**A PROJECT REPORT ON**

# FACE RECOGNITION USING OPENCV AND CNN

**ALGORITHM**

**Submitted to**

**JAWAHARLALNEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA**

***For the partial fulfillment for the award of the degree of***

# BACHELOR OF TECHNOLOGY

## Submitted By

**Y. DEEPU SAI VINAY (19X41A0571)**

**L.AJAY KUMAR (19X41A0586)**

**M.GHANA SYAMALA (19X41A0588)**

**M.NAVEEN SAI (19X41A0590)**

*Under the esteemed guidance*

**J. SIVA NAGA JYOTHI Associate Professor, Department of CSE**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

# S.R.K INSTITUE OF TECHNOLOGY

(Approved By AICTE, New Delhi & Affiliated To JNTU Kakinada)

(An Iso 9001:2015 Certified Institution & Accredited by NAAC with ‘A’ Grade)

Enikepadu, Vijayawada -521108

**APRIL 2023**

# S.R.K INSTITUE OF TECHNOLOGY

(Approved By AICTE, New Delhi & Affiliated To JNTU Kakinada)

(An Iso 9001:2015 Certified Institution & Accredited by NAAC with ‘A’ Grade)

Enikepadu, Vijayawada -521108

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



# CERTIFICATE

This is to certify that the Project Report entitled “**FACE RECOGNITION USING OPEN CV AND CNN ALGORITHM’’** is the Bonafide work of Y. Deepu Sai Vinay(19X41A0571), L. Ajay kumar(19X41A0586), M. Ghana Syamala(19X41A0588), M. Naveen Sai(19X41A0590) in partial fulfilment of the requirements for the award of the graduate degree of BACHELOR OF TECHNOLOGY during the academic year 2019-2023. This work has carried out under our supervision and guidance***.***

**Ms. J. Siva Naga Jyothi Dr. B. Asha Latha**

(PROJECT GUIDE) (HEAD OF THE DEPARTMENT)

**Signature of the External Examiner**

# DECLARATION

We Y. Deepu Sai Vinay, L. Ajay Kumar, M. Ghana Syamala, M. Naveen Sai hereby declare that the project report entitled “**FACE RECOGNITION USINGOPENCV AND CNN ALGORITHM’**” is an original work done in the Department of Computer Science & Engineering, SRK Institute Of Technology, Enikepadu, Vijayawada, during the academic year 2022-2023, in partial fulfillment for the award of the Degree of Bachelor Of Technology in Computer Science & Engineering. We assure you that this project is not submitted to any other College or University.

### Roll No Name of the Student Signature

**19X41A0571 Y.DEEPU SAI VINAY**

**19X41A0586 L.AJAY KUMAR**

**19X41A0588 M.GHANA SYAMALA**

**19X41A0590 M.NAVEEN SAI**

# ACKNOWLEDGEMENT

Firstly we would like to convey our heart full thanks to the Almighty for the blessings on us to carry out this project work without any disruption.

We are extremely thankful to **J. Siva Naga Jyothi**, our guide throughout the project. We also thank her for most independence and freedom throughout the given to us during various phases of the project.

We are also thankful for our project coordinator **Dr. K. Chaitanya**, for their valuable guidance which helped us to bring this project successfully.

We are very much grateful to **Dr. B. Asha Latha**, H.O.D of C.S.E Department, for her valuable guidance which helped us to bring out this project successfully. Her wise approach made us learn the minute details of the subject. Her matured and patient guidance paved away for completing our project with a sense of satisfaction and pleasure.

We are greatly thankful to our principal **Dr. M. Ekambaram Naidu** for his kind support and facilities provided at our campus which helped us to bring out this project successfully.

Finally, we would like to convey our heart full thanks to our Technical Staff, for their guidance and support in every step of this project. We convey our sincere thanks to all the faculty and friends who directly or indirectly helped us with the successful completion of this project

### PROJECT ASSOCIATES

Y.DEEPU SAI VINAY 19X41A0571 L.AJAY KUMAR 19X41A0586 M.GHANA SYAMALA 19X41A0588 M.NAVEEN SAI 19X41A0590

### CONTENTS

|  |  |
| --- | --- |
| **TITLE** | **Page. No** |
| List of Figures | i |
| Abstract | 1 |
| **Chapter1 :INTRODUCTION**   * 1. :Intelligent System   2. :Convolutional Neural Network   3. :Convolutional Neural Network Basic Structure   4. :Methodology and Algorithm   1.4.1 : LBPH Face Recognizer   * + 1. : Mobile Net   1.4.3 : CNN Layers | **2**  2  3  3  4  5  6  7 |
| **Chapter2: LITERATURE SURVEY** | **10** |
| **Chapter3 :SYSTEM ANALYSIS** | **14** |
| 3.1 :Existing System | 14 |
| 3.1.1 :Disadvantage of Existing System | 14 |
| 3.2 :Proposed System | 14 |
| 3.2.1 :Advantages of Proposed System | 15 |
| 3.2.2 :Modules | 15 |
| 3.3 :Feasibility study | 16 |
| 3.3.1 :Economic Feasibility | 16 |
| 3.3.2 :Operational Feasibility | 16 |
| 3.3.3 :Technical Feasibility | 17 |
| **Chapter4:SYSTEM SPECIFICATIONS** | **18** |
| 4.1:Hardware Requirement | 18 |
| 4.2 : Software Requirement | 18 |
| **Chapter5:SYSTEM DESIGN**  5.1:System Architecture | **19**  19 |
| 5.2 :UML Diagram | 20 |
| 5.3 :Use Case Diagram | 21 |
| 5.4 :Class Diagram | 22 |
| 5.5 :Sequence Diagram | 23 |
| 5.6 :Collaboration Diagram | 23 |
| 5.7: Deployment Diagram | 24 |

**Chapter 6: SYSTEM IMPLEMENTATON**

* 1. :Sample Code 25
  2. [:Input and Output Screens 33](#_bookmark1)
  3. [:System Testing 36](#_bookmark2)
  4. : Types of Tests 36
     1. [:Unit testing 36](#_bookmark3)
     2. :Integrational Testing 37
     3. : Function Testing 37
     4. [: System Testing 37](#_bookmark4)
     5. [: White Box Testing 37](#_bookmark5)
     6. [: Black Box Testing 38](#_bookmark6)
     7. [: Acceptance Testing 38](#_bookmark7)
     8. [: Testing Results 38](#_bookmark8)
  5. [: Test Cases 39](#_bookmark9)

[Chapter7 : TECHNOLOGY DESCRIPTION 40](#_bookmark10)

[Chapter 8 : CONCLUSION 44](#_bookmark11)

[FUTURE SCOPE 45](#_bookmark12)

REFERENCES 47

[SOFTWARES INCLUED 48](#_bookmark13)

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Name of the Figure** | **Page No.** |
| Figure 1.3.1 | Neural Networks | 4 |
| Figure 1.4.3 | Convolutional Image Figure | 7 |
| Figure 1.4.3 | Input Image and Convolutional Layers | 8 |
| Figure 1.4.4 | Convolutional Neural Scan Images | 9 |
| Figure 5.0 | System Architecture | 19 |
| Figure 5.3 | Use Case Diagram | 21 |
| Figure 5.4 | Class Diagram | 22 |
| Figure 5.5 | Sequence Diagram | 23 |
| Figure 5.6 | Collabration Diagram | 23 |
| Figure 5.7 | Deployment Diagram | 24 |
| Figure 6.2.1 | User Interface | 33 |
| Figure 6.2.2 | Input Page | 33 |
| Figure 6.2.3  Figure 6.2.4  Figure 6.2.5  Figure 6.2.6 | Training page  Image Trained  Output Screen  Error Results Screen | 34  34  35  35 |

**ABSTRACT**

Human face is the significant characteristic to identify a person. Everyone has their own unique face even for twins. Thus, a face recognition and identification are required to distinguish each other. Hence, we are proposing a method that which will able to recognize the deep learning based CNN model. Here we are collecting the dataset of different faces. Once after preprocessing it we train the data with the CNN algorithm. After training, we will test the results using the OpenCV and also can upload the image for recognition of faces.

**INDEX TERMS:** Convolutional neural network, Face recognition, OpenCV.

# CHAPTER 1

**INTRODUCTION**

**1.1Intelligent systems:**

It appears more and more in people's lives, and often need to be identified when using intelligent systems. Traditional methods of identification mainly identify individuals with some personal characteristics, such as identity documents, such as documents and keys, which have obvious shortcomings. They are easily forgotten, lost or faked. If you use some of the personal characteristics to identify the effect will be quite good, such as: face recognition, fingerprinting and so on. In terms of algorithms, there are sharing parameters between the convolution layer and the convolution layer of CNN. The advantage of this is that the memory requirements are reduced, and the number of parameters to be trained is correspondingly reduced. The performance of the algorithm is therefore improved. At the same time, in other machine learning algorithms, the pictures need us to perform preprocessing or feature extraction. However, we rarely need to do these operations when using CNN for image processing. This is something other machine learning algorithms cannot do. There are also some shortcomings in depth learning. One of them is that it requires a lot of samples to construct a depth model, which limits the application of this algorithm. Today, very good results have been achieved in the field of face recognition and license plate character recognition, so this topic will do some simple research on CNN-based face recognition technology.

