### PROJECT REPORT ON

Handwritten Hindi Character Recognition

Submitted In Fulfillment of The Requirements for The Award of The Degree Of

### BACHELOR OF ENGINEERING IN

**ELECTRONICS AND COMMUNICATION ENGINEERING**

Submitted by

### S. Aniroodh 1602-17-735-004

**D. Mruthyunjay** **1602-17-735-029**

**J. Vivekananda Reddy 1602-17-735-059**

UNDER THE GUIDANCE OF **CH. NEETHU** ASST.PROFESSOR

ELECTRONICS AND COMMUNICATION ENGINEERING



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

VASAVI COLLEGE OF ENGINEERING

(AFFILIATED TO OSMANIA UNIVERSITY) IBRAHIMBAGH, HYDERABAD-500031

# DEPARTMENT OF

**ELECTRONICS AND COMMUNICATION ENGINEERING**



**CERTIFICATE**

This is to certify that the Project Report entitled: “HANDWRITTEN HINDI CHARACTER RECOGNITION” is a Bonafide work done and submitted by

### S. Aniroodh 1602-17-735-004

**D. Mruthyunjay** **1602-17-735-029**

**J. Vivekananda Reddy 1602-17-735-059**

In fulfilment of requirement for the award of Bachelor of Engineering degree in Electronics and Communication Engineering during the year 2020-2021.The result embodied in this project report has not been submitted to any other university or institute for the award of any degree.

Head of the Department Internal Guide

### Dr. E. SREENIVASA RAO Ch. Neethu

Professor and HOD, ECE Dept. Asst.Professor, ECE Dept.

VCE-HYD VCE-HYD

External Examiner

# DECLARATION

We hereby declare that the results embodied in this dissertation entitled “**HANDWRITTEN HINDI CHARACTER RECOGNITION**” is carried out by us during the academic year 2020-2021 in fulfillment of the requirement for the award of B.E. (Electronics and Communication Engineering) from “**Vasavi College of Engineering**”.

We have not submitted the same to any other university or organization for the award of any other degree.

### S. Aniroodh D. Mruthyunjay J. Vivekananda Reddy

**(1602-17-735-004) (1602-17-735-029) (1602-17-735-059)**

**ACKNOWLEDGEMENT**

This satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mentioning of the people whose constant guidance and encouragement made it possible. We take pleasure in presenting before you, our project, which is result of studied blend of both research and knowledge.

It is our privilege to express our earnest gratitude and venerable regards to our internal guide **Ch. Neethu**, ASST.PROFESSOR, E.C.E. Department, Vasavi College of Engineering, IbrahimBagh, for abounding and able guidance during the preparation and execution of the project work. We are grateful for his cooperation and his valuable suggestions.

We record with pleasure our deep sense of gratitude to **Dr. E. SREENIVASA RAO**, Head of the Department, E.C.E. for his simulating guidance and profuse assistance we have received, which helped throughout the project. Also, we acknowledge with thanks for the technical help extended by one and all in shaping up our project.

### With Regards and Gratitude

**S. ANIROODH**

**D. MRUTHYUNJAY**

**J. VIVEKANANDA REDDY**

**ABSTRACT**

Character recognition of Hindi character is a little bit difficult as compared to character recognition of other languages like English, because of similarities in the characters, e.g., **अ, अः.** In every state of India there are two languages one is Hindi, and the other is English. For Optical Character Recognition (OCR) of such a bilingual document, it is necessary to identify the script before feeding the text words to the OCRs of individual scripts. In this paper, we are introducing a simple and efficient technique of script identification for Hindi text words of a printed document.

Convolution Neural Network (CNN) is turning out to be a very powerful tool for solving Machine Learning (ML) problems, especially in multiclass image classification. This project focuses on the task of recognizing handwritten Hindi characters using a Convolutional Neural Network (CNN) based Deep Learning model. The recognized characters can then be stored digitally in the computer or used for other purposes. The dataset used is obtained from the UC Irvine Machine Learning Repository which contains 92,000 images divided into training (85%) and test set (15%). Grayscale handwritten character images are used as input. Filters are applied on the images to extract different features at each layer. This is done by the Convolution operation. The two other main operations involved are Pooling and Flattening. The output of the CNN layers is fed to the fully connected layers. Finally, the chance or probability score of each character is determined and the character with the highest probability score is shown as the output.

**Keywords:** Convolutional Neural Network (CNN), Dataset.

# TABLE OF CONTENTS

|  |  |
| --- | --- |
| **CONTENTS** | **PAGE NO.** |
| ABSTRACT | V |
| TABLE OF CONTENTS | VI |
| LIST OF FIGURES | VIII |
| LIST OF TABLES | X |
| [**CHAPTER 1: INTRODUCTION**](#_bookmark0) | [**1**](#_bookmark0)**-9** |
| [1.1 Introduction](#_bookmark1) | [1](#_bookmark1) |
| [1.2 Software](#_bookmark2) | [1](#_bookmark2) |
| [1.2.1 PyCharm](#_bookmark3) | [1](#_bookmark3) |
| [1.2.2 Python 3.6](#_bookmark4) | [2](#_bookmark4) |
| [1.2.3 Libraries/Modules Used](#_bookmark5) | [2](#_bookmark5) |
| [1.2.4 Performance Metrics](#_bookmark6) | [4](#_bookmark6) |
| [1.3 Problem Statement](#_bookmark8) | [7](#_bookmark8) |
| [1.4 Flow Chart](#_bookmark9) | [8](#_bookmark9) |
| [1.5 Aims and Objectives](#_bookmark11) | [8](#_bookmark11) |
| [1.6 Scope of the Project](#_bookmark12) | [9](#_bookmark12) |
| **CHAPTER 2: LITERATURE SURVEY** | **10-15** |
| [2.1 Holistic methods](#_bookmark14) | [11](#_bookmark14) |
| [2.2 Feature based methods](#_bookmark15) | [12](#_bookmark15) |
| [2.3 Hybrid Methods](#_bookmark16) | [12](#_bookmark16) |
| **CHAPTER 3: DEEP NEURAL NETWORKS** | **16-28** |
| [3.1 Introduction to Deep Neural Networks (DNN)](#_bookmark18) | [16](#_bookmark18) |
| [3.2 Different types of Deep Neural Networks](#_bookmark21) | [18](#_bookmark21) |
| [3.2.1 Autoencoders](#_bookmark22) | [18](#_bookmark22) |
| [3.2.2 Deep Belief Network](#_bookmark23) | [19](#_bookmark23) |
| [3.2.3 Convolutional Neural Networks](#_bookmark24) | [20](#_bookmark24) |
| [3.2.4 Recurrent Neural Networks](#_bookmark25) | [20](#_bookmark25) |
| [3.2.5 Reinforcement Learning to Neural Networks](#_bookmark26) | [21](#_bookmark26) |

* 1. [CNN and its Architecture 21](#_bookmark27)
     1. [Convolution Layer 22](#_bookmark29)
     2. [Strides 23](#_bookmark31)
     3. [Padding 23](#_bookmark33)
     4. [Non-Linearity (ReLU) 23](#_bookmark34)
     5. [Pooling Layer 24](#_bookmark36)
     6. [Fully Connected Layer 25](#_bookmark38)
  2. [Siamese Network 26](#_bookmark41)
  3. [Applications of DNN 26](#_bookmark42)

CHAPTER 4: FACE DETECTION AND RECOGNITION FROM LIVE VIDEO 29-43

* 1. [Capture video 29](#_bookmark43)
  2. [Separate as frames from the video 29](#_bookmark44)
  3. [Face Detection 29](#_bookmark45)
  4. [Pre-Processing 30](#_bookmark48)
  5. [Model Training 32](#_bookmark51)
  6. [Face Recognition 37](#_bookmark60)
  7. [Post-Processing 38](#_bookmark63)
  8. Case in Face Recognition 39

[CHAPTER 5: CONCLUSION AND FUTURE SCOPE 44](#_TOC_250002)

[REFERENCES 45](#_TOC_250001)

[APPENDIX 46](#_TOC_250000)

**LIST OF FIGURES**

* 1. [Confusion Matrix 5](#_bookmark7)
  2. [Flow Chart 8](#_bookmark10)
  3. [Neural Network 17](#_bookmark19)
  4. [Activation Function 17](#_bookmark20)
  5. [Neural network with many convolutional layers 22](#_bookmark28)
  6. [Image matrix and Filter matrix 22](#_bookmark30)
  7. [Stride of 2 pixels 23](#_bookmark32)
  8. [ReLU Operation 24](#_bookmark35)
  9. [Max Pooling 25](#_bookmark37)
  10. [After pooling layer, flattened as FC layer 25](#_bookmark39)
  11. [Complete CNN architecture 26](#_bookmark40)
  12. [Human Faces in a Frame 29](#_bookmark46)
  13. [Face Detection 30](#_bookmark47)
  14. [Dataset 31](#_bookmark49)
  15. [Student Details 31](#_bookmark50)
  16. [Siamese Network (Type of CNN) 33](#_bookmark52)
  17. [Splitting of Dataset 33](#_bookmark53)
  18. [Second neural network 34](#_bookmark54)
  19. [DenseArchs 35](#_bookmark55)
  20. [Compile Keras model with an optimizer and loss function 35](#_bookmark56)
  21. [Fit Keras model to the data 36](#_bookmark57)
  22. [Evaluate Keras model 36](#_bookmark58)
  23. [Training Images 37](#_bookmark59)
  24. [Face Recognition 37](#_bookmark61)
  25. [Face Recognition with only Face Frame 38](#_bookmark62)
  26. [Attendance In Excel Sheet 38](#_bookmark64)
  27. [Performance Metrics 39](#_bookmark65)
  28. [Recognition under Normal Condition 39](#_bookmark66)
  29. [Recognition at a Distance 40](#_bookmark67)
  30. [No Detection in Low Light 40](#_bookmark68)
  31. [No Detection with Mask 41](#_bookmark69)
  32. [No Detection when facing side angles 41](#_bookmark70)
  33. [No Recognition under Bright Light 42](#_bookmark71)
  34. [No Recognition under Change of Expressions 42](#_bookmark72)
  35. [No Recognition beyond Certain Distance 43](#_bookmark73)

**LIST OF TABLES**

* 1. [Advantages & Disadvantages of Different Attendance System 11](#_bookmark13)
  2. [Advantages & Disadvantages of Face Detection Methods 14](#_bookmark17)

**CHAPTER 1 INTRODUCTION**

* 1. **Introduction**

Optical Character Recognition (OCR) is a process of converting printed or handwritten scanned documents into ASCII characters that a computer can recognize [1]. In other words, automatic text recognition using OCR is the process of converting an image of textual documents into its digital textual equivalent. The advantage is that the textual material can be edited, which otherwise is not possible in scanned documents in which these are image files. The document image itself can be either machine-printed or handwritten, or a combination of the two. Computer systems equipped with such an OCR system improve the speed of input operation, decrease some possible human errors and enable compact storage, fast retrieval and other file manipulations. The range of applications includes postal code recognition, automatic data entry into a large administrative system, banking, automatic cartography and, when interfaced with a voice synthesizer, reading devices for the visually handicapped.

* 1. **Software**
     1. **Jupyter Notebook**

JetBrains has developed PyCharm as a cross-platform IDE for Python. In addition to supporting versions 2.x and 3.x of Python, PyCharm is also compatible with Windows, Linux, and macOS. At the same time, the tools and features provided by PyCharm help programmers to write a variety of software applications in Python quickly and efficiently. The developers can even customize the PyCharm UI according to their specific needs and

preferences. Also, they can extend the IDE by choosing from over 50 plug-ins to meet complex project requirements.

## Python 3.9

Python is a[n interpreted](https://en.wikipedia.org/wiki/Interpreted_language), [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language) [programming language](https://en.wikipedia.org/wiki/Programming_language). Created by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) and first released in 1991, Python's design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with its notable use of [significant whitespace.](https://en.wikipedia.org/wiki/Off-side_rule) Its [language](https://en.wikipedia.org/wiki/Language_construct) [constructs](https://en.wikipedia.org/wiki/Language_construct) and [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is [dynamically typed](https://en.wikipedia.org/wiki/Dynamic_programming_language) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple [programming](https://en.wikipedia.org/wiki/Programming_paradigms) [paradigms,](https://en.wikipedia.org/wiki/Programming_paradigms) including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly, [procedural](https://en.wikipedia.org/wiki/Procedural_programming)), object-oriented, and [functional](https://en.wikipedia.org/wiki/Functional_programming) [programming.](https://en.wikipedia.org/wiki/Functional_programming) Python is often described as a "batteries included" language due to its comprehensive [standard library.](https://en.wikipedia.org/wiki/Standard_library)

Py[thon interpreters](https://en.wikipedia.org/wiki/Interpreter_(computing)) are available for many [operating systems.](https://en.wikipedia.org/wiki/Operating_system) A global community of programmers develops and maintains [CPython](https://en.wikipedia.org/wiki/CPython), an [open source](https://en.wikipedia.org/wiki/Open-source_software) [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation). A [non-profit organization](https://en.wikipedia.org/wiki/Nonprofit_organization), the [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation), manages and directs resources for Python and CPython development.

**Python** is a general-purpose high level programming language that is widely used in data science and for producing **deep learning** algorithms. This brief tutorial introduces **Python** and its libraries like NumPy, Scipy, Pandas, Matplotlib; frameworks like Theano, TensorFlow, Keras. **Python** is a general-purpose high level programming language that is widely used in data science and for producing **deep learning** algorithms. This brief tutorial introduces **Python** and its libraries like NumPy, Scipy, Pandas, Matplotlib frameworks like Theano, TensorFlow, Keras.

## Libraries/Modules Used

### OpenCV:

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

OpenCV uses machine learning algorithms to search for faces within a picture. Because faces are so complicated, there is not one simple test that will tell you if it found a face or not. Instead, there are thousands of small patterns and features that must be matched. The algorithms break the task of identifying the face into thousands of smaller, bite-sized tasks, each of which is easy to solve. These tasks are also called classifiers.

### Keras:

Keras is a[n open-source](https://en.wikipedia.org/wiki/Open-source_software) [neural-network](https://en.wikipedia.org/wiki/Artificial_neural_network) library written in [Python.](https://en.wikipedia.org/wiki/Python_(programming_language)) It is capable of running on top of [TensorFlow,](https://en.wikipedia.org/wiki/TensorFlow) [Microsoft Cognitive Toolkit](https://en.wikipedia.org/wiki/Microsoft_Cognitive_Toolkit), [R](https://en.wikipedia.org/wiki/R_(programming_language)), [Theano](https://en.wikipedia.org/wiki/Theano_(software)), or [PlaidML](https://en.wikipedia.org/wiki/PlaidML). Designed to enable fast experimentation with [deep neural networks,](https://en.wikipedia.org/wiki/Deep_learning) it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open- ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a [Google](https://en.wikipedia.org/wiki/Google) engineer. Chollet also is the author of the XCeption deep neural network model.

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, [objectives,](https://en.wikipedia.org/wiki/Objective_function) [activation functions,](https://en.wikipedia.org/wiki/Activation_function) [optimizers](https://en.wikipedia.org/wiki/Mathematical_optimization), and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides.

