the integrated webcam of the device where the app is being run or an external 4K webcam attached. These cameras will play an integral role in the system. There are other hardware requirements recommended for the devices that the app will be executed on, such as, the OS of the devices has to be Windows 7 or higher, MAC OS X v10.7 or higher and ubuntu Linux 14.x or higher. The system has to have an LAN or a wireless adapter (Wi-



Fi). The processor core has to be 2Ghz minimum but the recommended is of 3Ghz or more. The minimum RAM is of 4GB but for a smooth sailing of the app, 8GB or more RAM is preferred. The data stored in the database should at least have space of 32GB but it is advised to have space of 64GB or more to have a good user experience.

## 5.9 Help

Documentation showing the working of the system will be provided along with contact details of the project members for any technical queries. As for the software application, it is ideal for the user to run it in a well-lit environment for accurate, noise-free capture of the face.

# **CHAPTER-6**

## **LOW LEVEL DESIGN**

### **6.1 Introduction**

A detailed introduction to the low-level design of the periocular recognition model is detailed here. It emphasises the procedure, methods and steps followed to achieve the various functionalities included in the model.

#### **6.1.1 Overview**

The low-level design document contains several sections describing the design of the periocular recognition model. Further, the constraints, assumptions and dependencies of the model are clearly enunciated. The document gives a clear in-depth understanding of the various methods and the data members used through the use of clearly explained UML diagrams.

#### **6.1.2 Purpose**

Since the model developed is not a traditional facial recognition system, the purpose of the document is to give the internal logical design of the program. This document is created based on the high-level design of the system previously created. The document describes the modules so that a layman can create a low-level design of the system based on its contents. The code can then be developed with minimum debugging and testing. Additionally, this document would be a resource during maintenance of the designed system.

### 6.1.3 Scope

The document begins with an explanation of the constraints, dependencies and assumptions made while building the model. Then a modular description of the design of the model is given. Additionally, the document describes the model in terms of the system and the actors in the use case diagram. Each use case item and its description are provided in the form of a table. The next section describes the entire system's classes, data members and functions along with a clearly drawn class diagram. It is followed by the sequence diagram which describes the sequence of operations that take place once the user gives the input. It contains a description of the functions that interact between the classes. The deployment diagram then explains the system as tri-nodal architecture along with their artifacts.

## 6.2 Design Constraints, Assumptions, and Dependencies

### Constraints:

- The model is designed to handle only image inputs.
- The model performs pre-processing regardless of the quality of the input image. For example, even if the image has perfect lighting, no occlusions, and is unblurred, image resizing, and CLAHE techniques must be done.
- Constraints on response time due to Deep Learning-based convolutional neural network.
- Unsuitable Accuracy for large scale implementation.
- A fair prediction would not be possible in presence of extensive usage cosmetics

### Assumptions:

- The person to be recognized is in the visible spectrum (broad daylight) and not in a dark environment where IR would be necessary.
- The pose variation angles of the subject in view must not be high and each periocular region of the subject must be clearly visible

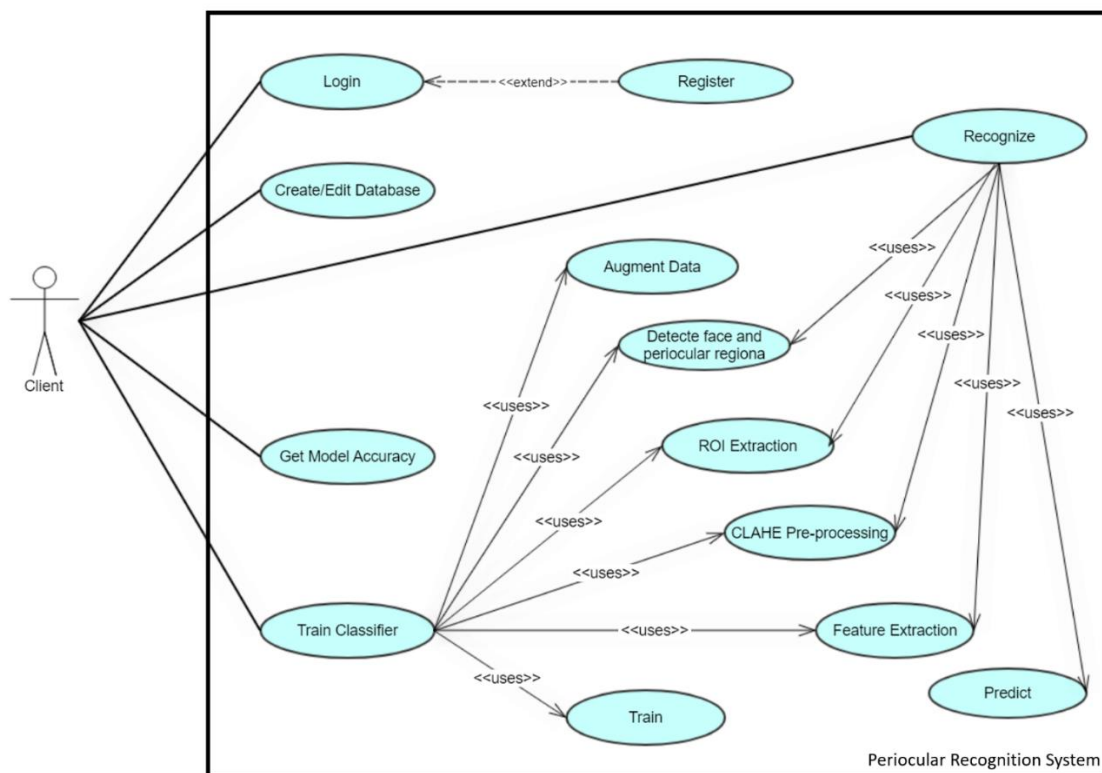
**Dependencies:**

- Languages: Python, HTML, Javascript, CSS
- Libraries: ImageDataGenerator, NumPy, dlib, Flask, TensorFlow, OpenCV, Keras
- CNN model with VGG16 architecture and OpenCV's Caffe model DNN

## 6.3 Design Description

### 6.3.1 Use Case Diagram

The figure 6.3.1.1 shows the use case diagram for the periocular recognition system. It represents all the dependencies the recognition function makes use of.



*Fig 6.3.1.1: Use case diagram for the system*

Use Case Item	Description
Client	The prime user of the product who performs various functions.
Login	Gives access to private data and various functionality only by user authentication
Register	Allows a new client to create and register an account thus gaining the service of the product
Create/Edit Database	Helps user in making changes such as adding or removing subjects or images from the database.
Get Model Accuracy	Retrieves the model accuracy of the trained model and displays to the user
Train Classifier	Performs all the pre-processing steps and feature extraction and trains the two classifiers for recognition.
Recognize	Acts as an intermediary between the Client and the server. It sends the input image to the server and performs recognition.
Augment Data	Expands the training data by creating 10 variants of each image
Detect Face and Periocular Regions	Using the OpenCV Caffe model and dlib 5 facial landmarks detector model, the face region and the 5 facial landmarks within are detected which aid in ROI extraction
ROI Extraction	Using the canthus points detected as facial landmarks, the region of interest bounding box coordinates are calculated and the two periocular regions are extracted or cropped.
CLAHE Pre-processing	Implements the CLAHE technique on all images in order to enhance the features in the periocular regions.
Feature Extraction	Feature maps, one from each periocular region are extracted and flattened for training with the classifier.

Train	Both the weighted KNN Classifiers are trained with the feature maps extracted along with their true labels.
Predict	It gives a percentage output of match based on the images stored in the Database.

Table 6.3.1.1: Use case Item Description

### 6.3.2 Class Diagram

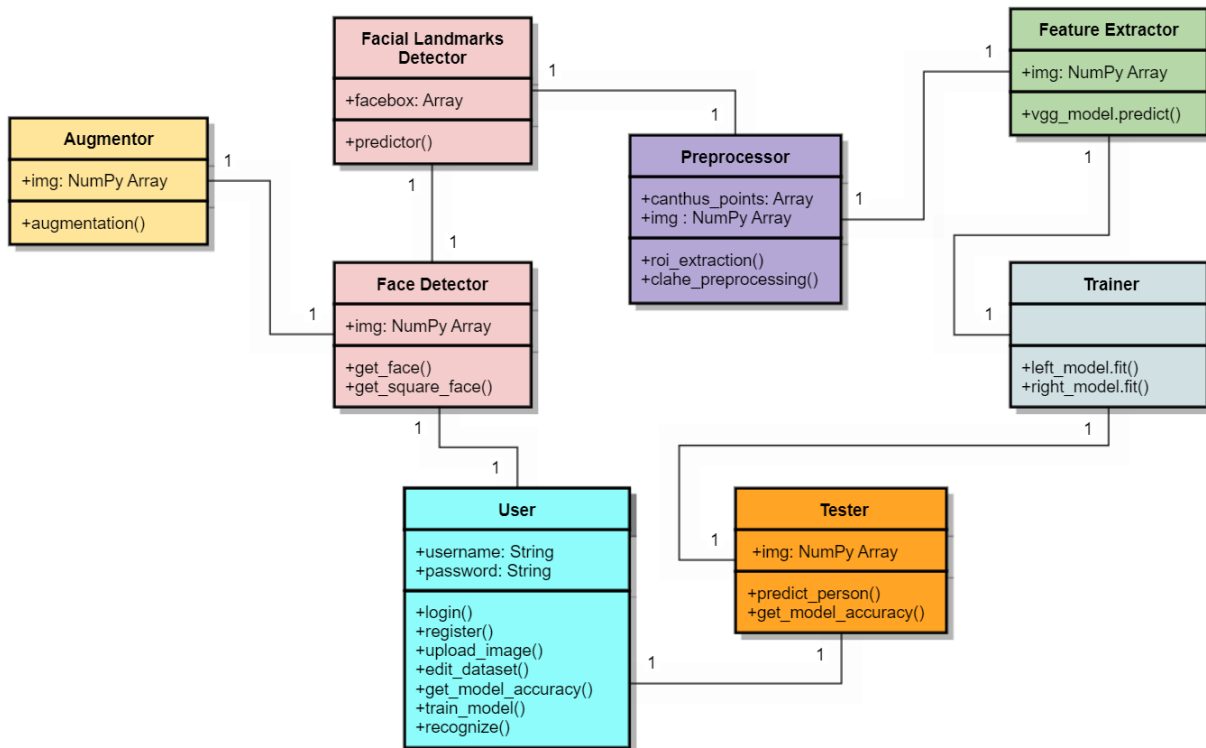


Fig: 6.3.2.1: Masterclass Diagram

#### 6.3.2.1 Class: User

After creating a profile for a user using the UI, the user can access various information like the training data, the model accuracy, among many others. The user is also provided with various