Because of advancements in AI, particularly the huge advancements in Generative Adversarial Networks, the current vision is untrustworthy. Even regular consumers without a background in professional photography can take high-quality inside images with the aid of Generative Adversarial Networks like Progressive Generative Adversarial Network, Style Generative Adversarial Network, and Star Generative Adversarial Network. Everything else remains the same, with the exception of the increasing use of unified interfaces with Generative Adversarial Networks. Moreover, these numerous phoney faces can easily fool even our own people. Deep learning has produced substantial advancements in computer vision, image processing, and the use of general languages over the last few years. Deep neural networks have occasionally done tasks better than humans. In a GAN (Generative Adversarial Network), two neural tissues—generators M and discriminators— compete to create better results, such as different sources of information.

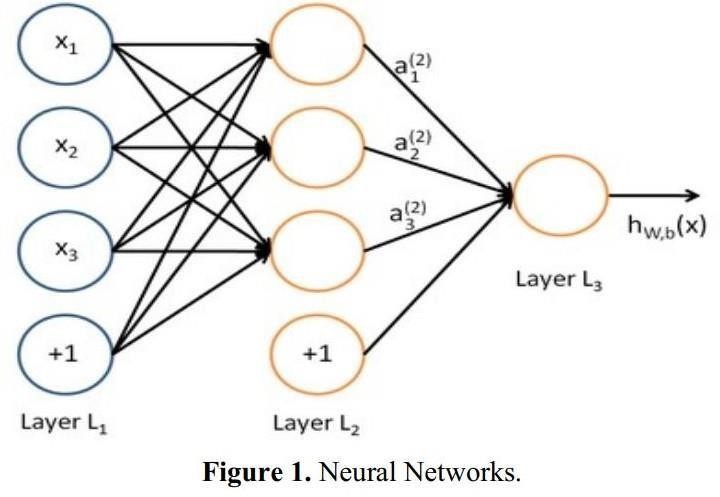
Generative Adversarial Networks are frequently used to produce fresh, insightful photos as well as to improve existing images. In either case, Generative Adversarial Networks and other AI algorithms may be abused to produce false data that misleads users. . For instance, Generative Adversarial Networks' false images can deceive both AI classifiers and humans. Also, cutting-edge photo editing software like Adobe Photoshop enables you to fix complex informational photographs and produce breath-taking new images.

**1.2 CONVOLUTIONAL NEURAL NETWORK**

**Convolutional neural network introduction:** With the development of convolutional neural networks, the achievements made in various competitions are getting better and better, making it the focus of research. In order to improve the training performance of the forward BP algorithm, an effective method is to reduce the number of learning parameters. This can be done by convolution of the spatial relationship of the neural network. Convolution neural network, the network structure is proposed, it minimizes the input data pretreatment. In the structure of convolution neural network, the input data is input from the initial input layer, through each layer processing, and then into the other hierarchy, each layer has a convolution kernel to obtain the most significant data characteristics. The previously mentioned obvious features such as translation, rotation and the like can be obtained by this method.

**1.3 Convolution neural network basic structure**: Neural network can be divided into two kinds, biological neural network is one of them, and artificial neural network is another kind. Here mainly introduces artificial neural network. An artificial neural network is a data model that processes information and is similar in structure to the synaptic connections in the brain. Neural network is composed of many neurons; the output of the previous neuron can be used as the input of the latter neuron. The corresponding formula is as follows:

This unit is also called Logistic regression model. When many neurons are linked together, and when they were layered, the structure can now be called a neural network model. Figure 1 shows a neural network with hidden layers.



**Fig** : 1.3.1 Neural Networks

In this neural network, X1, X2, X3 are the input of the neural network. +1 is the offset node, also known as the intercept term. The leftmost column of this neural network model is the input layer of the neural network, the rightmost column of which is the output layer of the neural network. The middle layer of the network model is a hidden layer, which is fully connected between the input layer and the output layer. The values of all the nodes in the network model cannot be seen in the training sample set. By observing this neural network model, we can see that the model contains a total of 3 input units, 3 hidden units and 1 output unit. Now, use nl to represent the number of layers in the neural network, and the number of layers in this neural network is 3. Now mark each layer, the first layer can be expressed by Ll, then the output layer of the neural network L1, its output layer is Lnl, in this neural network, the following parameters exist:



is the connection parameter between the jth cell of layer 1 and the ith cell of layer l+1, ambit l is the offset of the ith cell of layer 1+1. In this neural network model, set a (l) to

represent the output value of the first few cells in this layer. Let l denote this layer and i the first few cells in this layer. 3 1234567890 ‘’“” 2nd International Symposium on Resource Exploration and Environmental Science IOP Publishing IOP Conf. Series: Earth and Environmental Science 170 (2018) 032110 do i :10.1088/1755- 1315/170/3/032110 Given that the set of parameters W and b have been given, we can use the formula hw,b(x) to calculate the output of this neural network. The calculation of forward propagation is as shown in equation (3). Neural network training methods and Logistic regression model is similar, but due to the multi-layered neural network, but also the need for gradient descent chain derivation rule.

CNN Model Construction and Training 3.1 . CNN model At present, the typical architecture of neural network is divided into the following categories: LeNet5, AlexNet, ZFNet, GooLeNet, and VGGNet, the following will LeNet5 architecture for a detailed analysis. LeNet5 is a CNN classic structure that existed long ago, and it is mainly used in the recognition of handwritten fonts. It contains a total of seven layers of structure, except for the input layer, each of the other has training parameters, and each layer contains a plurality of Feature Maps, we can extract the input features through a convolution kernel.

**1.4 METHODOLOGY AND ALGORITHM**

**1.4.1 Mobile net:**

Mobile Net is a deep convolution neural network (CNN) architecture that is designed for efficient mobile devices, such as smart phones and tablets. It was developed by researchers at Google in 2017 and has since become a popular architecture for mobile vision applications. The main goal of Mobile Net is to reduce the number of parameters and computations required by a CNN while maintaining high accuracy. This is achieved through the use of depth wise separable convolutions, which split the standard convolutional filters into two separate operations: a depth wise convolution that filters each input channel separately, followed by a point wise convolution that combines the output of the depth wise convolution into a new feature map. By using depth wise separable convolutions, Mobile Net significantly reduces the number of parameters and computations required compared to traditional CNN architectures while maintaining high accuracy on image classification tasks. This makes it well-suited for resource-constrained environments such as mobile devices. Mobile Net has several variations, including Mobile Net V1,

Mobile Net V2, and Mobile Net V3. Each version has its own unique features and improvements, such as better accuracy, reduced computation, and improved efficiency. Overall, Mobile Net is an efficient and effective deep learning architecture for mobile vision applications.

### 1.4.2 LBPH face recognizer:

**working of LBPH face recognizer:**

The Local Binary Patterns Histograms (LBPH) is a face recognition algorithm that works by extracting features from an input face image and comparing them to a database of known faces to identify the person in the image. The LBPH algorithm follows these steps: **1.Preprocessing:** The input face image is first preprocessed to remove any noise and enhance the contrast. This can be done using techniques such as histogram equalization or Gaussian smoothing.

1. **Feature Extraction:** The preprocessed image is divided into a grid of cells, and a Local Binary Pattern (LBP) histogram is computed for each cell. LBP is a texture descriptor that captures the local patterns of an image by comparing the intensity of each pixel with its neighboring pixels. The LBP histogram for a cell summarizes the distribution of LBP codes within that cell.
2. **Face Recognition:** The LBP histograms for each cell are concatenated to form a feature vector for the input image. This feature vector is then compared to the feature vectors in the database using a distance metric such as Euclidean distance or cosine similarity. The closest match is considered to be the identity of the person in the input image.

The LBPH algorithm is simple yet effective in recognizing faces, especially in scenarios where the lighting and pose variations are not too extreme. However, it may struggle in cases where the face is partially occluded or if there is significant variation in the face appearance due to aging, facial hair, or makeup. CNN algorithm consists of 4 layers: Input layer, Convolution Layer, pooling layer, Flatten layer and dense layer.

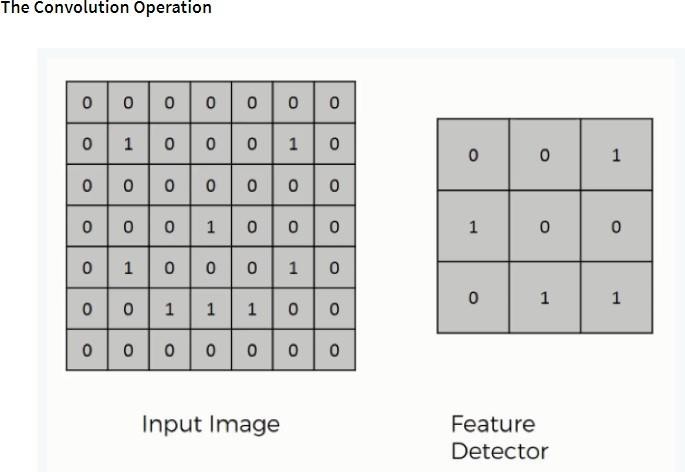
In input layer we consider images as input. In Convolution layer, we convert image into matric format. Here matrix size is 1024 X 1024 (rows X columns). In the pooling layer the numerical values will be stored. To change the numerical data to binary data, we use machine learning algorithm named SoftMax (supervised learning algorithm). In SoftMax layer we will convert the numerical data to binary. In flatten layer and dense the classes of total dataset (2 types) is stored which will be in the binary data format.

We use fit generator method for saving the data in the form of .h5. Here model is a format for storing the binary data.