### TensorFlow:

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy. If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the

Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

### NumPy:

NumPy is a library for the [Python programming language,](https://en.wikipedia.org/wiki/Python_(programming_language)) adding support for large, multi- dimensional [arrays](https://en.wikipedia.org/wiki/Array_data_structure) and [matrices](https://en.wikipedia.org/wiki/Matrix_(math)), along with a large collection of [high-](https://en.wikipedia.org/wiki/High-level_programming_language) [level](https://en.wikipedia.org/wiki/High-level_programming_language) [mathematical](https://en.wikipedia.org/wiki/Mathematics) [functions](https://en.wikipedia.org/wiki/Function_(mathematics)) to operate on the arrays.

### Pandas:

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the NumPy package, and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables. Using we can create csv files(excel) for uploading of attendance data in it.

### Matplotlib:

### TKINTER:

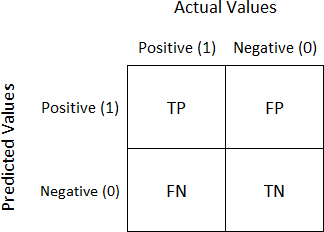
### PIL:

## Performance Metrics

**Confusion Matrix**

In the case of binary recognition or two class recognition, the system has to differentiate between face and nonface criteria. The true positive means the portion of face images to be detected by the system, while the false positive means the portion of nonface images to de detected as faces. The term true positive here has the same meaning as the detection rate and

recall. False positives imply wrongly matching the individuals with photos in the database, and false negatives means not catching people even when their photo is in the database. There are two main evaluation plots: the receiver operating characteristics (ROC) curve and the precision and recall (PR) curve. The ROC curve examines the relation between the true positive rate and the false positive rate, while the PR curve extracts the relation between detection rate (recall) and the detection precision.



## 1.1 Confusion Matrix

TP = True Positive FP = False Positive FN = False Negative TN = True Negative

### Accuracy:

Accuracy is the proportion of classifications, over all the N examples that were correctly detected. Accuracy is defined as “the fraction of quantity of correct classification over the entire number of samples.” The number of predictions in classification techniques relies upon the counts of the test records properly or incorrectly predicted by the model. These counts are tabulated into a confusion matrix (also referred as contingency).

Accuracy=No of correctly detected pattern / Total number of validation set

*Acc* 

*TP*  *TN*

*TP*  *TN*  *FP*  *FN*

### Loss

Precision is the fraction of the detected images that square measure relevant to the user’s wants. It is additionally referred to as reliability or repeatability and is that the degree to that recurrent measurements beneath unchanged conditions show an equivalent results.

Precison=No of truepositive / No of all detected patterns

*prec* 

*TP TP*  *FP*

### F1 Score

F-measure is additionally referred to as F-Score or F1-measure. It combines the exactness and recall. It computes the average of the precision and recall. A conventional F-measure is the harmonic mean of precision and recall.

*F*1*Score*  2  *recall* *prec*

*recall*  *prec*

## Problem Statement

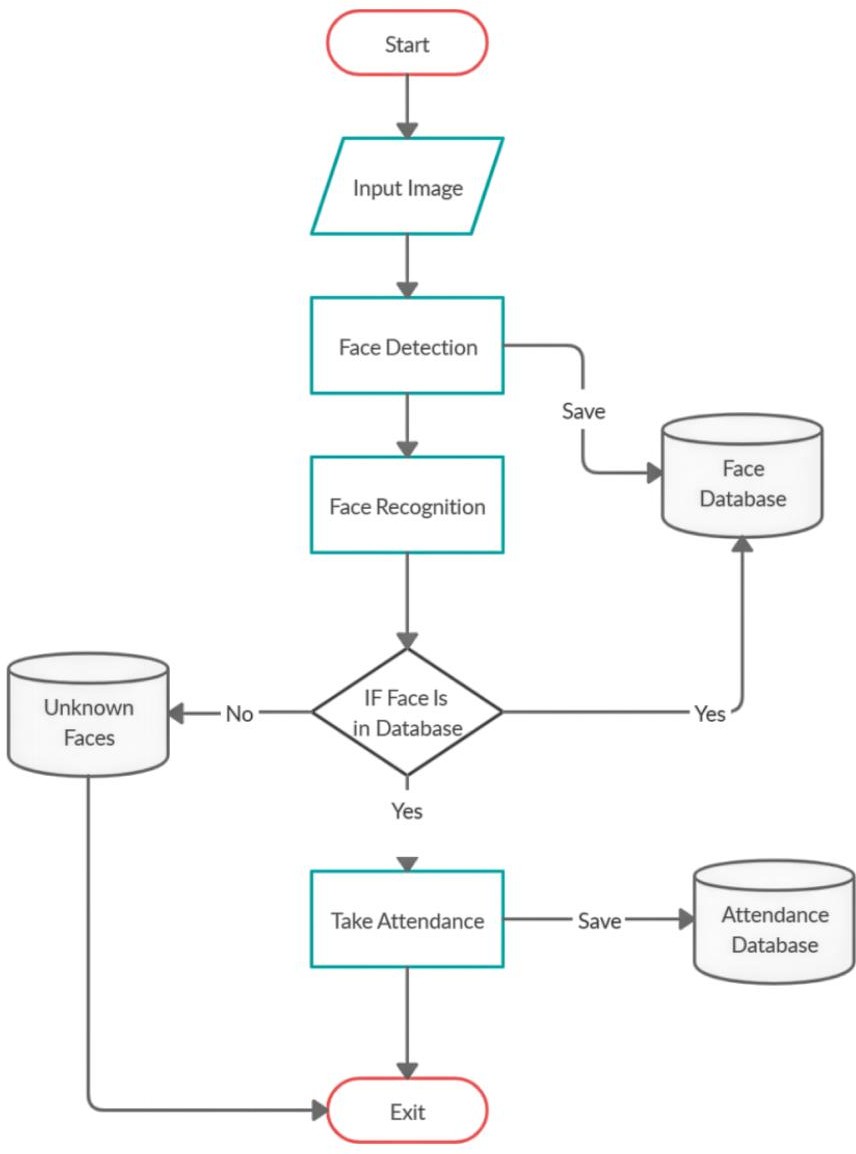
Traditional student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking technique such as calling student names or checking respective identification cards. There are not only disturbing the teaching process but also causes distraction for students during exam sessions. Apart from calling names, attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with a large number of students might find it difficult to have the attendance sheet being passed around the class. Thus, face recognition attendance system is proposed in order to replace the manual signing of the presence of students which are burdensome and causes students get distracted in order to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturers does not have to count the number of students several times to ensure the presence of the students.

the only option to digitize printed paper documents was by manually re-typing the text. Not only was this massively time-consuming, but it also came with inaccuracy and typing errors.

The paper proposed by Zhao, Wetal. (2003) has listed the difficulties of facial identification. One of the difficulties of facial identification is the identification between known and unknown images. In addition, paper proposed by Pooja G.Retal. (2010) found out that the training process for face recognition student attendance system is slow and time-consuming. In addition, the paper proposed by Priyanka Waghetal. (2015) mentioned that different lighting and head poses are often the problems that could degrade the performance of face recognition based student attendance system.

Hence, there is a need to develop a real time operating student attendance system which means the identification process must be done within defined time constraints to prevent omission. The extracted features from facial images which represent the identity of the students have to be consistent towards a change in background, illumination, pose and expression. High accuracy and fast computation time will be the evaluation points of the performance.

## Flow Chart



**1.2 Flow Chart**

* 1. **Aims and Objectives**

The objective of this project is to develop face recognition attendance system. Expected achievements in order to fulfill the objectives are:

* To detect the face segment from the video frame.
* To extract the useful features from the face detected.
* To classify the features by training in order to recognize the face detected.
* To record the attendance of the identified student.

## Scope of the Project

We are setting up to design a system comprising of two modules. The first module (face detector) is a mobile component, which is basically a camera application that captures student faces and stores them in a file using computer vision face detection algorithms and face extraction techniques. The second module is a desktop application that does face recognition of the captured images (faces) in the file, marks the students register and then stores the results in a database for future analysis.

# CHAPTER 2 LITERATURE SURVEY

1. **Deep Learning Based Large-Scale Handwritten Devanagari Character Recognition:** In this paper, Ashok Kumar Pant and his group worked on a image dataset for Devanagari script. The dataset contained 92000 images of 46 different classes of characters belonging to Devanagari script which are segmented from Handwritten files. They have also come across the challenges present in recognizing the handwritten characters of Devanagari script. The proposal of deep learning architecture has been done for recognition of characters. In this regard, Deep Convolutional Neural Network had shown significant results to commonly used networks in many recognition requirements. Keeping aside the character recognition by Deep CNN, they focused on dataset increment approach to improve test accuracy. By implementing these methods in Deep CNN, it has been possible to improve the test accuracy by 1%. This proposed architecture has given an accuracy of 98.47% on their dataset.
2. **Review of Research on Devanagari Character Recognition by Vikas Dhongre:** In this paper, they have explored and analyzed the Devanagari Optical Character Recognition and also investigated the disadvantages and limitations of all the methods for the systems which can divided depending upon two major factors: the data acquisition process and the text type(handwritten). The arrangement of this paper has been done to review the Devanagari Optical Character Recognition methods in the aspect of stages of the CR systems rather than the surveying of the complete solution. Handwritten character recognition was more focused in this article rather than machine printed text.
3. **Effective Techniques for the Detection, Extraction and Conversion of Devanagari Text from Traffic Panels by Miss. Gayatri H. Khobragade:** In this paper, they have put forward the methods for detection, extraction and conversion. The color segmentation for detection is either green or yellow from the background image that provided an accuracy of 95%. For text detection, object extraction and segmentation has been used which differentiates all the objects with an accuracy of 90%. Text recognition and extraction gave 99% result with own dictionary only. But with the help of OCR trained dataset, accuracy has been 80%, but with the help of additional dictionary it gave an accuracy of 99%. In order to convert the text, an algorithm named N-GRAM has been used to check the probability of single letter in the entire dictionary leading to an accuracy of 99%.
4. **Recognition of Handwritten Devanagari Characters Using Machine Learning Approach by Pankaj:** The proposed design is used to recognize 50 simple Devanagari characters. The process included, data collection, preprocessing, array formation, segmentation and classification. The creation of handwritten character database has been done and stored in 64\*64 size. 10 handwritten sentences have been taken into consideration to test. The very first step is to preprocess during which the phase images were filtered for noise that were present in the scanned documents followed by binarization of the image. The second step is segmentation in which the upper line(header line) in Devanagari script was excluded. The sum of all pixels in a row were found and were performed for all rows so as to get a column vector. The row containing maximum number belongs to header line. Then they have deleted the upper line and added background pixels. The segmented image was bounded to a size of 64\*64. Resize of images to 35\*35 pixels has been done and has been further converted to an array of 1225 elements. This array has been given as input to a neural network. The rest of the 50 characters were trained in the similar manner.
5. **Optical Character Recognition for Hindi Language Using a Neural-network Approach by Divakar Yadav**: In this paper, an OCR has been suggested for printed Hindi characters using Artificial Neural Network (ANN), because of which efficiency would be enhanced. One of the major reasons for acknowledgement rate is fault in character segmentation. Pre-processing, character segmentation, extraction, classification and recognition are the chief steps which are followed by an OCR. The preprocessing of the tasks is done to adapt to the conversion of grayscale images to binary images and also image rectification. The basic symbols from the segmentation are accepted by the neural classifier. For the forwardness of the neural classifier, the usage of back-propagation neural network with two hidden layers is done. The classifier is trained for Hindi texts. In this regard, 3 characteristic features are used for projection based on mean distance, projection based on pixel value and zero crossing(vertical). All these have been used to recover the swift of detection. These extraction methods are sufficient enough to drag all the features of indistinct characters or symbols. With the help of OCR, the accuracy has been 90%.
6. **Mr. Verma used radial basis function and multilayer perceptron neural networks for recognizing the handwritten characters of Devanagari script.** Back propagation error algorithm has been used to increase the recognition rate. In this paper, they compare the results that they got from radial basis Function networks and multi-layer perceptron. Dataset contained 245 sample, which are handwritten by five different users. The results then obtained showed that multilayer perceptron networks performed better when compared to radial basis functions. But networks training time is more as compared to radial basis function networks. Highest recognition rate that was obtained using radial basis function and multilayer perceptron networks are 85.01 percent and 70.8 percent respectively.
7. **Mr. Kapil Malhotra** introduced a technique in order to recognize the handwritten Devanagari characters. Besides the previous approaches that are based upon basic classifiers like SVM, HMM, ANN and trained on statistical, structural or spectral features, this method, based on CNN, allows writers to provide character inputs in any order and any number of strokes and is also keen on overwriting as well. The CNN architecture supports 42 Devanagari character classes. Experimental tasks undergone with 10 different configurations of CNN show highly accurate result. In the next development, the average of top 3 final layer neuron outputs has been taken into consideration and is used to take decision with respect to classification, leading to an accuracy of 99.82% on the training data and 98.19% on the testing data. This shows a significant improvement of 0.2% and 5.81%, for the train and test set data respectively. The data that is used for constructing the system has been taken from different parts of Devanagari writing states in India. Character data is extracted from the words that are collected using a hybrid approach and it considers all possible variations leading to the different way of writing and variable word formats.

**Student Attendance System Using Iris Detection**: In this proposed system the student is requested to stand in front of the camera to detect and recognize the iris, for the system to mark attendance for the student. Some algorithms like Gray Scale Conversion, Six Segment Rectangular Filter, Skin Pixel Detection is being used to detect the iris. It helps in preventing the proxy issues and it maintains the attendance of the student in an effective manner, but in one of the time-consuming process for a student or a staff to wait until the completion of the previous members.

**Automated Attendance System Using Face Recognition**: Automated Attendance System using Face Recognition proposes that the system is based on face detection and recognition algorithms, which is used to automatically detects the student face when he/she enters the class and the system is capable to mark the attendance by recognizing him. When it is compared to traditional attendance marking this system saves the time and also helps to monitor the students.

Face recognition has been highlighted in many research papers throughout journals. The other classifier used for image detection is the Haar Cascade Classifier. Haar features are described and the technique is proposed in the paper Rapid Object Detection using a Boosted Cascade of Simple Features by Paul Viola and Michael Jones in 2001.

There were many approaches used for dealing with disparity in images subject to illumination changes and these approaches were implemented in object recognition systems and also by systems that were specific to faces. A method for dealing with such variations was using

gray-level information to extract a face or an object from shading approach. The main reason why gray scale representations are used for extracting descriptors instead of operating on colour images directly is that gray scale simplifies the algorithm and reduces computational requirements. Here in our case, colour is of limited benefit and introducing unnecessary information could increase the amount of training data required to achieve good performance. Being an ill-posed problem, these proposed solutions assumed either the object shape and reflectance properties or the illumination conditions. These assumptions made are too strict for general object recognition and therefore it didn’t prove to be sufficient for face recognition.

## 2.1: Advantages & Disadvantages of Different Attendance System

|  |  |  |
| --- | --- | --- |
| **System Type** | **Advantages** | **Disadvantages** |
| RFID card system | Simple | Fraudulent usage |
| Fingerprint system | Accurate | Time-consuming |
| Voice recognition system |  | Less accurate compared to Others |
| Iris recognition system | Accurate | Privacy Invasion |

In the last few decades many face recognition methods have been proposed by researchers with different background, which lead to vast and diverse literature. Due to this diverse view in a single system of face recognition it becomes difficult to classify based on the techniques employed. Face recognition techniques have been classified into three categories based on the information from physiological studies as follows.

