

# **MAR IVANIOS COLLEGE (AUTONOMOUS)**

**Mar Ivanios Vidya Nagar, Nalanchira**  
**Thiruvananthapuram-695015**

**B.Sc. Computer Science Major Project (Phase II) Report**

## **FORENSIC FACE-PHOTO SKETCH RECOGNITION SOFTWARE USING DEEP CNN**

Submitted in partial fulfillment of the requirement for the Fifth Semester

B.Sc. Computer Science (Career Related)



### **SUBMITTED BY**

<b>NITHIN A L</b>	<b>-</b>	<b>(Regno:2190625)</b>
<b>SOORAJ J S</b>		<b>(Regno:2190631)</b>
<b>RIDHIK T T</b>		<b>(Regno:2190628)</b>
<b>BELWIN JOE ABRAHAM</b>		<b>(Regno:2190619)</b>
<b>MEGHA RAJENDRAN</b>		<b>(Regno:2190603)</b>

*Under the guidance of*

**Mr. Vinodh M R**

**DEPARTMENT OF COMPUTER SCIENCE**  
**B.Sc Computer Science**  
**April 2022**

# **MAR IVANIOS COLLEGE (AUTONOMOUS)**

**Mar Ivanios Vidya Nagar, Nalanchira ,Thiruvananthapuram-695015**

## **DEPARTMENT OF COMPUTER SCIENCE**



### **CERTIFICATE**

This is to certify that the project entitled “**FORENSIC FACE-PHOTO SKETCH RECOGNITION SOFTWARE USING DEEP CNN**” is a bonafide record of the work done by **NITHIN A L(Reg no: 2190625), SOORAJ J S (Reg no: 2190631), RIDHIK T T(Reg no.:2190628),BELWIN JOE ABRAHAM(Reg no: 2190619), MEGHA RAJENDRAN(Reg no:2190603)** in partial fulfillment of the requirements for the award of Bachelor of Science Degree in Computer Science by the University of Kerala.

**INTERNAL GUIDE**

**HEAD OF THE DEPARTMENT**

**EXTERNAL EXAMINERS**

**1.**

**2.**

# **ACKNOWLEDGEMENT**

First of all We thank God Almighty for keeping us safe and healthy in order to successfully complete our project work.

We express our sincere thanks and a deep sense of gratitude to our Principal **Dr. JIJIMON K. THOMAS** and our Director (in-charge) **Dr. K L ANANDAVALLY** for providing all necessary facilities.

We wish to express our sincere gratitude and respect to **Ms. TINU C PHILIP**, Head, Department of Computer Science, for all the words of inspiration.

We have great pleasure to express our deep sense of gratitude and obligation to our project guide **Mr. VINODH M R** for his valuable guidance and suggestions throughout the entire project work.

Last but certainly not the least, we would also like to thank all the faculty of the Department of Computer Science for their help and support.

**NITHIN A L**

**SOORAJ J S**

**RIDHIK T T**

**BELWIN JOE ABRAHAM**

**MEGHA RAJENDRAN**

## **ABSTRACT**

Forensic is the method of using scientific tests to find out about a crime. In Forensics face sketches play a vital role to identify the face of the culprit. The existing technology uses face sketches drawn by the face sketch artist who were appointed by the government. Hence to avoid manual searching machine were trained to identify the face sketch of the person by simply uploading to the machine and identifying. The common issue with all the previously done software where is that they compared the face sketches with human face which were usually front facing making it easier to be mapped in drawn sketch but when a sketch collected had their faces in different direction the algorithms were less likely to map it and match with a face from the database which is front facing.

We use CNN algorithm for our proposed software. CNN is an efficient recognition algorithm which is widely used in pattern recognition and image processing. It has many features such as simple structure, less training parameters and adaptability. In CNN, we can add new layers like feature learning layer, normalization layer, so that the full feature of dataset is learned by CNN architecture, it increases the ability to find matching image from the dataset which contains mugshots of people in different angles.

# TABLE OF CONTENTS

## ABSTRACT

## LIST OF ABBREVIATIONS

## LIST OF FIGURES

### 1. INTRODUCTION

1.1 Motivation .....	3
1.2 Objective .....	3
1.3 Organization of report.....	4

### 2. LITERATURE REVIEW.....5

### 3. PROBLEM DEFINITION AND ALGORITHM

3.1 Task Definition.....	7
3.2 Existing System.....	8
3.3 Proposed System.....	11

### 4. SYSTEM SPECIFICATION

4.1 Software Requirement.....	19
4.2 Hardware Requirement.....	19
4.3 Language Description.....	19

### 5. SYSTEM DESIGN

6.1 Data flow diagram.....	20
6.2 Use case diagram.....	23

### 6. SYSTEM IMPLEMENTATION.....24

### 7. TESTING

7.1 Unit testing.....	29
7.2 Integration testing.....	30
7.3 System testing.....	30

### 8. SYSTEM MAINTENANCE.....31

### 9. FUTURE ENHANCEMENT.....32

### 10. CONCLUSION.....33

### 11. APPENDIX.....34

### 12. BIBLIOGRAPHY

12.1 References.....	49
----------------------	----

## LIST OF ABBREVIATIONS

Abbreviations	Definitions	No.
CNN	Convolutional neural network	1
SIFT	Scale invariant feature transform	5
FERET	Face recognition technology	5
SVM	Support vector machine	8
ML	Machine learning	8
ANN	Artificial neural network	10
ReLU	Rectified linear activation unit	16
CCTV	Closed-circuit television	32

# LIST OF FIGURES

Figure No.	Figure Name	Page No.
3.3.1	Array of RGB Matrix	12
3.3.2	Neural network with many convolutional layers	13
3.3.3	Image matrix multiplies kernel or filter matrix	13
3.3.4	Image matrix multiplies kernel or filter matrix14	14
3.3.5	3 x 3 Output matrix14	14
3.3.6	Some common filters	15
3.3.7	Stride of 2 pixels	15
3.3.8	ReLU	16
3.3.9	Max Pooling	17
3.3.10	After pooling layer, flattened as FC layer	17
5.1	Data flow diagram	20
5.2	Use case diagram	23
7.2.5.1	Opening operation	26
11.1	Screenshots	40

## 1. INTRODUCTION

In this modern age, the overall crime rate is increasing day-by-day and to cope up with this law enforcement departments too should find ways that would speed up the overall process and help them in bringing them to justice. Forensic is the method of using scientific tests to find out about a crime. Forensics face sketches play a vital role to identify the face of the culprit. The existing technology uses face sketches drawn by the face sketch artist who were appointed by the government. Hence to avoid manual searching, machine were trained to identify the face sketch of a person by simply uploading it (mugshots) to the machine and identifying it. A **mugshot** is a photographic portrait of a person from the shoulders up, typically taken after a person is arrested. The original purpose of the mugshot was to allow law enforcement to have a photographic record of an arrested individual to allow for identification by victims, the public and investigators. A criminal can be easily identified and brought to justice using a face sketch drawn based on the description provided by the eye-witness.

Numerous methods that automatically identify subjects depicted in sketches as described by eyewitnesses have been implemented, but their performance often degrades when using real-world forensic sketches. The common issue with all the previously done software where is that they compared the face sketches with human face which were usually front facing making it easier to be mapped in drawn sketch but when a sketch collected had their faces in different direction the algorithms were less likely to map it and match with a face from the database which is front facing.

The machine learning method that proposed software used is a deep CNN. CNN is feature learning and classification part. Here we can add new layers like feature learning layer, normalization layer so the full feature of dataset is learned by CNN architecture which increases the ability to find matching image and also by using mug shots of people in different angle to train the dataset helps system to recognize the face of people in different direction.

Here we propose an application that recognizes the face sketch of the criminal using deep learning models.

Our application consist of

- ❖ Database administration
- ❖ User management



- ❖ Password reset
- ❖ Signup and Login
- ❖ Upload sketch
- ❖ View criminals profile
- ❖ Search for matching profiles with the image input
- ❖ Returns an identification of a criminal

The steps involved in this approach are data collection, data pre-processing, face detection, face sketching and classification. A dataset of criminal's face is created by collecting their faces that is captured in all position. We proposed a dataset containing faces of 10 persons. It should contain at least 20 images of each person which is captured in all position.

Pre-processing is done to clean the image data for model input. The images in the data are required to be the same sized arrays. Image pre-processing may also decrease model training time and increase model inference speed.

The sketch of the detected faces created by using image processing techniques which involves bitwise not operation, Gaussian blurring and image normalisation.

Recognition is performed by using a convolutional neural network to recognize the face sketches. We will be doing this as an application using django framework.

The proposed software consists of three modules

1. user
2. admin
3. detection

User module consists of all the information of user in time when login the whole data is stored in database. By using hashing algorithm it can try to increase security of the device. Admin module try to manage the software adding new features, parameters, new faces. Detection module is the place where we upload sketch and train it. By using deep CNN is used to preprocess and extract features of images and try to find the face from the given sketch. System can add different mug shots of people in different angle as the dataset and train these dataset to find the best match images to our sketches that are drawn by the forensic artist manually or digitally.

## 1.1 Motivation

Forensics face sketches play a vital role to identify the face of the culprit. The existing technology uses face sketches drawn by the face sketch artist who were appointed by the government. Hence to avoid manual searching, machine were trained to identify the face sketch of the person by simply uploading to the machine and identifying.

The common issue with all the previously done software where is that they compared the face sketches with human face which were usually front facing making it easier to be mapped in drawn sketch but when a sketch collected had their faces in different direction the algorithms were less likely to map it and match with a face from the database which is front facing.

We are using CNN architecture which increases the ability to find matching image and also by using mugshots of people in different angle to train the dataset helps system to recognise the face of people in different direction.

By using deep CNN is used to preprocess and extract features of images and try to find the face from the given sketch. System can add different mug shots of people in different angle as the dataset and train these dataset to find the best match images to our sketches that are drawn by the forensic artist manually or digitally.

## 1.2 Objective

Objective of our proposed system is to create a forensic face sketch recognition software which can find images of the culprit from the databases by using a face sketch drawn in any face angle of the culprit as define by the eye witness, we can add new layers in CNN which can increase the accuracy of the software.

### 1.3 Organization of report

The report of the project is organized as follows: Chapter 1 is used to introduce the project topic and it leads into the motivations behind selecting the project and what hoped to achieve in pursuing it. Chapter 2 contains the literature survey of the project topic where it lists the findings of existing implementations and its drawbacks. Chapter 3 Problem definition and algorithm of the project. Chapter 4 discusses System specification of the project. Chapter 5 deals with System design of the project. Chapter 6 describes the testing. Chapter 7 describes System implementation. Chapter 8 describes system maintenance. Chapter 9 describes Future enhancement. Chapter 10 describes conclusion. Chapter 11 describes appendix. Chapter 12 describes Bibliography.

## 2. LITERATURE REVIEW

*Wang, Xiaogang & Tang, Xiaoou* [1] proposed a recognition method of photo-sketch synthesized using a Multiscale Markov Random Field Model the project could synthesis a give sketch into photo or a given photo in to sketch and then search the database for a relevant match for this the model divided the face sketch in to patches. In this they first synthesized the available photos in to sketch and then trained the model making the model to decrease the difference between photos and sketch this enhanced the overall efficiency of the recognition model. For testing this they took few samples in which the photos where synthesized in to sketch and the same faces where drawn from sketch artist and then the model was trained from 60% data and remaining 40% data for testing the model. The overall results where impressive but not up to the mark as expected.