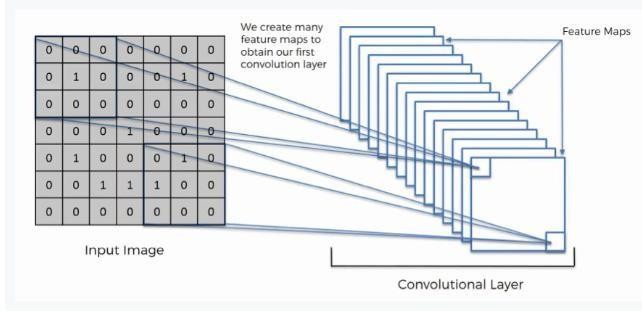
### 1.4.3 Convolutional Neural Network (CNN)

**Step 1: convolutional operation**

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.



**Fig**:1.4.3 Convolution Images



**Fig** :1.4.3 Input image and Covolutional Layer

### Step (1b): ReLu Layer

The second part of this step will involve the Rectified Linear Unit or Relook. We will cover Relook layers and explore how linearity functions in the context of Convolutional Neural Networks.

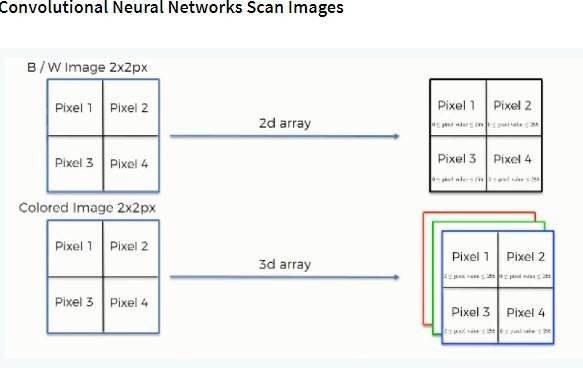
Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve your skills.

Fig : 1.4.4 Convolutional Neural Scan Images

**Step 2: Pooling Layer**

In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

### Step 3: Flattening

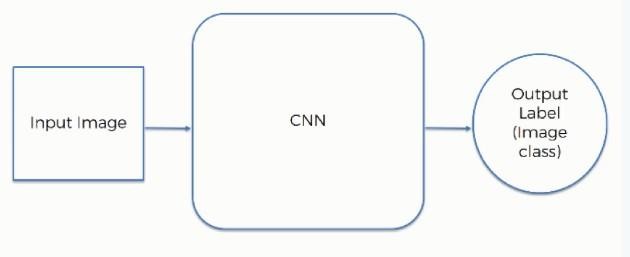
This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolutional Neural Networks.

### Step 4: Full Connection

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolutional Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.

### Summary

In the end, we'll wrap everything up and give a quick recap of the concept covered in the section. If you feel like it will do you any benefit (and it probably will), you should check out the extra tutorial in which Soft ax and Cross-Entropy are covered. It's not mandatory for the course, but you will likely come across these concepts when working with Convolutional Neural Networks and it will do you a lot of good to be familiar with them.



# CHAPTER 2

**LITERATURE REVIEW**

Literature [survey](http://www.blurtit.com/q876299.html) is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the [programmers](http://www.blurtit.com/q876299.html) start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from [book](http://www.blurtit.com/q876299.html) or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

1. **Yao L S, Xu G M, Zhap F. Facial Expression Recognition Based on CNN Local Feature Fusion[J]. Laser and Optoelectronics Progress, 2020, 57(03): 032501.**

With the transition of facial expression recognition (FER) from laboratory-controlled to challenging in-the-wild conditions and the recent success of deep learning techniques in various fields, deep neural networks have increasingly been leveraged to For the state of the art in deep FER, we review existing novel deep neural networks and related training strategies that are designed for FER based on both static images and dynamic image sequences, and discuss their advantages and limitations. Competitive performances on widely used benchmarks are also summarized in this section. We then extend our survey to additional related issues a finally, we review the remaining challenges and corresponding opportunities in this field as well as future directions for the design of robust deep FER systems. And application scenarios.

### Zhang Chen. Research on some key technologies of facial micro- expression recognition [D]. 2019.

Facial Expression Recognition (FER) can be widely applied to various research areas, such as mental diseases diagnosis and human social/physiological interaction detection. With the emerging advanced technologies in hardware and sensors, FER systems have been developed to support real-world application scenes, instead of laboratory environments. Although the laboratory-controlled FER systems achieve very high accuracy, around 97%, the technical transferring from the laboratory to real-

world applications faces a great barrier of very low accuracy, approximately 50%. In this survey, we comprehensively discuss three significant challenges in the unconstrained real-world environments, such as illumination variation, head pose, and subject-dependence, which may not be resolved by only analyzing images/videos in the FER system. We focus on those sensors that may provide extra information and help the FER systems to detect emotion in both static images and video sequences. We introduce three categories of sensors that may help improve the accuracy and reliability of an expression recognition system by tackling the challenges mentioned above in pure image/video processing. The first group is detailed-face sensors, which detect a small dynamic change of a face component, such as eye-trackers, which may help differentiate the background noise and the feature of faces. The second is non-visual sensors, such as audio, depth, and EEG sensors, which provide extra information in addition to visual dimension and improve the recognition reliability for example in illumination variation and position shift situation. The last is target-focused sensors, such as infrared thermal sensors, which can facilitate the FER systems to filter useless visual contents and may help resist illumination variation. Also, we discuss the methods of fusing different inputs obtained from multimodal sensors in an emotion system. We comparatively review the most prominent multimodal emotional expression recognition approaches and point out their advantages and limitations. We briefly introduce the benchmark data sets related to FER systems for each category of sensors and extend our surveyto the open challenges and issues. Meanwhile, we design a framework of an expression recognition system, which uses multimodal sensor data (provided by the three categories of sensors) to provide complete information about emotions to assist the pure face image/video analysis. We theoretically analyze the feasibility and achievability of our new expression recognition system, especially for the use in the wild environment, and point out the future directions to design an efficient, emotional expression recognition system.

### Xu Linlin, Zhang Shumei, Zhao Junli. Expression recognition algorithm for constructing parallel convolutional neural networks [J]. Journal of Image and Graphics,2019, 24 (02): 0227-0236.

Facial expression recognition is one of the very important research topics in computer vision. Studies on nonverbal communication have shown that 55% of intentional information is conveyed through facial expressions

In this paper we have proposed a parallel Convolutional Neural Network (CNN) structure for detection of expression from frontal faces. The CNNs are trained on two most important sub facial patches. The overall feature vector will be the features concatenated from the parallel models. We have experimentally found applying such a strategy provides better results than the models which take the entire facial image. We have also compared our performance with other benchmark CNN structures like AlexNet and VGG16.

### Li Siquan, Zhang Xuanxiong. Research on Facial Expression Recognition Based on Convolutional Neural Networks [J]. Journal of Software, 2018, v.17; No.183 (01): 32-35

Facial expression recognition has been an active research area over the past few decades, and it is still challenging due to the high intra-class variation. Traditional approaches for this problem rely on hand-crafted features such as SIFT, HOG and LBP, followed by a classifier trained on a database of images or videos. Most of these works perform reasonably well on datasets of images captured in a controlled condition, but fail to perform as good on more challenging datasets with more image variation and partial faces. In recent years, several works proposed an end-to-end framework for facial expression recognition, using deep learning models. Despite the better performance of these works, there still seems to be a great room for improvement. In this work, we propose a deep learning approach based on attentional convolutional network, which is able to focus on important parts of the face, and achieves significant improvement over previous models on multiple datasets, including FER-2013, CK+, FERG, and JAFFE. We also use a visualization technique which is able to find important face regions for detecting different emotions, based on the classifier’s output. Through experimental results, we show that different emotions seems to be sensitive to different parts of the face.

### Zhai Junkui, Liu Jian. Research on Transfer Convolutional Neural Network for Facial Expression Recognition [J]. Signal Processing, 2018, 34 (6): 729-738.

In recent years, with the development of artificial intelligence and human–computer interaction, more attention has been paid to the recognition and analysis of facial expressions. Despite much great success, there are a lot of unsatisfying problems,

FACE RECOGNITION USING OPENCV AND CNN ALGORITHM

because facial expressions are subtle and complex. Hence, facial expression recognition is still a challenging problem. In most papers, the entire face image is often chosen as the input information. In our daily life, people can perceive other’s current emotions only by several facial components (such as eye, mouth and nose), and other areas of the face (such as hair, skin tone, ears, etc.) play a smaller role in determining one’s emotion. If the entire face image is used as the only input information, the system will produce some unnecessary information and miss some important information in the process of feature extraction. To solve the above problem, this paper proposes a method that combines multiple sub-regions and the entire face image by weighting, which can capture more important feature information that is conducive to improving the recognition accuracy. Our proposed method was evaluated based on four well-known publicly available facial expression databases: JAFFE, CK+, FER2013 and SFEW. The new method showed better performance than most state-of-the-art methods.

**CHAPTER 3**

# SYSTEM ANALYSIS

### EXISTING SYSTEM

Face live ness detection is a traditional method for real vs. fake face detection. It involves capturing several images of the user's face from different angles and analyzing themto detect spoofing attacks. This method relies on various cues, such as eye blinking, head movements, and facial expressions, to differentiate between real and fake faces. The choice of system will depend on the specific needs and context of the application. It's important to consider factors such as accuracy, convenience, user experience, and cost when selecting a real vs. fake face recognition system.

### DISADVANTAGES OF EXISTING SYSTEM

* + - * Passive liveness detection
      * Multi-factor authentication
      * Artificial intelligence-based face recognition.