functionalities such as training the model, viewing or editing the database, and recognition. The recognition process requires the user to upload the image that he/she would want to be recognize. The uploaded image is processed in the backend and the results is returned back to the user.

### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
String	username	public	null	Stores the username for a particular client
String	password	public	null	Stores the password for the respective username

*Table 6.3.2.1.1: Class User data members*

### Methods

#### ➤ Method 1: login()

- *Purpose:* To ensure that the data privacy of the client is met, since facial data is considered private, this method enables only authenticates users to access the product.
- *Input:* The username and the password from the form are extracted and validated to authenticate the user.
- *Output:* A notification toast is shown when successfully logged in and the user can now access all data and functionality of the product.

#### ➤ Method 2: registered()

- *Purpose:* When a new client would want to register an account and obtain services from the product, the registration page is opened where the user is prompted to enter details. Then this method is called to save these details in the backend database, so that the new client can login into the account and use the product.
  - *Input:* Form values such as first name, last name, email id, password and phone number are extracted and store in the database.
  - *Output:* A notification toast is shown when successfully registered and the new user can now login into the product.
- **Method 3: loadFile()**
- *Purpose:* This method is used to load the image file that the user drags and drops or selects as the subject for recognition.
  - *Input:* An event such as a user click or file drop in the space is taken as input and triggers this method.
  - *Output:* The file is loaded and passed to the backend and the image chosen by the user is displayed on the page.
- **Method 4: get\_dataset()**
- *Purpose:* Lets the client add or remove images from the database based on their needs.
  - *Input:* The event of user click triggers this event resulting in an opening of a new window the training data folder opened.
  - *Output:* No output (the changes made are automatically saved into the database)
- **Method 5: get\_model\_accuracy()**
- *Purpose:* Calculates the percentage accuracy of the saved model and displays it to the user.
  - *Input:* Saved model in the database



- *Output:* The calculates accuracy percentage is the output of this method and is displayed to the user on the web app.

➤ **Method 6: train\_model()**

- *Purpose:* Trains the two KNN classifiers with the feature amps extracted from the entire training data passed along with their corresponding true labels.
- *Input:* Feature Maps of all the images in the training data and their corresponding true labels.
- *Output:* The two classifiers are trained and saved in the database.

➤ **Method 7: recognize()**

- *Purpose:* To recognize the user inputted image.
- *Input:* Image file uploaded by the user.
- *Output:* The input image along with the predicted label.

### 6.3.2.2 Class: Face Detector

This module assists in detecting the face using the `get_face()` function and providing a square bounding box using the `get_square_face()` function.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
NumPy Array	img	public	—	Stores the pixel values of the image

*Table 6.3.2.2.1: Class Face Detector data members*

## Methods

### ➤ Method 1: `get_face()`

- *Purpose:* To detect the face from the image and obtain the coordinates of the bounding box enclosing the face region.
- *Input:* User inputted image during the recognition phase or the training data image in the training phase
- *Output:* Top left and the bottom right coordinates of the face region in the image.

### ➤ Method 2: `get_square_face()`

- *Purpose:* To convert the bounding box obtained into a square shape
- *Input:* Coordinates of the bounding box obtained in the `get_face()` method.
- *Output:* Coordinates of the square shaped bounding box.

### 6.3.2.3 Class: Facial Landmarks Detector

It detects 5 facial landmarks in the image using the dlib library. 4 canthus points and one coordinate for the nose is detected. The nose coordinate is discarded as it is not enclosed in the periocular regions.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
Array	facebox	public	null	Stores the coordinates of the square bounding box

*Table 6.3.2.3.1: Class Facial Landmarks detector data members*

## Methods

### ➤ Method 1: predictor()

- *Purpose:* Retrieve the five coordinates of the facial landmarks detected in the face region.
- *Input:* The square shaped bounding box coordinates enclosing the face region
- *Output:* The five coordinates of the facial landmarks detected.

### 6.3.2.4 Class: Augmentor

This class performs augmentation, thus creating 10 variants of each image for each test subject.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
NumPy Array	img	public	null	Stores the pixel values of the image

*Table 6.3.2.4.1: Class Augmentor data members*

## Methods

### ➤ Method 1: augmentation()

- *Purpose:* To generate more training data from existing data for achieving better accuracy.
- *Input:* Each image from the training data in each test subject folder.
- *Output:* The 10 variants for each image are obtained and then stored in the training folder.

### 6.3.2.5 Class: Pre-Processor

The entire pre-processing for each image is performed. Pre-processing includes extracting the region of interest and enhancing the image using CLAHE algorithm.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
Array	canthus_points	public	null	Stores the coordinates of the four canthus points
NumPy Array	img	public	null	Stores the pixel values of the image

*Table 6.3.2.5.1: Class Pre-processor data members*

#### Methods

##### ➤ Method 1: roi\_extraction()

- *Purpose:* Extracting the periocular regions from the image.
- *Input:* The four canthus points along with the original image from the database during the training phase or the user inputted image during the recognition phase.
- *Output:* The two cropped images consisting of the periocular regions from the input image are obtained and stored in a separate directory in the database.

##### ➤ Method 2: clahe\_preprocessing()

- *Purpose:* Enhancing the features and edges in the periocular regions.
- *Input:* Each periocular image extracted by the roi\_extracted method.

- *Output:* Contrast enhanced image is obtained and is stored in the database.

### 6.3.2.6 Class: Feature Extractor

The class helps extract the feature vector for each contrast enhanced image in the database resulting in a feature vector of shape (1,7,7,512) which is later flattened.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
NumPy Array	img	public	null	Stores the pixel values of the image

Table 6.3.2.6.1: Class Feature Extractor data members

#### Methods

##### ➤ Method 1: vgg\_model.predict()

- *Purpose:* Retrieve the feature map of the image.
- *Input:* Contrast enhanced image from training data during te training phase or from the user inputted image during the recognition phase.
- *Output:* Feature map of the shape (1,7,7,512).

### 6.3.2.7 Class: Trainer

The class trains the classifiers with all the feature maps extracted along with their true labels which on completion is then ready for recognition.

There are no data members for this method.

## Methods

### ➤ Method 1: `left_model.fit()`

- *Purpose:* Train the classifier for left eye images.
- *Input:* An array consisting of the feature vectors of the left eye periocular regions along with their true labels.
- *Output:* No output, the model is just saved in the database.

### ➤ Method 2: `right_model.fit()`

- *Purpose:* Train the classifier for right eye images.
- *Input:* An array consisting of the feature vectors of the right eye periocular regions along with their true labels.
- *Output:* No output, the model is just saved in the database.

### 6.3.2.8 Class: Tester

Performs the recognition and fetched model accuracy when required.

#### Data members

Data Type	Data Name	Access Modifiers	Initial Value	Description
NumPy Array	img	public	null	Stores the pixel values of the image

*Table 6.3.2.8.1: Class Tester Data Members*

---

## Methods

➤ **Method 1: predict\_person()**

- *Purpose:* To recognize the inputted image.
- *Input:* The user inputted image
- *Output:* The predicted label of the predicted image.

➤ **Method 2: get\_model\_accuracy()**

- *Purpose:* Retrieve the recognition accuracy of the model.
- *Input:* The saved models in the database.
- *Output:* The recognition accuracy percentage is retrieved and displayed to the user.