## Holistic methods

In these methods, whole face is given as input to a recognition module. The face image is represented in a lower dimension using principal component analysis (PCA) without losing much information, and then reconstructing it. The Eigen pictures are determined from the

correlation matrix. These Eigen pictures are the optimal set to represent any picture. Face recognition system was developed based on Eigen pictures. Other techniques like Independent component analysis which is a generalization of principal component analysis and Linear Discriminate Analysis which retrieves vectors to discriminate the classes by increasing the between-class differences, reducing the within- class ones also use whole face image as input.

## Feature based methods

The locations and local statistics of the local features on face such as eyes, mouth and nose are extracted and fed into a structural classifier. Kanade in 1974 developed Face recognition system which extracts the local features of the face and defined a face model based on the position, size and the relation between the features. Wiskott et.al use Gabor wavelet transform to represent the local features. Lawrence had proposed a face recognition system which uses local image sample representation, Self Organizing Map (SOM), and Convolution neural network (CNN). Ahonen et al. divided the Face image into sub regions and calculate the Linear Binary Pattern (LBP) histograms. Later these LBP histograms are combined into global histogram. Liao et al introduced a variation of LBP where they calculate the average values of block sub regions instead of individual pixel.

## Hybrid Methods

These methods are based on the human vision system which considers both local features and whole face. Pentand in 1994 used both Eigen faces and Eigen modules like Eigen mouth, Eigen eyes, and Eigen nose. Huang et al. in 2003 has used face region and components.

### Viola-jones

Viola-Jones was proposed by Paul Viola and Michael Jones in 2001. Viola- Jones is based on object detection, but its main application is for face detection. The detection rate of the Viola-Jones algorithm is high (true-positive level), and the false-positive rate is low and could detect face rapidly. Although this method has a drawback, the training for this system is slow and less effective on non-frontal face.

For face detection, the Viola-Jones algorithm goes throughthree main stages:

### Computing Integral Image

This stage is used to convert the image into an integral image. An integral image is a concept of Summed-Area Table, that is used to compute the sum of values in a subset of rectangular boxes. The integral image that is located at (x, y) is the sum result of pixels located above it, and pixels located to its left. By creating an integral image, computation for Haar features can be done rapidly.

### Adaboost Algorithm

AdaBoost is one of a machine learning algorithm that is used for detecting face. Classifiers are created by selecting few essential features computed in the previous stage. An AdaBoost algorithm is used to select these original featuresand train classifiers that would be using those features. The AdaBoost algorithm is aiming to construct a robust classifier from the linear combination of weak classifier.

### Cascading Classifiers

In this process, classifiers are combined in order to increases the detector’s speed by focusing on face regions. This works in a way that the initial classifiers are more straightforward and are used to remove most rejectedsub-windows and finally get multiple classifiers that can achieve a low false positive level.

### Eigenface

Eigenface was introduced in 1987 by Sirovich and Kirby.When an image is used as input, the image has a lot of noise, such as poses, background colours, light effects, etc. However, all the images that contain face have several patterns that appear. These patterns are facial features. Facial features of a face are mouth, nose, and eyes and the distance between them. These facial features are referred to as "eigenface" or the principal components in general. In order to extract these facial features, a mathematical method called Principal \Component Analysis (PCA) is used.

### Neural network

Neural Network is inspired by the human brain, which consists of neurons or perceptron that are connected in severalsame or different layers. Neural Network is self-learning, and this can be achieved by training it. In case of face detection, neural network scans every matrix in an

input image, to determine the existence of face. This approach is considered efficient because there is no need to train images which haveno face in it. The process of face detection is divided into two.The first step is to use an area of the image as an input for the filter that made up of the neural network. The result of the filter is an array of -1 or 1, which represents the absence or presence of a face in the image. The second step is to omit false detection in the first step in order to get a better result. To achieve this, all overlapping detection are combined.

## 2.2: Advantages & Disadvantages of Face Detection Methods

|  |  |  |
| --- | --- | --- |
| **Face Detection Method** | **Advantages** | **Disadvantages** |
| Viola Jones Algorithm | 1. High detection Speed. 2. High Accuracy. | 1. Long Training Time. 2. Limited Head Pose. 3. Not able to detect dark faces. |
| Local Binary Pattern Histogram | 1.Simple computation. 2.High tolerance against the monotonic illumination changes. | 1. Only used for binary and grey images. 2. Overall performance is inaccurate compared to Viola- Jones Algorithm. |
| Ada Boost Algorithm | Need not to have any prior knowledge about face structure. | The result highly depends on the training data and affected by weak classifiers. |
| SMQT Features and SNOW Classifier Method | 1. Capable to deal with lighting problem in object detection. 2. Efficient in computation. | The region contain very similar to grey value regions will be misidentified as face. |
| Neural-Network | High accuracy only if large size of image were trained. | 1. Detection process is slow and computation is complex. |

|  |  |  |
| --- | --- | --- |
|  |  | 2. Overall performance is weaker than Viola-Jones algorithm. |

**CHAPTER 3**

**DEEP NEURAL NETWORKS**

**3.1 Introduction to Deep Neural Networks(DNN)**

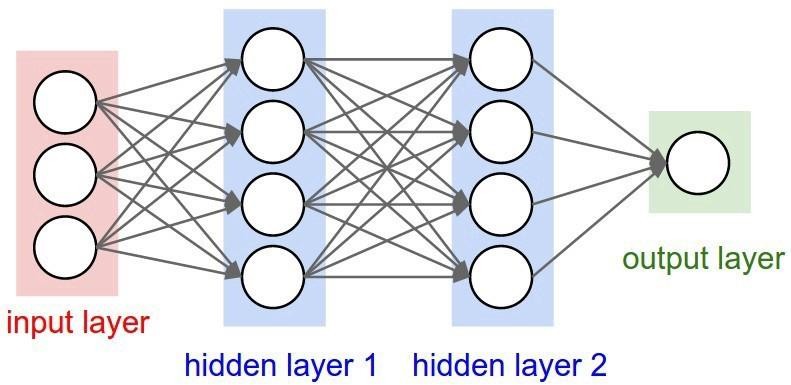
The field of artificial intelligence is essentially when machines can do tasks that typically require human intelligence. It encompasses machine learning, where machines can learn by experience and acquire skills without human involvement.

Deep learning is a subset of machine learning where artificial neural networks, algorithms inspired by the human brain, learn from large amounts of data. Similarly to how we learn from experience, the deep learning algorithm would perform a task repeatedly, each time tweaking it a little to improve the outcome. We refer to ‘deep learning’ because the neural networks have various (deep) layers that enable learning. Just about any problem that requires “thought” to figure out is a problem deep learning can learn to solve.

The amount of data we generate every day is staggering—currently estimated at [2.6](https://web-assets.domo.com/blog/wp-content/uploads/2017/07/17_domo_data-never-sleeps-5-01.png) [quintillion bytes](https://web-assets.domo.com/blog/wp-content/uploads/2017/07/17_domo_data-never-sleeps-5-01.png)—and it’s the resource that makes deep learning possible. Since deep- learning algorithms require a ton of data to learn from, this increase in data creation is one reason that deep learning capabilities have grown in recent years. In addition to more data creation, deep learning algorithms benefit from the stronger computing power that’s available today as well as the proliferation of Artificial Intelligence (AI) as a Service. AI as a Service has given smaller organizations access to artificial intelligence technology and specifically the AI algorithms required for deep learning without a large initial investment.

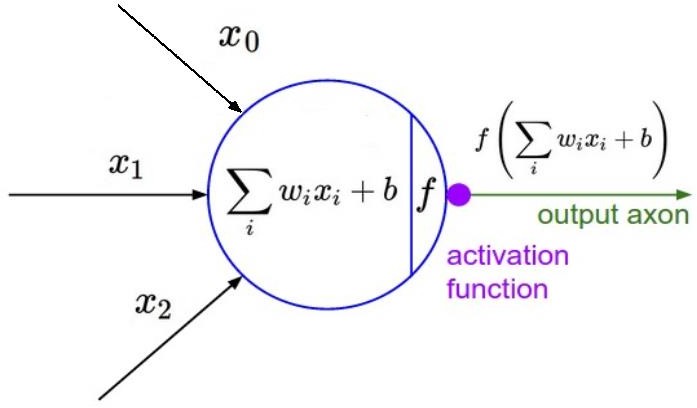
Deep learning allows machines to solve complex problems even when using a data set that is very diverse, unstructured and inter-connected. The more deep learning algorithms learn, the better they perform.

Deep Learning Algorithms use something called a neural network to find associations between a set of inputs and outputs. The basic structure is seen below:



## Neural Network

A neural network is composed of input, hidden, and output layers — all of which are composed of “nodes”. Input layers take in a numerical representation of data (e.g. images with pixel specs), output layers output predictions, while hidden layers are correlated with most of the computation.



## Activation Function

The major points to keep note of here are the tunable weight and bias parameters — represented by w and b respectively in the function above. These are essential to the actual “learning” process of a deep learning algorithm.

After the neural network passes its inputs all the way to its outputs, the network evaluates how good its prediction was (relative to the expected output) through something called a loss function. As an example, the “Mean Squared Error” loss function is shown below.

1 *n* 

^ 2

*n* *Yi* *Yi* 

*i* 1  

Y hat represents the prediction, while Y represents the expected output. A mean is used if batches of inputs and outputs are used simultaneously (n represents sample count).

The goal of my network is ultimately to minimize this loss by adjusting the weights and biases of the network. In using something called “back propagation” through gradient descent, the network backtracks through all its layers to update the weights and biases of every node in the opposite direction of the loss function — in other words, every iteration of back propagation should result in a smaller loss function than before.

Without going into the proof, the continuous updates of the weights and biases of the network ultimately turns it into a precise function approximator — one that models the relationship between inputs and expected outputs.

The “deep” part of deep learning refers to creating deep neural networks. This refers a neural network with a large amount of layers — with the addition of more weights and biases, the neural network improves its ability to approximate more complex functions.

## Different types of Deep Neural Networks

* + 1. **Autoencoders**

An autoencoder is an artificial neural network that is capable of learning various coding patterns. The simple form of the autoencoder is just like the multilayer perceptron, containing an input layer or one or more hidden layers, or an output layer. The significant difference between the typical multilayer perceptron and feedforward neural network and autoencoder is in the number of nodes at the output layer. In the case of the autoencoder, the output layer contains the same amount of nodes as in the input layer. Instead of predicting target values as per the output vector, the autoencoder has to predict its inputs. The broad outline of the learning mechanism is as follows.

For each input x,

* Do a feedforward pass to compute activation functions provided at all the hidden layers and output layers
* Find the deviation between the calculated values with the inputs using appropriate error function
* Backpropagate the error to update weights
* Repeat the task till satisfactory output.

If the number of nodes in the hidden layers is fewer than the input/output nodes, then the activations of the last hidden layer are considered as a compressed representation of the inputs. When the hidden layer nodes are more than the input layer, an autoencoder can potentially learn the identity function and become useless in the majority of the cases.

## Deep Belief Network

A deep belief network is a solution to the problem of handling non-convex objective functions and local minima while using the typical multilayer perceptron. It is an alternative type of deep learning consisting of multiple layers of latent variables with connection between the layers. The deep belief network can be viewed as restricted Boltzmann machines (RBM), where each subnetwork’s hidden layer acts as the visible input layer for the adjacent layer of the network. It makes the lowest visible layer a training set for the adjacent layer of the network. This way, each layer of the network is trained independently and greedily. The hidden variables are used as the observed variables to train each layer of the deep structure. The training algorithm for such a deep belief network is provided as follows:

* Consider a vector of inputs
* Train a restricted Boltzmann machine using the input vector and obtain the weight matrix
* Train the lower two layers of the network using this weight matrix
* Generate new input vector by using the network (RBM) through sampling or mean activation of the hidden units
* Repeat the procedure till the top two layers of the network are reached

The fine-tuning of the deep belief network is very similar to the multilayer perceptron. Such deep belief networks are useful in acoustic modeling.

## Convolutional Neural Networks

A convolutional neural network (CNN) is another variant of the feedforward multilayer perceptron. It is a type of feedforward neural network, where the individual neurons are ordered in a way that they respond to all overlapping regions in the visual area.

Deep CNN works by consecutively modeling small pieces of information and combining them deeper in the network. One way to understand them is that the first layer will try to identify edges and form templates for edge detection. Then, the subsequent layers will try to combine them into simpler shapes and eventually into templates of different object positions, illumination, scales, etc. The final layers will match an input image with all the templates, and the final prediction is like a weighted sum of all of them. So, deep CNNs can model complex variations and behavior, giving highly accurate predictions.

Such a network follows the visual mechanism of living organisms. The cells in the visual cortex are sensitive to small subregions of the visual field, called a receptive field. The subregions are arranged to cover the entire visual area, and the cells act as local filters over the input space. The backpropagation algorithm is used to train the parameters of each convolution kernel. Further, each kernel is replicated over the entire image with the same parameters. There are convolutional operators which extract unique features of the input. Besides the convolutional layer, the network contains a rectified linear unit layer, pooling layers to compute the max or average value of a feature over a region of the image, and a loss layer consisting of application-specific loss functions. Image recognition and video analysis and natural language processing are major applications of such a neural network.

The area of computer vision has witnessed frequent progresses in the past few years. One of the most stated advancements is CNNs. Now, deep CNNs form the core of most sophisticated fancy computer vision applications, such as self-driving cars, gesture recognition, auto- tagging of friends in our Facebook pictures, facial security features, and automatic number plate recognition.

## Recurrent Neural Networks

The convolutional model works on a fixed number of inputs, generates a fix-sized vector as output with a predefined number of steps. The recurrent networks allow us to operate over sequences of vectors in input and output. In the case of recurrent neural network, the connection between units forms a directed cycle. Unlike the traditional neural network, the

recurrent neural network input and output are not independent but related. Further, the recurrent neural network shares the standard parameters at every layer. One can train the recurrent network in a way that is like the traditional neural network using the backpropagation method.

Here, calculation of gradient depends not on the current step but previous steps also. A variant called a bidirectional recurrent neural network is also used for many applications. The bidirectional neural network considers not only the previous but also the expected future output. In two-way and straightforward recurrent neural networks, deep learning can be achieved by introducing multiple hidden layers. Such deep networks provide higher learning capacity with lots of learning data. Speech, image processing, and natural language processing are some of the candidate areas where recurrent neural networks can be used.

## Reinforcement Learning to Neural Networks

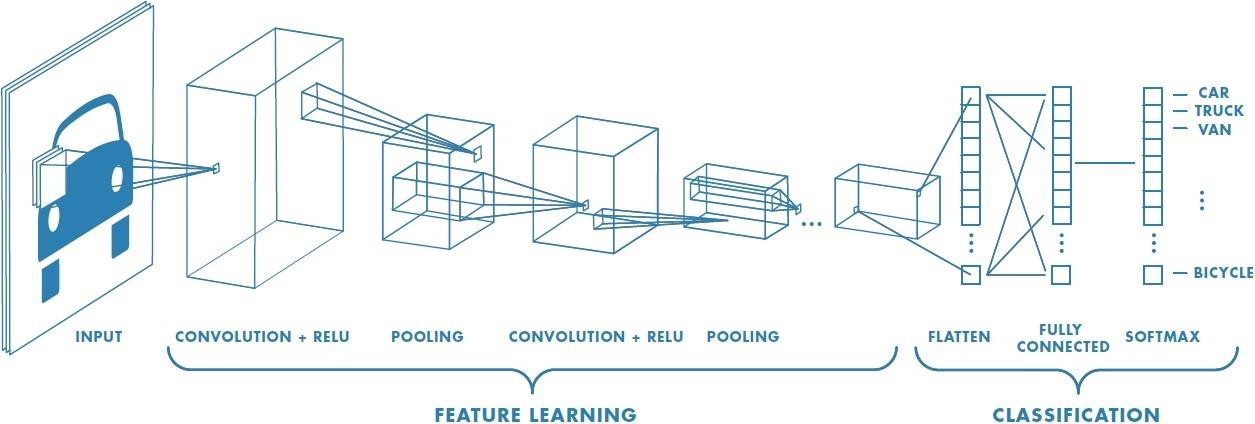
Reinforcement learning is a kind of hybridization of dynamic programming and supervised learning. Typical components of the approach are environment, agent, actions, policy, and cost functions. The agent acts as a controller of the system; policy determines the actions to be taken, and the reward function specifies the overall objective of the reinforcement learning problem. An agent, receiving the maximum possible reward, can be regarded as performing the best action for a given state.