### PROPOSED SYSTEM

Real or fake face detection is a complex problem, but with a well-designed system that leverages deep learning and advanced image processing techniques, it is possible to reliably identify real faces from fake ones. Train a binary classification model using the extracted features and labels indicating whether each image is real or fake. The model should be trained on a large and diverse dataset to ensure that itcan generalize well to unseen data. In this, here, we are using LBPH face recognizer to detect the faces and CNN Algorithm to recognize. And it shows whether it is real face or fake face and gives the result.

### ADVANTAGES OF PROPOSED SYSTEM

* + - * High Accuracy
      * Performance is high
      * Time minimizing

### MODULES

**1.Use**: First, person will enter their detail and, it will store the data in to data base and then it will start screening to capture image.

**2.Applicant/Trainer:** Applicant will load the dataset then he will preprocess the

dataset and train the model of algorithm.

**3.Create Dataset:** Here we are searching the dataset with the help of internet. For example, we consider the kaggle website mainly for the datasets. If data is unavailable in the website we our self-create our data with a technique called data augmentation.

**4.Data Pre-Processing:** Data pre-processing is a data mining technique which is

used to transform the raw data in a useful and efficient format.

1. Missing Data: This situation arises when some data is missing in the data. It can be handled in various ways.
2. Noisy Data: Noisy data is a meaningless data that can’t be interpreted by machines. It can be generated due to faulty

data collection, data entry errors etc.

**5.Training (CNN with Mobile Net):** We are using the pre-processed training dataset to train our model using CNN algorithm. In the pooling layer the numerical values will be stored. To change the numerical data to binary data, we use machine learning algorithm named SoftMax (supervised learning algorithm). In SoftMax layer we will convert the numerical data to binary.

**6.Classify the face:** The system will classify and detect the face of the person.

### FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

* Economical Feasibility
* Social Feasibility
* Technical Feasibility

### ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

# CHAPTER 4 SYSTEM SPECIFICATIONS

### HARDWARE REQUIREMENTS

**Processor - I3/Intel Processor**

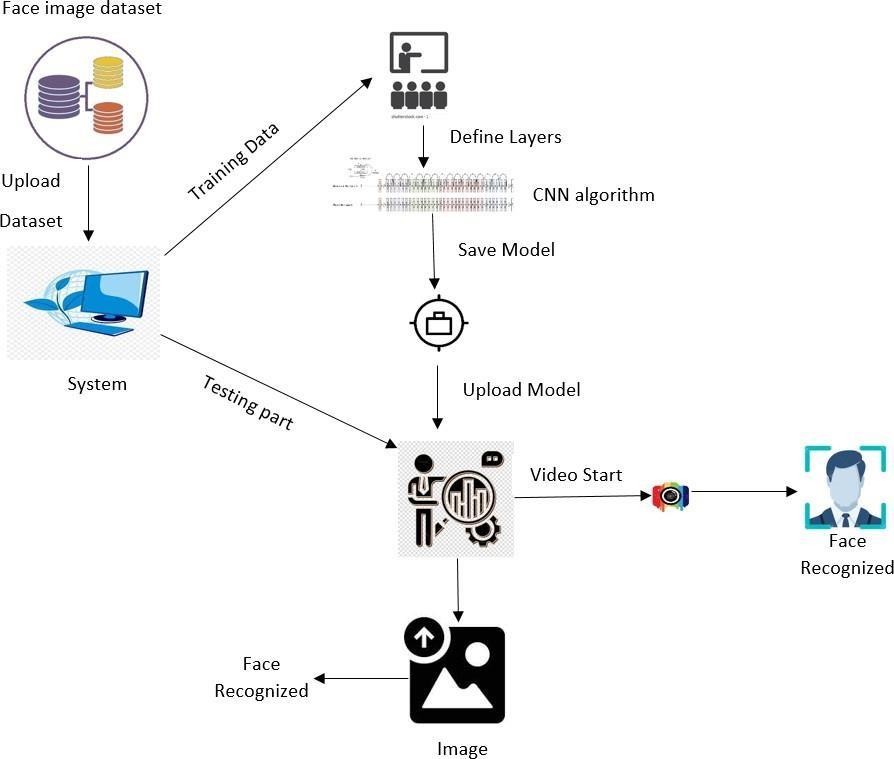
* + - Hard Disk -160GB
    - Key Board - Standard Windows Keyboard
    - Mouse - Two or Three Button Mouse
    - Monitor - SVGA
    - RAM - 8Gb

### SOFTWARE REQUIREMENTS

* + - Operating System : Windows 7/8/10
    - IDE : Pycharm
    - Libraries Used : Numpy, CV2, Os, Face-recognition
    - Technology : Python 3.6+
    - Camera : Webcam

# CHAPTER 5 SYSTEM DESIGN

### 5.1 SYSTEM ARCHITECTURE



**Fig 5. 1: System Architecture**

### 5.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object- oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added

to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

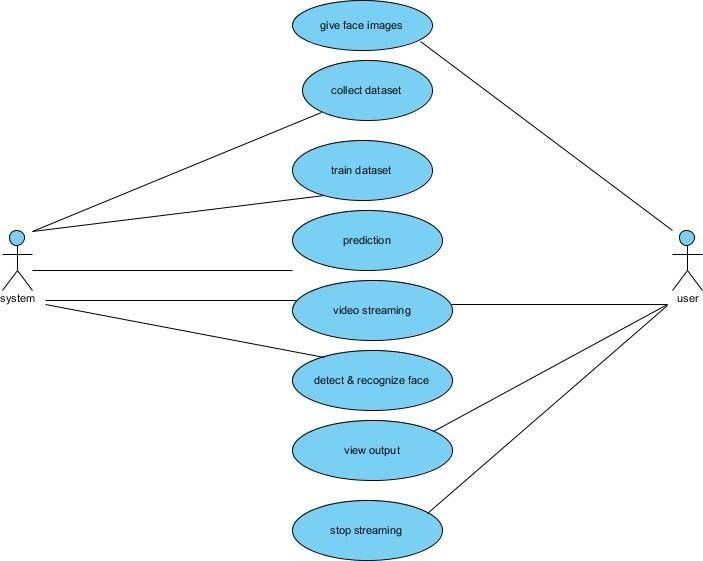
1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

### 5.3 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

▶ Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

▶ The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**Fig 5.3:Use Case Diagram**

### 5.4 CLASS DIAGRAM

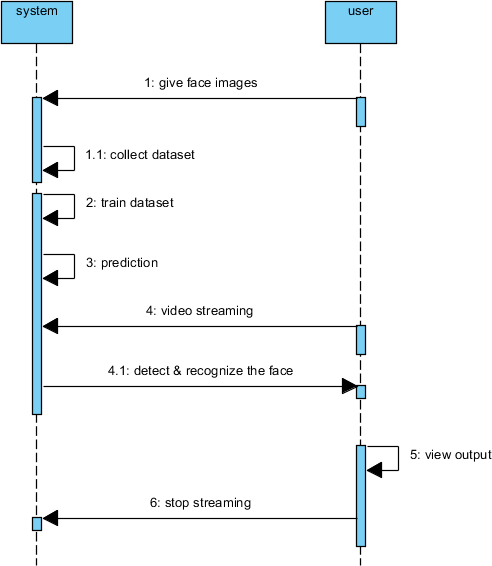
In software engineering, a class diagram in the Unified Modeling Language (UML) isa type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**Fig 5.4: Class Diagram**

### 5.5 SEQUENCE DIAGRAM

* + - A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.
    - It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



**Fig 5.5: Sequence Diagram**

### 5.6 COLLABORATION DIAGRAM

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features.

Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.



**Fig 5.6: Collaboration Diagram**

### 5.7 DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application**.**



**Fig 5.7: Deployment Diagram**

### CHAPTER 6 SYSTEM IMPLEMENTATON

### SAMPLE CODE

import tKinter as tk import cv2, os import csv

import numpy as np from PIL import image import pandas as pd import datetime import time

from tkinter import messagebox from sklearn.preprocessing import LabelEncoder

import pickel

from tensorflow.keras.applications.mobilenet\_v2 import mport load\_model

from imutils.video import VideoStream import numpy as np

import imutils window = tk.Tk()

window.title("Face Spoofing")

facedata = cv2.data.haarcascades + "haarcascade\_frontalface\_default.xml" cascade = cv2.CascadeClassifier(facedata)

font = cv2.FONT\_HERSHEY\_SIMPLEX

recognizer = cv2.face.LBPHFaceRecognizer\_create() # cv2.createLBPHFaceRecognizer()

window.configure(background='#e0ffff',width=1200, height=700)

window.grid\_rowconfigure(0, weight=10) window.grid\_columnconfigure(0, weight=1) x\_cord = 75;

y\_cord = 20;

checker = 0; print('Welcome')

message = tk.Label(window, text="FACE RECOGNITION USING DEEP LEARNING", bg="#e0ffff", fg="black",

width=40, height=1, font=('Times New Roman', 35, 'bold underline')) message.place(x=50, y=70)

lbl1 = tk.Label(window, text="Enter Name,Roll no", width=20, fg="black", bg="#e0ffff", height=2, font=('Times New Roman', 18, ' bold '))

lbl1.place(x=300 - x\_cord, y=240 - y\_cord)

txt1 = tk.Entry(window, width=36, bg="white", fg="blue", font=('Times New Roman', 15, ' bold ')) txt1.place(x=250 - x\_cord, y=300 - y\_cord)

lbl3 = tk.Label(window, text="Notification", width=15, fg="black", bg="#e0ffff", height=2, font=('Times New Roman', 18, ' bold '))

lbl3.place(x=920 - x\_cord, y=240 - y\_cord)

message = tk.Label(window, text="", bg="white", fg="blue", width=30, height=1, activebackground="white",

font=('Times New Roman', 15, ' bold ')) message.place(x=850 - x\_cord, y=300 - y\_cord)

lbl4 = tk.Label(window, text="STEP 1", width=20, fg="black", bg="#e0ffff", height=2, font=('Times New Roman', 20, ' bold '))

lbl4.place(x=160 - x\_cord, y=375 - y\_cord)

lbl5 = tk.Label(window, text="STEP 2", width=20, fg="black", bg="#e0ffff", height=2, font=('Times New Roman', 20, ' bold '))

lbl5.place(x=545 - x\_cord, y=375 - y\_cord)

lbl6 = tk.Label(window, text="STEP 3", width=20, fg="black", bg="#e0ffff", height=2, font=('Times New Roman', 20, ' bold '))

lbl6.place(x=925 - x\_cord, y=375 - y\_cord)

def clear2(): txt1.delete(0, 'end') res = ""

message.configure(text=res) def TakeImages():