*Klare, Brendan & Jain, Anil* [2] proposed a method was sketch to photo matching which used SIFT Descriptor, the method proposed displayed result based on the measured SIFT Descriptor distance between the face photos in the database and the sketches. The algorithm first converts the face photos using linear transformation and then the sketch was used to measure the SIFT descriptor distance compared to the face photo and in some cases distance between images in the databases too where measured for better accuracy.

*Lahlali, S.E. & Sadiq, Abdelalim & Mbarki, Samir* [3] proposed a method to search human faces using sketches, this method converted sketches to mug shots and then matched those mugshots to faces using some local and global variables been declared by the face matching algorithms. However, in some cases the mugshots where hard to be matched with the human faces in the databases like FERET Database and Japanese Database. The proposed method showed an accuracy of about 70% in the experimental results, which was fair decent but still lacked the accuracy needed by the law enforcement department.

*Abhijit Patil, Akash Sahu, Jyoti Sah* [4] designed a standalone application for constructing and identifying the facial composites, the initial system was found to be time consuming and confusing as the traditional method, later switching to a new

approach in which the victim was given option of faces and was made to selected similar face resembling the suspect and at the end the system would combine all the selected face and try to predict automatically the criminal's facial composite. The Results where promising and 10 out of 12 composite faces where named correctly out of which the results 21.3% when the witness was helped by the department person to construct the faces and 17.1% when the witness tried constructing faces by themselves.

### 3. PROBLEM DEFINITION

#### 3.1 Task Definition

The common issue with all the previously done software where is that they compared the face sketches with human face which were usually front facing making it easier to be mapped in drawn sketch but when a sketch collected had their faces in different direction the algorithms were less likely to map it and match with a face from the database which is front facing.

The machine learning method that proposed software used is a deep CNN. CNN is feature learning and classification part. Here we can add new layer s like feature learning layer, normalisation layer so the full feature of dataset is learned by CNN architecture which increases the ability to find matching image and also by using mugshots of people in different angle to train the dataset helps system to recognise the face of people in different direction.

The proposed software consists of three modules 1).user 2).admin 3) sketch recognition. User module consists of all the information of user in time when login the whole data is stored in database. By using hashing algorithm it can try to increase security of the device. Admin module try to manage the software adding new features, parameters, new faces. Sketch recognition module is the place where we upload sketch and train it. System can add different mug shots of people in different angle as the dataset and train these dataset to find the best match images to our sketches that are drawn by the forensic artist manually or digitally.

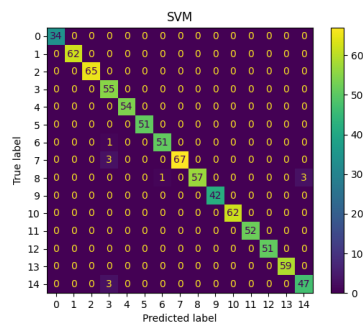
## 3.2 Existing System

### 3.2.1 Support Vector Machine (SVM)

This method is the most popular Supervised Learning algorithm, which is employed for Analysis as well as Regression problems. It is employed for Classification problems in ML. The objective of this algorithm is to construct the best line or decision boundary that can divide n-dimensional space into categories so that we can simply put the new data point in the correct classes in the future. SVM chooses the extreme points or vectors that help in developing the hyperplane. These extreme cases are called support vectors, therefore the algorithm is termed as Support Vector Machine.

Pros and Cons associated with SVM

- **Pros:**
  - It works really well with a clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where the number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- **Cons:**
  - It doesn't perform well when we have large data set because the required training time is higher
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
  - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.



### 3.2.2 RANDOM FOREST

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

#### Steps involved in Random Forest algorithm:

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

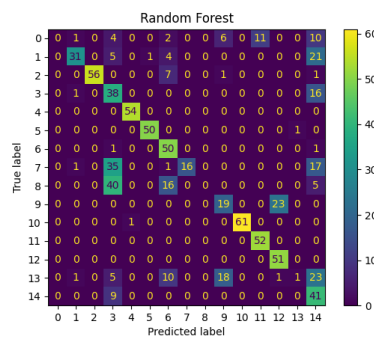
Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on **Majority Voting or Averaging** for Classification and regression respectively.

- **Pros:**
  - Random Forest is capable of performing both Classification and Regression tasks.
  - It is capable of handling large datasets with high dimensionality.
  - It enhances the accuracy of the model and prevents the over fitting issue.
- **Cons:**
  - Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.





### 3.2.3 ARTIFICIAL NEURAL NETWORKS (ANN)

ANN or artificial neural networks are information processing systems that are inspired by our biological nervous system, such as how our brain processes information. Our human brain has around 100 billion neurons, and each neuron has a connection point between 1000 to 10000. The human brain is designed to store information and extract it in a way where we can extract more than one piece of information from our memory whenever needed. The human brain is made up of thousands of powerful parallel processors.

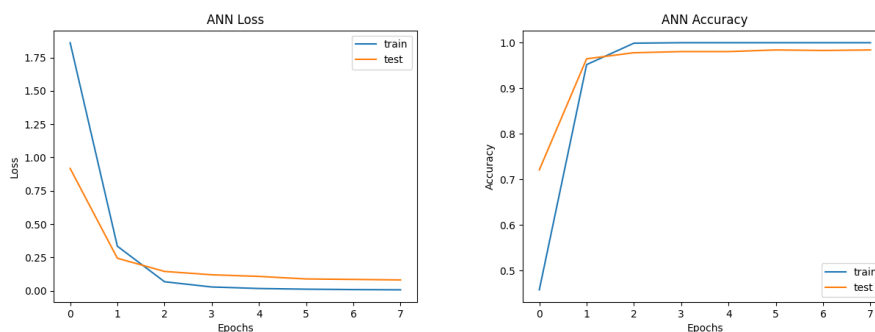
Similarly, artificial neural networks have neurons placed similarly to the human mind. Each neuron is connected with other neurons with certain coefficients. When these networks are put through training, information gets passed through these connecting points, which helps the network learn.

- **Pros:**

- You can store information on the entire network
- It has fault tolerance –
- The feature of gradual corruption
- It can make machine learning
- It has parallel processing capability

- **Cons:**

- It is hardware dependent
- Lack of determining factors of proper network structure
- ANNs has delicate showing the problem to the network
- Unjustified behavior of the network.



### 3.3 Proposed System

#### 3.3.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Convolutional Neural Networks belong to a category of deep learning neural networks. It follows supervised learning. They are excellent in the fields of object classification, recognition and localisation. They work on the mathematical principle of convolution. A kernel or a matrix is applied on the image to obtain a convoluted output of the original image and depending on the number of kernels operated or “filter” used, several outputs are obtained for a single input. However, there an issue arises, as the size and volume of the input images increases, the process becomes more computationally expensive. This is overcome by applying a pooling function which reduces the output image size. Once successive rounds of convolution and pooling(subsampling) are performed, the features learned so far can be utilized by applying it into a fully connected layer to vote for which class the input belongs to. If the predicted results are inaccurate, back propagation and gradient descent is used to adjust the weights of the parameters, until the predicted output is sufficiently close to the target output.

Using CNNs for deep learning is popular due to three important factors:

- CNNs eliminate the need for manual feature extraction—the features are learned directly by the CNN.
- CNNs produce highly accurate recognition results.
- CNNs can be retrained for new recognition tasks, enabling you to build on pre-existing networks.

- In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

CNN image classifications takes an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see  $h \times w \times d$  ( $h$  = Height,  $w$  = Width,  $d$  = Dimension ). Eg., An image of  $6 \times 6 \times 3$  array of matrix of RGB (3 refers to RGB values) and an image of  $4 \times 4 \times 1$  array of matrix of grayscale image.

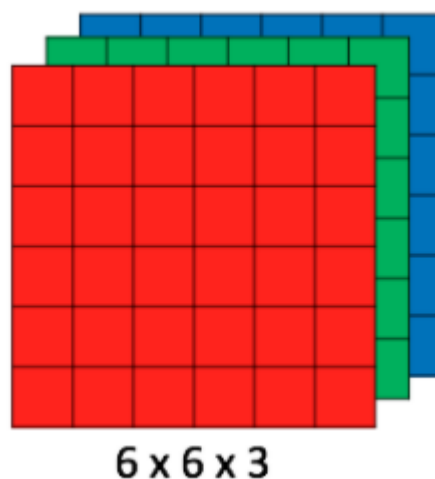


Figure 3.3.1 : Array of RGB Matrix

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

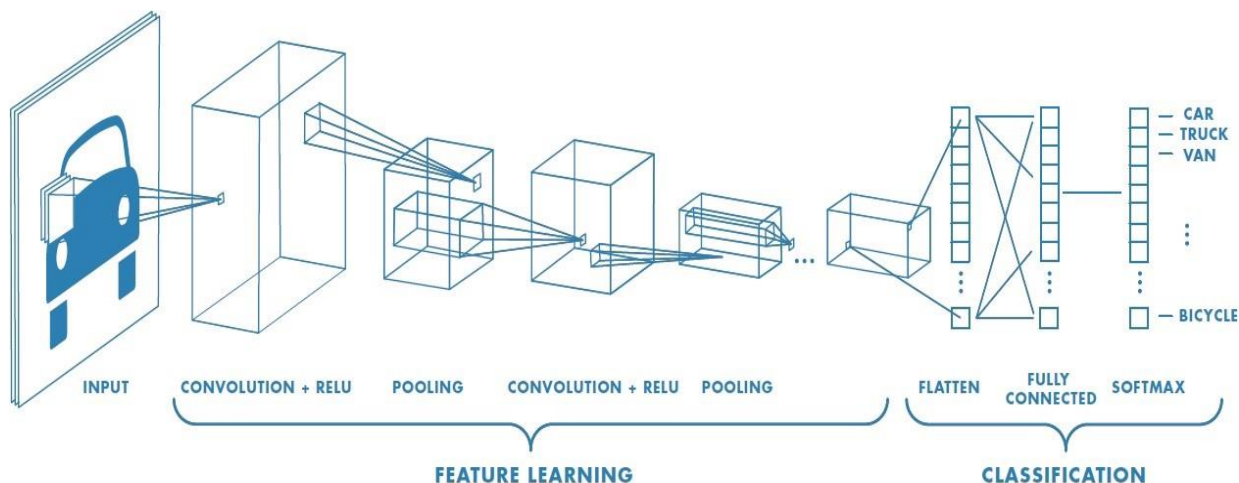


Figure 3.3.2 : Neural network with many convolutional layers

### Convolution Layer

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

- An image matrix (volume) of dimension **(h x w x d)**
- A filter **(f<sub>h</sub> x f<sub>w</sub> x d)**
- Outputs a volume dimension **(h - f<sub>h</sub> + 1) x (w - f<sub>w</sub> + 1) x 1**

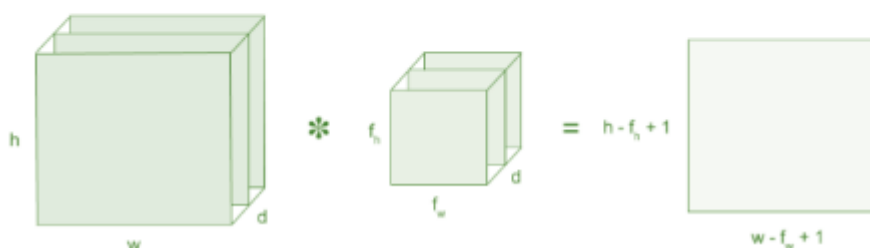


Figure 3.3.3: Image matrix multiplies kernel or filter matrix

Consider a 5 x 5 whose image pixel values are 0, 1 and filter matrix 3 x 3 as shown in below

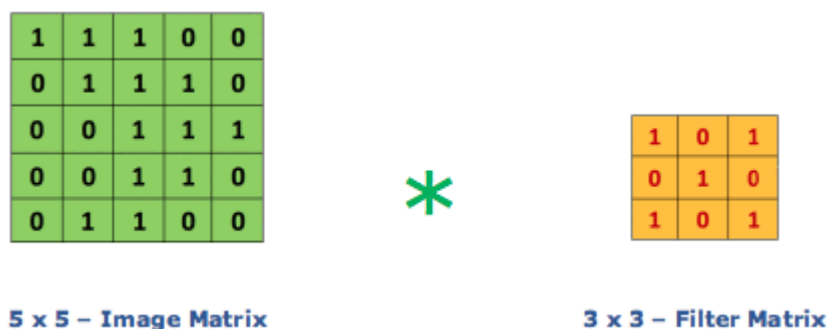


Figure 3.3.4: Image matrix multiplies kernel or filter matrix

Then the convolution of 5 x 5 image matrix multiplies with 3 x 3 filter matrix which is called **“Feature Map”** as output shown in below

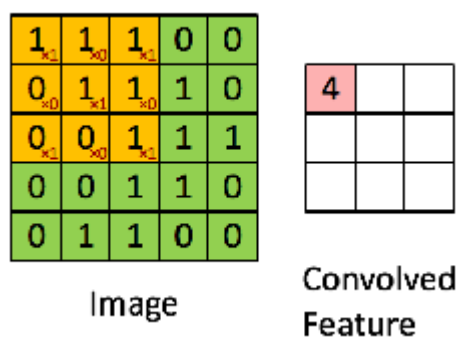


Figure 3.3.5: 3 x 3 Output matrix

Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example shows various convolution image after applying different types of filters (Kernels).



