A Project Report on

DOUBLE AUTHENTICATION SYSTEM INTEGRATING FACE RECOGNITION AND EYE BLINK COUNT RECOGNITION

is submitted in partial fulfillment of the requirement for the award of the Degree of

Bachelor of Technology

to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTHAPURAMU

by

SK. RESHMA
(19711A04A3)

MD. RAHILA SULTHANA
(20715A0406)

T. SRIVANI
(19711A04B3)

M. SUVARNA
(19711A0474)

Under the Guidance of

Mr. V. SUDHEER

Assistant Professor, Dept. of ECE

Department of Electronics and Communication Engineering

NARAYANA ENGINEERING COLLEGE::NELLORE

AUTONOMOUS

(Affiliated to Jawaharlal Nehru Technological University Anantapur, Ananthapuramu)

MAY 2023

NARAYANA ENGINEERING COLLEGE::NELLORE
AUTONOMOUS
(Affiliated to Jawaharlal Nehru Technological University Anantapur, Ananthapuramu)
Department of Electronics and Communication Engineering

CERTIFICATE

This is to certify that the project report entitled **"DOUBLE AUTHENTICATION SYSTEM INTEGRATING FACE RECOGNITION AND EYE BLINK COUNT RECOGNITION"** is being submitted by **SK. RESHMA (19711A04A3), MD. RAHILA SULTHANA (20715A0404), T. SRIVANI (19711A04B3), M. SUVARNA (19711A0474)** in partial fulfillment for the award of the Degree of Bachelor of Technology in Electronics & Communication Engineering Department to the Jawaharlal Nehru Technological University Anantapur, Ananthapuramu is a record of bonafide work carried out by them under my guidance and supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Mr. V. SUDHEER,
Assistant Professor
Project Supervisor

Dr.K.MURALI, M.Tech., Ph.D
HOD
Department of ECE

Date of Viva-Voce_____

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. P. NARAYANA, Ph.D.Founder**, Narayana Educational Institutions, Andhra Pradesh for the kind blessings. We are extremely thankful to **Mr. R. Sambasiva Rao B.Tech, Registrar of** Narayana Engineering College, Nellore.

We are much obliged to **Dr. A.V.S Prasad, Ph.D. Director,** Narayana Engineering & Pharmacy Colleges, for the continuous encouragement and support. We owe indebtedness to our **Principal Dr. G. Srinivasulu Reddy, M.Tech., Ph.D.,** Narayana Engineering College, Nellore for providing us with the required facilities.

We express our deep sense of gratitude and sincere thanks to **Dr. K. Murali, M. Tech, Ph.D., Professor & HOD,** Department of Electronics and Communication Engineering, Narayana Engineering College, Nellore for providing the necessary facilities and encouragement towards the project work.

We thank our project guide, **Mr. V. Sudheer, Assistant Professor, Department of Electronics and Communication Engineering** for his guidance, valuable suggestions, and support in the completion of the project.

We gratefully acknowledge and express our thanks to the teaching and non-teaching staff of the E.C.E Department. We would like to express our love and affection to our parents for their encouragement throughout this project.

Project Associates

SK. RESHMA	19711A04A3
MD. RAHILA SULTHANA	20715A0404
T. SRIVANI	19711A04B3
M. SUVARNA	19711A0474

ABSTRACT

Authentication plays a major role in maintaining security in data access. It is the process of recognizing a user's identity and providing data access. It is a mechanism of associating an incoming request with a set of identifying credentials and providing access to data. Traditional password authentication could not provide enough security. Even though many other authentication methods were introduced but unfortunately they couldn't prevent all types of security attacks, because traces like fingerprints, touch marks, etc., were left during authentication.

Hence we come up with an idea named "Double Authentication System Integrating Face Recognition and Eye Blink Count Recognition". This system provides double-layered authentication for a user, which makes authentication more secure. The two layers of authentication are face recognition followed by eye blink count recognition. This system first verifies the user's face and then takes the eye blink count from the user.