Id = (txt1.get()) if not Id:

res = "Please enter first Name" message.configure(text=res)

MsgBox = tk.messagebox.askquestion("Warning", "Please enter first name properly , press yes if you understood",

icon='warning')

if MsgBox == 'no':

tk.messagebox.showinfo('Your need', 'Please go through the readme file properly') cam = cv2.VideoCapture(0)

sampleNum = 0 while (True):

ret, img = cam.read()

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) faces = cascade.detectMultiScale(gray, 1.3, 5)

for (x, y, w, h) in faces:

cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2) sampleNum = sampleNum + 1

cv2.imwrite("TrainingImage/ " +Id + '.' +str(sampleNum) + ".jpg", gray[y:y + h, x:x + w])

cv2.imshow('frame', img)

if cv2.waitKey(100) & 0xFF == ord('q'): break

elif sampleNum > 200:

break cam.release()

cv2.destroyAllWindows()

tk.messagebox.showinfo('Completed', 'Captured images successfully!!')

def TrainImages():

le = LabelEncoder()

faces, Id = getImagesAndLabels("TrainingImage") Id=le.fit\_transform(Id)

output = open('model/encoder.pkl', 'wb') pickle.dump(le, output)

output.close()

recognizer = cv2.face.LBPHFaceRecognizer\_create() recognizer.train(faces, np.array(Id)) recognizer.save(r"model\Trainner.yml")

res = "Image Trained" clear2(); message.configure(text=res)

tk.messagebox.showinfo('Completed', 'Your model has been trained successfully!!') def getImagesAndLabels(path):

imagePaths = [os.path.join(path, f) for f in os.listdir(path)] faces = []

Ids = []

for imagePath in imagePaths:

pilImage = Image.open(imagePath).convert('L') imageNp = np.array(pilImage, 'uint8')

Id = str(os.path.split(imagePath)[-1].split(".")[0]) faces.append(imageNp)

Ids.append(Id) return faces, Ids

def detect\_and\_predict\_mask(frame, faceNet, maskNet): (h, w) = frame.shape[:2]

blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),(104.0, 177.0, 123.0)) faceNet.setInput(blob)

detections = faceNet.forward() print(detections.shape)

faces = [] locs = [] preds = []

for i in range(0, detections.shape[2]): confidence = detections[0, 0, i, 2] if confidence > 0.5:

box = detections[0, 0, i, 3:7] \* np.array([w, h, w, h]) (startX, startY, endX, endY) = box.astype("int") (startX, startY) = (max(0, startX), max(0, startY)) (endX, endY) = (min(w - 1, endX), min(h - 1, endY)) face = frame[startY:endY, startX:endX]

face = cv2.cvtColor(face, cv2.COLOR\_BGR2RGB) face = cv2.resize(face, (32, 32))

face = img\_to\_array(face) face = preprocess\_input(face) faces.append(face)

locs.append((startX, startY, endX, endY))

if len(faces) > 0:

faces = np.array(faces, dtype="float32")

preds = maskNet.predict(faces, batch\_size=32) return (locs, preds)

def TrackImages():

prototxtPath = r"face\_detector\deploy.prototxt"

weightsPath = r"face\_detector\res10\_300x300\_ssd\_iter\_140000.caffemodel" faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

maskNet = load\_model("model/my\_model2.h5") print("[INFO] starting video stream...")

vs = VideoStream(src=0).start() recognizer.read(r"model/Trainner.yml") pkl\_file = open('model/encoder.pkl', 'rb')

le = pickle.load(pkl\_file) pkl\_file.close()

while True:

frame = vs.read()

frame = imutils.resize(frame, width=1000)

(locs, preds) = detect\_and\_predict\_mask(frame, faceNet, maskNet) for (box, pred) in zip(locs, preds):

(startX, startY, endX, endY) = box (fake, real) = pred

if np.argmax(np.array(pred)) == 0: color = (255, 0, 0)

val = "Fake" else:

color = (0, 255, 0) val = "Real"

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY) faces = cascade.detectMultiScale(gray, 1.2, 5)

for (x, y, w, h) in faces:

# cv2.rectangle(frame, (x, y), (x + w, y + h), (225, 0, 0), 2) Id, conf = recognizer.predict(gray[y:y + h, x:x + w]) print(conf)

if (conf < 48):

tt = le.inverse\_transform([Id])

tt = tt[0]

r='Name & Rollno: ' tt=r+ str(tt)

else:

tt = "Unknown"

cv2.putText(frame, str(tt), (x, y + h), font, 1, (255, 255, 255), 2) label = "Fake" if fake > real else "Real"

color = (0, 0, 255) if label == "Fake" else (0, 255, 0) label = "{}: {:.2f}%".format(label, max(fake, real) \* 100) cv2.putText(frame, label, (startX, startY - 10),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.45, color, 2)

cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2) cv2.imshow("Frame", frame)

key = cv2.waitKey(1) & 0xFF if key == ord("q"):

break cv2.destroyAllWindows() res = "face recognized" message.configure(text=res)

tk.messagebox.showinfo('Completed', 'Congratulations ! Your face successfully detected!!')

def quit\_window():

MsgBox = tk.messagebox.askquestion('Exit Application', 'Are you sure you want to exit the application',

icon='warning')

if MsgBox == 'yes':

tk.messagebox.showinfo("Greetings", "Thank You very much for using our software. Have a nice day ahead!!")

window.destroy()

takeImg = tk.Button(window, text="IMAGE CAPTURE BUTTON", command=TakeImages, fg="white", bg="blue", width=25, height=2,

activebackground="pink", font=('Times New Roman', 15, ' bold ')) takeImg.place(x=165 - x\_cord, y=425 - y\_cord)

trainImg = tk.Button(window, text="MODEL TRAINING BUTTON", command=TrainImages, fg="white", bg="#8e09db", width=25,

height=2, activebackground="pink", font=('Times New Roman', 15, ' bold ')) trainImg.place(x=545 - x\_cord, y=425 - y\_cord)

trackImg = tk.Button(window, text="FACE RECOGNITION BUTTON", command=TrackImages, fg="white", bg="red", width=28,

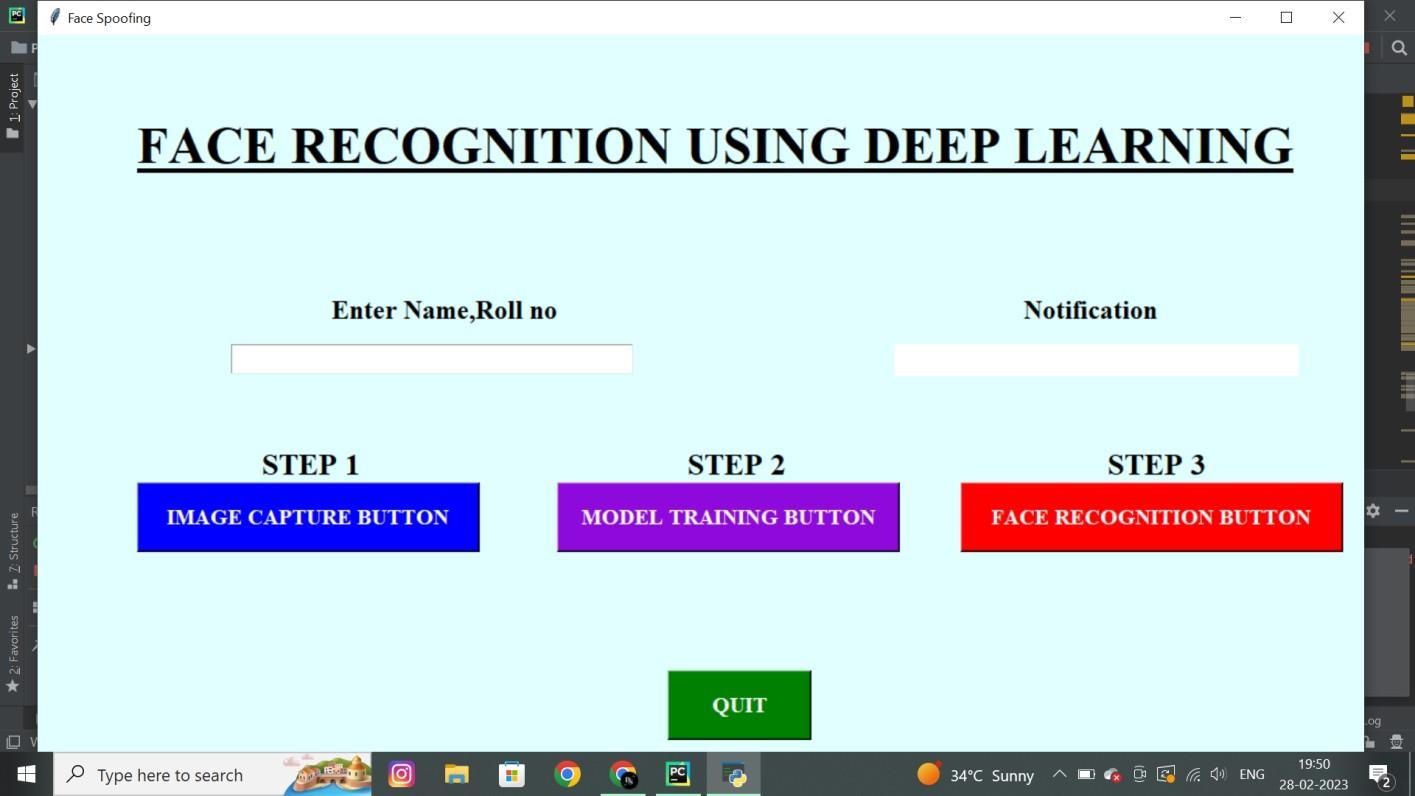
height=2, activebackground="pink", font=('Times New Roman', 15, ' bold ')) trackImg.place(x=910 - x\_cord, y=425 - y\_cord)