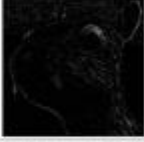




Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Figure 3.3.6 : Some common filters

## Strides

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. The below figure shows convolution would work with a stride of 2.

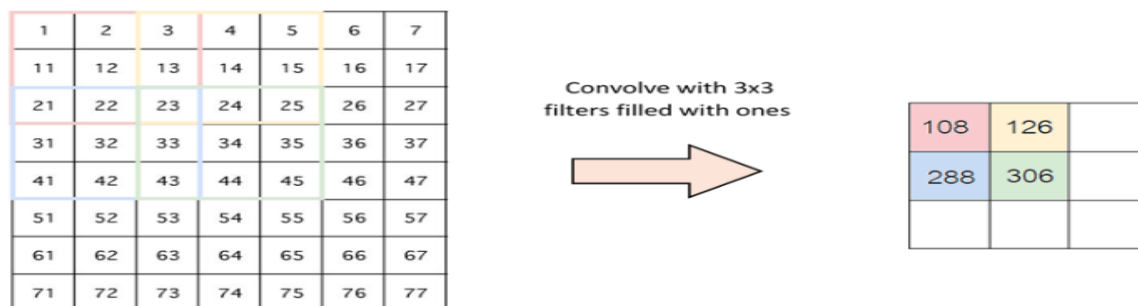


Figure 3.3.7 : Stride of 2 pixels

## Padding

Sometimes filter does not fit perfectly fit the input image. We have two options:

- Pad the picture with zeros (zero-padding) so that it fits
- Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

## Non Linearity (ReLU)

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is  $f(x) = \max(0, x)$ . Why ReLU is important : ReLU's purpose is to introduce non-linearity in our ConvNet. Since, the real world data would want our ConvNet to learn would be non-negative linear values.

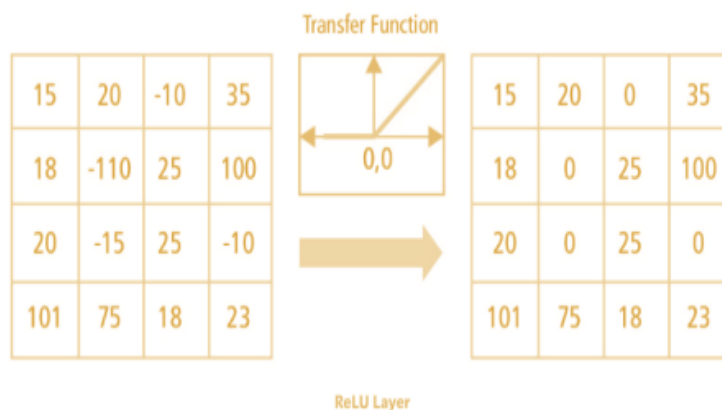


Fig 3.3.8:ReLU

There are other non linear functions such as tanh or sigmoid that can also be used instead of ReLU. Most of the data scientists use ReLU since performance wise ReLU is better than the other two.

## Pooling Layer

Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or downsampling which reduces the dimensionality of each map but retains important information. Spatial pooling can be of different types:

- Max Pooling

- Average Pooling
- Sum Pooling

Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.

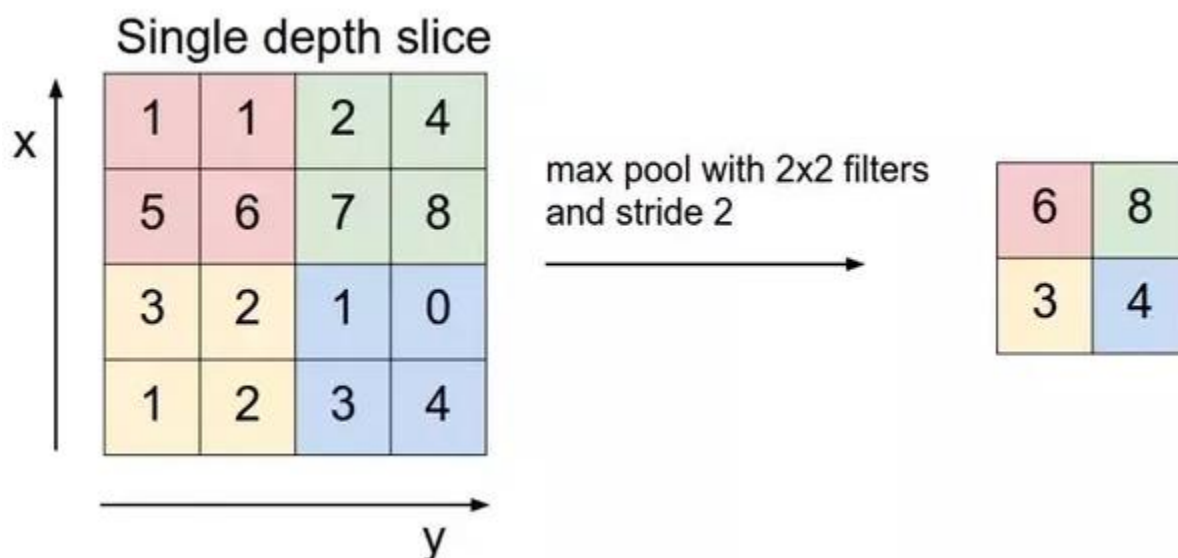


Figure 3.3.9 : Max Pooling

### Fully Connected Layer

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network.

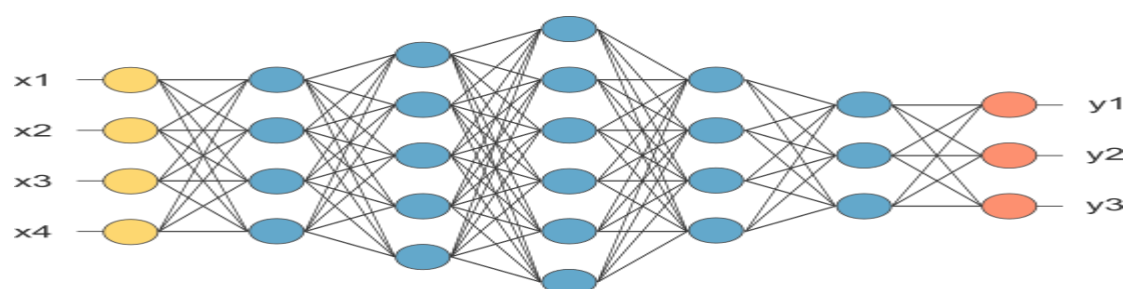
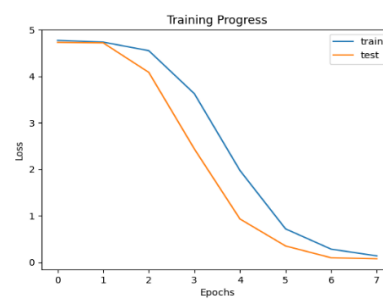
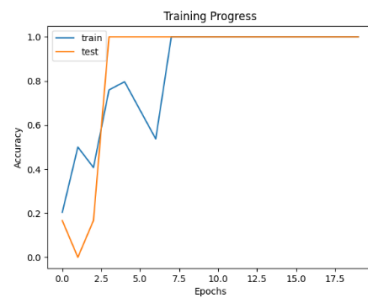


Figure 3.3.10 : After pooling layer, flattened as FC layer



**FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN**

## 4. SYSTEM SPECIFICATION

### 4.1 Software Requirements

- Operating System : Windows 10
- Processor : Intel i3 or above
- RAM : 4GB or above
- Language : Python
- IDE : VS Code

### 4.2 Hardware requirements

- Processor : intel i3 or above
- RAM : 4GB or above

### 4.3 Language description

#### 4.3.1 Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is often described as a "batteries included" language due to its comprehensive standard library.

Following are important characteristics of Python Programming:

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

## 5. SYSTEM DESIGN

### 5.1 Data flow diagram

#### Level 0 DFD

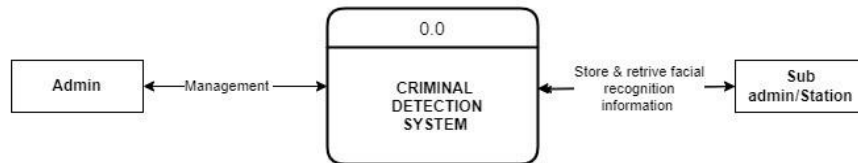


Fig 5.1.1.