CONTENTS

LIST OF FIGURES AND TABLES

LISTS OF TABLES

LISTS OF FORMULAE

CHAPTER 1

INTRODUCTION

Biometrics is part of cutting-edge technology. Biometrics are the metrics related to human features. As an emerging technology, biometric systems can add great convenience by replacing passwords, helping law enforcement catch criminals, and even in organizations in posting the attendance of an employee. Several types of biometric types are available Face Recognition, Iris Recognition, Fingerprint Scanner, Voice Recognition, Hand Geometry, and Behaviour Characteristics.

1.1 Importance of Face Recognition

Face Recognition generally measures the unique patterns of a person's face by comparing and analyzing facial contours. It is not only used in security and law enforcement but also used as a way to authenticate identity and unlock devices like smartphones and laptops. Nowadays, we are observing many smartphones and laptops having a face recognition system to unlock the device.

The face is one of the most important biometric features of a human. A human can recognize different faces without difficulty. Yet it is a challenging task to design a robust computer system for face identification. The inadequacy of automated face recognition systems is especially apparent when compared to our own innate face recognition ability. Humans perform face recognition, an extremely complex visual task, almost instantaneously and our own recognition ability is far more robust than any computer can hope to be. Humans can recognize a familiar individual under very adverse lighting conditions, from varying angles or viewpoints. While research into this area dates back to the1960ss, it is only very recently that acceptable results have been obtained. However, face recognition is still an area of active research since a completely successful approach or model has not been proposed to solve the face recognition problem. The next-generation surveillance systems are expected to take the human face as an input pattern and extract useful information such as face recognition from it

1.2 Problem Definition

Face Recognition is one of the most widely used features in many fields. It is used in providing security to data, Fraud detection of Passports and Visas, Track attendance, detecting criminals, etc.

The Face Recognition system is being used by some organizations to track the attendance of employees. The system collects and records the facial fine points of the employees in the database. Once the process is done, the employee only needs to look at the camera, and the attendance is automatically marked in the face recognition attendance system.

The project develops a novel CNN model which is used for Face recognition. The face that is detected and recognized using this CNN model will be provided the attendance based on the time at which the person faces the camera.

1.3 Aim and Objective

The main aim of this project is to construct an authentication system that will not leave any traces on authentication by integrating face recognition and eye blink count models which leads to a new type of authentication system by providing double-layer authentication

The main objective of the project is to build a custom CNN model that can recognize the faces and post the appropriate attendance into the database for the designated periods based on the timings of the college hours.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

A literature survey is the most important step in the software development process. Before developing the tools it is necessary to determine the time factor, economic, and company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need a lot of external support. This support can be obtained from senior programmers, books, or from websites. Before building the system the above consideration is taken into account for developing the proposed system.

2.2 LITERATURE SURVEY:

[1] Liang, L., Ai, H.: Summary on Face Detection Research. Journal of Computers 25(5),449–459 (2004)

This paper studies two critical technical links between face detection and face recognition in the recognition of face and conducts analyses and comparisons on some commonly used face detection and recognition algorithms. Through the collation of this information and the combination of related knowledge in digital image processing, it proposes a face recognition algorithm based on the singular value decomposition. After preprocessing face images, by the use of the projection method, it obtains the positions of five sense organs, and then extracts local feature values near five sense organs as the main features of faces by using the singular value decomposition. For different face images of the same individual, the matchingdegree of feature values will be very high.

[2] Zhao, M.: Study on Coding Algorithm of Wavelet-based Color Face Images. Master Thesis. Computer Academy of Sciences of Sichuan Normal University, Sichuan (2005) A face recognition system is one of the biometric information processes, its applicability iseasier and the working range is larger than others, i.e.; fingerprint, iris scanning, signature, etc. A face recognition system is

designed, implemented, and tested at Atılım University, Mechatronics Engineering Department. The system uses a combination of techniques in two topics; face detection and recognition. Face detection is performed on live acquiredimages without any application field in mind. Processes utilized in the system are whitebalance correction, skin-like region segmentation, facial feature extraction, and face image extraction on a face candidate. Then a face classification method that uses FeedForward Neural Network is integrated with the system. The system is tested with a database generated in the laboratory with 26 people. The tested system has the acceptable performance to recognize faces within intended limits. The system is also capable of detecting and recognizing multiple faces in live acquired images.