Here, an agent refers to an abstract entity, either an object or a subject (autonomous cars, robots, humans, customer support chatbots, etc.), which performs actions. The state of an agent refers to its position and state of being in its abstract environment; for example, a specific position in a virtual reality world, a building, a chessboard, or the position and speed on a racetrack. Deep reinforcement learning holds the promise of a very generalized learning procedure that can learn useful behavior with very little feedback. It is an exciting and challenging area, which will undoubtedly be an essential part of the future AI landscape.

## CNN and its Architecture

Convolutional Neural networks are Artificial Deep Neural Networks. In Deep neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories

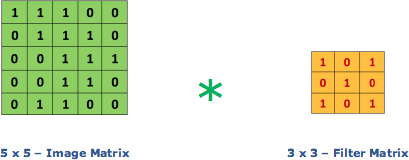
to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, 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.



## 13.3 Neural network with many convolutional layers

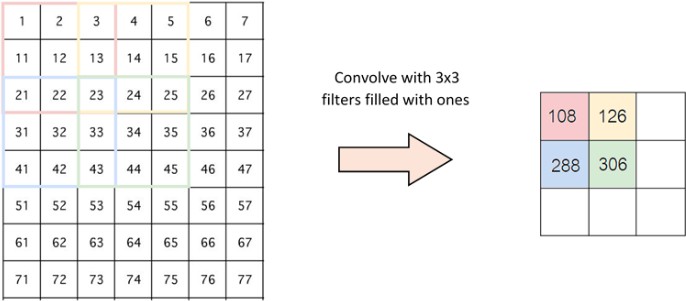
## Convolution Layer

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.

Consider a 5 x 5 whose image pixel values are 0, 1 and filter matrix 3 x 3, then the convolution of 5 x 5 matrices which is called “Feature Map” .Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters.

## Image matrix and Filter matrix

## 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.

## Stride of 2 pixels

## Padding

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

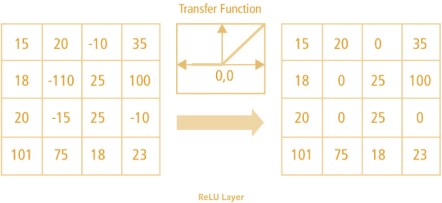
* Pad the picture with zeros (zero-padding) so that it fits
* Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

## Non Linearity (ReLU)

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is

*f* *x*  max0, *x*

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.



## ReLU Operation

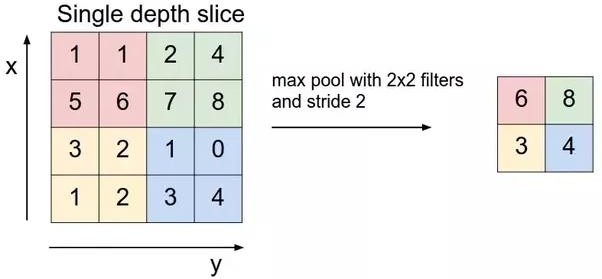
There are other non linear functions such as tanh or sigmoid that can also be used instead of ReLU. Most of the data scientists use ReLU since performance wise ReLU is better than the other two.

## 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

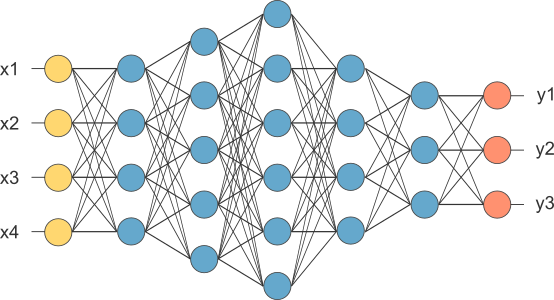
Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.



## Max Pooling

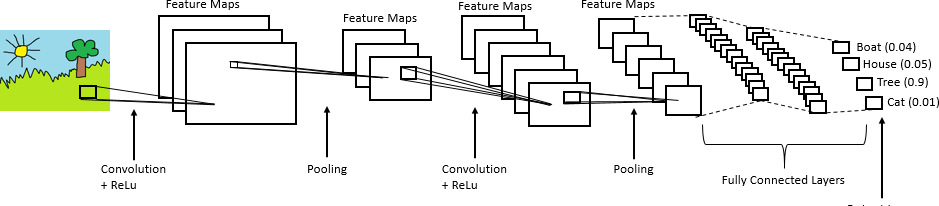
## Fully Connected Layer

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.



## After pooling layer, flattened as FC layer

In the above diagram, the feature map matrix will be converted as vector (x1, x2, x3, …). With the fully connected layers, we combined these features together to create a model. Finally, we have an activation function such as softmax or sigmoid to classify the outputs.



## Complete CNN architecture

## Siamese Network

The word “Siamese” means joined or connected. Siamese Networks could consist of [Convolutional layers](https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8) as well as [Dense](https://forums.fast.ai/t/dense-vs-convolutional-vs-fully-connected-layers/191) or [LSTM layers](https://skymind.ai/wiki/lstm). Since we are going to deal with images for detecting the faces, we will utilize a Convolutional Siamese Network.

Yes, we can build a standard CNN to classify the images into a fixed number of classes ( or persons ) in our case.

### But, there could be a number of difficulties here:

1. We need a large number of images which belong to different people.
2. High computation power is required to train such a model.
3. If the number of people whose face needs to be classified changes, we need to reconstruct the whole CNN from scratch.
4. Moreover it requires high end System with GPU and CUDA.

### But, with Siamese networks:

1. We need only a handful of images from each class or of a person.
2. They learn similarity functions. Hence, they don’t classify, but, differentiate images.
3. They are easy to train and have a simple architecture.

## Applications of DNN

Here are just a few of the [tasks that deep learning supports](https://bernardmarr.com/default.asp?contentID=1742) today and the list will just continue to grow as the algorithms continue to learn via the infusion of data.

### Virtual assistants

Whether it’s Alexa or Siri or Cortana, the virtual assistants of online service providers use deep learning to help understand your speech and the language humans use when they ineract with them.

### Translations

In a similar way, deep learning algorithms can automatically translate between languages. This can be powerful for travelers, business people and those in government.

### Vision for driverless delivery trucks, drones and autonomous cars

The way an autonomous vehicle understands the realities of the road and how to respond to them whether it’s a stop sign, a ball in the street or another vehicle is through deep learning algorithms. The more data the algorithms receive, the better they are able to act human-like in their information processing—knowing a stop sign covered with snow is still a stop sign.

### Chatbots and service bots

Chatbots and service bots that provide customer service for a lot of companies are able to respond in an intelligent and helpful way to an increasing amount of auditory and text questions thanks to deep learning.

### Image colorization

Transforming black-and-white images into color was formerly a task done meticulously by human hand. Today, deep learning algorithms are able to use the context and objects in the images to color them to basically recreate the black-and-white image in color. The results are impressive and accurate.

### Facial recognition

Deep learning is being used for facial recognition not only for security purposes but for tagging people on Facebook posts and we might be able to [pay for items in a store just by](https://www.technologyreview.com/s/603494/10-breakthrough-technologies-2017-paying-with-your-face/) [using our faces](https://www.technologyreview.com/s/603494/10-breakthrough-technologies-2017-paying-with-your-face/) in the near future. The challenges for deep-learning algorithms for facial recognition is knowing it’s the same person even when they have changed hairstyles, grown or shaved off a beard or if the image taken is poor due to bad lighting or an obstruction.

### Medicine and pharmaceuticals

From disease and tumor diagnoses to personalized medicines created specifically for an individual’s genome, deep learning in the medical field has the attention of many of the largest pharmaceutical and medical companies.

### Personalized shopping and entertainment

Ever wonder how Netflix comes up with suggestions for what you should watch next? Or where Amazon comes up with ideas for what you should buy next and those suggestions are exactly what you need but just never knew it before? Yep, it’s deep-learning algorithms at work.

The more experience deep-learning algorithms get, the better they become. It should be an extraordinary few years as the technology continues to mature.

# CHAPTER 4

**FACE DETECTION AND RECOGNITION FROM LIVE VIDEO**

## Capture video

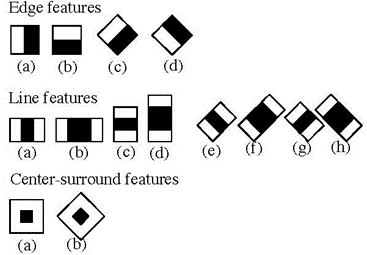
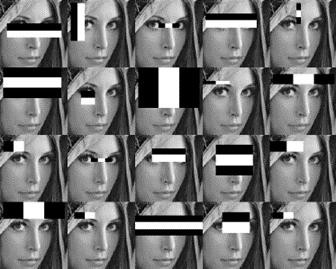
The Camera is fixed at a specific distance inside a classroom to capture videos of the frontal images of the entire students of the class.

## Separate as frames from the video

The captured video needs to be converted into frames per second for easier detection and recognition of the students' face to generate the attendance database.

## Face Detection

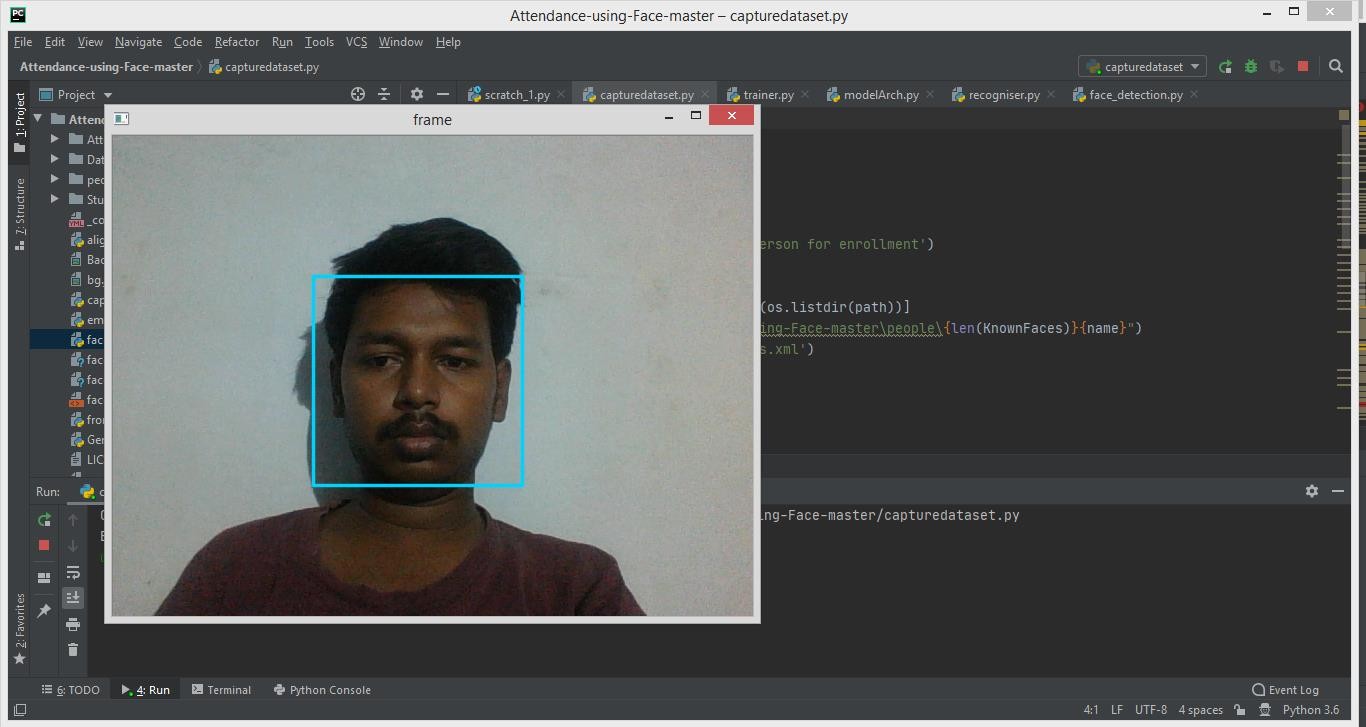
Face detection is acheived by using haar cascades of opencv. Algorithm used for finding the location of the human faces in a frame or image. All human faces shares some universal properties of the human face like the eyes region is darker than its neighbor pixels and nose region is brighter than eye region.



## Human Faces In a Frame

The haar-like algorithm is also used for feature selection or feature extraction for an object in an image, with the help of edge detection, line detection, center detection for detecting eyes, nose, mouth, etc. in the picture. It is used to select the essential features in an image and extract these features for face detection.

The next step is to give the coordinates of x, y, w, h which makes a rectangle box in the picture to show the location of the face or we can say that to show the region of interest in the image. After this, it can make a rectangle box in the area of interest where it detects the face. There are also many other detection techniques that are used together for detection such as smile detection, eye detection, blink detection, etc.