#### Level 1 DFD

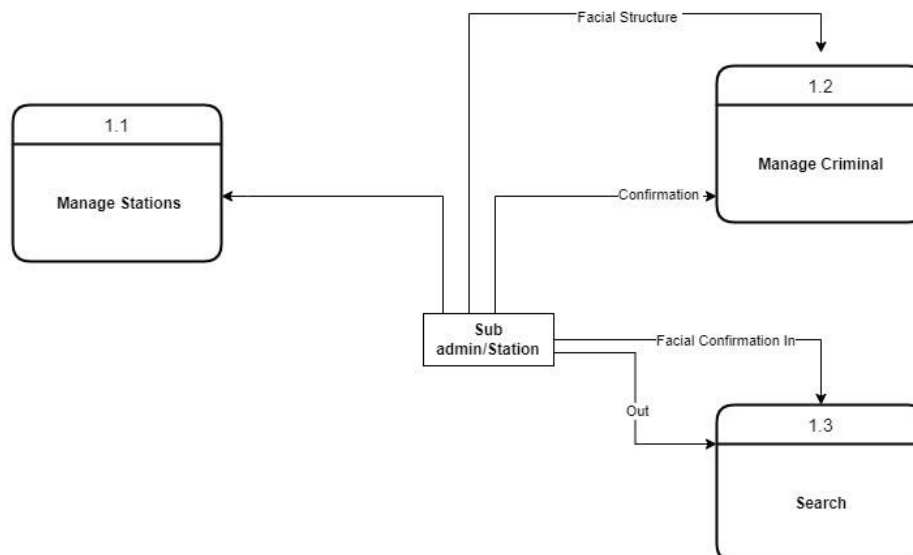


Fig 5.1.2

## Level 2 DFD

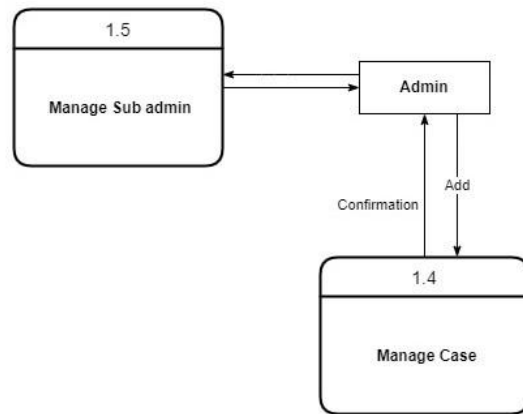


Fig 5.1.3

## Level 3 DFD

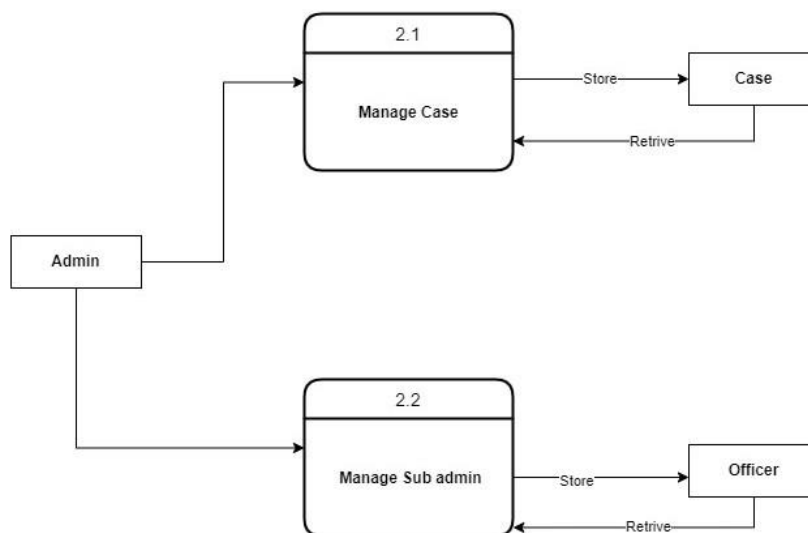


Fig 5.1.4

## Level 4 DFD

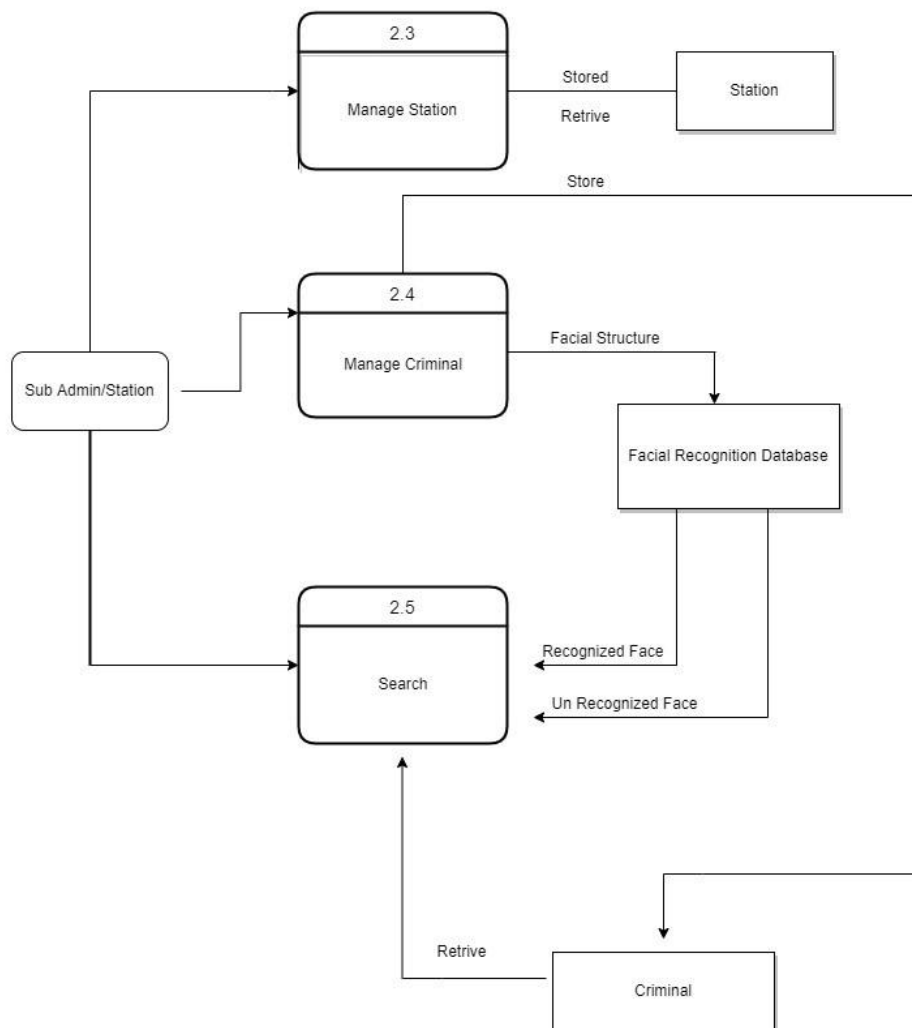


Fig 5.1.5

## 5.2 Use case diagram

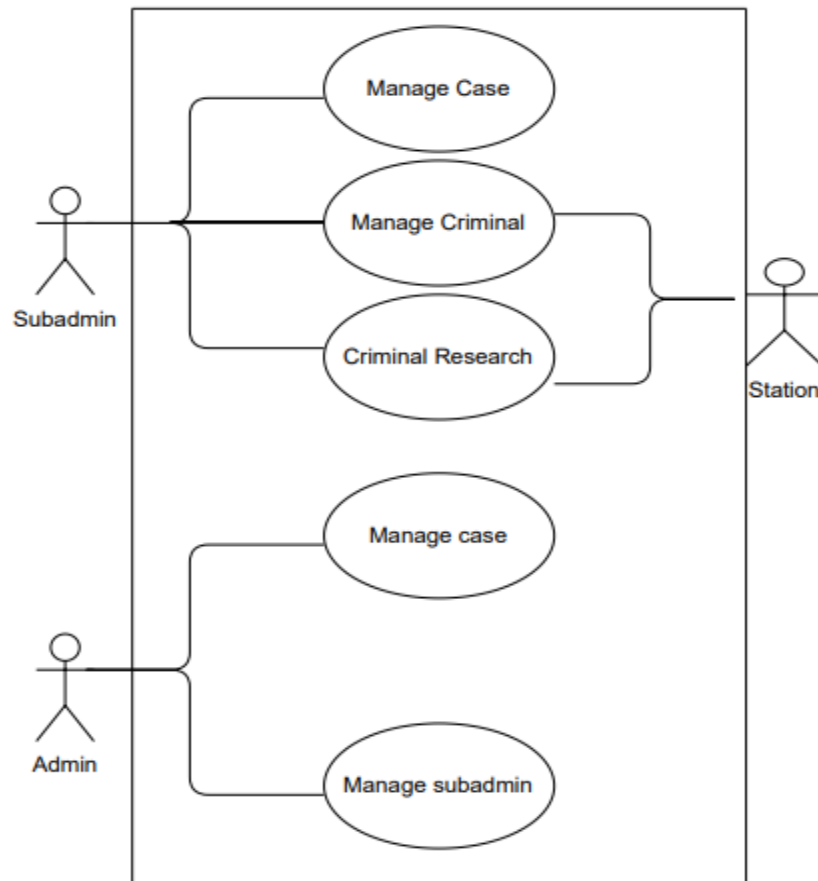
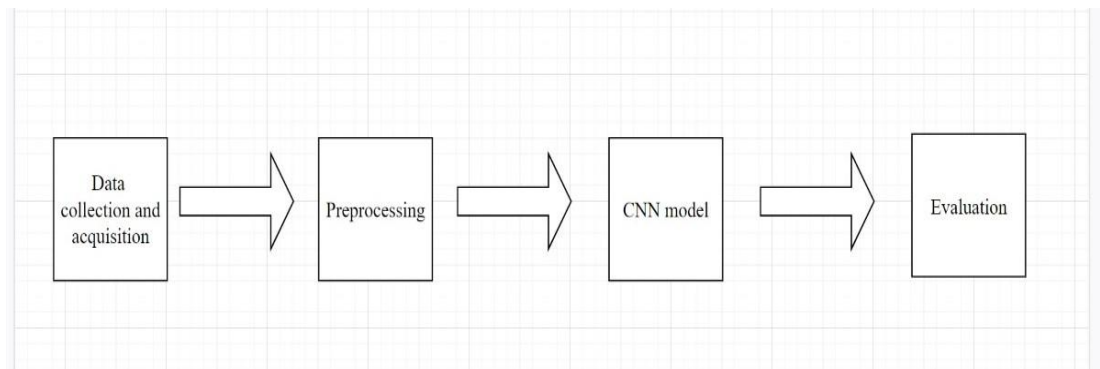


Fig 5.2.1

## 6. SYSTEM IMPLEMENTATION



### 6.1 Data collection

To perform object recognition first we need to collect or gather data required for that, and for that purpose we can capture images directly from our live environment and save it to its corresponding section or category. Another way of collecting images of objects for recognition are download it directly from the platforms like Kaggle, GitHub, etc. these platforms consists of either separate dataset of the collection of objects or else we can download large dataset like FERET dataset or such which consists of collection of objects. The images that we capture for object recognition are then divided to test and train images which are set within our dataset for various processing

We have taken HeadPoseImageDatabase for our project. A dataset of criminal's face is created by collecting their faces that is captured in all position. It contains 120 images of each person which is captured in all position.

### 6.2 Preprocessing

#### 6.2.1.RGB to Grayscale

The images in the dataset are comprised of color images. The color images are converted to gray scale for easy computation. There are a number of commonly used methods to convert an RGB image to a grayscale image such as average method.

#### Average Method

The Average method takes the average value of R, G, and B as the grayscale value.

$$\text{Grayscale} = (R + G + B) / 3.$$

Theoretically, the formula is 100% correct. But when writing code, you may encounter uint8 overflow error — the sum of R, G, and B is greater than 255. To avoid the exception, R, G, and B should be calculated respectively.

$$\text{Grayscale} = R / 3 + G / 3 + B / 3.$$

The average method is simple but doesn't work as well as expected. The reason being that human eyeballs react differently to RGB. Eyes are most sensitive to green light, less sensitive to red light, and the least sensitive to blue light. Therefore, the three colors should have different weights in the distribution. That brings us to the weighted method.

### 6.2.3. Image resizing

Image resizing refers to the scaling of images. Scaling comes in handy in many image processing as well as machine learning applications. It helps in reducing the number of pixels from an image and that has several advantages e.g. It can reduce the time of training of a neural network as more is the number of pixels in an image more is the number of input nodes that in turn increases the complexity of the model. It also helps in zooming in images. Many times we need to resize the image i.e. either shrink it or scale up to meet the size requirements. OpenCV provides us several interpolation methods for resizing an image.

### 6.2.4. Adaptive thresholding

Adaptive thresholding is the method where the threshold value is calculated for smaller regions and therefore, there will be different threshold values for different regions.

### 6.2.5. Opening operations

- Opening is generally used to restore or recover the original image to the maximum possible extent
- Opening is a process in which first erosion operation is performed and then dilation operation is performed.