[3] Liang, L., Ai, H.: Face Detections Based on Multi-Template Matching. Journal ofChina's Image Graphics 44(10), 623–630 (2004)

Face recognition has become relevant in recent years because of its potential applications. Theaim of this paper is to find out the relevant techniques which give not only better accuracy but also efficient speed. There are several techniques available for face detection which give much better accuracy but the execution speed is not efficient. In this paper, a normalized cross-correlation template matching technique is used to solve this problem. According to the proposed algorithm, first different facial parts are detected like the mouth, eyes, and nose. If anyof the two facial parts are found successfully then the face can be detected. For matching thetemplates with the target image, the template rotates at a certain angle interval.

[4] Chen, M.: Studies on the Face Image Detection and Classification System. Ph.D. Thesis. Computer Science and Engineering Department of Shanghai Jiaotong University,Shanghai (2003)

The reliability of the face recognition system has the characteristics of fuzziness, randomness, and continuity. In order to measure it in unconstrained scenes, we find out and quantify key broad-sense and narrow-sense influencing factors of reliability on the basis of analyzing operation states for six dynamic face recognition systems in the practical use of six public security bureaus. In this article, we propose a novel evaluation method with a True Positive Identification

Rate in dynamic and M: N mode and create a novel evaluation model of system reliability with the improved Fuzzy Dynamic Bayesian Network. Subsequently, we infer to solve the fuzzy reliability state probabilities of the six systems with Netica and get the two most important factors with the improved fuzzy C-means algorithm. We verify the model by comparing the evaluation results with the actual achievements of these systems. Finally, we findseveral vulnerabilities in the system with the least reliability and put forward a few optimization strategies. The proposed method combines the advantages of the improved fuzzy C-means model with those of the dynamic Bayesian network to evaluate the reliability of the dynamic face recognition systems, making the evaluation results more reasonable and realistic. It starts new research on face recognition systems in unconstrained scenes and contributes to the research on face recognition performance evaluation and system reliability analysis. Besides, the proposed method is of practical significance in improving the reliabilityof the systems in use.

[5] Lu, C.: Study on Several Problems of Automatic Face Recognition and the System Implementation. Ph.D. Thesis. Computer Academy of Sciences of Tsinghua University,Beijing (1998)

The paper analyses the multiple kernel learning-based face recognition in the pattern matching area. Based on the analysis of the basic theory of multiple kernel SVM, this thesis focuses onthe multiple kernel SVM algorithm based on the semi-infinite linear program (SILP), including SILP based on column generation (CG) and SILP based on chunking algorithm (CA). The two SILP-improved algorithms are applied to several classification problems, including UCI binary classification problem datasets and multi-classification problem datasets. Furthermore, the two SILP-improved algorithms are applied to the actual problems of face recognition. Theexperiment data shows that with the multiple kernel learning-based method, the performanceof face recognition can be obviously improved.

[6] Su, G.: Summary on Face Recognition Technology. Journal of China's ImageGraphics 5(11), 220–238 (2000)

Face recognition presents a challenging problem in the field of image analysis and computervision, and as such has received a great deal of attention over the last few years because of its many applications in various domains. Face recognition

techniques can be broadly divided into three categories based on the face data acquisition methodology: methods that operate onintensity images; those that deal with video sequences; and those that require other sensory data such as 3D information or infra-red imagery. In this paper, an overview of some of the well-known methods in each of these categories is provided and some of the benefits and drawbacks of the schemes mentioned therein are examined. Furthermore, a discussionoutlining the incentive for using face recognition, the applications of this technology, and some of the difficulties plaguing current systems with regard to this task have also been provided. This paper also mentions some of the most recent algorithms developed for this purpose and attempts to give an idea of the state of the art of face recognition technology.