quitWindow = tk.Button(window, text="QUIT", command=quit\_window, fg="white", bg="green", width=10, height=2,

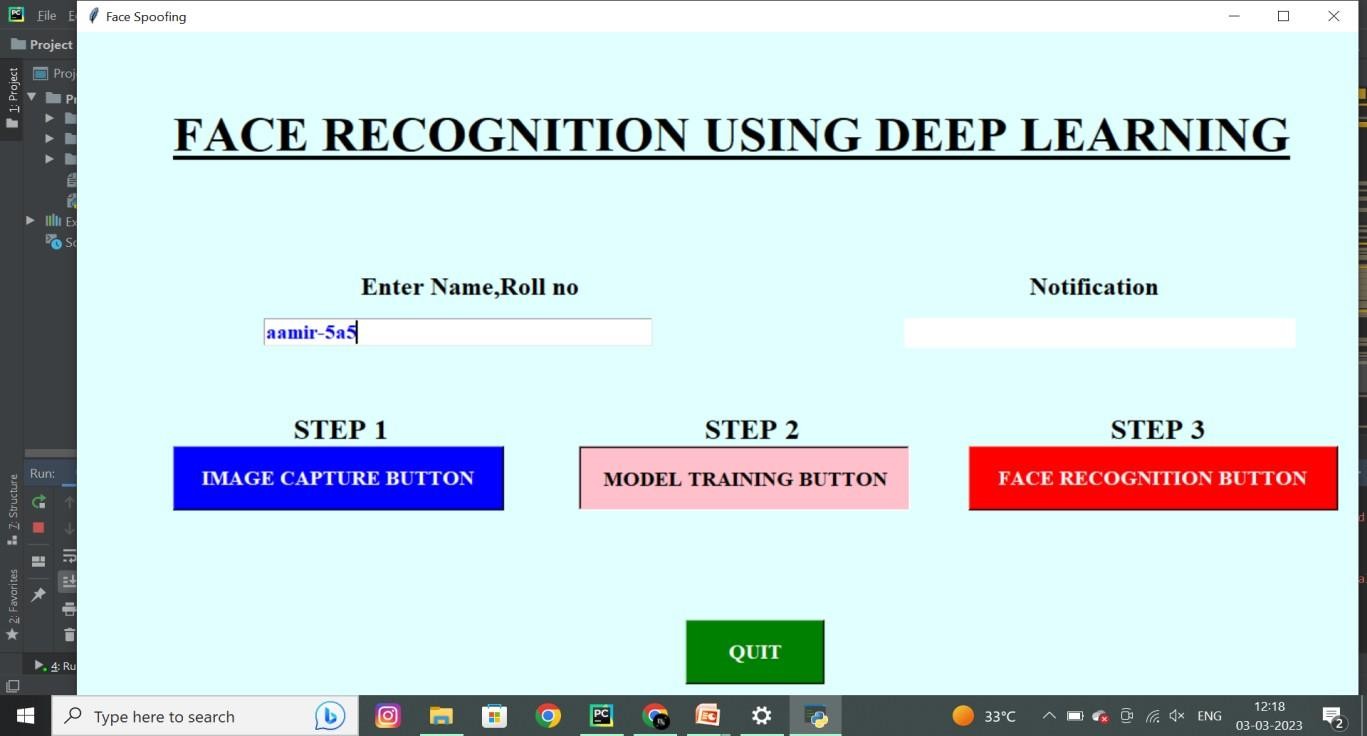
activebackground="pink", font=('Times New Roman', 15, ' bold ')) quitWindow.place(x=570, y=595 - y\_cord)

window.mainloop()