Opening is denoted by:

$$A \circ B = (A \ominus B) \oplus B$$

where A - image



B - structuring element

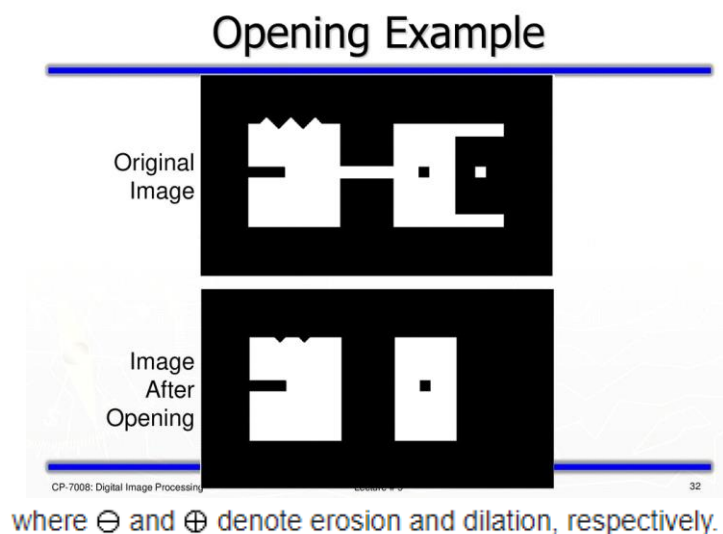


Fig 7.2.5.1

#### 6.2.6. Apply image negative

negative images are useful for enhancing white or gray detail embedded in dark regions of an image.

negative images is denoted by:

$$s = L - 1 - r$$

$L-1$  = maximum pixel value

$r$  = pixel value of an image

#### 6.2.7. Multiply image negative with original image

### 6.3. Convolutional Neural Network (CNN) Model

A convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery.

There is no feature extraction technique is applied in the project to get face features. However, the features are obtained from the convolution layer of CNN. We are directly inputting the pre-processed image to CNN Model.

CNN layers used in our project:

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 217, 217, 64)	7808
max_pooling2d_1 (MaxPooling2)	(None, 108, 108, 64)	0
conv2d_2 (Conv2D)	(None, 104, 104, 96)	153696
max_pooling2d_2 (MaxPooling2)	(None, 52, 52, 96)	0
conv2d_3 (Conv2D)	(None, 49, 49, 256)	393472
max_pooling2d_3 (MaxPooling2)	(None, 24, 24, 256)	0
flatten_1 (Flatten)	(None, 147456)	0
dense_1 (Dense)	(None, 50)	7372850
dense_2 (Dense)	(None, 50)	2550
dense_3 (Dense)	(None, 5)	255
Total params: 7,930,631		
Trainable params: 7,930,631		
Non-trainable params: 0		

The layers used are convolution layer, maxpooling layer and fully connected layer.

The proposed CNN consist of an input layer, 3 convolutional layer followed by maxpooling layers, a flatten layer, 2 fully connected layer and an output layer. The images of criminals are stored in an array and is fed to the input layer of CNN. The input of the CNN model is images with dimension  $227 \times 227 \times 1$  pixels. In the first convolutional layer, 64 filters of size  $11 \times 11$  are applied to the input images, generating 64 feature maps.

These feature maps are passed to rectified linear unit (ReLu) function for non-linear operation. The output of ReLu are selected by the first maxpooling layer with a stride of 2 pixels, using a neighborhood of  $2 \times 2$  pixels. This reduces the size of feature maps by factor of two, thus retaining the most useful information while discarding the less important details.

Next, we use second convolutional layer, 96 filters of size  $5 \times 5$  generating 96 feature maps. ReLu function and maxpooling with a stride of 2 pixels, using a neighborhood of  $2 \times 2$  pixels are applied. The third convolutional layer consist of 256 filters of size  $4 \times 4$  followed by ReLu function and maxpooling layer with a stride of 2

pixels, using a neighborhood of 2\*2 pixels. The output of the third maxpooling layer is fed to the flatten layer to flatten the matrix into vector and inputted to the fully connected layer. In a fully connected layer each neuron is connected to every neuron in the previous layer, and each connection has its own weight. The output of the fully connected layer is fed directly into the softmax function to calculate the probabilities of the target classes.

#### 6.4 Evaluation

**Accuracy:** Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$\text{Accuracy} = \text{Correctly classified instances} / \text{Total instances}$$

**Precision:** Also called as positive predictive value is the fraction of relevant instances among the retrieved instances. The precision is intuitively the ability of the classifier not to label a sample as positive if it is negative. High the value, better the classifier.

$$\text{Precision} = \text{tp} / (\text{tp} + \text{tn} + \text{fp} + \text{fn})$$

**Sensitivity (Recall)** is the fraction of the total amount of relevant instances that were actually retrieved. The recall is intuitively the ability of the classifier to find all the positive samples. Higher the value higher is the accuracy.

$$\text{Recall} = \text{tp} / (\text{tp} + \text{fn})$$

**F1 score:** The F1 Score is the  $2 * ((\text{precision} * \text{recall}) / (\text{precision} + \text{recall}))$ .

## 7. TESTING

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to actual requirements.

### Testing levels and types

The aim of the system testing process was to determine all defects in our project. The program was subjected to a set of test inputs various observations were made and based on these observations it will be decided whether the program behaves as expected or not. Our project went through three levels of testing

- Unit Testing
- Integration Testing
- System Testing

### 7.1 Unit testing

Unit testing is undertaken when a module has been created and successfully reviewed. In order to test a single module we need to provide a complete environment ie beside the module we would require.

- The procedure belonging to other modules under test calls
- Unit testing was done on each and every module that is described under module description
- In unit testing we test the modules in python independently to ensure its accuracy and performance.

## 7.2 Integration testing

In this type of testing we test various integrations of the project module by providing the input. The primary objective is to test the module interfaces in order to ensure that no errors are occurring when no module invokes the other module. In this testing we combine different modules together and test them.

## 7.3 System testing

To verify and validate the behavior of the entire system against the original system objectives. Software testing is a process that identifies the correctness, completeness, and quality of software. Following is a list of various types of software testing and their definitions in a random order.

- **Manual Testing:** That part of software testing that requires human input, analysis, evaluation.
- **Black box Testing:** Testing software without any knowledge of the back-end of the system, structure or language of the module being tested. Black box test cases are written from a definitive source document, such as a specification or requirements document.
- **White box Testing:** Testing in which the software tester has knowledge of the back- end, structure and language of the software, or at least its purpose.

## 8. SYSTEM MAINTENANCE

Any work done to change the system after it is in operation

This include:

- Error corrections
- Enhancement of capabilities
- Deletion of obsolete capabilities
- Optimization
- The purpose is to preserve the value of software over time

**Corrective Maintenance:** refers to modifications initiated by defects in the software

**Adaptive Maintenance:** include any work initiated as consequence of moving the software to a different hardware or software platform-compiler, operating system or new processor.

**Perfective Maintenance:** refers to enhancements such as making the product better, faster, smaller, better documented, cleaner structured.

**Preventive Maintenance:** the purpose is making program easier to understand and hence facilitating further maintenance work.

## 9. FUTURE ENHANCEMENT

Our project “**Forensic Face Sketch Recognition Software using deep CNN**” is currently designed to work on very few scenarios like on face sketches and matching those sketches with the face photos in the law enforcement records.

The platform can be much enhanced in the future to work with various technologies and scenarios enabling it to explore various media and surveillances medium and get a much wider spread and outputs. The platform can be modified to match the Face sketch with the human faces from the video feeds by using the 3D mapping and imaging techniques and same can be implemented to the CCTV surveillances to perform face recognition on the Live CCTV footage using the Face Sketch.

The platform can further be connected to social media has social media platforms acts has a rich source for data in today’s world, this technique of connecting this platform with the social media platform would enhance the ability of the platform to find a much more accurate match for the face sketch and making the process much more accurate and speeding up the process.

In all the platform could have features which could be different and unique too and easy to upgrade, when compared to related studies on this field, enhancing the overall security and accuracy by standing out among all the related studies and proposed systems in this field.

## 10. CONCLUSION

Our project “Forensic Face Photo-Sketch Recognition Software using deep CNN” is been designed, developed and finally tested keeping the real-world scenarios from the very first login screen to the final screen to fetch data from the records keeping security, privacy and accuracy as the key factor in every scenario.

Here we proposed a photo-sketch recognition software for law enforcement departments to identify a criminal by the details given by the victim. Our approach is tested on HeadPoseImageDatabase which contains images of 15 persons and around 130 images of a single person. The platform showed a good accuracy and speed while face sketch recognition, provided an average accuracy of more than 90% which means a very good rate according to elated studies on this field.

The platform even has features which are different and unique too when compared to related studies on this field, enhancing the overall security and accuracy by standing out among all the related studies and proposed systems in this field.



## 11. APPENDIX

### 11.1 Source code

#### model1.py

```
import tensorflow as tf
import numpy as np
import os
import cv2
import random
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from keras import layers, models
from keras.models import load_model
from sklearn.metrics import classification_report,
confusion_matrix, accuracy_score, plot_confusion_matrix, ConfusionMatrixDisplay

path1=os.getcwd()
path=os.path.join(path1,'dataset','HeadPoseImageDatabase')

data=os.listdir(path)
kernel = np.ones((2,1),np.uint8)

out=[]
##out=np.zeros((60,227,227))
label=[]

d=0
k=0
m=0

for i in data:
    path1=os.path.join(path,i)
    print(path1)
    class_data=os.listdir(path1)
    print(class_data)
    m=d
    d=d+1

    da=[m for x in range(len(class_data))]
    label.extend(da)
    m=m+1
    for j in class_data:
```

```

    imag=cv2.imread(os.path.join(path1,j),0)
    imag=cv2.resize(imag,(227,227))

    ou =
cv2.adaptiveThreshold(imag,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.TH
RESH_BINARY,7,5)
    print('ou',ou)
    ou = cv2.morphologyEx(ou, cv2.MORPH_OPEN, kernel)
    ou= ou.reshape(227,227)
##    cv2.imshow('face1',ou)
##    cv2.waitKey(1)
    ou=255-ou

    ou=np.multiply(imag,ou)
    print(ou)
##    cv2.imshow('face',ou)
##    cv2.destroyAllWindows()
##    cv2.waitKey(1)
    out.append(ou)

    k=k+1

##cv2.destroyAllWindows()
out=np.array(out)

out=out/255
u=out.reshape(k,227,227,1)
from keras.utils import np_utils
label = np_utils.to_categorical(label)

print(label)
x_train, x_valid, y_train, y_valid = train_test_split(u, label, test_size=0.30,
random_state= 42)

model = models.Sequential()
model.add(layers.Conv2D(64, (11, 11), activation='relu', input_shape=(227, 227, 1)))
model.add(layers.MaxPooling2D((2, 2),strides=(2,2)))
model.add(layers.Conv2D(96, (5, 5), activation='relu'))
model.add(layers.MaxPooling2D((2, 2),strides=(2,2)))
model.add(layers.Conv2D(256, (4, 4), activation='relu'))
model.add(layers.MaxPooling2D((2, 2),strides=(2,2)))
model.add(layers.Flatten())
model.add(layers.Dense(50))
model.add(layers.Dense(50))
model.add(layers.Dense(114, activation='softmax'))

model.summary()