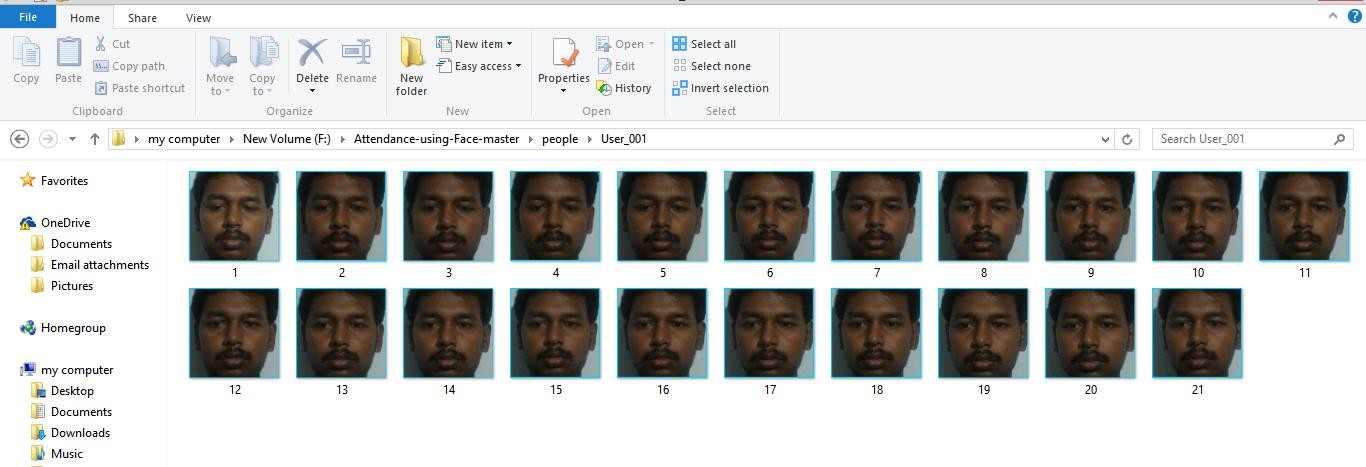


## Face Detection

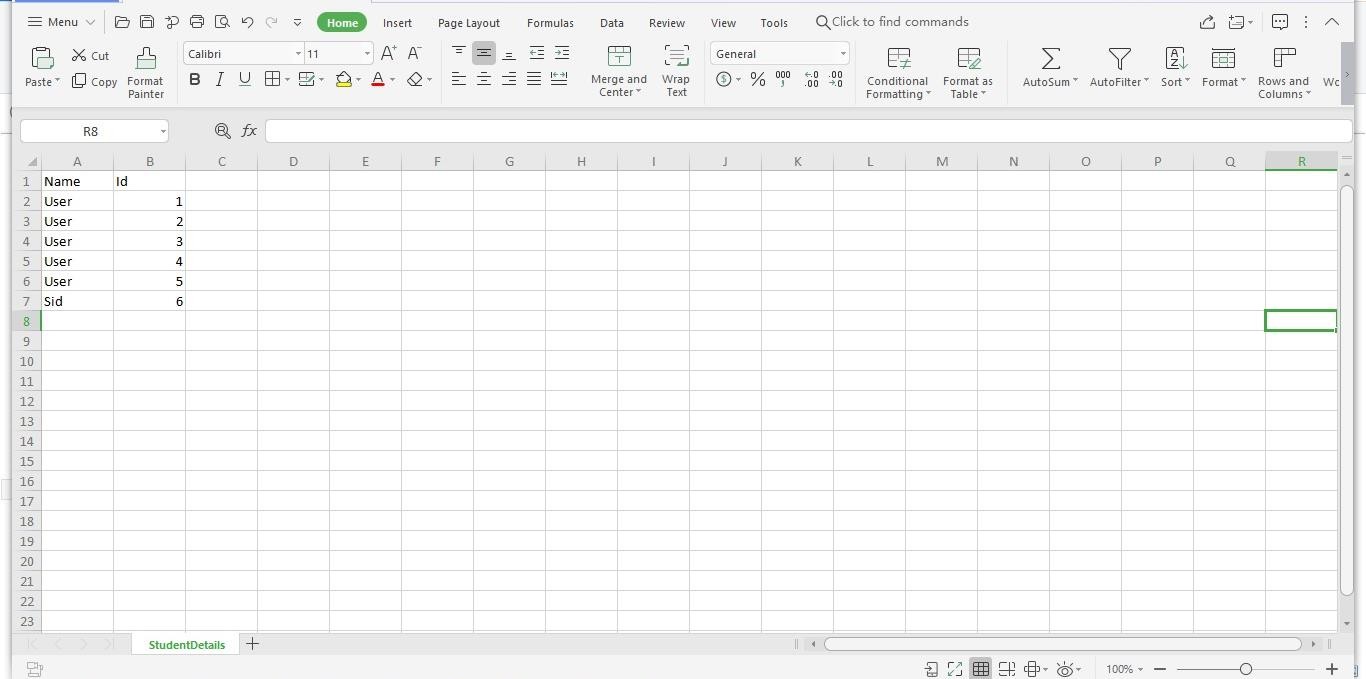
## Pre-Processing

Face detection algorithms work on greyscale images as color is not an important attribute. This step uses OpenCV methods which convert training images to grayscale. These greyscale images are used for further processing down the pipeline. Firstly the image is captured from video frame. Then the picture is transformed from RGB to Grayscale because it is easy to detect faces in the grayscale. After that, the image manipulation used, in which the resizing, cropping, blurring and sharpening of the images done if needed. The next step is image

segmentation, which is used for contour detection or segments the multiple objects in a single image so that the classifier can quickly detect the objects and faces in the picture. These pictures are stored as a dataset inside the folder with Username,id as folder name in people folder.At the same time these Username with Id is updated in the StudentDetails sheet in StudentDetails folder.



## Dataset

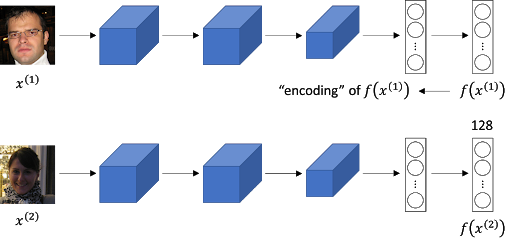


* 1. **StudentDetails**

## Model Training

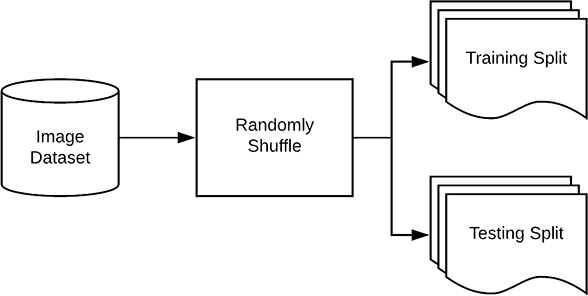
The system is based on a special type of cnn architecture known as a siamese network.It is a network which uses two Neural Networks parallely. One Neural Network iterates through images and the other Neural Network makes prediction on the images it sees. The second Neural Network Tries to find corelation from the weights of the first Neural Network if the weights match to some extent we finally get the label. Such a network is trained to generate a very accurate and almost unique 128 vector given that the images of face which a are fed to the network are properly aligned and cropped.Training a Siamese neural network is slightly different than training a regular CNN. Recall that when training a CNN, the training samples are arrays of images, along with the corresponding class label for each image. In contrast, to train a Siamese neural network we need to use pairs of arrays of images, along with the corresponding class label for the pairs of images. It is an approach to getting a neural net to do one-shot classification is to give it two images and train it to guess whether they have the same category. Then when doing a one-shot classification task described above, the network can compare the test image to each image in the support set, and pick which one it thinks is most likely to be of the same category. So we want a neural net architecture that takes two images as input and outputs the probability they share the same class.

If we just concatenate two examples together and use them as a single input to a neural net, each example will be matrix multiplied(or convolved) with a different set of weights, which breaks symmetry. Sure it’s possible it will eventually manage to learn the exact same weights for each input, but it would be much easier to learn a single set of weights applied to both inputs. So we could propagate both inputs through identical twin neural nets with shared parameters, then use the absolute difference as the input to a linear classifier - this is essentially what a siamese net is. Two identical twins, joined at the head, hence the name. Then another dense neural network is trained taking input these embeddings. The second neural network is only for classification purposes. Then the person who is identified by the system, his/her attendance in the system is incremented by 1.When the system is closed, a excel file consisting of attendance of all the students is generated.



## Siamese Network(Type of CNN)

Firstly, the images in dataset are split as we need to construct our training and testing splits



## Splitting of Dataset

It is typical to allocate a percentage of your data for training and a smaller percentage of your data for testing. The scikit-learn provides a handy train\_test\_split function which will split the data for us.Both trainX and testX make up the image data itself while trainY and testY make up the labels.Our class labels are currently represented as strings; however, Keras will assume that both:

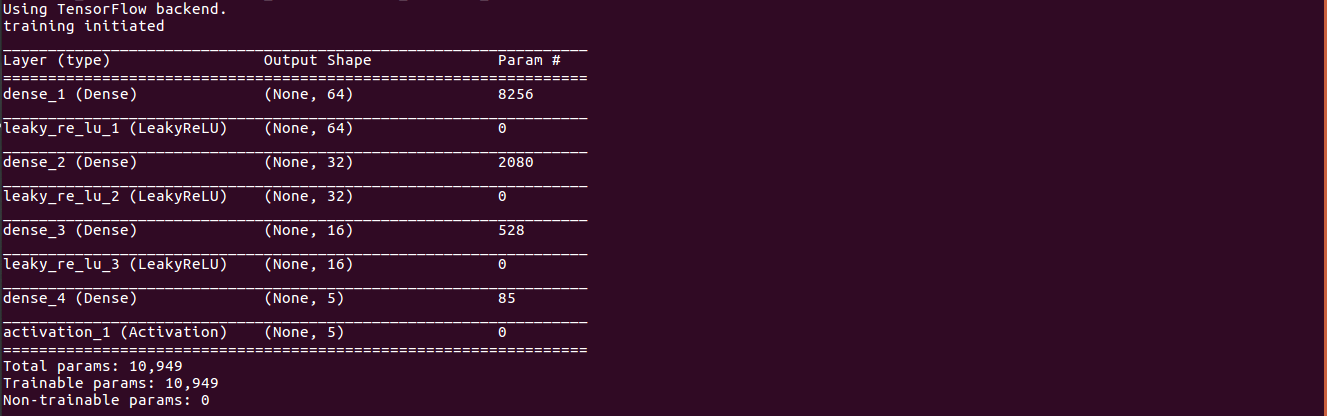
* Labels are encoded as integers
* And furthermore, one-hot encoding is performed on these labels making each label represented as a vector rather than an integer

A call to fit\_transform finds all unique class labels in trainY and then transforms them into one-hot encoded labels.

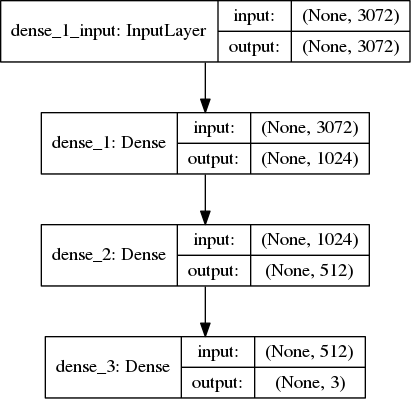
A call to just .transform on testY performs just the one-hot encoding step — the unique set of possible class labels was already determined by the call to .fit\_transform.

Now Keras model is defined using these encodings using embedding.py which loads the ‘facenet\_keras.h5’ model. This network model is pretrained on a pretty large dataset, and produces a unique 128 dimensional vector for a particular face given the images fed to it are cropped to only the face region and are aligned. The input size of image for this network is 160X160X3.

The second neural network has a dense architecture and is used for classification. The second neural network take input the 128 dimensional vector and ouputs the probability of the face to be one of the student. The architecture of the second neural network is in the figure below.

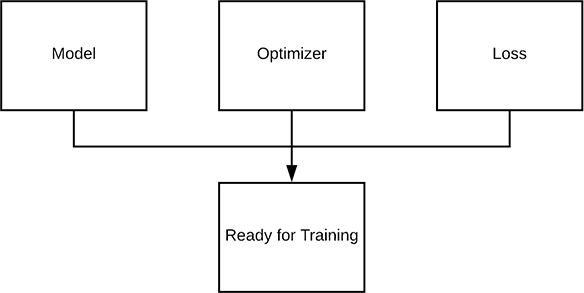


## Second neural network



* 1. **DenseArchs**

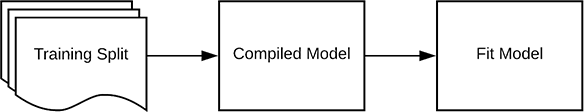
Once we have defined our deep neural network architecture, the next step is to “compile” it:



## Compile Keras model with an optimizer and loss function

First, we initialize our learning rate and total number of epochs to train for training images. Then we compile our model using the Stochastic Gradient Descent (SGD) optimizer with "categorical\_crossentropy" as the loss function.

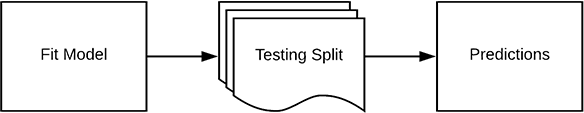
Now that our Keras model is compiled, we can “fit” (i.e., train) it on our training data:



## Fit Keras model to the data

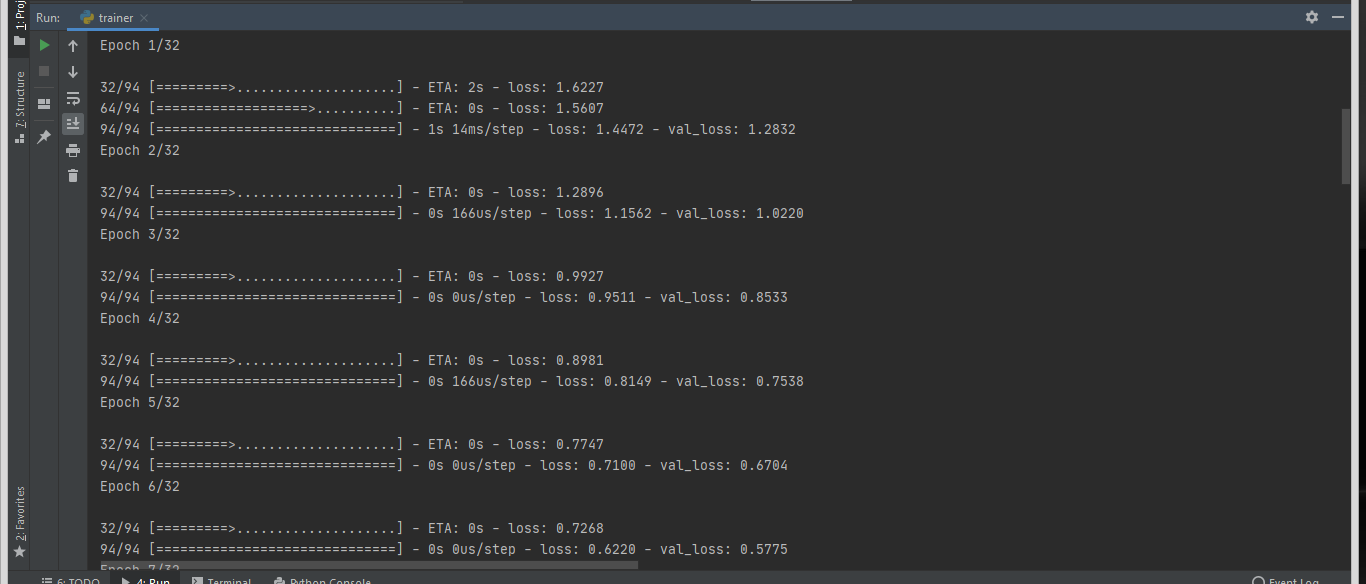
The batch\_size controls the size of each group of data to pass through the network. Larger GPUs would be able to accommodate larger batch sizes. I recommend starting with 32 or 64 and going up from there.

We have trained our actual network model but now we need to evaluate it on our testing data.



## Evaluate Keras model

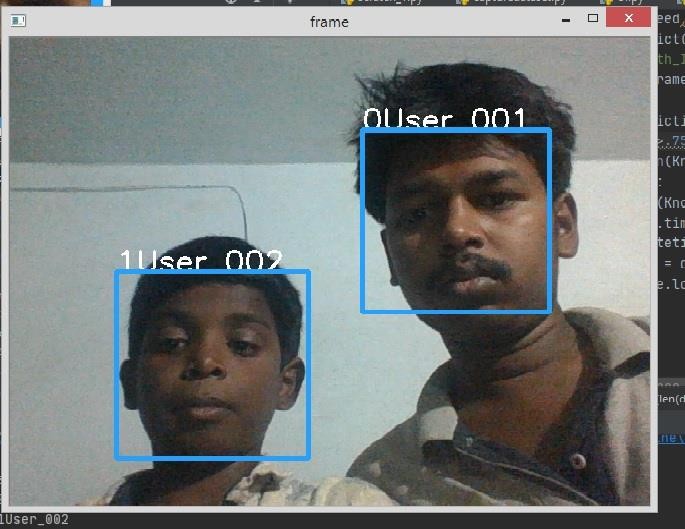
It’s important that we evaluate on our testing data so we can obtain an unbiased (or as close to unbiased as possible) representation of how well our model is performing with data it has never been trained on.