### 6.3.3 Sequence Diagram

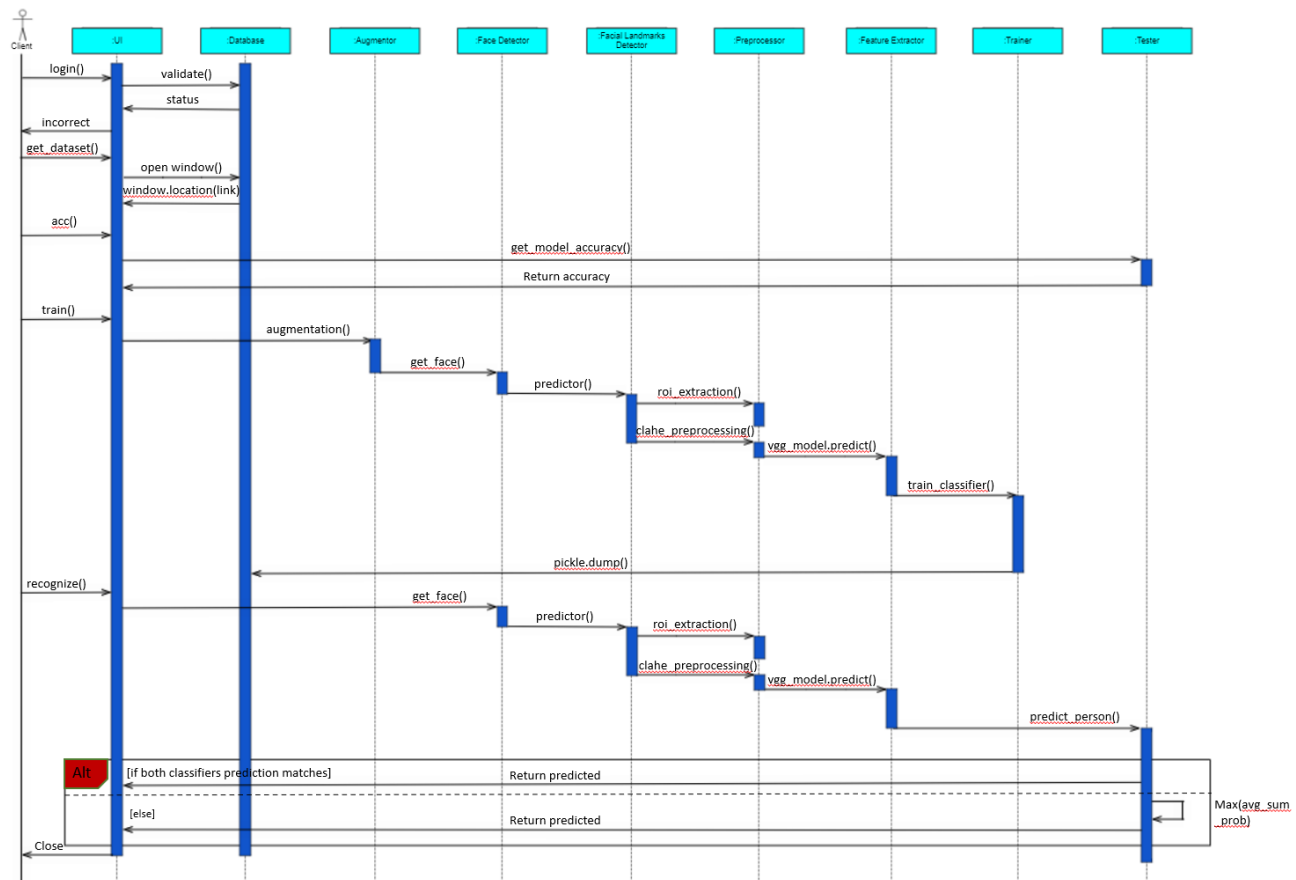


Fig 6.3.3.1: Sequence Diagram for the system

The figure 6.3.3.1 shows the sequence of classes, functions and phases during run-time.

First, the client opens the web app and enters the details to login. These details are validated using the Database. After receiving a successful validation status, the user can utilise the various functionalities of the product.

One of the functionalities that the client can access is the ability to create or alter the database. Using this feature the client can input a custom database of their choice. This is done using the `get_dataset()`



function. Once the edits or creation of a database is made, the client can then start the training phase of the model using the `train()` function which calls the `train_classifier()` in the backend.

Internally in the backend, the Augmentor is called using the `augmentation()` function. This gives 10 augmented images or variants for each of the original images in the training data.

After augmentation the `get_face()` function and the `get_square_face()` function are utilised to obtain the coordinates of the bounding box enclosing the periocular regions in the image. The dlib's 5 facial landmarks detector is called, thus retrieving the canthus points coordinates making use of the `predictor()` function.

Further, the set of images are cropped based on the calculated ROI from the canthus points in order to obtain the periocular images. This is performed by executing `roi_extraction()` function on each image.

On obtaining the periocular region from the image, for further enhancement the images are treated with the CLAHE algorithm. The `clahe_preprocessing()` function is performed on the ROI extracted images to enhance the contrast in each periocular image to highlight the features and edges.

Then `vgg_model.predict()` obtains the feature map of each periocular image which is directed to the respective classifiers with their true labels.

The `train_classifier()` function takes care of the entire training process of the two weighted K Nearest Neighbour Classifier models. The train models are finally saved to the database to avoid repetitive training procedures during recognition.

Following this series of steps, the client can then request for recognition on some inputted image using the `recognize()` function. This executes the same chain of pre-processing procedures namely, face detection, facial landmarks detection, region of interest extraction, CLAHE pre-processing and feature extraction. Following feature extraction, the feature map is sent to the tester and the recognition is done using the `predict_person()` function that calls the `model.predict()` internally.

The two predictions made by the classifiers are then matched. Succeeding in the match, the predicted person is returned to the user. On failure of match, the average sum of recognition probabilities in the left and right models is calculated for each label. The corresponding label with the highest average

sum probability is returned to the user as the final predicted output. This method is shown in the alternative section in the sequence diagram.

The user also has the option to request the accuracy of the model using the `acc()` function which then calls the `get_model_accuracy()` function in the backend that validates and retrieves the model accuracy. This model accuracy is returned to the user as a percentage value.

### 6.3.4 Deployment Diagram

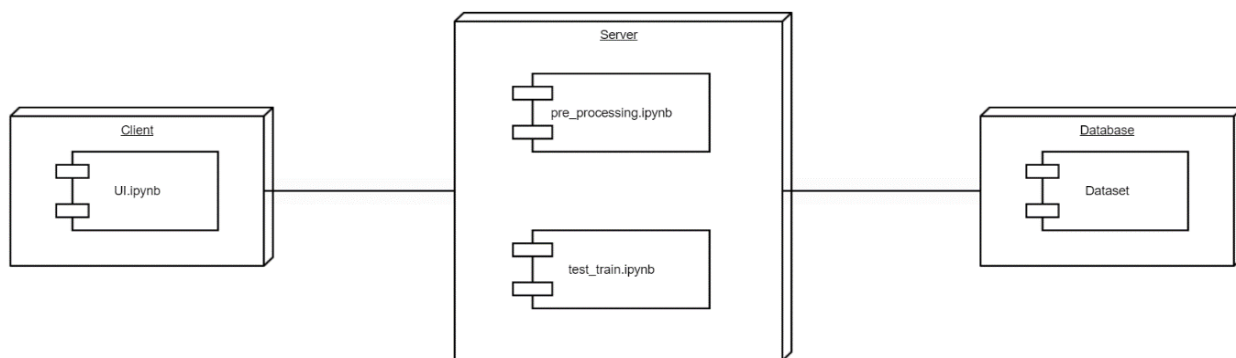
The figure shows a tri-nodal system for the Periocular recognition model.

**Client Node:** The client node contains the UI which will be visible to the user. This UI takes the input image from the user.

**Server Node:** The server node contains the integral part of the code. It has the pre-processor and the trainer to get the features from the input image and the tester to predict percentage match.

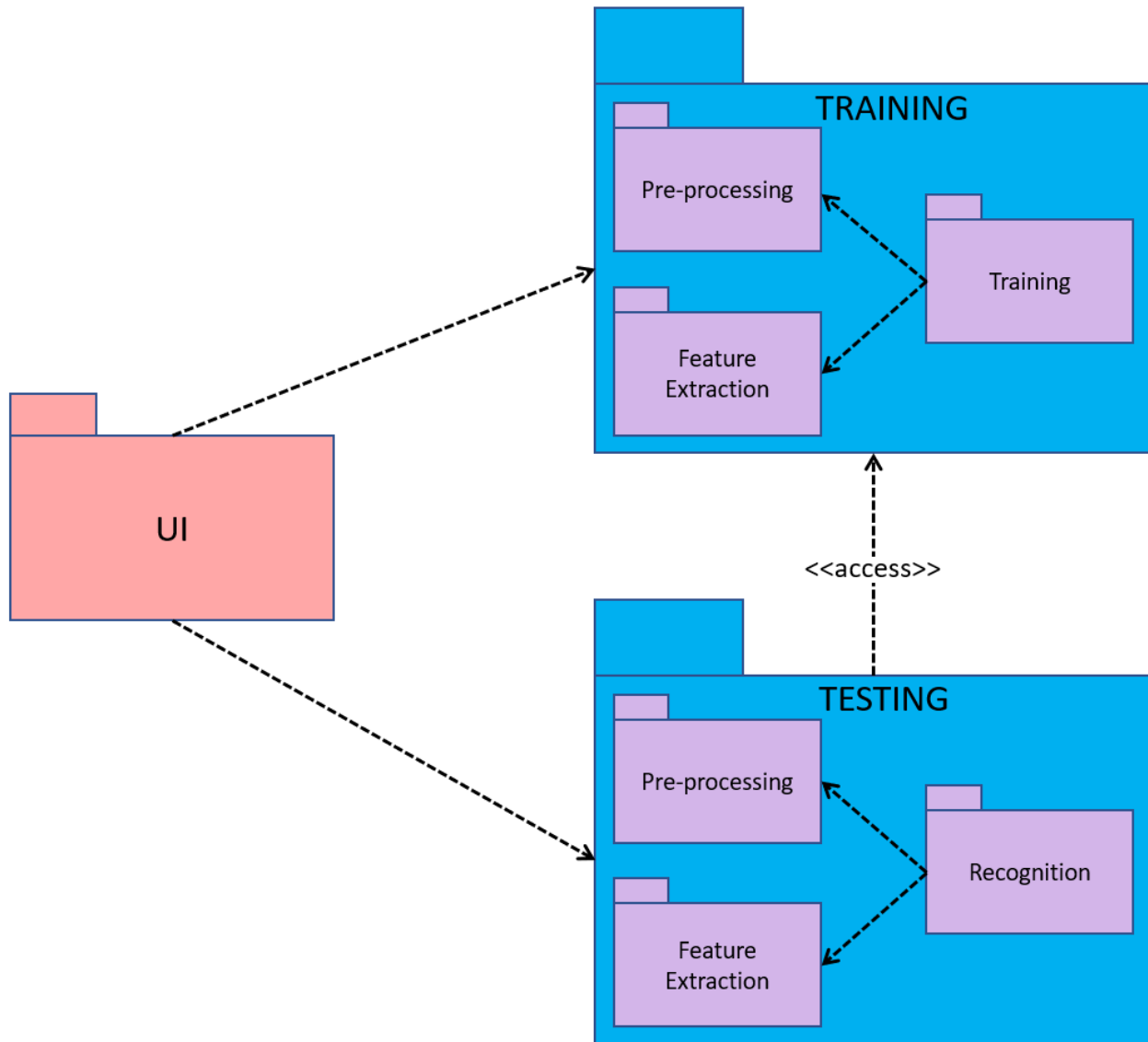
**Database Node:** Contains the dataset of images which are used for testing and calculating percentage match.