[7] Li, J.: Research Progress of New Face Recognition Technology. Journal ofComputers 31(10), 293–295 (2004)

Over the past few decades, interest in theories and algorithms for face recognition has been growing rapidly. Video surveillance, criminal identification, building access control, and unmanned and autonomous vehicles are just a few examples of concrete applications that are gaining attraction among industries. Various techniques are being developed including local, holistic, and hybrid approaches, which provide a face image description using only a few faceimage features or the whole facial features. The main contribution of this survey is to review

some well-known techniques for each approach and to give the taxonomy of their categories.In the paper, a detailed comparison between these techniques is exposed by listing the advantages and the disadvantages of their schemes in terms of robustness, accuracy, complexity, and discrimination. One interesting feature mentioned in the paper is about the database used for face recognition. An overview of the most commonly used databases, including those of supervised and unsupervised learning, is given. Numerical results of the most interesting techniques are given along with the context of experiments and challenges handled by these techniques. Finally, a solid discussion is given in the paper about future directions in terms of techniques to be used for face recognition.

2.3 Existing System

The existing system for Face Recognition uses some built-in datasets and some built-in architectures. Even though some custom data sets have been prepared and used they have been using the architectural models that have been already present. Some of the models that have been used in the existing systems are:

- Local Binary Histogram Pattern (LBHP)
- Fischer Face Recognition
- Eigen Face Recognition

1. Local Binary Histogram Pattern (LBHP)

In the LBHP approach for texture classification, the occurrences of the LBHP codes in an image are collected into a histogram. The classification is then performed by computing simple histogram similarities. However, considering a similar approach for facial image representation results in a loss of spatial information, and therefore one should codify the texture information while retaining also their locations. One way to achieve this goal is to use the LBHP texture descriptors to build several local descriptions of the face and combine them into a global description.

The facial image is divided into local regions and LBHP texture descriptors

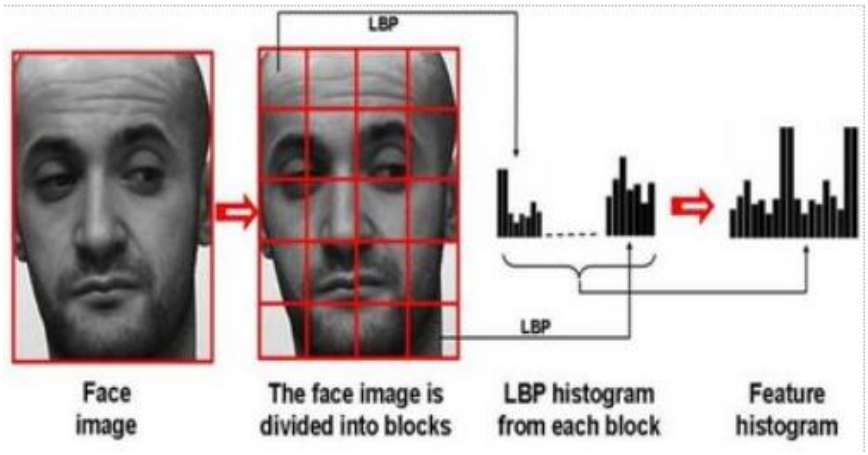


Fig. 2.1 Face Description with Local Binary Patterns

are extracted from each region independently. The descriptors are then concatenated to form a global description of the face, as shown in Fig. 2.1

This histogram effectively has a description of the face on three different levels of locality: the LBHP labels for the histogram contain information about the

patterns on a pixel level, the labels are summed over a small region to produce information on a regional level and the regional histograms are concatenated to build a global description of the face.

2. Fischer Face Recognition

The input generally given to a face recognition system is always an image or video stream and the output is an identification of the subject or subjects that appear in the image or video.

Fisher Face is one of the popular algorithms used in face recognition and is widely believed to be superior to other techniques, such as eigenface because of the effort to maximize the separation between classes in the training process. Image recognition using the Fischer Face method is based on the reduction of face space dimension using Principal Component Analysis (PCA) method, then applying Fisher's Linear Discriminant (FLD) method or also known as Linear Discriminant