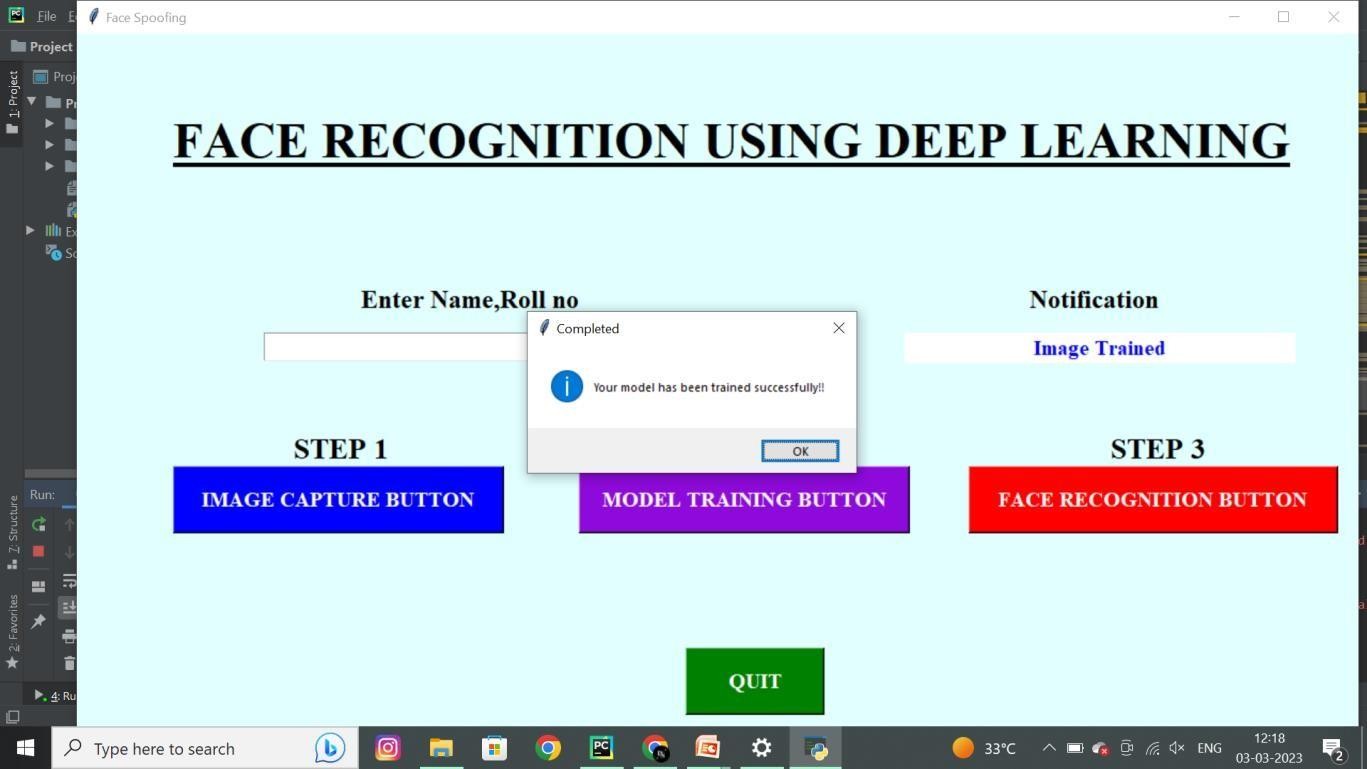
# INPUT AND OUTPUT SCREENS



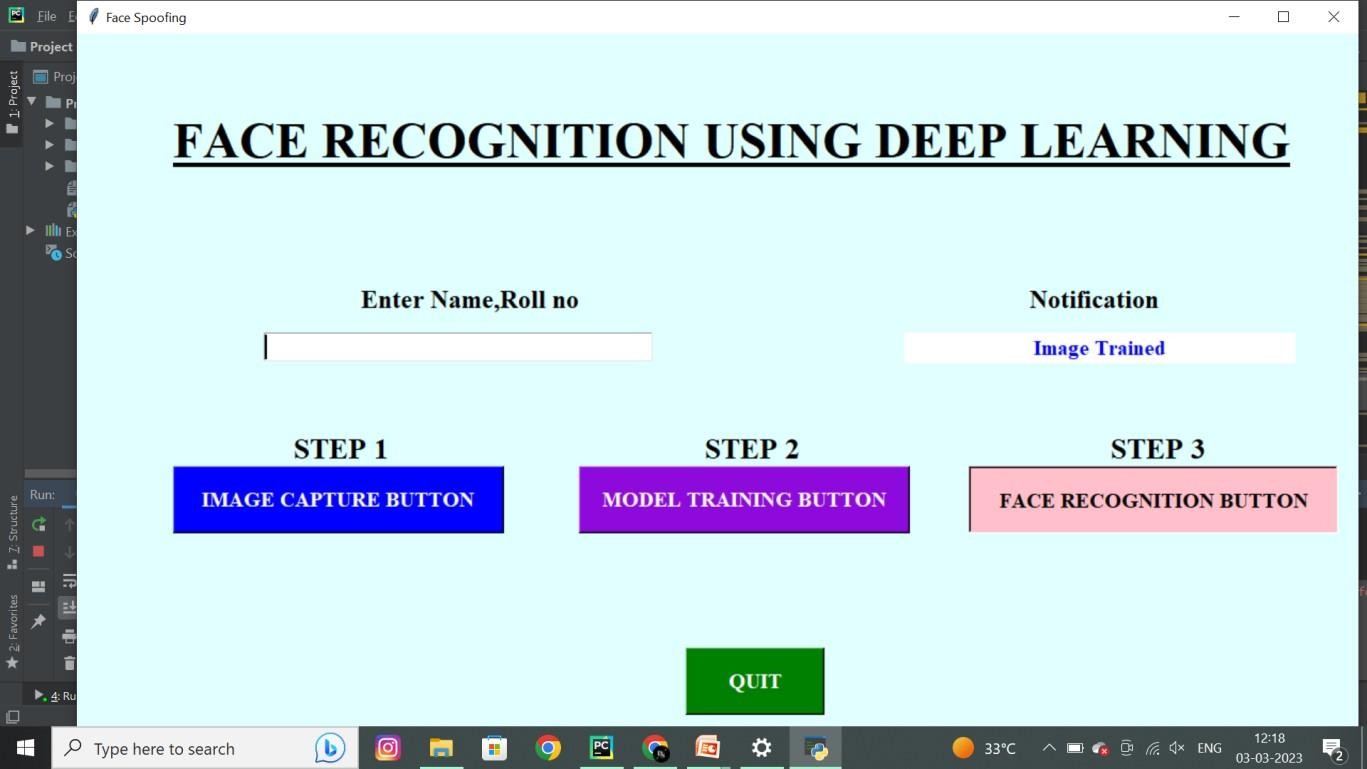
**Figure 6.2.1**: User interface



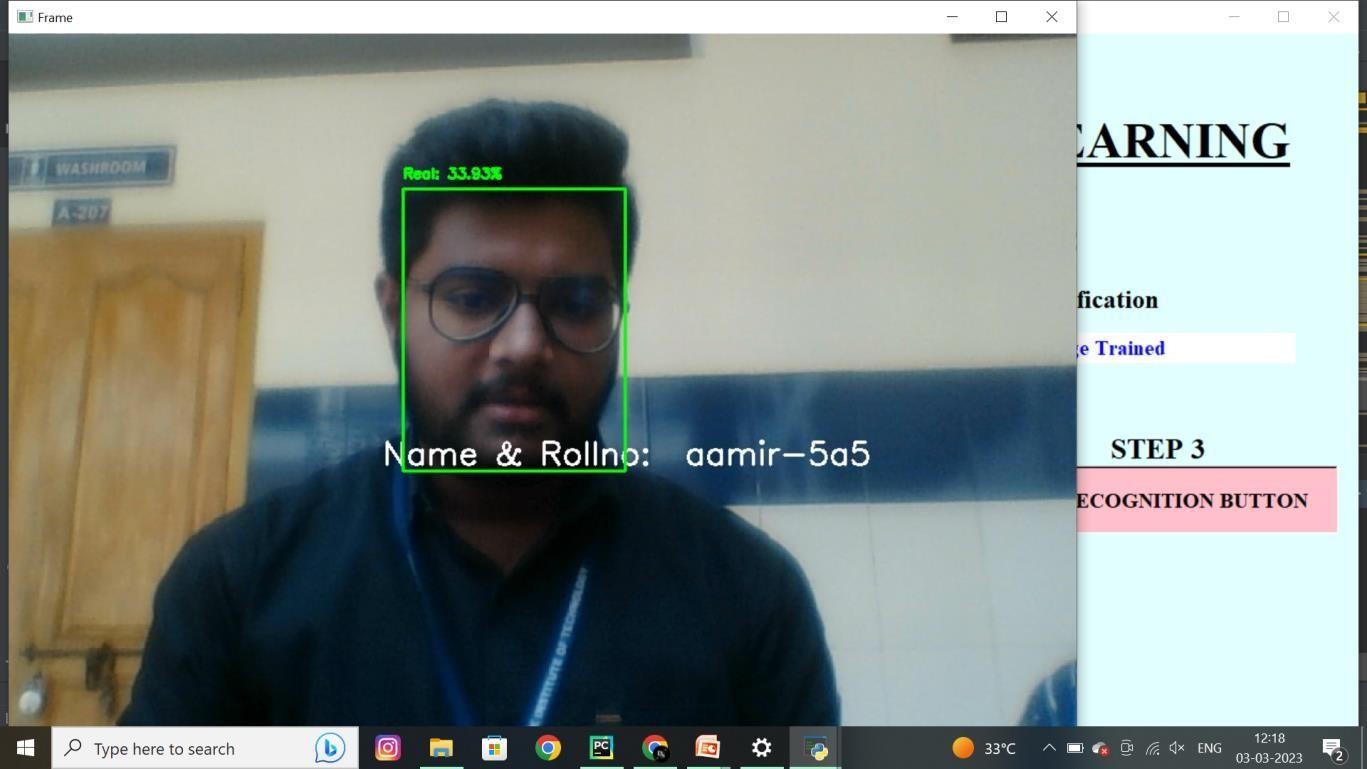
**Figure 6.2.2** :Enter the data of the user to capture the image of the user.



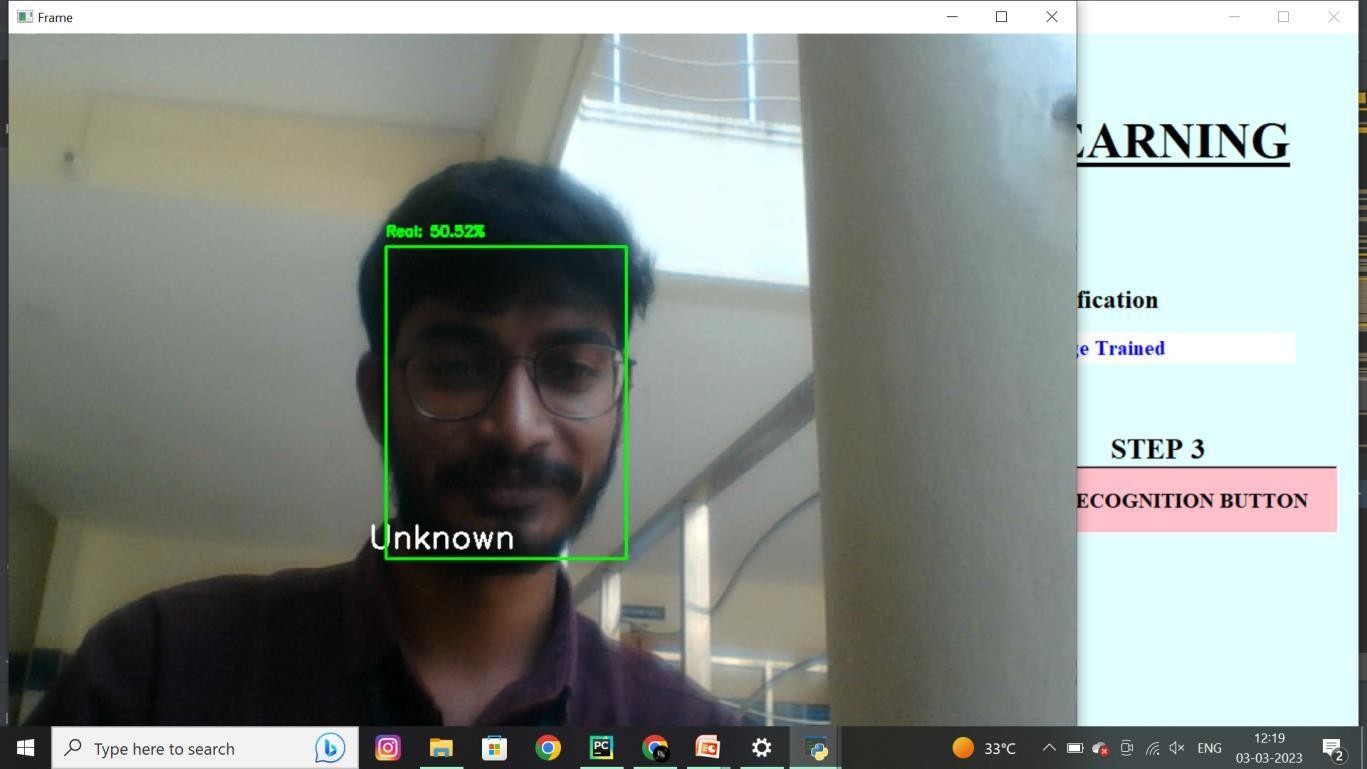
**Figure 6.2.3** After the capturing image the image will trained.