The Client and the Database nodes are dependent on the Server Node for the complete working of the system.



*Fig 6.3.4.1: Deployment Diagram*

### 6.3.5 Package Diagram



*Fig 6.3.5.1: Package Diagram*

Figure 6.3.5.1 depicts the package diagram for the periocular recognition system. This system comprises of three packages namely UI, Training and Testing as show.

**UI package:**

This package is dependent on the training phase to facilitate the user with functionalities that allow him/her to alter the database and perform training. The testing phase is one of the main dependencies since the recognition process is entirely a part of the testing package.

**Training package:**

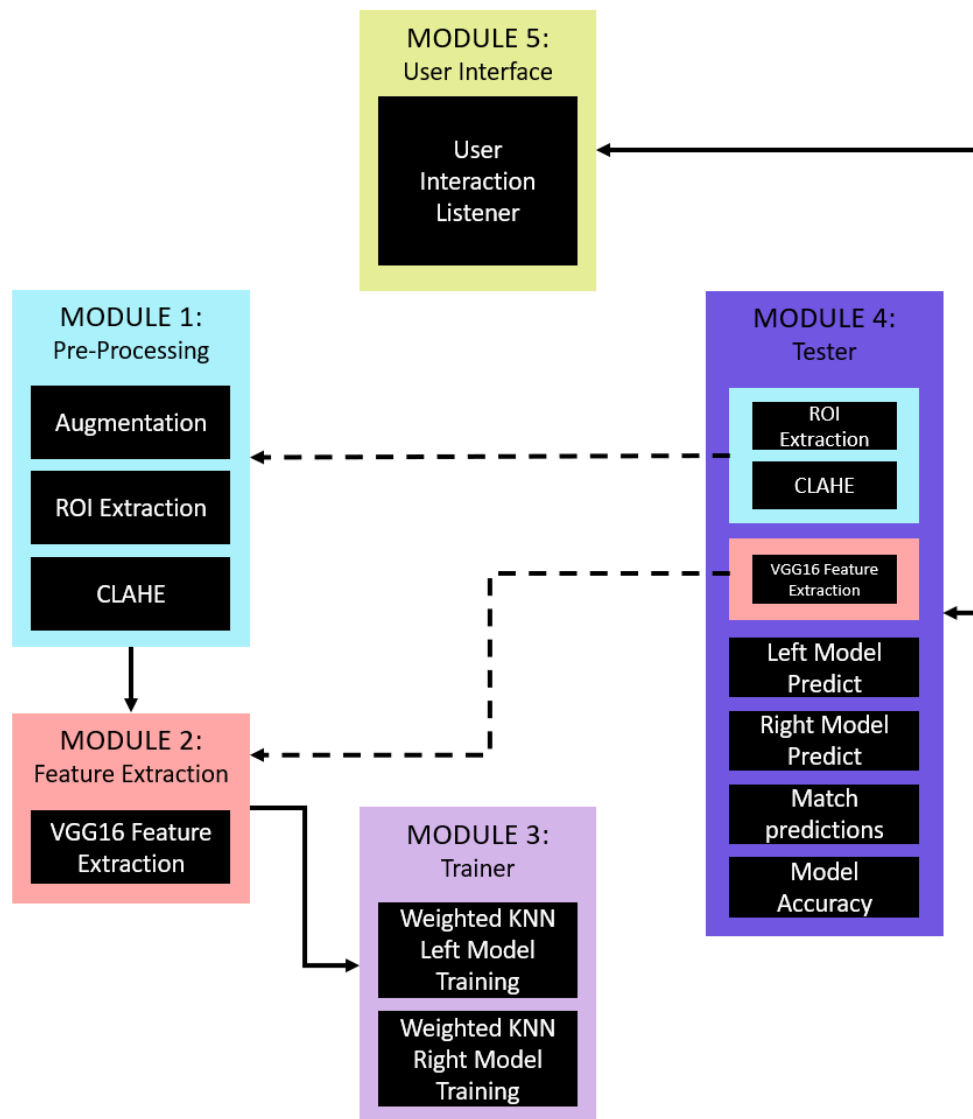
This package includes modules such as pre-processing, which has functions performing region of interest extraction, image enhancement using the CLAHE algorithm, feature extraction, and finally training. Inside this package, all of these modules are correlated since training function requires the output from both the pre-processing and feature extraction functions.

**Testing package:**

This package also includes modules namely pre-processing, which has functions performing region of interest extraction, image enhancement using the CLAHE algorithm, and feature extraction. This package accesses the pre-processing and feature extraction function from the training package. Internally, the recognize function is dependent on both pre-processing and vector extraction since effective feature maps derive using both the procedures is required for accurate recognition.

# CHAPTER 7

## SYSTEM DESIGN



*Fig 7.1: Modular System Design*

The technology employs an updated version of the facial recognition system that only considers the periocular area since it has been discovered to provide a compromise between conventional iris-based recognition and face recognition. For ease of interpretation, the concept has been rendered modular.

## 7.1 Modules

### 7.1.1 Module 1: Pre-processing

- **Augmentation:** The procedure begins with pre-processing, in which the database is supplemented in order to produce additional training data from the current data. This results in ten variants for each image of the test subject and aids in accuracy amplification for the model.
- **ROI:** For each image, augmentation is followed by face and facial landmarks detection. The coordinates of the bounding rectangle enclosing the face region is first detected and then the two periocular regions in each face region within the image are also obtained. Following that, the region of interest is extracted by cropping the periocular regions from each image, resulting in two intermediate images from each augmented image in the database.
- **CLAHE:** Contrast Limited Adaptive Histogram Equalisation (CLAHE) is then performed on each periocular region image extracted, which aids by enhancing the features and edges in the Region of Interest. This approach addresses issues with low-quality images and helps in a better accuracy due to the highlighted features and edges in the periocular region.

### 7.1.2 Module 2: Feature extraction

- **Feature extraction:** Moving on to the most important part of the method, a feature extraction step is performed using a Convolutional Neural Network, namely VGG16, that has been pre-trained on the ImageNet dataset. This CNN functions as a feature extractor, producing a feature vector with the shape (1,7,7,512) for each periocular image which is then flattened to a shape of (25088,) for consistency and suiting the training process in the used classifier.

### 7.1.3 Module 3: Training

- **Classifier:** Two weighted K Nearest Neighbours models, one for the left eye and the other for the right eye images are used in the system's classification process, with the k value set to 3. In order to train the outputted feature maps from the feature extraction module, the true labels of each periocular image are also retrieved from the database. Two arrays, one consisting of all the feature maps extracted by the VGG16 network and the other consisting of the corresponding true label of the periocular image are fed to the respective weighted KNN classifiers. It is fed in such a way that all the left periocular images with their corresponding true labels are fed to the left eye classifier KNN model and all the right periocular images with their corresponding true labels are fed to the right eye classifier KNN. After receiving the input, the two KNN's are trained by using the fit function and then stored into the database. This brings the entire training process to a stop.

### 7.1.4 Module 4: Testing

- **Recognition:** On successful completion of training on both models and exporting to the database, the user can proceed to recognize various subjects in the database. This step calls

for all the functions in the pre-processing module excluding the augmentation function, and feature extraction module in order to extract the region of interest, enhance the features and edges in the detected periocular regions of the test image by CLAHE and extract the feature maps corresponding to the two periocular regions in the test image for recognition. Following that, the derived feature map is fed into the appropriate classifier depending on the location of the periocular area, i.e., left or right. On this input feature map, the predict function in the classifiers is called, and the predicted person is outputted by both classifiers. The models' predictions are tested and matched. When a perfect match is found, the output is returned to the user. In the exception of perfect match, the resulting average sum of recognition probabilities is calculated, and the label with the highest average sum probability is returned as the output.

### 7.1.5 Module 5: User Interface

- **UI:** The whole system is integrated with a clean user-friendly User Interface that was built using HTML, CSS, JavaScript, and flask. A basic user authentication is provided in order to protect private data of each client. Each function in the model that include retrieving model accuracy, training model, recognition results, editing the database to name a few are all integrated with several buttons which are accessible to the users, thus providing full functionality and transparency.