```

```

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

history1=model.fit(u, label, epochs=8, validation_data=(x_valid, y_valid),verbose=2)

loss, acc = model.evaluate(x_valid,y_valid, verbose=2,batch_size=24)

model.save('modelcnncv1.h5')

print("accuracy is:",acc)

plt.title('loss')
plt.plot(history1.history['loss'], label='train')
plt.plot(history1.history['val_loss'], label='test')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.title('Training Progress')
plt.legend()
plt.show()

plt.title('accuracy')
plt.plot(history1.history['acc'], label='train')
plt.plot(history1.history['val_acc'], label='test')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Training Progress')
plt.legend()
plt.show()

y_pred = model.predict_classes(x_valid)
cm=confusion_matrix(y_valid,y_pred)
print(cm)
print(classification_report(y_valid,y_pred))
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=['0','1'])
disp.plot()
plt.show()

```

### **single\_test.py**

```

import warnings
warnings.filterwarnings("ignore")
import keras
import tensorflow as tf
import cv2
import numpy as np
from tkinter import Tk
from tkinter.filedialog import askopenfilename

```

```

def get_criminal(file):
    fileName = file
    if fileName == "":
        print('No file selected')
        print('Program Completed')
        exit()
    kernel = np.ones((2,1),np.uint8)

    imag=cv2.imread(fileName,0)

    imag=cv2.resize(imag,(227,227))

    ou =
cv2.adaptiveThreshold(imag,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,7,5)

    ou = cv2.morphologyEx(ou, cv2.MORPH_OPEN, kernel)
    ou= ou.reshape(227,227)

    ou=255-ou
    ou=np.multiply(imag,ou)

    model = keras.models.load_model('main/modelcnncv1.h5')

    OUT=model.predict(ou.reshape(-1,227,227,1)/255)

    X=np.argmax(OUT,axis=1)

    d={0:'aesculus flava',1:'ailanthus altissima',2:'chamaecyparis thyoides',3:'evodia
daniellii',4:'fraxinus pennsylvanica',5:'juniperus virginiana',6:'phellodendron
amurense',
    7:'pinus parviflora',8:'quercus falcata',9:'quercus macrocarpa',10:'taxodium
distichum'}
    keras.backend.clear_session()
    print(X[0])
    return X[0]

```

### views.py

```

from django.shortcuts import render, redirect
from .forms import CaseTypeForm, CaseType
from accounts.forms import UserForm, OfficerForm
from accounts.models import Officer, User

```

```

# Create your views here.
def index(request):
    return render(request, 'admin/index.html')

def add_case_type(request):
    if request.method == 'GET':
        context = {}
        context['form'] = CaseTypeForm()
        context['case_types'] = CaseType.objects.all()
        return render(request, 'admin/add_case_type.html', context)
    elif request.method == 'POST':
        form = CaseTypeForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('add_case_type')
        else:
            return render(request, 'admin/add_case_type.html', {'form': form})

def remove_case_type(request, case_type_id):
    case_type = CaseType.objects.get(id=case_type_id)
    case_type.delete()
    return redirect('add_case_type')

def add_sub_admin(request):
    if request.method == 'GET':
        context = {}
        context['form1'] = UserForm()
        context['form2'] = OfficerForm()
        return render(request, 'admin/add_sub_admin.html', context)
    elif request.method == 'POST':
        form1 = UserForm(request.POST)
        form2 = OfficerForm(request.POST)
        if form1.is_valid() and form2.is_valid():
            obj = form1.save(commit=False)
            obj.set_password(form1.cleaned_data['password'])
            obj.is_staff=True
            obj.save()
            officer = form2.save(commit=False)
            officer.user = obj
            officer.save()
            return redirect('add_sub_admin')
        else:
            context = {}
            context['form1'] = form1
            context['form2'] = form2
            return render(request, 'admin/add_sub_admin.html', context)

def manage_sub_admins(request):
    context = {}

```

```
context['users'] = Officer.objects.all()
return render(request, 'admin/manage_sub_admins.html', context)

def remove_sub_admin(request, sub_id):
    sub_admin = User.objects.get(id=sub_id)
    sub_admin.delete()
    return redirect('manage_sub_admins')
```

### **manage.py**

```
import os
import sys

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE",
        "criminaldetection.settings")
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)
```

## 11.1 Screenshots

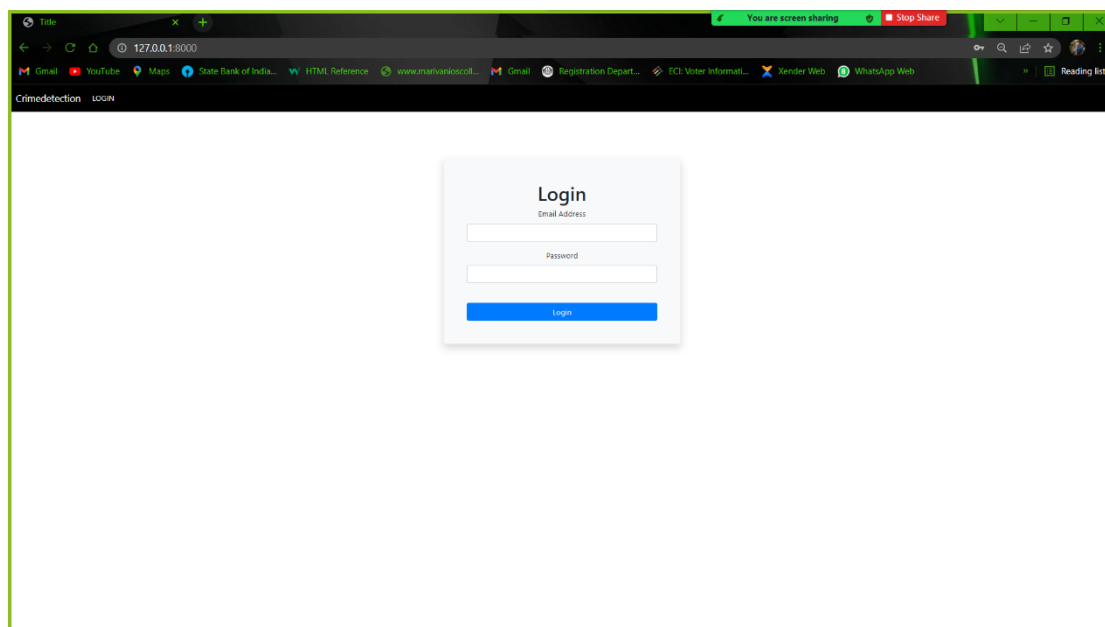


Fig 11.1.1. Login Page  
(Login page for both admin and sub admin)

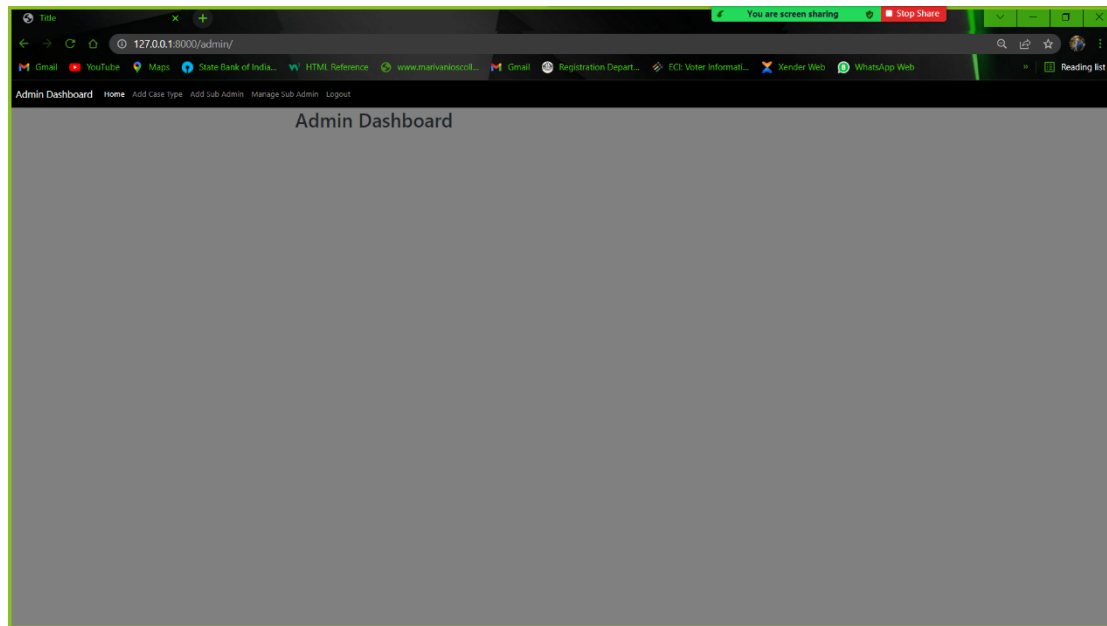


Fig11.1.2 Admin Dashboard

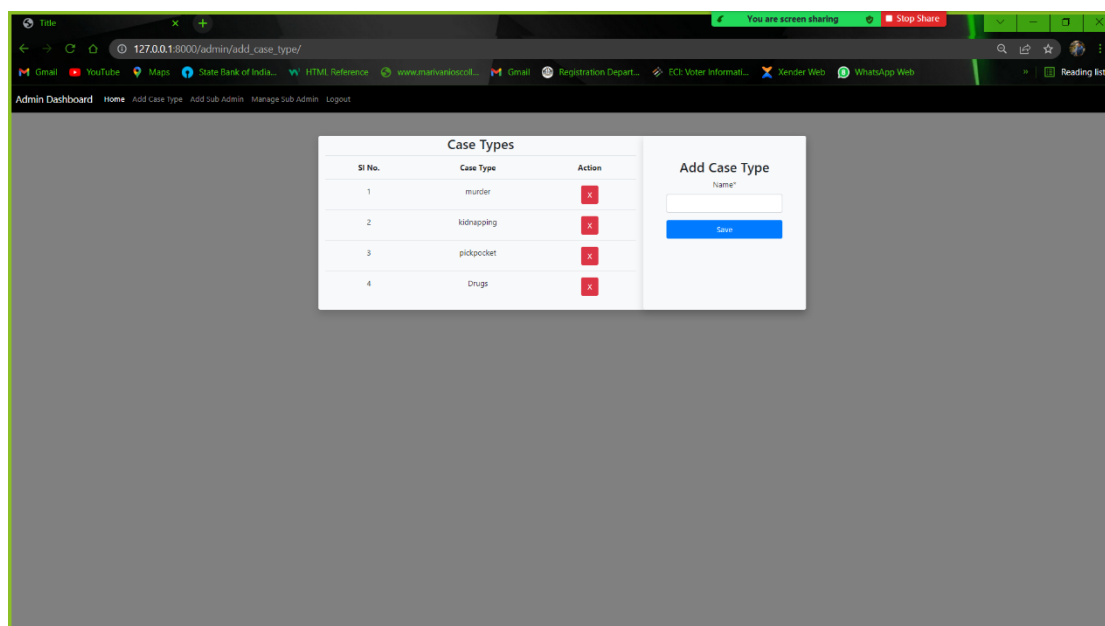


Fig 11.1.3 Add case type



## FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN

Sub Admin Registration

First name

Last name

Email address

Username\*

Required: 100 characters or fewer. Letters, digits and @/./\_/- only

Password\*

Phone

Rank

Register

Fig 11.1.4 Add sub admin

Sub Admin List

Sl No.	Name	Email Id	Phone	Rank	Action
1	Akesh Mohanan	akeshd@gmail.com	9876543210	Police Sub Inspector (P.S.I)	
2	Manu S	manu@gmail.com	9876543210	Police Inspector (P.I)	
3	sooraj j s	soorajjs937@gmail.com	8590613803	Police Chief	
4	belvin joe	belvinabraham@gmail.com	8590613803	Assistant Police Sub Inspector (A.S.I)	
5	Megha Rajendran	megha24rajendran@gmail.com	6282962354	Police Chief	
6	nithin nithin	nithinalis@gmail.com	5656565	Police Sub Inspector (P.S.I)	