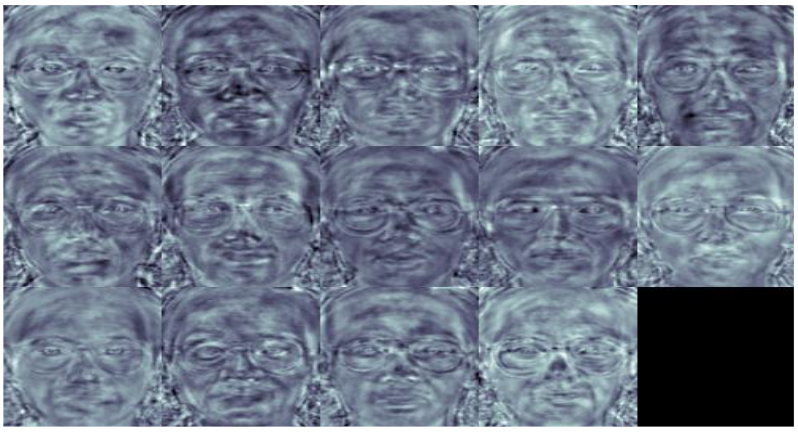


Fig. 2.2 Fischer Faces

Analysis (LDA) method to obtain feature of image characteristic dimension using Principal Component Analysis (PCA) method, then apply Fisher's Linear Discriminant (FLD) method or also known as the Linear Discriminant Analysis (LDA) method to obtain features of image characteristics.

The Fischer Face method learns a class-specific transformation matrix, so they do not capture illumination as obviously as the Eigenfaces method. The Discriminant Analysis instead finds the facial features to discriminate between the persons. The Fischer Face is especially useful when facial images have large variations in illumination and facial expressions. An example of Fischer’s face representation is illustrated in Figure 2.2.

3. Eigen Face Recognition

An eigenface is the name given to a set of eigenvectors when used in computer vision problems of human face recognition. The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenfaces themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. Classification can be achieved by comparing how faces are represented by the basis set.

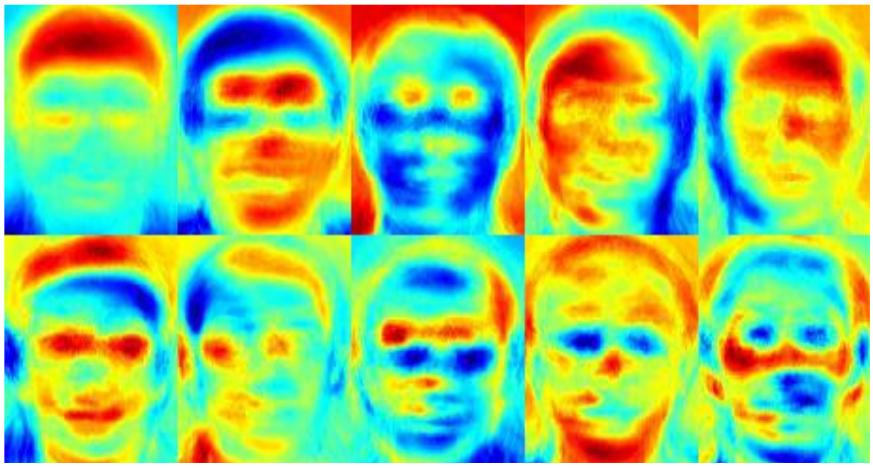


Fig. 2.3 Eigen Faces

A set of eigenfaces can be generated by performing a mathematical process called Principal Component Analysis (PCA) on a large set of images depicting different human faces. Informally, eigenfaces can be considered a set of "standardized face ingredients", derived from statistical analysis of many pictures of faces. Any human face can be considered to be a combination of these standard faces. For example, one's face might be composed of the average face plus 10% from eigenface 1, 55% from eigenface 2, and even −3% from eigenface 3. Remarkably, it does not take many eigen faces combined together to achieve a fair approximation of most faces. An example of Fischer's face representation is illustrated in Figure 2.3.

CONS OF THE EXISTING SYSTEM:

- The **traditional authentication** which uses a personal identification number (PIN) for login, leaves the touch marks on the keypad, that can be used to trace out the PIN.
- **Fingerprint authentication** leaves the thumb impression of the user on the sensor, nowadays which can be easily traced out and easily perform unauthorized access.
- The **face recognition authentication** system can be easily cheated by using a face picture of the user.
- The **voice recognition authentication** system also be easily cheated by recorded audio of the user or through mimicry

2.4 Proposed System

The project proposes a pipeline to build a Deep Learning model for Face Recognition which is motivated by the state-of-the-art architectures in Computer Vision. The contributions of the project are:

- Develops a novel Deep Learning model to detect and recognize human faces.
- Develops a web application to post the attendance using the novel Deep Learning model developed.
- The process flow to develop a new face dataset is proposed which is furtherly used to train the Deep Learning model.