All these modules are linked sequentially, and the process begins when the user opens the web app and either uploads an image to the UI, makes changes or edits to the database, trains the model, or tries to obtain the model accuracy. The UI responds to each user action and displays the results that could be the recognition results on user selecting to recognize, the percentage of model accuracy on user retrieving model accuracy, database display to edit the database and so on.



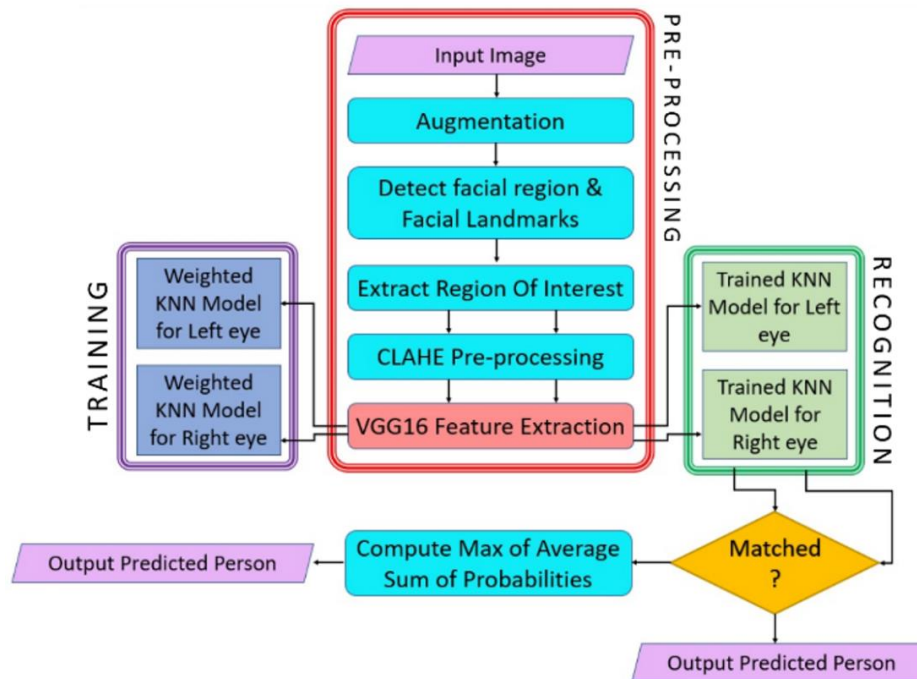
## CHAPTER 8

### PROPOSED METHODOLOGY

The proposed approach for periocular recognition involves a series of steps, namely augmentation, face identification, facial landmark detection, extraction of region of interest (ROI), image enhancement using the CLAHE algorithm, feature extraction, classification using weighted KNNs, and finally the output of the test subject prediction is displayed on a web user interface.

A custom dataset consisting of 24 test subjects was captured in the visible spectrum in a stable environment is used by the model. The test subjects were photographed in a variety of poses, both with and without occlusions. The training dataset contains test subjects without masks, while the test dataset contains the same test subject with masks. Some subjects have eye wear included as well.

The complete framework of the proposed methodology is displayed in the figure 8.1.



*Fig 8.1: Overall Procedure of the Model*

## 8.1 Augmentation

The project initiates the process with augmentation in order to generate more training data from the collected dataset. Augmentation is performed with the use of ImageDataGenerator in the TensorFlow library in python. Various image manipulations are performed such as rotation, zooming in and out, shear effect, flipping, brightness variations, height and width shifts, to name a few. It is recommended to complete this step-in order to amplify the overall efficiency of the model. Ten variants of each database image are obtained and stored in the database to undergo pre-processing followed by training.

## 8.2 Face and facial landmarks detection

Next, from each augmented image the bounding box enclosing the face needs to be retrieved in order to detect the facial landmarks within the face region.

OpenCV's Caffe Model of the Deep Neural Network (DNN) module is used to retrieve the coordinates of the bounding box enclosing the face region in each augmented image. Among the other detectors such as Haar Cascades, dlib's face detector, MTCNN detector, the Caffe model is found to work the best in cases of variation in poses and occlusions such as masks.

The output of the Caffe Model for face detection is an array consisting of the coordinates of the bounding box that enclose the face region in the image. This output is converted into a square bounding box and then to a dlib rectangle in order to be compatible with the dlib library for facial landmark detection.

Dlib's 5 facial landmark detector model is used to get the 5 facial landmarks within the face region in each image. This model takes the dlib compatible rectangle which is an array with the coordinates of

the face region and performs the landmark detection within this face region. Five coordinates are outputted by the model, four canthus points, two for each eye and a single coordinate for the nose. The coordinate for the nose is discarded for this project since the nose coordinate is not enclosed in the periocular region and that we consider only the regions around the eye.

## 8.3 Region of Interest Extraction

The four canthus points are prime for the process of region of interest extraction. For each eye, we calculate various parameters to obtain the bounding box enclosing each of the periocular regions in the image.

The algorithm followed for feature extraction is detailed below:

### 8.3.1 ROI extraction algorithm

**Input:**  $(x_1, y_1)$  are the coordinate for medial canthus points and  $(x_2, y_2)$  are the coordinate for lateral canthus points.

**Output:** Coordinated of the rectangle enclosing ROI

**Step 1:** Calculate Euclidean distance between medial and lateral canthus points

$$D((x_1, y_1)(x_2, y_2)) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

**Step 2:** Compute  $L_p = (L_{px}, L_{py})$ , midpoint of the line which connects medial and lateral canthus points.

$$L_{px} = (x_1 + x_2)/2 \quad (2)$$

$$L_{py} = (y_1 + y_2)/2 \quad (3)$$

**Step 3:** Compute the top left coordinate  $(x_3, y_3)$  and bottom right coordinate  $(x_4, y_4)$  of rectangle enclosing the desire region of interest

$$(x_3, y_3) = (L_{px} - aD, L_{py} - bD) \quad (4)$$

$$(x_4, y_4) = (L_{px} + aD, L_{py} + bD) \quad (5)$$

Here, a and b are constants whose values are chosen to cover the required periocular region.

**Step 4:** Use the calculated points  $(x_3, y_3)$  and  $(x_4, y_4)$  to extract rectangular ROI.

This procedure is performed for both pairs of the canthus points in each eye, thus this method outputs two regions enclosing the periocular region.

Using OpenCV's image manipulation techniques, both the regions are cropped thus resulted in two images from each input image.

## 8.4 Contrast Limited Histogram Equalisation (CLAHE)

Each periocular image extracted is undergone with a contrast enhancement technique known as the Contrast Limited Histogram Equalisation (CLAHE) technique. This technique aids in enhancement of the features and highlights the edges in the periocular image. It is said to be an advancement of a method called Adaptive Histogram Equalisation (AHE). CLAHE is preferred as the latter over amplifies the noise in the image as well. CLAHE on the other hand splits each input image into sub images of user defined tile size and clip limit and enhances the contrast locally. Then the neighbouring tiles are combined using bilinear interpolation methods thus highlighting the edges and addressing the over amplification of noise issues.

The periocular image is split into its respective hue, saturation, and value channels. Then the enhancement is done on the value channel to enhance contrast, and later emerged back to give the resultant image.

## 8.5 Feature extraction

The project proposes to use a convolutional neural network model for feature extraction. The model used here is a pre-trained VGG16 model that was trained on the ImageNet dataset. The classification layer of the VGG16 is discarded as we intend to obtain only the feature vectors of each contrast enhanced periocular image. The resultant images from the CLAHE procedures are inputted to the VGG16 network which extracts a 4-dimension feature vector of the shape (1,7,7,512), this later is flattened for training and testing purposes.

## 8.6 Classifier training/testing:

Two weighted KNNs are used as classifiers in this model for the left and right eye, respectively. The  $k$  value indicating the number of neighbours to consider for output prediction is set to 3. During the training phase the flattened feature maps of shape (25088,) are fed into the respective model based on the position of eye along with their true labels from the database. The KNN models are then trained with this data. The weights considered here are calculated with the standard formula  $\text{distance}^{-1}$ . The distance measure used here is the Euclidean distance. After training the model is saved in the database using a library called pickle to help fasten the process of recognition by reducing the repetitive training process for each user iteration.

During the testing or recognition phase, the flattened two feature maps for the user inputted image are outputted from the VGG16 neural network and then fed to the respective weighted KNN models. Each KNN model then predicts the output based on the weight and highest recognition probabilities. This results in two predicted outputs from each of the weighted KNNs. If the predicted or recognized person matches for both the right eye and left eye models then the output is returned to the user immediately, failing which two dictionaries are created with the labels and their corresponding recognition probabilities. An average sum of the probabilities is calculated and stored in another dictionary. The highest probability in this resultant dictionary is returned to the user as final output.

## CHAPTER 9

### IMPLEMENTATION AND PSEUDOCODE

The project is implemented entirely in a google Collab environment, mounting google drive as the database. All necessary libraries are initially imported into the working environment. Namely, OpenCV, OS, NumPy, TensorFlow, Keras, ImageDataGenerator, sklearn, pickle and flask for the UI to run.

Next the OpenCV's Caffe Model in the Deep Neural Network is loaded into the database for face detection in the image. Dlib's 5 facial landmarks detector is also loaded into the database for facial landmarks detection within the detected face region.

Each subject in the dataset with their corresponding details are stored in order to retrieve on successful recognition.