Fig 11.1.5 Manage sub admin

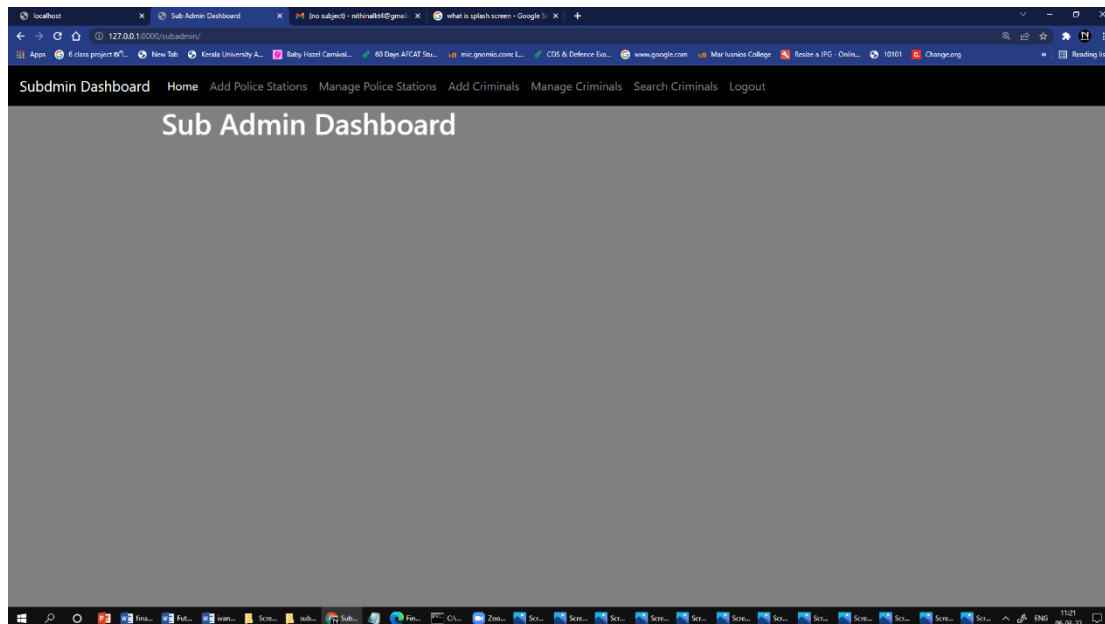


Fig 11.2.1 Sub Admin Dashboard

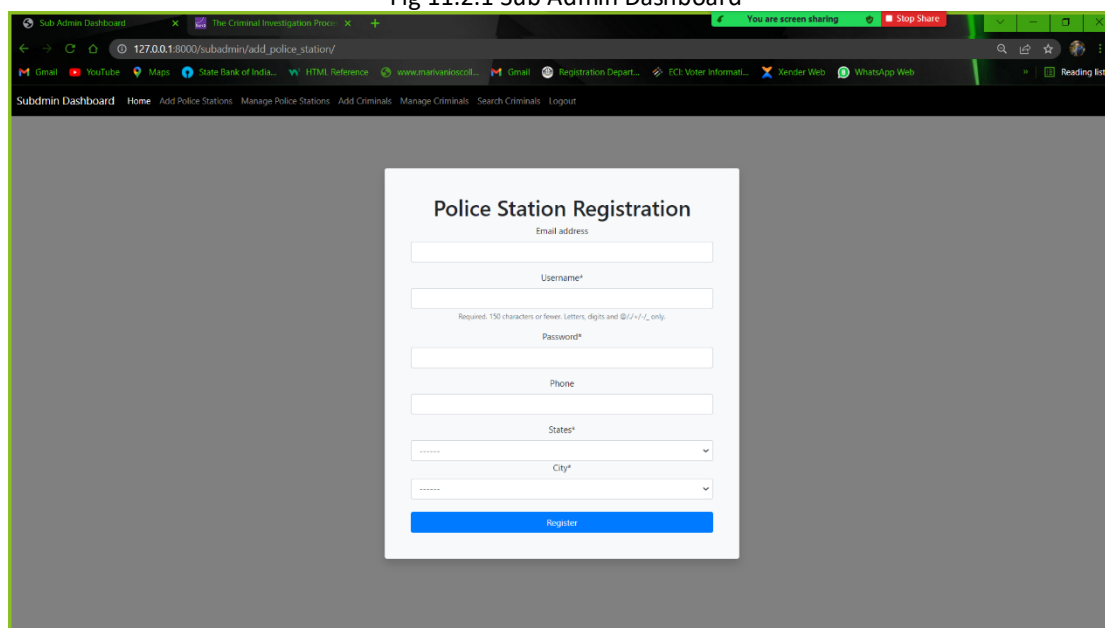


Fig 11.2.2 Add police station

## FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN

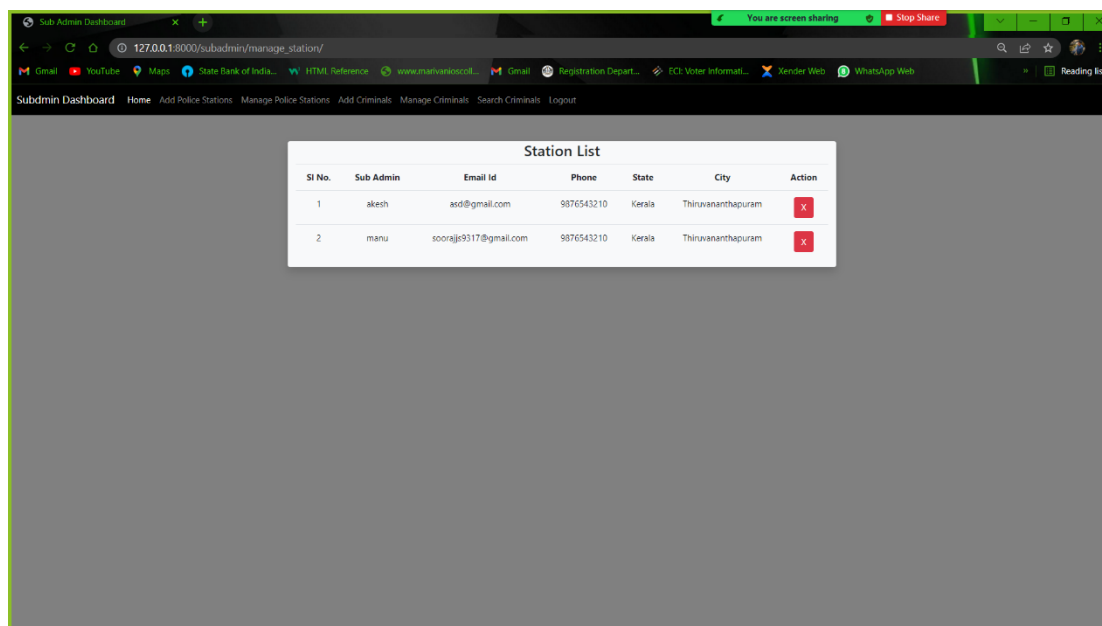


Fig 11.2.3 Manage police station

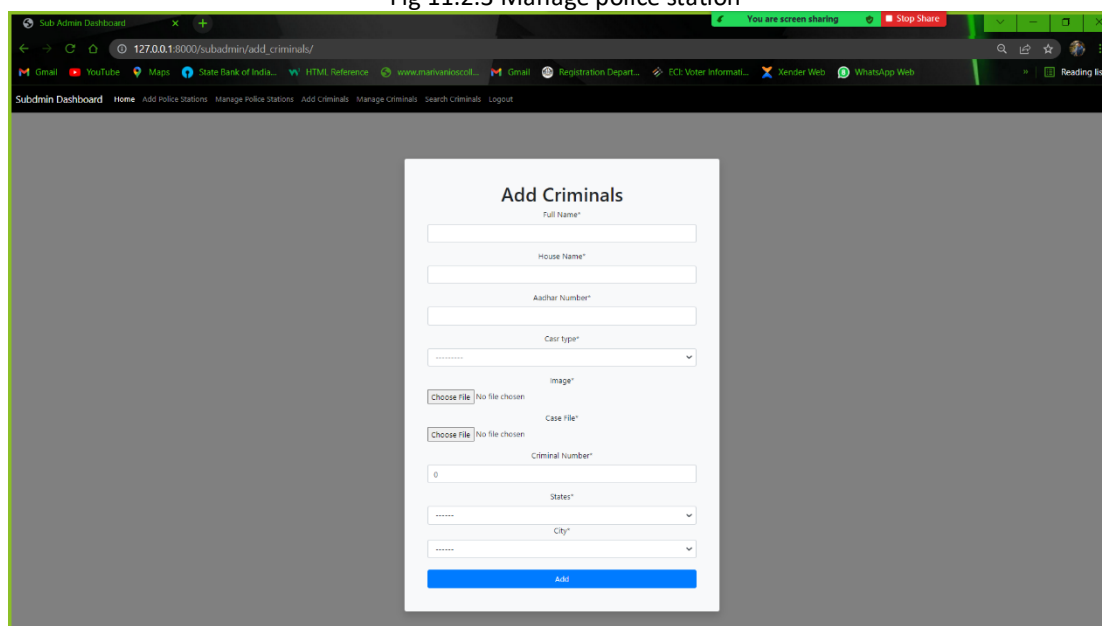


Fig 11.2.4 Add criminals

## FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN

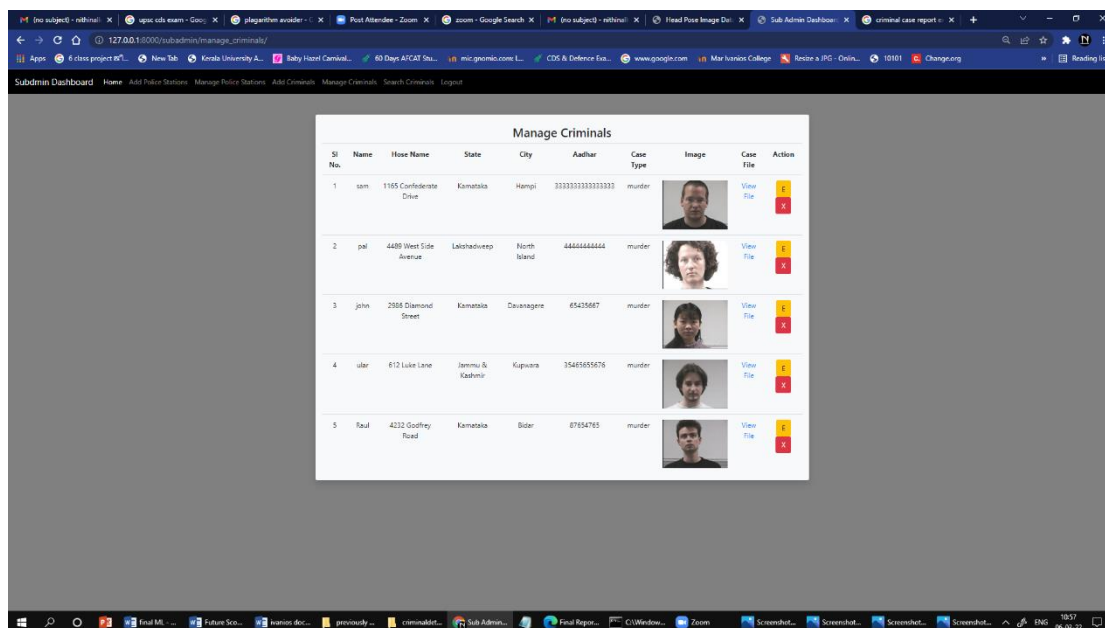


Fig 11.2.6 Search Criminal

## FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN

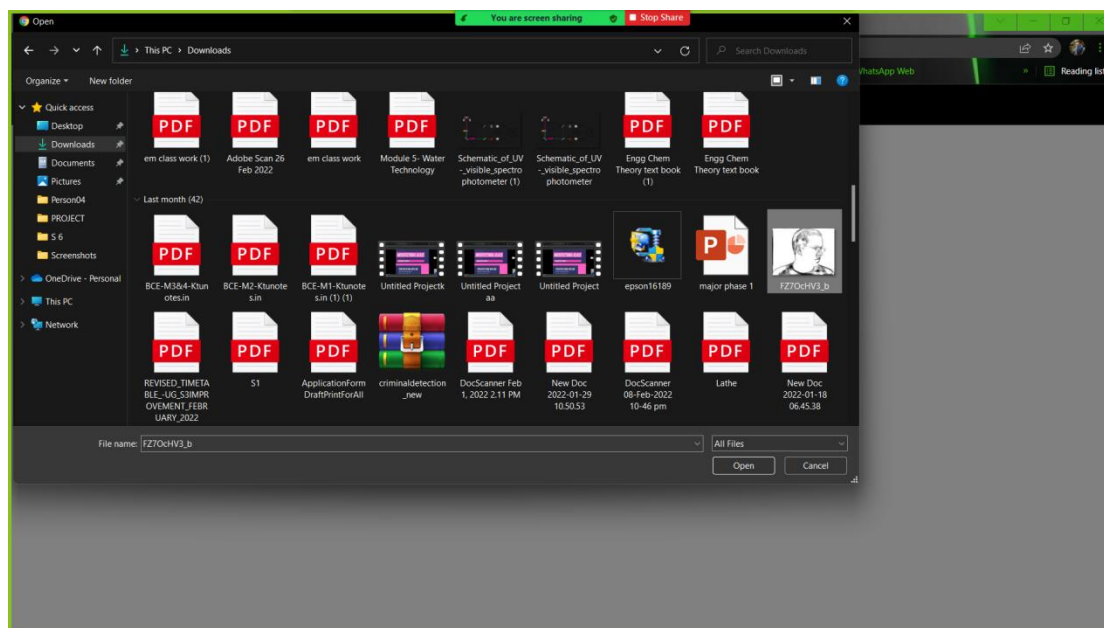


Fig 11.2.7 uploading a sketch

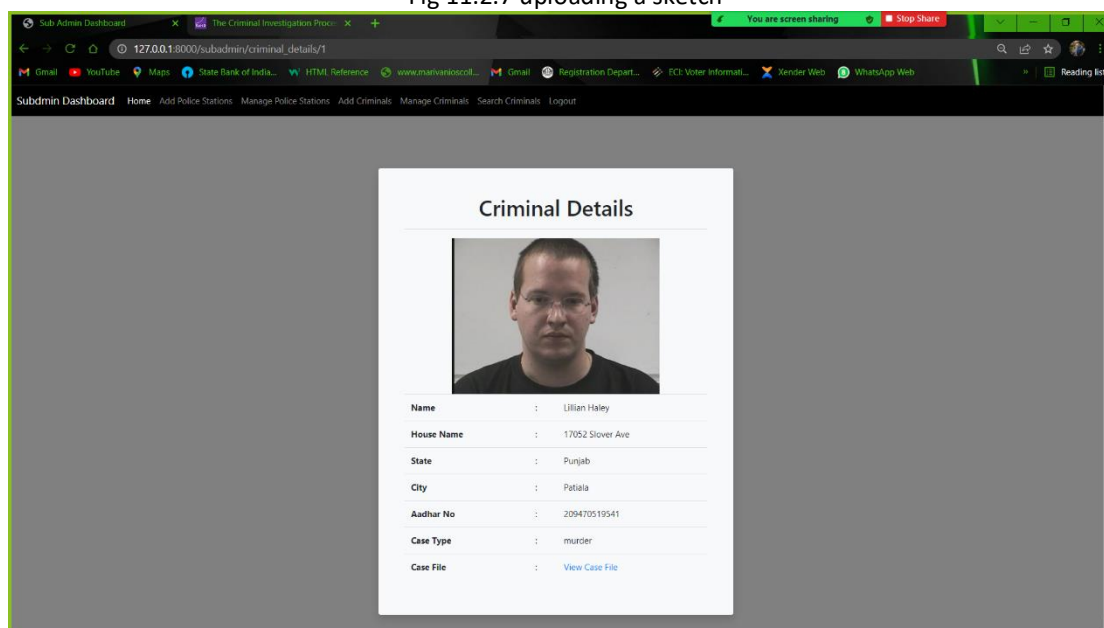


Fig 11.2.8 face sketch matched to the database record

## FORENSIC FACE SKETCH RECOGNITION USING DEEP CNN

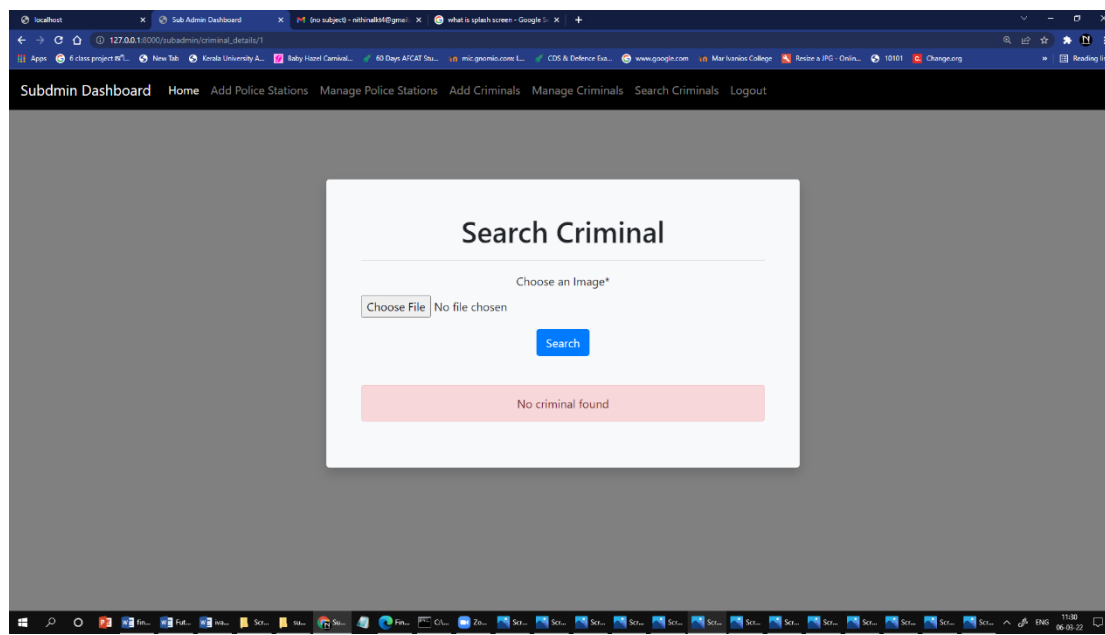


Fig 11.2.9 face sketch not matched to the database record

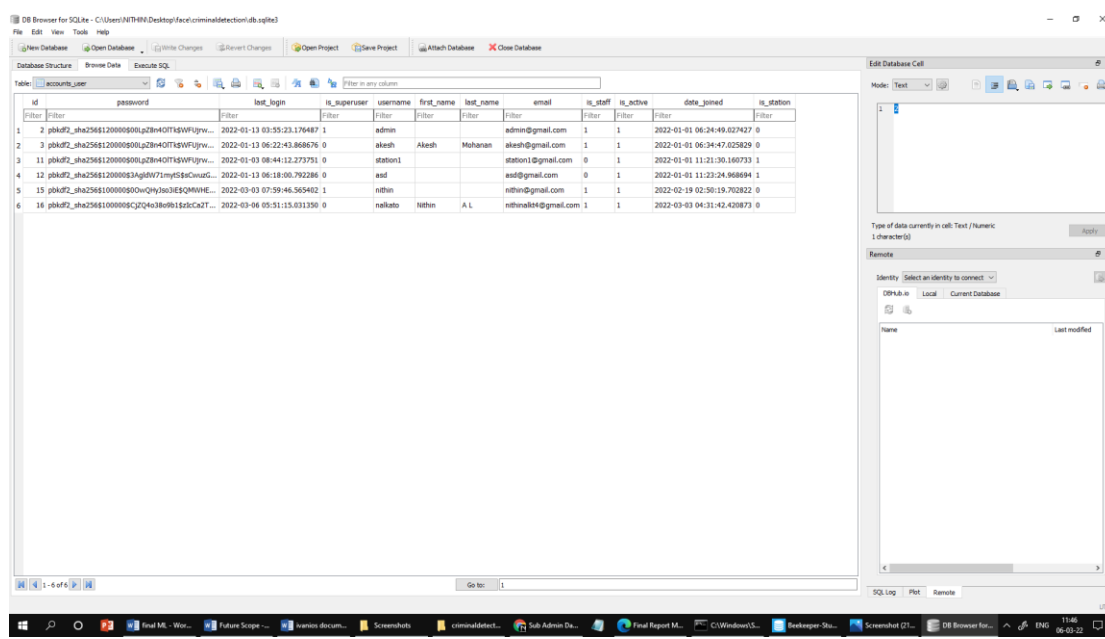


Fig 11.2.10 Database with user credentials  
(The User Credentials Management Dashboard)

SQL Log

Show SQL submitted by Application

Clear

```
1 PRAGMA foreign_keys = '1';
2 PRAGMA database_list;
3 SELECT type,name,sql,tbl_name FROM "main".sqlite_master;
4 PRAGMA encoding;
5 PRAGMA database_list;