**Figure 6.2.4** After trained the image, we have to click on recognition button.



**Figure 6.2.5** After click on button we get the screen with result



**Figure 6.2.6**:If the recognized person was fake it shown as unknown.

### SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### TYPES OF TESETING

### UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Unit testing is usually conducted as part of a combined code and unit test phase of the software life cycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail. Test objectives

* + - * All field entries must work properly.
      * Pages must be activated from the identified link.
      * The entry screen, messages and responses must not be delayed. Features to be tested
      * Verify that the entries are of the correct format
      * No duplicate entries should be allowed
      * All links should take the user to the correct page.

### INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised. Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

### SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

### BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

### ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

### TESTING RESULTS

All the test cases mentioned above passed successfully. No defects encountered.

### TEST CASES

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Result** |
| Input image | Model recognize the face successfully | Success |

**Table No: 1** Input image

### 6.5.1TEST CASE MODEL BUILDING

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **Test cases** | | **I/O** | | **Expected O/T** | | **Actual O/T** | **P/F** |
| 1 | Read dataset. | the | Dataset path. | | Dataset need to read  successfully. | | Dataset fetched successfully. | P |
| 2 | Performing | | Pre- | | Pre- | | Pre- | P |
|  | pre- | | processing | | processing | | processing |  |
|  | processing | | part | takes | should | be | successfully |  |
|  | on | the | place | | performed on | | completed. |  |
|  | dataset | |  | | dataset | |  |  |
| 3 | Model Building | | Model Building for the clean data | | Need to create model using required algorithms | | Model Created Successfully. | P |
|  | Face | | Video | | Face | | Model | P |
| recognition | | streaming | | recognized | | recognize the |  |
|  | | Or upload | image | successfully | | faceimage successfully |  |

**Table No: 2** Expected and Actual outputs.

### CHAPTER 7 TECHNOLOGY DESCRIPTION

Language Used: Python

* **What is Python?**

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

* **Scripts are reusable:**

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

* **Scripts are editable:**

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

* **You will need a text editor:**

Just about any text editor will suffice for creating Python script files.

You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

**Difference between a script and a program:**

Script:

* Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

Program:

* The program has an executable form that the computer can use directly to execute the instructions.
* The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

**Python**

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don’t know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you. Python concepts

If you’re not interested in the how’s and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it’s a great one to start programming with.

* + Open-source general-purpose language.
  + Object Oriented, Procedural, Functional
  + Easy to interface with C/ObjC/Java/Fortran
  + Easy-is to interface with C++ (via SWIG)
  + Great interactive environment
  + Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

### 7.1 History of Python:

* Python was developed by Guido van Possum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.
* Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and UNIX shell and other scripting languages.
* Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).
* Python is now maintained by a core development team at the institute, although Guido van Possum still holds a vital role in directing its progress.

### 7.2 Python Features

Python's features include −

* Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* Easy-to-read − Python code is more clearly defined and visible to the eyes.
* Easy-to-maintain − Python's source code is fairly easy-to-maintained.
* A broad standard library − Python's bulk of the library is very portable and cross- platform compatible on UNIX, Windows, and Macintosh.
* Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* Extendable − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* Databases − Python provides interfaces to all major commercial databases.
* GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* 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.Dynamic vs. Static

### CHAPTER 8 CONCLUSION

In this project we have successfully developed an application, that which can detect and recognize the faces. Here we developed the two types of methods like image and video based by using CNN algorithm. Once after trained the dataset results were tested by uploading image and also video streaming with face inputs.

### FUTURE SCOPE

In future, we should consider training the model on large amount of data to have better accuracy and predictability. This process can help the machines to recognize the facial expressions.

### REFRENCES

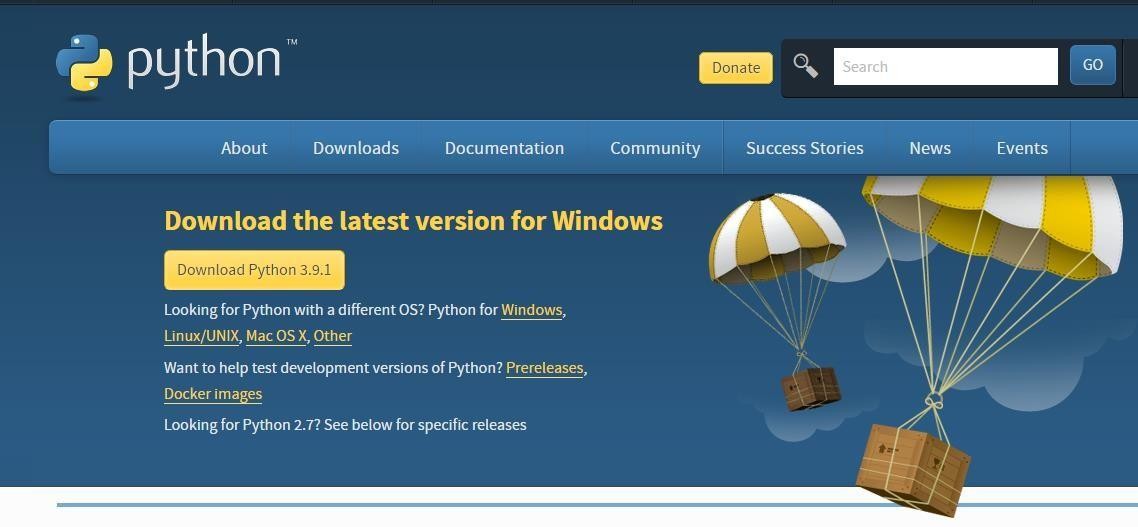
1. Liu Jianwei, Liu Yuan, Luo Xionglin. Progress in Deep Learning Research [J]. Application Research of Computers, 2014, 31 (7): 1921-1942.
2. Shen Huijun. Research and implementation of face recognition image preprocessing method [J]. Science and Technology and Innovation, 2014 (18): 119-120.
3. Zhang Chen. Research on some key technologies of facial micro-expression recognition [D]. 2019.
4. Liu Mingqi, Ni Guoqiang, Chen Xiaomei. Research on Pretreatment Algorithm of Dorsal Vein Image [J]. Optics Technology, 2007, 33: 255-256.
5. Li Siquan, Zhang Xuanxiong. Research on Facial Expression Recognition Based on Convolutional Neural Networks [J]. Journal of Software, 2018, v.17; No.183 (01): 32-35.
6. HouYuqingyang, QuanJicheng, Wang Hongwei. Overview of the development of deep learning [J]. Ship Electronic Engineering, 2017, 4: 5-9.
7. Liu Sijia, Chen Zhikun, Wang Fubin, et al. Multi-angle face recognition based on convolutional neural network [J]. Journal of North China University of Technology (Natural Science Edition), 2019, 41 (4): 103-108.
8. Li Huihui. Research on facial expression recognition based on cognitive machine learning [D]. Guangzhou: South China University of Technology, 2019.
9. Li Yong, Lin Xiaozhu, Jiang Mengying. Facial expression recognition based on cross- connection LeNet-5 network [J]. Journal of Automation, 2018,44 (1): 176-182.
10. Yao L S, Xu G M, Zhap F. Facial Expression Recognition Based on CNN Local Feature Fusion[J]. Laser and Optoelectronics Progress, 2020, 57(03): 032501.
11. Xie S, Hu H. Facial expression recognition with FRR-CNN [J]. Electronics Letters, 2017, 53 (4): 235-237.
12. Zou Jiancheng, Deng Hao. An automatic facial expression recognition method based on convolutional neural network [J]. Journal of North China University of Technology, 2019,31 (5): 51-56. [13] Xue Liang, Huang Meichuan. Emotional analysis of lyrics in Chinese popular music——Big data analysis method based on new media music terminal [D]. Music Culture Industry, 2017. (4): 77-81.
13. GuoYanhong. Research on collaborative filtering algorithm and application of recommendation system [D]. Dalian: Dalian University of Technology, 2008: 1-41.
14. Mao Xu, Wei Cheng, Qian Zhaoet al. Facial expression recognition based on transfer learning from deep convolutional networks[C]. 2015 11th Int. Conf. Nat. Comput.,2015: pp.702 - 708

### APPENDIX

**SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECTS:**

**Installing Python:**

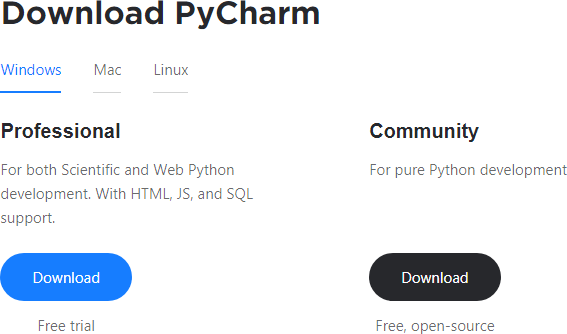
1. To download and install Python visit the official website of Python <https://www.python.org/downloads/>and choose your version.



1. Once the download is complete, run the exe for install Python. Now click on Install Now.
2. You can see Python installing at this point.
3. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

**Installing PyCharm:**

* 1. To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/>and click the "DOWNLOAD" link under the Community Section.



* 1. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
  2. On the next screen, Change the installation path if required. Click “Next”.
  3. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
  4. Choose the start menu folder. Keep selected Jet Brains and click on “Install”.
  5. Wait for the installation to finish.
  6. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
  7. After you click on "Finish," the Following screen will appear.



* 1. You need to install some packages to execute your project in a proper way.
  2. Open the command prompt/ anaconda prompt or terminal as administrator.
  3. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, sea born, scikit-learn, Matplotlib, Pyplot)

Ex: Pip install NumPy