The entire model is processed in several stages as detailed along with the pseudocode below:

#### 9.1 Dataset Creation

The dataset used in the model was entirely custom made in a controlled environment consisting of 24 subjects in the visible spectrum. In order to ensure the model works with different cameras, two different cameras were used for the dataset generation, one being the iPhone 11 pro max camera shot in the portrait mode with autofocus, Auto ISO settings providing a resolution of 3042 X 4032 pixels of width and height respectively and another being a DSLR Sony ILCE-6000 shot with a 35mm focal length lens, auto focus, auto iso, fixed aperture, exposure time set to 1/60 seconds, providing a resolution of 6000 X 3376 pixels of width and height respectively i.e., high quality images.

Both cameras were made to save the images in the JPEG format. The subject was made to stand at a 3-4 feet distance from the lens for data acquisition. Various pose angles were shot for each test subject with and without occlusions in order to deliver a remarkable accuracy.

A unique point which led to creation of the dataset was due to the unavailability of clear images posed at different angles masked and unmasked for the same subject. The images captured are composed of the person looking straight into the lens and at a few small acute angles. A sample of the dataset with and without the mask of a test subject is displayed below.



*Fig 9.1.1: Unmasked Images in the Database*





*Fig 9.1.2: Masked Images in the Database*

The entire dataset is stored in google drive. All the unmasked images are stored into the 24 respective files, named with the subject's name which is entirely stored in the folder called "trainset". All the unmasked images are stored in one folder called the "testset" without any labels.

In case of obtaining accuracy of the model, the same unmasked images are also stored with their respective labels in 24 files which are stored in a folder called "valset".



## 9.2 Augmentation

Augmentation is performed in order to generate more data from the existing data. This procedure is done to generate more training data to amplify the overall efficiency of the model. Dataset is expanded by performing various image manipulations such as rotation, width and height shifts, shear, flipping, brightness variations, zooming, to name a few. Tensorflow's ImageDataGenerator is used to generate more training data using all the manipulations mentioned.

*Pseudocode:*

```
1 augmentor = ImageDataGenerator (  
2     rotation_range=20  
3     width_shift_range=0.2  
4     height_shift_range=0.2  
5     shear_range=0.15  
6     vertical_flip=True  
7     brightness_range=[0.5,1.5]  
8     zoom_range = 0.3  
9     fill_mode="nearest")
```

This pseudocode shows the instantiation of the augmentor that aids in dataset expansion.

Line 2-9 indicate the values of various attributes. It is seen that the rotation range has been defined to 20 to make sure the rotation angle is not too high to ensure effective detection of facial landmarks. The width shift and height shift range are set to 0.2 with a shear range of 0.15 and zoom range of 0.3. Vertical flip is set to true in order to include the generation of mirror images as well. The brightness range is 0.5 to 1.5 to change the brightness conditions to ensure model working in various lighting conditions within the visible spectrum and “fill\_mode” is set to nearest so that the nearest pixels are taken during height and width shift or while zooming.

*Pseudocode:*

```
1  function augmentation(training_directory):
2      for each test_subject_folder in the training_directory:
3          for each image in test_subject_folder:
4              img_path= get path of image
5              to_augment = load image
6              total_images = 0
7              generator = generate_images(to_augment)
8              for to_augment in generator:
9                  total_images += 1
10                 if total_images == 10:
11                     break
```

After instantiation when the user called the augmentation function, the whole augmentation procedure is done which is detailed above.

Line 2-3 State that each image in each test subject folder image is read from the training dataset. This is done using OpenCV which consists of the custom captured images of the test subjects.

Line 4-11 Depicts that each of this image read is loaded into the generator that creates the various variants. In line 8, a loop is created such that the image is passed and checked in the generator. As defined the image is iterated 10 times, each time creating a variant. When total\_images exceed 10 the loop is exited.

The input and observed output for this data expansion procedure are shown in figure 9.2.1 and 9.2.2 respectively.



*Fig 9.2.1: Input for Augmentation*

This input fed to the augmentation() function produces the following results.



*Fig 9.2.2: Output for Augmentation*

This procedure is done for each image in the training data thus increasing it 10-fold.

### 9.3 Face Detection

The four canthus points in the face play a prime role in region of interest extraction. These facial landmarks can be detected only after face detection. So, we initiate this process by attempting to find the face region in each image. The input for this method is taken from the output of the augmentation phase, i.e., the original and augmented images in the training data directory.

OpenCV's Caffe model from the Deep Neural Network module is used for face detection purposes. It is found that it works fairly well in the presence of face occlusions and pose angle variations as well.

*Pseudocode:*

```
1  function get_face(image, model):  
2      model.detect_face(image)  
3      face = get coordinates of the face from model  
4  return face  
5  
6  function get_square_face(image, face):  
7      facebox = convert face coordinates to a perfect square  
8  return facebox
```

In Line 1 we have a function called `get_face` which can be called to obtain an array consisting the top left and bottom right coordinates of the rectangle enclosing the face region in the detected image.

In Line 2 and 3, the face is detected from the image and the coordinates are retrieved and stores in the variable "face".

This array consisting of these coordinates is the output of this function and thus returned in line 4.

Next, in order for this to be used for further facial landmarks detection, this bounding box of the face must be converted into a square which is done by the `get_square_function` defined in line 6.

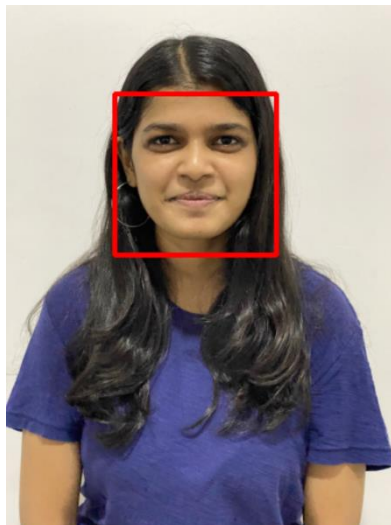
Line 7 consists of the facebox which consists of the squares shaped bounding box coordinates enclosing the face which is then returned as shown in line 8.

An example of the input and output is depicted in figures 9.3.1 and 9.3.2 respectively.



*Fig 9.3.1: Input for face detection*

Images like this are passed to the `get_face()` and `get_square_face` function, which results in a bounding box enclosing the face region shown in figure 9.3.2.



*Fig 9.3.2: Output for face detection*

## 9.4 Facial Landmarks Detection and region of interest extraction

On obtaining the bounding box of the face region, we then use dlib's 5 facial landmarks detection model in order to retrieve the 5 facial landmarks. Four canthus points, 2 for each eye and the point for the nose are detected by this model. Since we do not need the coordinate of the nose as it is not enclosed in the periocular region, we do not process that coordinate for our region of interest extraction.

After obtaining the four coordinates of the canthus points in each image, two bounding boxes are calculated enclosing the two periocular regions in the image.

For each pair of medial and lateral canthus points in the image, we calculate the Euclidean distance between them and calculate the midpoint joining the two points.

Then the top left coordinate and right left coordinate are calculated by the formula stated.

$$(x_l, y_l) = (Midpoint_x - a \times Euclidean\ distance, Midpoint_y - b \times Euclidean\ distance)$$

Where a and b are values chosen based on the extent of area of periocular region required for that model.

Here we have set the values of a and b as (1.25, 0.95).

*Pseudocode:*

```
1 function roi_extraction(directory):
2     for each image in the test subject folder in the training data:
3         img = read the image
4         face = get_face(img, model)
5         facebox = get_square_face(img, face)
6         landmarks = predict the facial landmarks
7         D_left = get Euclidean distance between inner and outer canthus point of left eye
8         D_right = get Euclidean distance between inner and outer canthus point of right eye
9         Midpoint_left = get mid point between inner and outer canthus point of left eye
10        Midpoint_right = get mid point between inner and outer canthus point of right eye
11        topleft_l = Midpoint_left[0] - a * D_left, Midpoint_left[1] - b * D_left
12        bottomright_l = Midpoint_left[0] + a * D_left, Midpoint_left[1] + b * D_left
13        topleft_r = Midpoint_right[0] - a * D_right, Midpoint_right[1] - b * D_right
14        bottomright_r = Midpoint_right[0] + a * D_right, Midpoint_right[1] + b * D_right
15        crop the periocular images
16        save the images
```

In Line 1, the `roi_extraction` function takes the input of the augmented dataset directory which is the and results in creation of another directory called “roiset” which consists of the cropped images of each subject in their respective right or left folders based on the position of eye which is present in the respective label folder.

As shown in lines 3-5, each image is read from the directory and the `get_face` function is executed on this to obtain the face region coordinates which is converted to a square shaped bounding box using the `get_square_face` function.

In Line 6, the `dlib` model is called on this face box to obtain the canthus points from the face region. The 5 facial landmarks are retrieved by this step.

Lines 7-14 depict the calculation required in order to obtain the coordinates of the bounding box enclosing the two periocular regions in the image. The Euclidean distance is first found between the inner and outer canthus of both eye in lines 7 and 8. Then the midpoint of the line joining the inner and outer canthus of each eye is computed in lines 9 and 10. Finally, the bounding box enclosing the periocular regions are found in order to extract the region of interest using a simple formula as shown in lines 11-14.

Line 15, 16 shows that the retrieved coordinates are used to crop the two periocular regions in the image which is done by OpenCV and then saved into the “roiset” directory.