The result of the project will be the provision of attendance for the corresponding period after recognizing the face based on the time at which the person's face has been captured.

PROS OF THE PROPOSED SYSTEM:

- The proposed system will not leave any traces.
- Provides more security than the existing system.
- Provides double-layered authentication.
- Cost effective.
- Easily deployable.

The proposed system was built by integrating the face recognition model and the eye blink countrecognition model. The flow of authentication is, firstly the face of

the user was recognized by using the face recognition model and then the password will be taken from the user through eye blinks, this part willbe done by the eye blink count recognition model.

Face Detection

There are different packages used to detect the face of the user and they are mentioned below:

- cv2
- FaceMeshDetector

cv2:

OpenCV is a great tool for image processing and performing computer vision tasks.It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It supports multiple languages including Python.

In our project cv2 is mainly used for image processing. By using cv2 we convert the images into grayscale images which are very helpful in training the model.

FaceMeshDetector:

Face Mesh Detection can be done using Python OpenCV directly via an Image file, webcam, or video file. This package is used to detect different parts ofthe face.

First of all, a model should be created and that can be done by using the LBPHalgorithm.

LBPH Algorithm:

Local Binary Pattern Histograms (LBPS) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is

combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Using the LBP combined with histograms we can represent the face images with a simple data vector.

As LBP is a visual descriptor it can be used for face recognition, as can be seen in the following step-by-step explanation.

Parameters:

the LBPH uses 4 parameters:

- **Radius**: the radius is used to build the circular local binary pattern andrepresents the radius around the central pixel. It is usually set to 1.
- **Neighbours**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, and the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y**: the number of cells in the vertical direction. The more cells, the fine the grid, and the higher the dimensionality of the resulting feature vector. It is usually set to 8.

Training the Algorithm:

First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same personmust have the same ID.

With the training set already constructed, let's see the LBPH computational steps.

Applying the LBP operation:

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. Todo so,

the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

The following image will show the procedure:

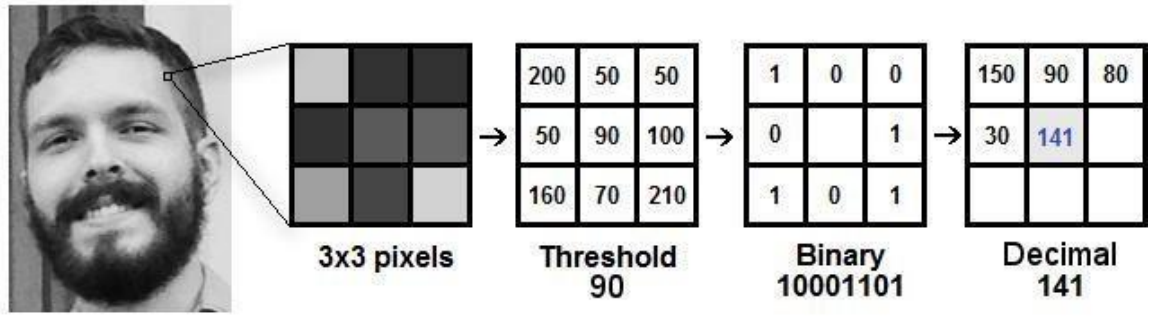


Fig. 2.4 Procedure for dividing a pixel

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of eachpixel (0~255).
- Then, we need to take the central value of the matrix to be used as thethreshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal to or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrixline by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the final result will be the same.

- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image that represents better the characteristics of the original image.
- **Note**: The LBP procedure was expanded to use a different number of radii and neighbors, it is called Circular LBP.