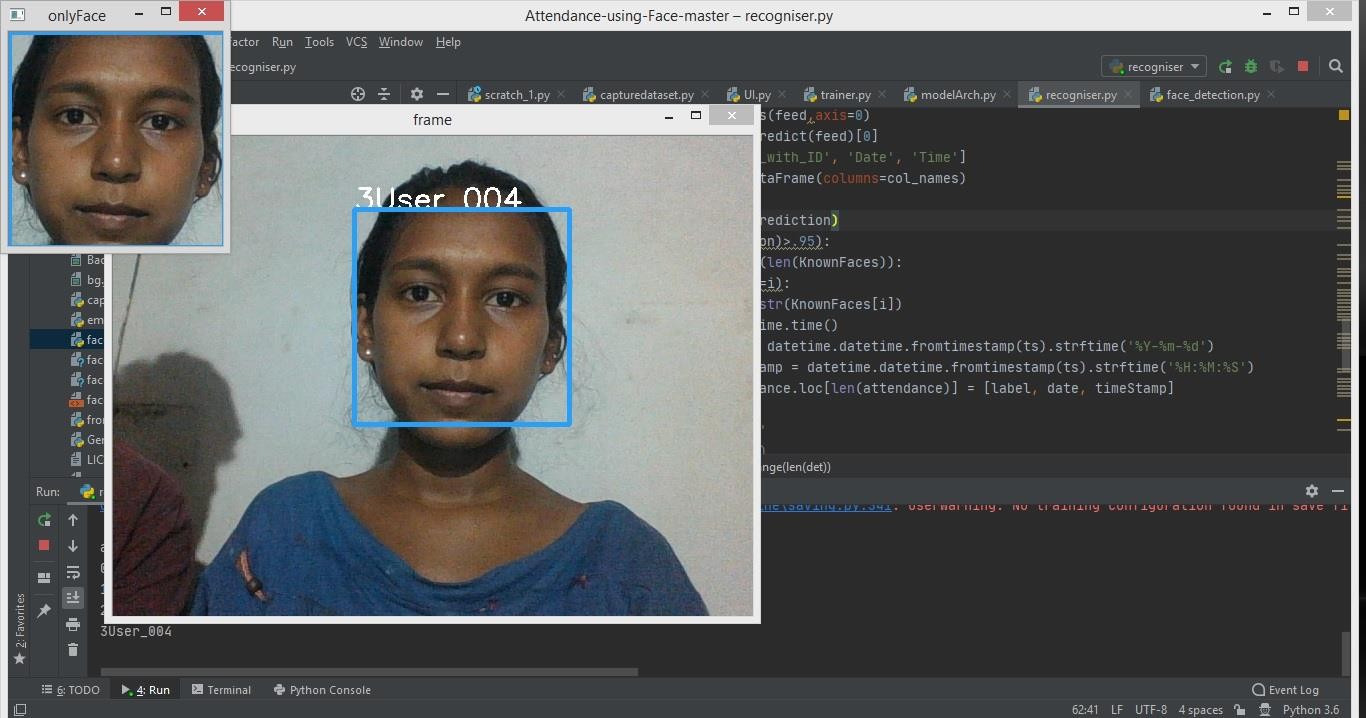
## Training Images

## Face Recognition

Now the frames of images are captured from frame and tested with the trained data model to predict the output as the User or Unknown based on the threshold and produce on the frame.



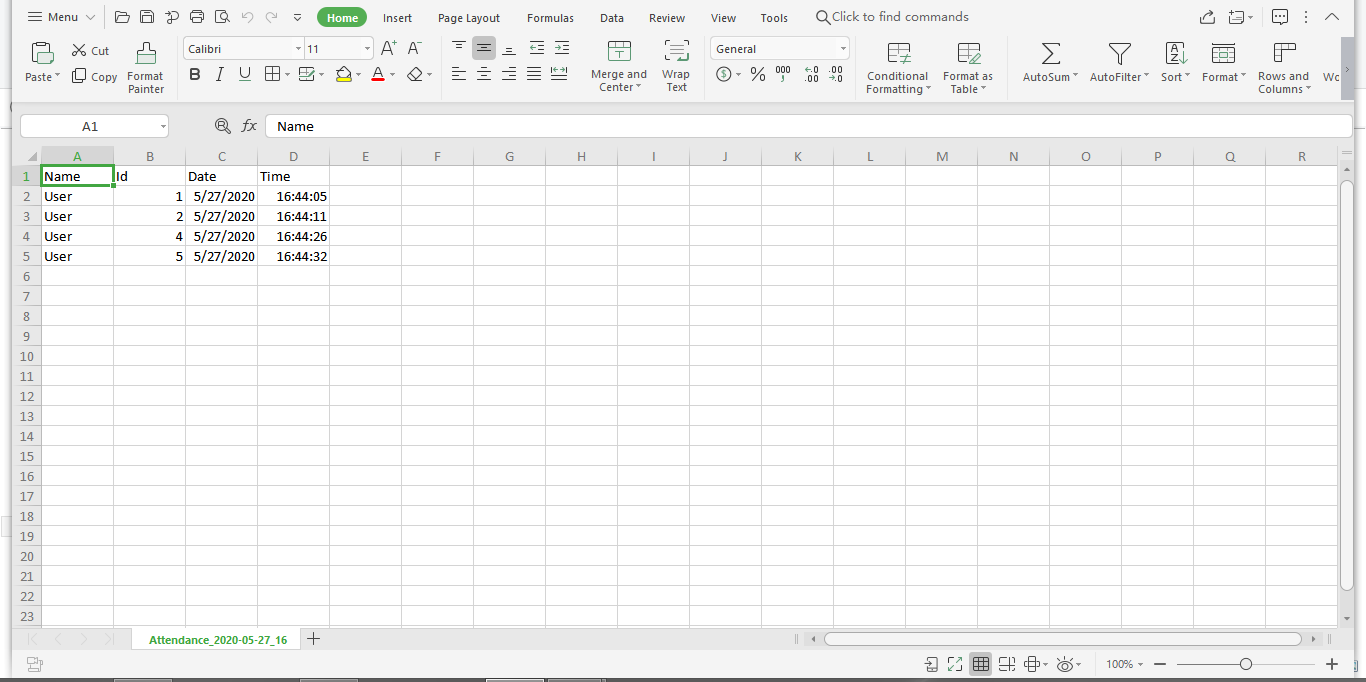
## Face Recognition



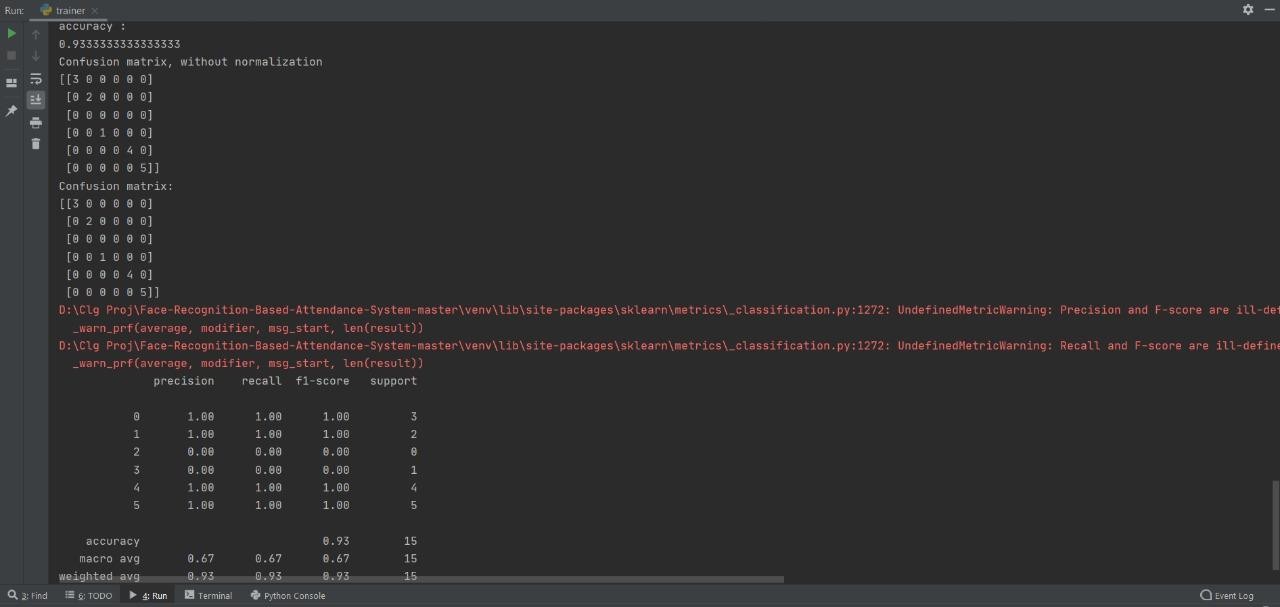
* 1. **Face Recognition with only Face Frame**

## Post-Processing

The post-processing mechanism involves the process of updating the names of the student into an excel sheet.After the application is closed, an excel file is generated. This excel file contains the attendance of all the student. The excel sheet can be maintained on a weekly basis or monthly basis to record the students' attendance. This attendance record can be sent to parents or guardians of students to report the performance of the student.

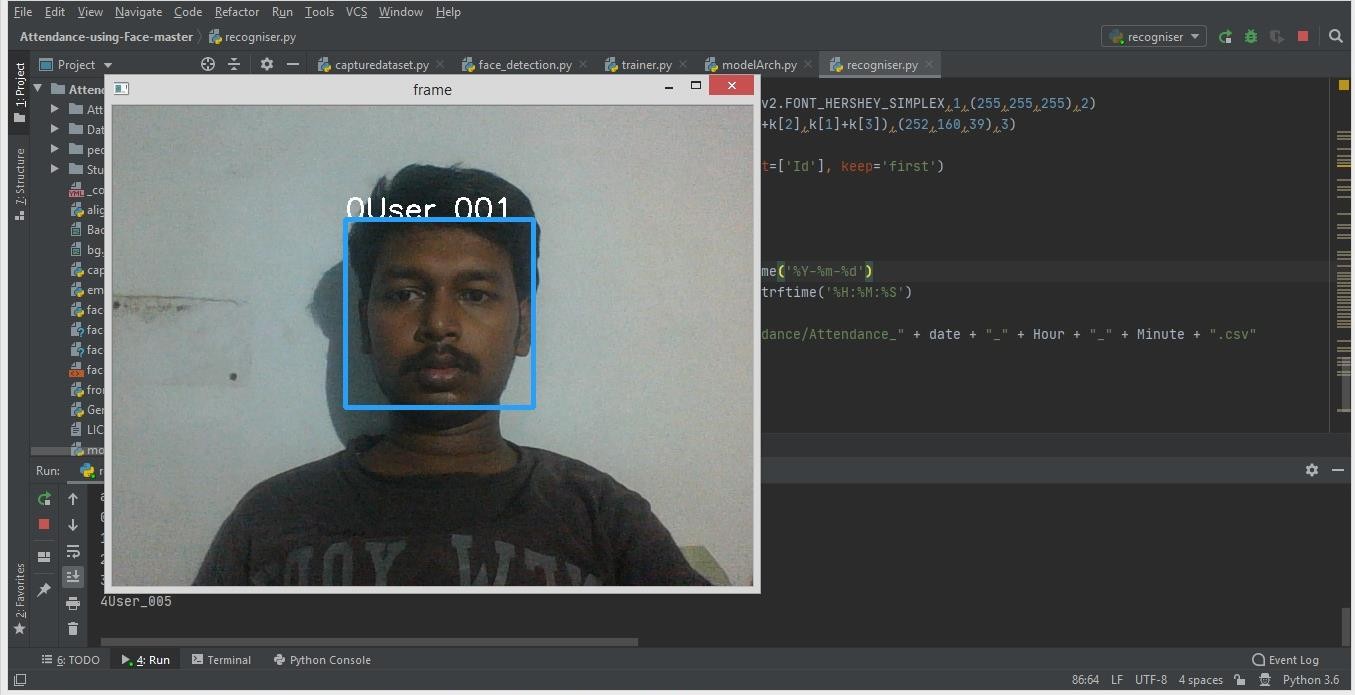


## Attendance In Excel Sheet

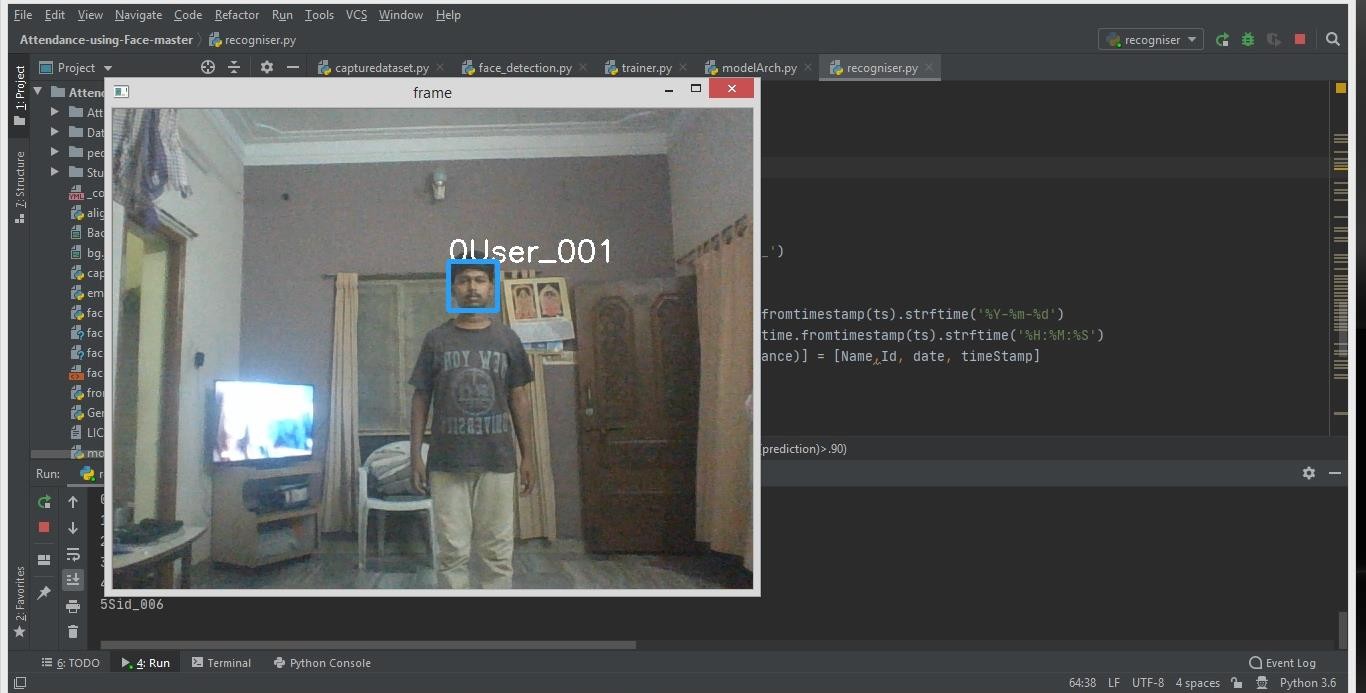


* 1. **Performance Metrics**
  2. **Cases in Face Recognition Able to Recognise:**

Under Medium Light conditions with Straight Face Posture without any turns and Distance upto 2-3 metres.