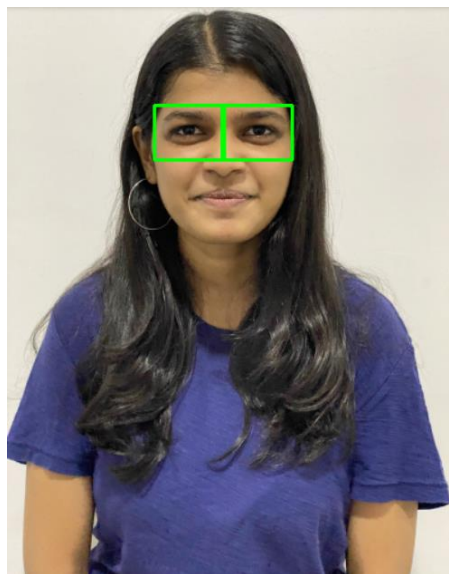


For the detection of the face landmarks, the coordinates of the bounding box enclosing the face are sent as input, which then results in an output array consisting of the facial landmarks. The visual depiction of the canthus points extracted is shown below. The 5<sup>th</sup> coordinate for the nose is discarded.



*Fig 9.4.1: Output for facial landmarks detection*

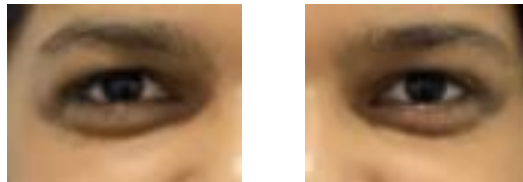
Next using these canthus points the region of interest bounding box coordinates is found. The figure below shows the bounding boxes of the two periocular regions in the image.



*Fig 9.4.2: Intermediate output for ROI*



Finally, the periocular images are extracted, cropped and saved as shown in figure 9.4.3.



*Fig 9.4.3: ROI Extraction Output*

## 9.5 Contrast Limited Adaptive Histogram Equalisation (CLAHE)

After the region of interest extraction, each periocular image must be enhanced with respect to contrast to highlight the edges and features in the periocular region.

*Pseudocode:*

```
1  function clahe_preprocessing(directory):
2      for each test_subject in the training data:
3          clahe_eye(person_dir_left, 0, person)
4          clahe_eye(person_dir_right, 1, person)
5
6  function clahe_eye(directory, left_right, person):
7      for each image in test_subject:
8          read the image
9          convert bgr image into hsv
10         split the hsv into its respective h,s,v channels
11         clahe = createCLAHE(cliplimit, tileGridSize)
12         apply clahe on the value channel
13         merge the h,s,v channels back to hsv
14         convert the image back to bgr
15         save the images
```

In Lines 1-4, we have a function `clahe_preprocessing` that calls the `clahe_eye` to perform the CLAHE procedure on each image within each test subject folder in the “roiset” directory.

In line 6, the `clahe_eye` function is defined which performs the actual CLAHE procedure.

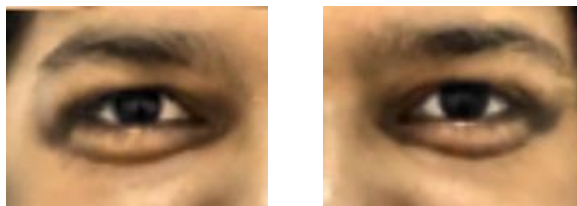
Lines 7-14 depict the actual procedure where, each image is read, converted into the HSV colour scale from BGR, then the channels are split. The OpenCV CLAHE function is define in line 11 with

a certain tile grid size and clip limit. This method is called on the value channel of the image as seen in line 12, following which the channels are merged.

The two contrast enhanced periocular images are saved into a directory called “claheSet” as mentioned in line 15.

The input for this method is are the pictures from the output of the ROI extraction as shown in figure 9.4.3.

After the CLAHE procedure the resulting images are saved into the database as displayed in figure 9.5.1.



*Fig 9.5.1: CLAHE Output*

## 9.6 Feature Extraction and Training

Succeeding the pre-processing procedure, a VGG16 model that is pre-trained on the ImageNet dataset is used as a feature extractor and two weighted K Nearest Neighbours models are used as classifiers.

*Pseudocode:*

```
1  function train_classifier():
2      for each test_subject in the clahe directory:
3          for each image in the test_subject folder:
4              load image
5              preprocess the image
6              feature_vector = vggmodel.predict(preprocessed image)
7              flatten feature vector
8              left_model.fit(all_features_vectors, labels)
9              right_model.fit(all_features_vectors, labels)
10
11 function create_model()
12     perform ROI extraction
13     perform CLAHE pre-processing
14     train_classifier()
15     save left and right classifiers in the database
```

In Line 1, the `train_classifier()` function is defined which trains the two weighted K Nearest Neighbours Classifiers.

Each image is loaded and preprocessed from the `test_subject` folder in the CLAHE directory as mentioned in lines 2-5.

As stated in Lines 6-9, the feature vector is extracted using the “`vgg_model`” which is then flattened for compatibility with KNNs. Both the classifiers are then trained using the `model.fit` function.

Later in lines 11-15, it is shown how to create a model from the start which perform ROI extraction, CLAHE Preprocessing, classifiers training and then saves the two classifiers into the database.

The input for this method is shown in figure 9.5.1.

The output is a feature map which is trained in the two KNN classifiers.

## 9.7 Recognition

The main functionality of the project is to perform recognition. This step is carried out only after the model is created successfully and stored in the database. Here the User inputs an image which is processed and finally the recognition results are returned back to the user.

*Pseudocode:*

```
1 function predict_person(image path):
2     load the left eye classifier
3     load the right eye classifier
4     read image
5     perform roi extraction on image
6     perform clahe pre-processing on image)
7     perform feature extraction
8     prediction1 = loaded_left_model.predict(feature map of left periocular image)
9     prediction2 = loaded_right_model.predict(feature map of right periocular image)
10    if prediction1 == prediction 2:
11        return prediction 1 to the user
12    else:
13        avg_prob = calculate average sum of recognition probabilities for each label
14        predicted = corresponding label of max(avg_prob)
15        return predicted to the user
```

Line 1 defines the predict\_person function that helps recognize the person in the user input image.

Line 2,3 help in loading the saved and trained models from the database.

Then the image is read, ROI extraction and CLAHE pre-processing are performed following by the two feature maps extraction as detailed in the lines 4-7.

Then in Lines 8,9 the prediction is done by each classifier which results in the name of the test subject recognized.

If both classifiers predict the same test subject, then the output is immediately returned to the user as states in lines 10,11.

The unlikely condition is states in line 12-15 where the predictions made by the left and right classifier do not match. Thus, the average sum of recognition probabilities is found and the corresponding label of the highest of the averages calculates is returned to the user as the recognition result.

For this method the user may upload any image as shown in the figure 9.7.1, and the recognition results along with the uploaded picture are displayed in the UI.



*Fig 9.7.1: User Input Image*

## 9.8 Model Accuracy:

A function is created in the project just to retrieve the accuracy of the trained model to notify the user as to how well the model can recognize subjects.

*Pseudocode:*

```
1 function get_model_accuracy():
2   true_labels=[]
3   predicted_labels=[]
4   for each image in the training data directory:
5     true_labels.append(name of person)
6     predicted_person = predict_person(image_path)
7     pred_labels.append(predicted_person)
8     accuracy = calculate accuracy_score(true_labels, pred_labels) * 100
9   return accuracy to the user
```

The `get_model_accuracy()` function is define in line 1, which helps retrieve the model accuracy.

An empty array is initialised for the true labels and predicted labels by the model in line 2,3.

Line 4-9 detail the procedure of retrieving accuracy. It is shown that each image in the training data directory must be read and the true label must be stored in the true\_labels array initialised, then the prediction must be performed on that image and the predicted label must be stores in the predicted\_labels array. Then the accuracy function is performed on these two arrays which returns the accuracy percentage to the user.

## CHAPTER 10

### RESULTS AND DISCUSSIONS

The experiment stipulates that the canthus points are essential components for periocular identification, since using the iris would reduce matching accuracy. Iris based regions of interest extraction are found to fail with partially closed eyes of the subject. As compared to other approaches in the literature, the DNN model is capable of substantially neutralising the effect of pose variance and achieving a reasonably good match. Thus, the Caffe Model of the DNN (Deep Neural Network) module is found to be the best in face detection in the presence of numerous pose angles and occlusions. The region of interest extraction process included the selection of two variables based on the necessity of the area of the periocular region. These values were found on the trial-and-error basis and by constant tweaking with every advancement in the process to understand the extent of area required for accurate and effective recognition. This experiment laid out the required values of these a and b constants and they were set to (1.25,0.95) respectively. The CLAHE (Contrast Limited Adaptive Histogram Equalization) methods are found to be extremely useful in order to highlight the features and edges in the periocular region. It is observed that there is a substantial drop in accuracy when this method is not performed.

Next, VGG16 pretrained on the ImageNet dataset is found to be of extreme help as the feature extractor. The project was performed using a K Nearest Neighbours classifier initially which did give a considerably good accuracy percentage, but it was found that the accuracy could be amplified with the use of a weighted KNN in this project. The k value was set to 3 in this case. In order to find this result, various iterations of tweaking the k values was done and it was found that the value of 3 gives the best result. Further, after prediction of both the weighted KNN models, a match was made to check if both models predicted the same label. Initially the result was shown as required to the user only on successful matches. But this also did not deliver a great accuracy. Thus, this method was

enhanced by performing an average of the sum of corresponding recognition probabilities from each model. The highest average sum probability was taken and its corresponding label was returned to the user. This method boosted the accuracy by over 20%. In the overall function of the project, a method was described to retrieve the model accuracy.

On performing the overall proposed method, it was observed that an overall accuracy level of 90.01% was obtained for the 24 test subjects.