6 SELECT type,name,sql,tbl_name FROM "main".sqlite_master;
7 SELECT COUNT(*) FROM "main"."accounts_casetype"
8 SELECT "_rowid_",* FROM "main"."accounts_casetype" LIMIT 0, 49999;
9 PRAGMA database_list;
10 SELECT type,name,sql,tbl_name FROM "main".sqlite_master;
11 SELECT COUNT(*) FROM "main"."accounts_user"
12 SELECT "_rowid_",* FROM "main"."accounts_user" LIMIT 0, 49999;
13
```

Fig 11.2.11 Database user credentials schema  
(The User Credentials Schema)

[illegible]

Fig 11.2.12 Database schema

## 12. BIBLIOGRAPHY

### 12.1 References

[1] Wang, Xiaogang & Tang, Xiaoou. (2009). *Face Photo-Sketch Synthesis and Recognition*. *IEEE transactions on pattern analysis and machine intelligence*. 31. 1955-67. 10.1109/TPAMI.2008.222.

[2] Klare, Brendan & Jain, Anil. (2010). *Sketch to Photo Matching: A Feature-based Approach*. *Proceedings of SPIE - The International Society for Optical Engineering*. 10.1117/12.849821.

[3] Lahlali, S.E. & Sadiq, Abdelalim & Mbarki, Samir. (2015). *A review of face sketch recognition systems*. 81. 255-265.

[4] *Forensic Face Sketch Construction and Recognition* Asst. Prof. Abhijit Patil [1] , Akash Sahu [2] , Jyoti Sah [3] , Supriya Sarvade [4] , Saurabh Vadekar [5]

[5] <http://crowley-coutaz.fr/Head%20Pose%20Image%20Database.html>