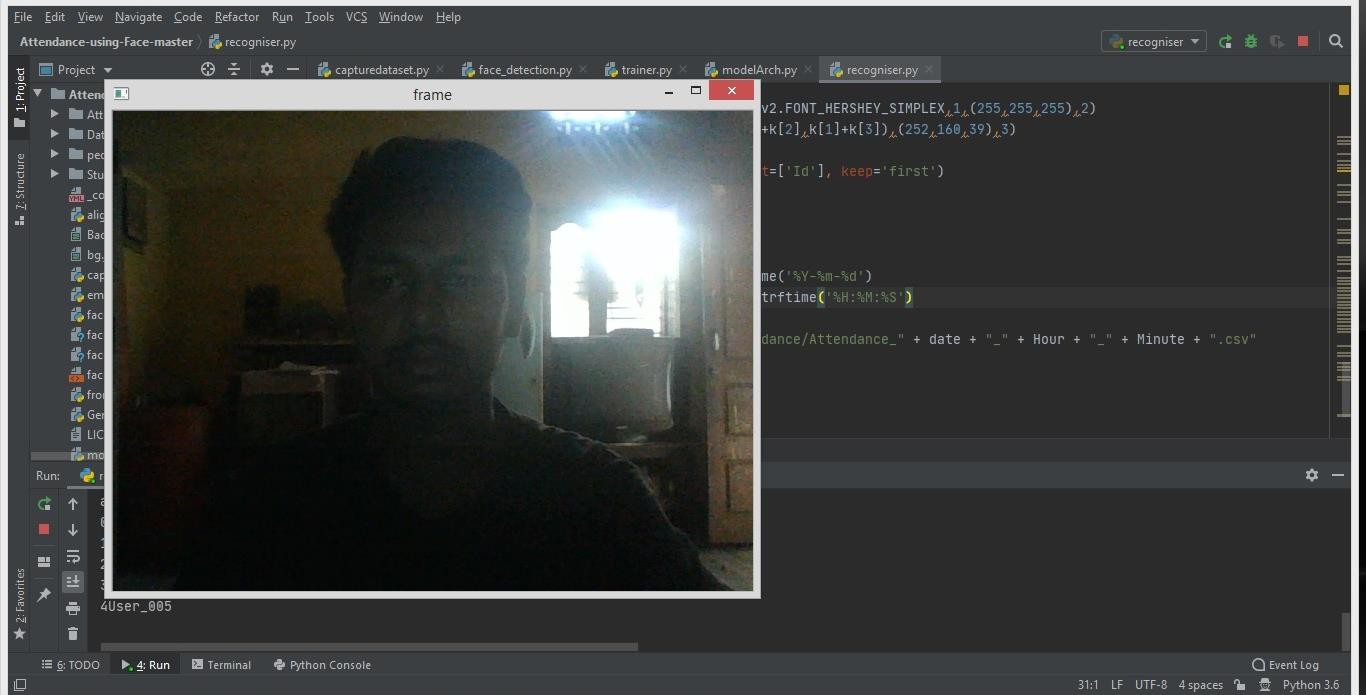
## Recognition under Normal Condition



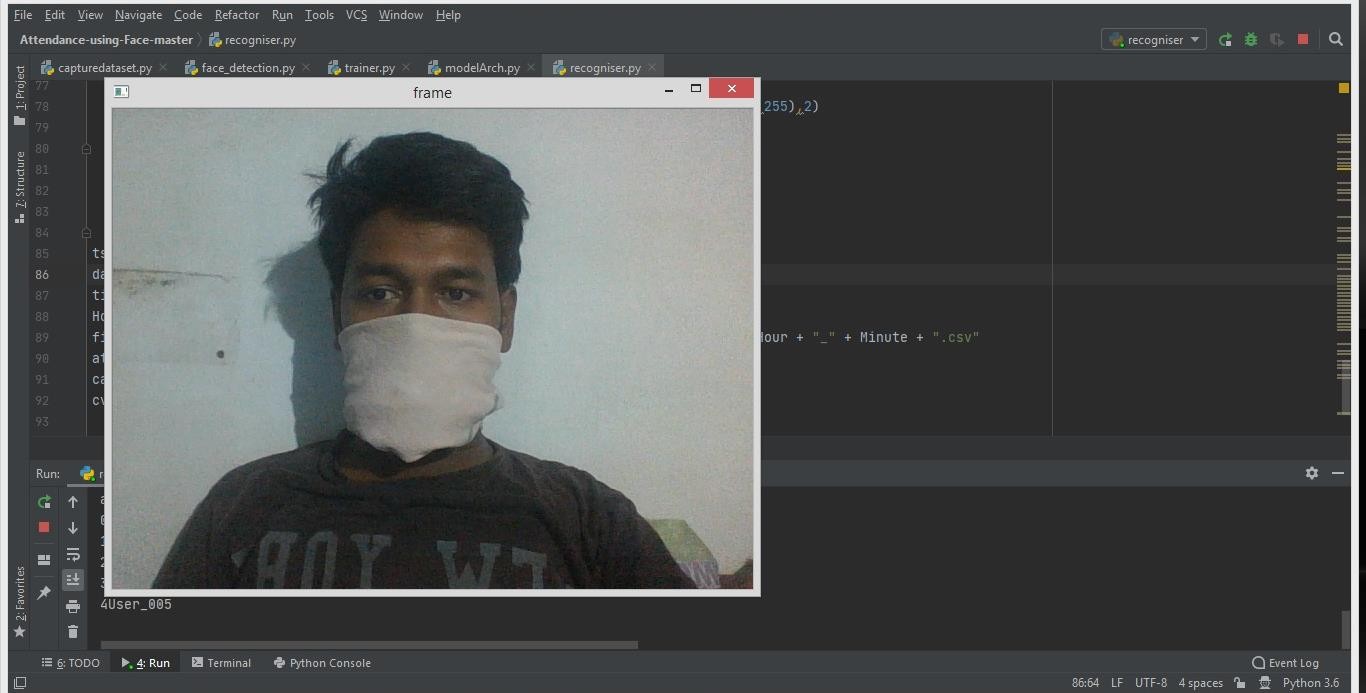
* 1. **Recognition at a Distance**

**Not able to Recognise or Detect:**

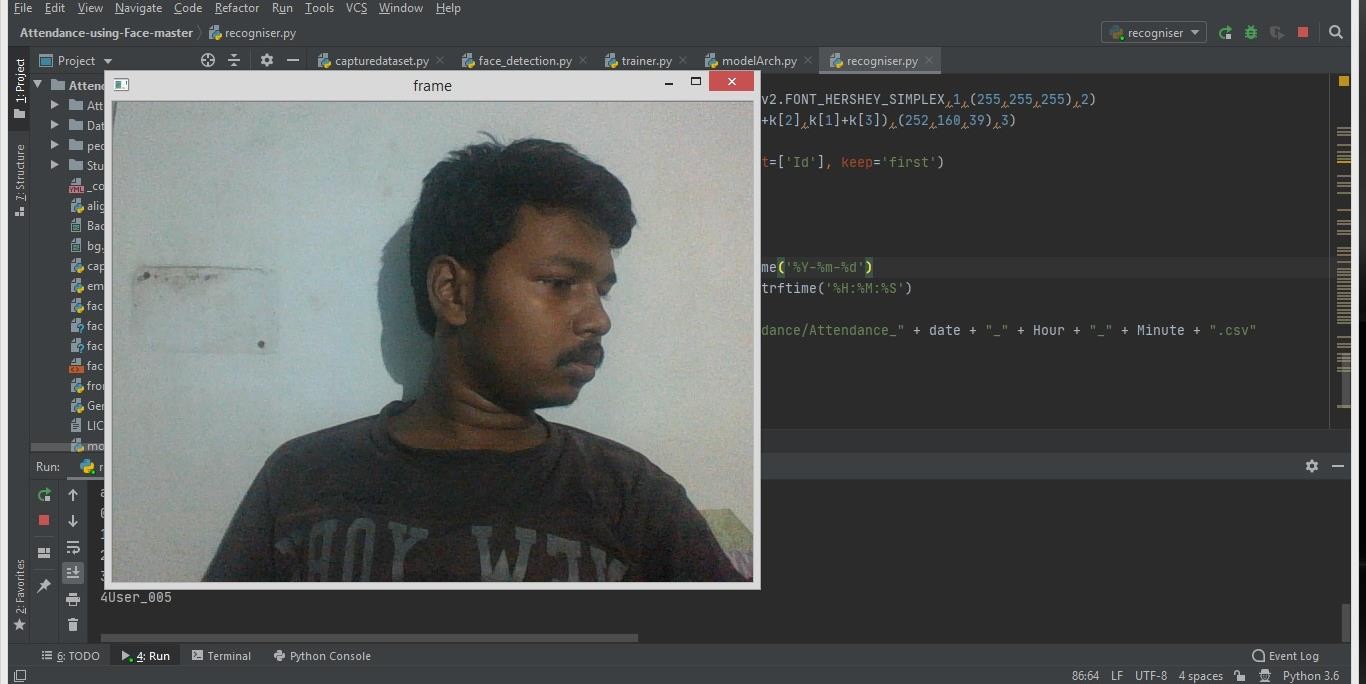
No Detection of Face in cases like Low light condition, wearing mask, facing side angles.