# CHAPTER 11

## CONCLUSIONS AND FUTURE WORK

Facial recognition is one of the most widely used applications today, however, the currently existing facial recognition models cannot handle cases like changes in expression, aging, occlusions etc. So, from the results obtained it is possible to observe that this study proposes a reliable periocular region-based biometric authentication scheme to persist during the COVID-19 pandemic.

The proposed system uses a VGG16 model along with two weighted KNN classifiers for image matching. The model also performs well in non-optimal situations, such as subjects wearing spectacles (which can cover up to a certain periocular region), cloaking of the eye region (effect of partial/full closure of eyes), and pose variance (images with rotated head on a plane perpendicular to the centre of gravity).[41]. The experimental results provide very strong support to the proposed VGG16 model along with weighted KNNs that are precise of its kind where the design is best suited in the area of periocular biometrics/recognition. This above proposed methodology was decided to be used for developing this problem because it will provide the most optimized way of taking into consideration a variety of distinct parameters like space, time, GPU etc. Also, the tremendous literature survey that has been done for all the neural networks along with various other algorithms for periocular recognition. The neural network used to be pre-trained on ImageNet dataset and also a custom dataset was built that would be best suited for the project by collecting high resolution images from 24 distant test subjects. The dataset includes numerous images per person in which some of the images are associated with masks for validation and testing purposes, while the rest were for training. By the end of the project a neatly displayed user interface is created which is consigned with input images that are then sent to the server where it involves a series of phases, starting with augmentation followed by extracting the region of interest (ROI) then by using Contrast-Limited Adaptive Histogram Equalization (CLAHE) where the contrast, saturation and hue of the images are

being enhanced accordingly to compensate variability in local illumination. Whilst the process is further continued by extracting the feature with the help of neural networks and then accompanied by a classifier to label the images which are then compared with the existing labels in the database and matched. The end result of the match is then returned to the user via the user interface.

The proposed approach's output will be improved in the future by providing it with multispectral details for periocular recognition. It is also intended to study the impact of cosmetics on texture of the periocular region and the resulting effect on recognition accuracy. Other areas to explore are the cases of blurred or distorted images using restoration algorithms and usage of Recurrent Neural Networks model for exploiting dynamic information.

## REFERENCES

- [1] M. C. Kim, J. H. Koo, S. W. Cho, N. R. Baek and K. R. Park, "Convolutional Neural Network-Based Periocular Recognition in Surveillance Environments," in *IEEE Access*, vol. 6, pp. 57291-57310, 2018, doi: 10.1109/ACCESS.2018.2874056.
- [2] Y. Wong, S. Chen, S. Mau, C. Sanderson, and B. C. Lovell, "Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Colorado Springs, CO, USA, Jun. 2011, pp. 74–81.
- [3] ChokePoint Database. Accessed: Feb. 21, 2018. [Online]. Available: <http://arma.sourceforge.net/chokepoint/>
- [4] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 1867–1874.
- [5] B. J. Kang and K. R. Park, "A robust eyelash detection based on iris focus assessment," *Pattern Recognit. Lett.*, vol. 28, no. 3, pp. 1630–1639, 2007.
- [6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1097–1105.
- [7] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014.

- [8] D. Usher, Y. Tosa, and M. Friedman, "Ocular biometrics: simultaneous capture and analysis of the retina and iris," *Advances in Biometrics: Sensors, Algorithms and Systems*, Springer Publishers, pp. 133-155, 2008.
- [9] R. Derakhshani and A. Ross, "A texture-based neural network classifier for biometric identification using ocular surface vasculature," *Proc. of International Joint Conference on Neural Networks (IJCNN)*, pp.2982-2987, 2007.
- [10] U. Park, R. R. Jillela, A. Ross and A. K. Jain, "Periocular Biometrics in the Visible Spectrum," in *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 1, pp. 96-106, March 2011, doi: 10.1109/TIFS.2010.2096810.
- [11] J. Matey, D. Ackerman, J. Bergen, and M. Tinker, "Iris recognition in less constrained environments," *Advances in Biometrics: Sensors, Algorithms and Systems*, Springer Publishers, 2008.
- [12] S. Crihalmeanu, A. Ross and R. Derakhshani, "Enhancement and registration schemes for matching conjunctival vasculature," *Proc. of the 3rd IAPR/IEEE International Conference on Biometrics (ICB)*, pp.1240-1249, 2009.
- [13] C. Boyce, A. Ross, M. Monaco, L. Hornak, and X. Li, "Multispectral iris analysis: a preliminary study," *Proc. of IEEE Computer Society Workshop on Biometrics at CVPR*, pp. 51-59, 2006.
- [14] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," 22(12), pp. 1349-1380, 2000.
- [15] C. Schmid and R. Mohr, "Local Grayvalue Invariants for Image Retrieval," *IEEE TPAMI*, 19(5), pp. 530-535, 1997.

- 
- [16] Krystian Mikolajczyk and Cordelia Schmid, “A performance evaluation of local descriptors,” *IEEE TPAMI*, 27(10), pp. 1615-1630, 2005.
  - [17] F. Alonso-Fernandez and J. Bigun, “A survey on periocular biometrics research,” *Pattern Recognition Letters*, vol. 82, pp. 92–105, 2016.
  - [18] J. M. Smereka, V. N. Boddeti, and B. V. K. V. Kumar, “Probabilistic deformation models for challenging periocular image verification,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp.1875–1890, Sept 2015.
  - [19] F. Juefei-Xu, K. Luu, M. Savvides, T. Bui, and C. Suen, “Investigating age invariant face recognition based on periocular biometrics,” *Proc Intl Joint Conf Biometrics, IJCB*, Oct 2011.
  - [20] R. Jillella and A. Ross, “Mitigating effects of plastic surgery: Fusing face and ocular biometrics,” *Proc Intl Conf Biometrics: Theory, Applications and Systems, BTAS*, pp. 402–411, Sep 2012.
  - [21] G. Mahalingam, K. Ricanek, and A. Albert, “Investigating the periocular-based face recognition across gender transformation,” *IEEE Trans Information Forensics and Security*, vol.9, no. 12, pp. 2180–2192, Dec 2014.
  - [22] O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Deep face recognition,” *Proc British Machine Vision Conference, BMVC*, 2015.
  - [23] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, “A convolutional neural network cascade for face detection,” in *Proc Intl Conf on Computer Vision and Pattern Recognition, CVPR*, June 2015, pp. 5325–5334.
  - [24] K. Nguyen, C. Fookes, A. Ross, and S. Sridharan, “Iris recognition with off-the-shelf CNN features: A deep learning perspective,” *IEEE Access*, vol. 6, pp. 18 848–18 855, 2018.
  - [25] B. Bhanu and A. Kumar, Eds., *Deep Learning for Biometrics*. Springer, 2017.

- [26] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: An astounding baseline for recognition,” in Proc IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW, June 2014, pp. 512–519.
- [27] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, 2002.
- [28] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” Proc Intl Conf Computer Vision and Pattern Recogn, CVPR, 2005.
- [29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems 25, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>
- [30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in Proc IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 2015, pp. 1–9.
- [31] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 2016, pp. 770–778.
- [32] Hernandez-Diaz, Kevin & Alonso-Fernandez, Fernando & Bigun, Josef. (2018). Periocular Recognition Using CNN Features Off-the-Shelf.
- [33] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” CoRR, vol. abs/1409.1556, 2014.
- [34] U. Park, R. R. Jillela, A. Ross, and A. K. Jain, “Periocular biometrics in the visible spectrum,” IEEE Trans Information Forensics and Security,

vol. 6, no. 1, pp. 96–106, 2011.

[35] C. Padole and H. Proenca, “Periocular recognition: Analysis of performance degradation factors,” Proc Intl Conf Biometrics, ICB, pp. 439– 445, Mar 2012.

[36] K. Zuiderveld, “Graphics gems iv,” pp. 474–485, 1994. [Online].

Available: <http://dl.acm.org/citation.cfm?id=180895.180940>

[37] F. Alonso-Fernandez and J. Bigun, "Periocular biometrics: databases, algorithms and directions," 2016 4th International Conference on Biometrics and Forensics (IWBF), 2016, pp. 1-6, doi: 10.1109/IWBF.2016.7449688.

[38] P. Viola, M. J. Jones. Robust Real-Time Face Detection. IJCV, 57(2), 2004.

[39] R. Jillela et al. Handbook of Iris Recognition, ch. Iris Segmentation for Challenging

[40] Vol.:(0123456789)

Journal of Ambient Intelligence and Humanized Computing

<https://doi.org/10.1007/s12652-020-02814-1> “A novel periocular biometrics solution for authentication during Covid-19 pandemic situation”

[41] Yamini Cherukuri, Rohit Motamarri, Avi Bansal, Ganesha KS, “Periocular Recognition in the Visible Spectrum”, 2021.

## Appendix A: Definitions, Acronyms, and Abbreviations

- CNN = Convolutional Neural Network

In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analysing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.

- UI = User Interface

A communication medium between the user and the backend which aids the user with easy-to-use visual displays.

- ROI = Region of interest

The area or region where the entire focus is laid on and deemed to be necessary for a model.

- KNN = K Nearest Neighbours

A machine learning algorithm that may classify various data considering distance of the nearest neighbours as the main criterion for the classification.

- CLAHE = Contrast Limited Adaptive Histogram Equalisation

Image contrast enhancing technique that helps highlight the features and edges in an image locally without over amplifying the noise in the image.