## No Detection in Low Light

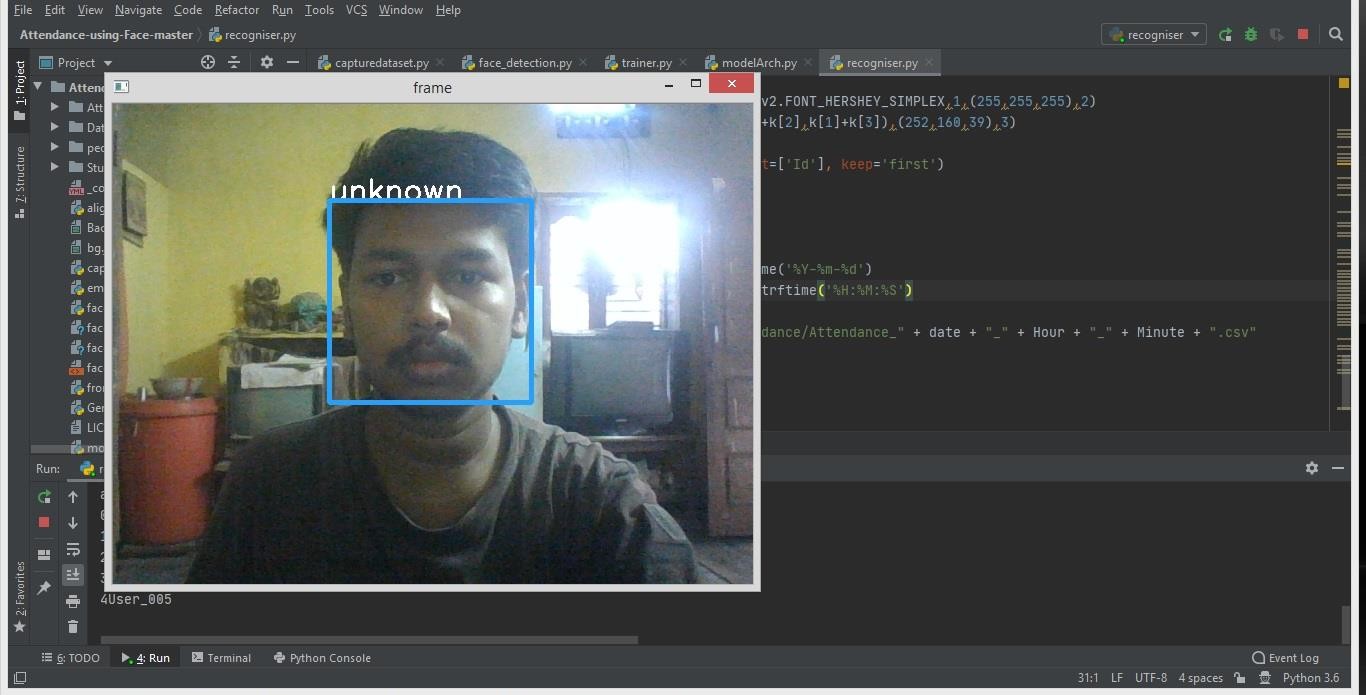


* 1. **No Detection with Mask**

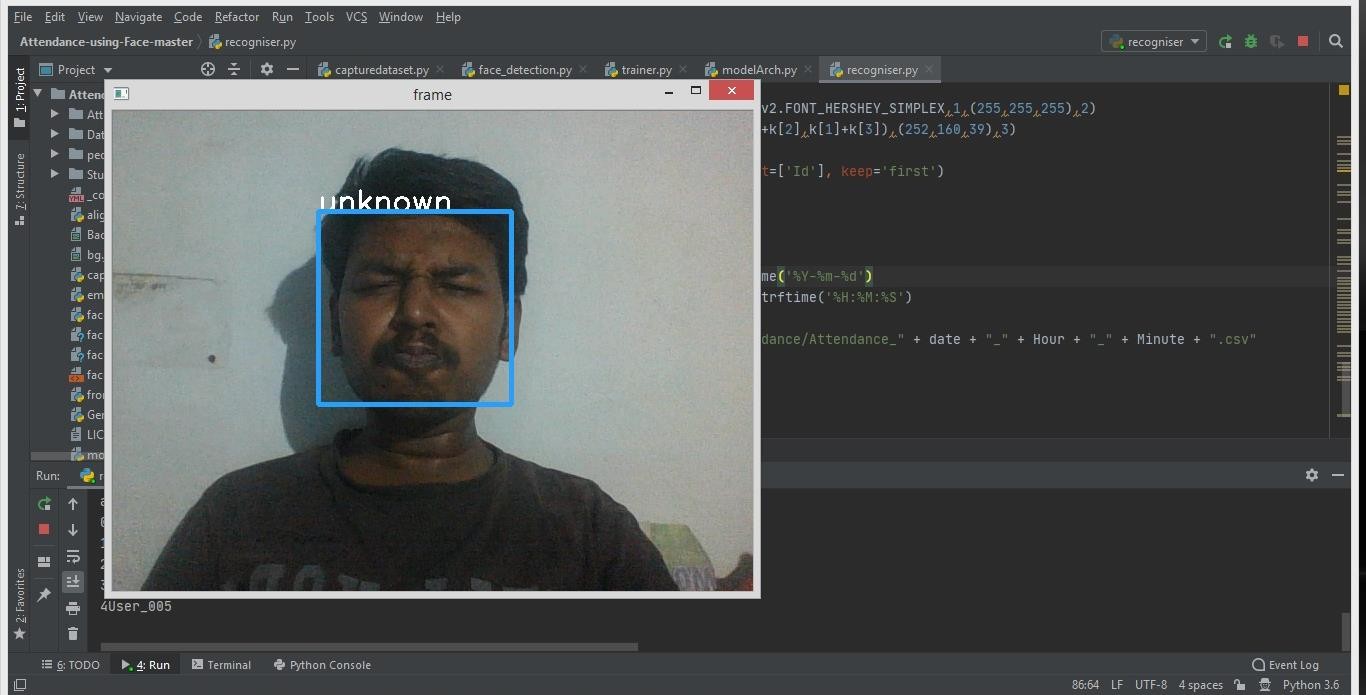


* 1. **No Detection when facing side angles**

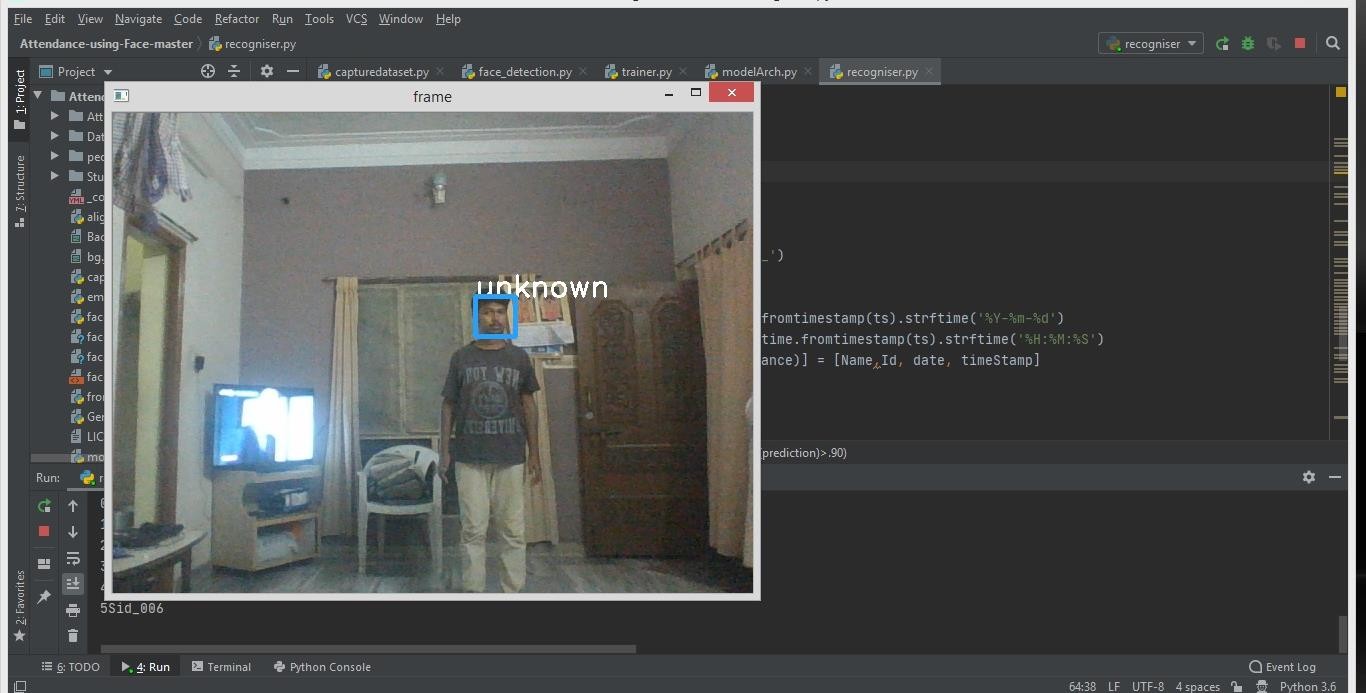
No Recognition of Face in cases like Bright Light conditions, change of facial expressions and Distance greater than 3 metres.



## No Recognition under Bright Light



* 1. **No Recognition under Change of Expressions**



* 1. **No Recognition beyond Certain Distance**

# CHAPTER 5 CONCLUSION AND FUTURE SCOPE

The smart and automated attendance system can be proven as an efficient system for classroom attendance. By using this system, the chances of fake attendance and proxies can be reduced. There are a lots of Biometric Systems which can be used for managing attendance, but the face recognition has the best performance. So we need to implement a reliable and efficient attendance system for classroom attendance which can work for multiple face recognition at one time. Also to implement this system, no any specialized hardware is required. A camera device and a standalone PC, database servers are sufficient for constructing the smart attendance system. With the help of a divergent combination of algorithms, this system helps us to achieve desired results with better accuracy and less time consumption.

Using Machine Learning, Python3, the Face Recognition Attendance System has been developed. All modules from the front end, the processing python script and the back end database are working perfectly. The teachers were provided a way of digitally updating attendance without the hassle of paperwork to mark and recall current attendance status. This system can take over the existing methods followed in institutions and can turn out to be very efficient and with many advantages, providing ease to the teachers, students and education as a whole.

Deep learning is ultimately an expansive field, and is far more complex than I’ve described it to be. Various types of neural networks exist for different tasks (e.g. Convolutional NN for computer vision, Recurrent NN for NLP).

Deep learning though is being applied on many of the AI related areas for better performance, its ability is still largely untapped. There is an ample opportunity to apply DL in the field of medicine, precision agriculture,etc. NVIDIA has increased the lifespan of DL by creating CUDA equivalent for Deep Neural Network (cuDNN). Thus DL has a scope to tackle wide variety of problem in near future.

# REFERENCES

1. Deep learning with python by Francois Chollet
2. keras.io
3. Dan Wang, Rong Fu, Zuying Luo,"Classroom Attendance Auto-management Based on Deep Learning", Advances in Social Science, Education and Humanities Research, volume 123,ICESAME 2017.
4. Pyimagesearch.com(Adrian Rosenberg)
5. Lawrence, S., Giles, C. L., Tsoi, A. C., & Back, A. D. (1997). Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks, 8(1), 98-113.
6. Modern Face Recognition with Deep learning. WebsiteReference: [https://medium.com/@ageitgey/machinelearning-is-fun-part-4-modern-face-recognitionwith-](https://medium.com/%40ageitgey/machinelearning-is-fun-part-4-modern-face-recognitionwith-) deep- learning.
7. Mala Sundaram and Ambika Mani (July 6th 2016). Face Recognition: Demystification of Multifarious Aspect in Evaluation Metrics, Face Recognition - Semisupervised Classification, Subspace Projection and Evaluation Methods, S. Ramakrishnan, IntechOpen, DOI: 10.5772/62825. Available from: [https://www.intechopen.com/books/face-recognition- semisupervised-classification-subspace-projection-and-evaluation-methods/face-recognition- demystification-of-multifarious-aspect-in-evaluation-metrics](https://www.intechopen.com/books/face-recognition-semisupervised-classification-subspace-projection-and-evaluation-methods/face-recognition-demystification-of-multifarious-aspect-in-evaluation-metrics)
8. H. Wu, Z. Xu, J. Zhang, W. Yan and X. Ma, "Face recognition based on convolution siamese networks," 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Shanghai, 2017, pp. 1-5, doi: 10.1109/CISP-BMEI.2017.8302003.

# APPENDIX

**Capture Images:**

import os import cv2 import csv

path= "people" pic\_no=0

print('Enter the name of the person for enrollment') name=input()

KnownFaces= [f for f in sorted(os.listdir(path))]

os.makedirs(f"F:\Attendance-using-Face-master\people\{len(KnownFaces)}{name}") fa=cv2.CascadeClassifier('faces.xml')

cap=cv2.VideoCapture(0) ret=True

while ret: ret,frame=cap.read() frame=cv2.flip(frame,1)

gray=cv2.cvtColor(frame,cv2.COLOR\_BGR2GRAY) faces=fa.detectMultiScale(gray,1.3,5)

for (x,y,w,h) in faces: cropped=frame[y:y+h,x:x+w]

cv2.rectangle(frame,(x,y),(x+w,y+h),(255,210,0),2,cv2.LINE\_AA) pic\_no=pic\_no+1 cv2.imwrite(f"{path}\{str(len(KnownFaces))}{name}\{str(pic\_no)}"+'.jpg',cropped)

cv2.imshow('frame',frame) cv2.waitKey(100)

if(pic\_no>20): break

cap.release() cv2.destroyAllWindows() Name,Id=name.split('\_') row=[Name,Id]

with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile: writer = csv.writer(csvFile)

writer.writerow(row) csvFile.close()

**Face Detection:**

import cv2 class face:

def init (self): self.cascade=cv2.CascadeClassifier('faces.xml') self.x=None

self.y=None self.w=None self.h=None

def detectFace(self,img): cropped=None

grey=cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) faces=self.cascade.detectMultiScale(grey,1.3,5) cropped=[]

coor=[]

for (self.x,self.y,self.w,self.h) in faces: cropped.append(img[self.y:self.y+self.h,self.x:self.x+self.w]) coor.append([self.x,self.y,self.w,self.h])

return cropped,coor

**Training Images:**

from modelArch import DenseArchs import cv2

import numpy as np import os

import tensorflow as tf

from embedding import emb

from keras.optimizers import Adam

from sklearn.model\_selection import train\_test\_split from keras.utils import to\_categorical

import matplotlib.pyplot as plt

from sklearn.metrics import confusion\_matrix from sklearn.metrics import classification\_report from sklearn.metrics import accuracy\_score import itertools

n\_classes=5

e=emb() arc=DenseArchs(n\_classes) face\_model=arc.arch()

x\_data=[] y\_data=[]

learning\_rate=0.001 epochs=32 batch\_size=32

people=os.listdir('people')

path= 'people' for x in people:

for i in os.listdir('people/' + x): img=cv2.imread('people'+'/'+x+'/'+i,1) img=cv2.resize(img,(160,160)) img=img.astype('float')/255.0 img=np.expand\_dims(img,axis=0) embs=e.calculate(img) x\_data.append(embs) y\_data.append(int(x[0]))

x\_data=np.array(x\_data,dtype='float') y\_data=np.array(y\_data) y\_data=y\_data.reshape(len(y\_data),1)

x\_train,x\_test,y\_train,Y\_test=train\_test\_split(x\_data,y\_data,test\_size=0.1,random\_state=77) y\_train=to\_categorical(y\_train,num\_classes=n\_classes) y\_test=to\_categorical(Y\_test,num\_classes=n\_classes)

o=Adam(lr=learning\_rate,decay=learning\_rate/epochs) face\_model.compile(optimizer=o,loss='categorical\_crossentropy') face\_model.fit(x\_train,y\_train,batch\_size=batch\_size,epochs=epochs,shuffle='true',validation

\_data=(x\_test,y\_test)) face\_model.save('face\_reco2.MODEL') print(x\_data.shape,y\_data.shape)

predicted =np.array(face\_model.predict(x\_test)) ynew = face\_model.predict\_classes(x\_test)

Acc=accuracy\_score(Y\_test, ynew) print("accuracy : ")

print(Acc)

cnf\_matrix=confusion\_matrix(np.array(Y\_test), ynew) y\_test1 = to\_categorical(Y\_test, 20)

def plot\_confusion\_matrix(cm, classes,

normalize=False, title='Confusion matrix', cmap=plt.cm.Blues):

"""

This function prints and plots the confusion matrix. Normalization can be applied by setting `normalize=True`. """

if normalize:

cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis] #print("Normalized confusion matrix")

else:

print('Confusion matrix, without normalization')

#print(cm)

plt.imshow(cm, interpolation='nearest', cmap=cmap) plt.title(title)

plt.colorbar()

tick\_marks = np.arange(len(classes)) plt.xticks(tick\_marks, classes, rotation=45) plt.yticks(tick\_marks, classes)

fmt = '.2f' if normalize else 'd' thresh = cm.max() / 2.

for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])): plt.text(j, i, format(cm[i, j], fmt),

horizontalalignment="center",

color="white" if cm[i, j] > thresh else "black")

plt.tight\_layout() plt.ylabel('True label')

plt.xlabel('Predicted label') plt.show()

print('Confusion matrix, without normalization') print(cnf\_matrix)

print("Confusion matrix:\n%s" % confusion\_matrix(Y\_test, ynew)) print(classification\_report(Y\_test, ynew))

**Training Model :**

from keras.layers import Dense,Activation from keras.layers import LeakyReLU from keras.models import Sequential

class DenseArchs:

def init (self,classes): print('training initiated') self.model=Sequential() self.classes=classes

def arch(self): self.model.add(Dense(64,input\_dim=128)) self.model.add(LeakyReLU(alpha=0.1)) self.model.add(Dense(32)) self.model.add(LeakyReLU(alpha=0.1)) self.model.add(Dense(16)) self.model.add(LeakyReLU(alpha=0.1)) self.model.add(Dense(self.classes)) self.model.add(Activation('softmax'))

return self.model

**Recognition Code:**

import os import cv2

from face\_detection import face

from keras.models import load\_model import numpy as np

import datetime import time

import pandas as pd

from embedding import emb

label=None e=emb() fd=face()

print('attendance till now is ') #data.view()

model=load\_model('face\_reco2.MODEL') peoplePath= "people"

KnownFaces= []

KnownFaces= [f for f in sorted(os.listdir(peoplePath))]

for i in KnownFaces: print(i)

def test():

for FolderName in os.listdir(peoplePath):

for filenumber in os.listdir(f'{peoplePath}\{FolderName}'):

test\_run = cv2.imread(f"{peoplePath}\{FolderName}\{filenumber}", 1) test\_run = cv2.resize(test\_run, (160, 160)) #test\_run=np.rollaxis(test\_run,2,0)

test\_run = test\_run.astype('float') / 255.0

test\_run = np.expand\_dims(test\_run, axis=0) test\_run = e.calculate(test\_run)

test\_run = np.expand\_dims(test\_run, axis=0) test\_run = model.predict(test\_run)[0]

cap=cv2.VideoCapture(0) ret=True

test() path='people'

col\_names = ['Name','Id', 'Date', 'Time'] attendance = pd.DataFrame(columns=col\_names)

while ret: ret,frame=cap.read() frame=cv2.flip(frame,1)

det,coor=fd.detectFace(frame)

if(det is not None):

for i in range(len(det)): detected=det[i] k=coor[i] f=detected

detected=cv2.resize(detected,(160,160)) #detected=np.rollaxis(detected,2,0) detected=detected.astype('float')/255.0 detected=np.expand\_dims(detected,axis=0) feed=e.calculate(detected) feed=np.expand\_dims(feed,axis=0) prediction=model.predict(feed)[0] result=np.argmax(prediction) if(np.max(prediction)>.95):

for i in range(len(KnownFaces)): if(result==i):

label=str(KnownFaces[i]) Name,Id=label[1:].split('\_') Id=int(Id)

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d') timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S') attendance.loc[len(attendance)] = [Name,Id, date, timeStamp]

else:

label='unknown' #data.update(label)

cv2.putText(frame,label,(k[0],k[1]),cv2.FONT\_HERSHEY\_SIMPLEX,1,(255,255,255),2) cv2.rectangle(frame,(k[0],k[1]),(k[0]+k[2],k[1]+k[3]),(252,160,39),3) cv2.imshow('onlyFace',f)

attendance = attendance.drop\_duplicates(subset=['Id'], keep='first') cv2.imshow('frame',frame)

if(cv2.waitKey(1) & 0XFF==ord('q')): break

ts = time.time()

date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d') timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S') Hour, Minute, Second = timeStamp.split(":")

fileName = "F:/Attendance-using-Face-master/Attendance/Attendance\_" + date + "\_" + Hour

+ "\_" + Minute + ".csv" attendance.to\_csv(fileName, index=False) cap.release()

cv2.destroyAllWindows()