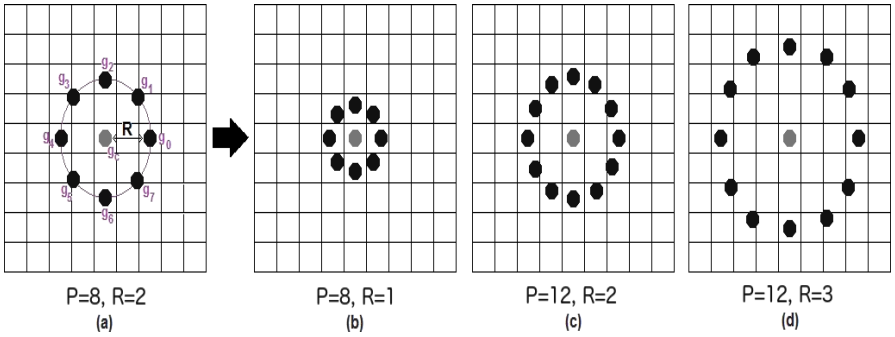


Fig. 2.5 Finding neighbours and radius

It can be done by using **bilinear interpolation**. If some data point is between the pixels,it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

Extracting the Histograms:

Now, using the image generated in the last step, we can use the **Grid X** and **Grid Y** parameters to divide the image into multiple grids, as can be seen in the following

Based on the image above, we can extract the histogram of each region as follows:

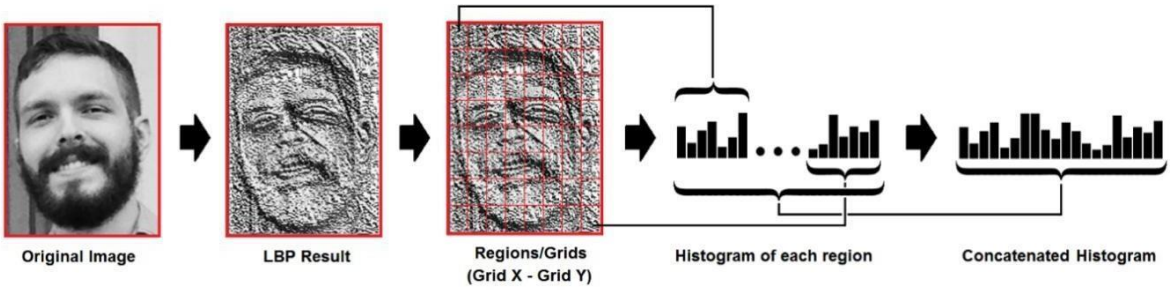


Fig. 2.6 Grayscale Conversion

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixelintensity.
- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the characteristics of the image original image.

Performing face recognition:

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and create a histogram that represents the image.

- So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram. We can use various approaches to compare the histograms (calculate the distance between two histograms), for example, **euclidean distance**, **chi-square absolute value**, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

Formula: 2.1 Euclidean Distance

- So the algorithm output is the ID from the image with the closest histogram.The algorithm should also return the calculated distance, which can be used as a '**confidence**' measurement.

Note: don't be fooled about the 'confidence' name, as lower confidences are better because it means the distance between the two histograms is closer.

We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined.

After training done its time to recognize the face so we call a function named **face_check().**In this function, we compare the trained model results and present the input image. Then with the help of a mathematical formula, we calculate the percentage of confidence. If the user is valid or not. If the confidence is more than 85% the user is valid and returns "1" else user is not valid then the function return "0".

Eye Blink Count Recognition

For recognizing the eye blink count of a user we calculate the **Euclidean distance** between upper lid and lower lid of an eye of a user with the help of a user-defined function named **blink()**. The Euclidean distance becomes zero if the user closes both lids and becomes some positive value if he opens the lids of an eye.

The Euclidean distance is calculated with the help of pre-defined land marks on face, which was developed by Python for locating different points on the face. We will consider a list of points to locate eye position accurately and those list of points are given below

[22, 23, 24, 26, 110, 157, 158, 159, 160, 161, 130, 243]

The pair of consecutive one close and one open of lids is considered as a blink. In order to calculate the Euclidean distance of eye lids of user, we should be able to recognize the landmarks (which includes eyes) of a face. So we are using above given points as landmarks and we are taking horizontal and vertical distance between the eyelids and then calculate the average between the resultant values using the below formula

(Vertical length / Horizontal length) * 100

Formula: 2.2 Average

Now when the user's face was identified as an authorized user, then only the system starts taking the numeric password through the eye blink count. Here user needs to give sometime delay between the numbers and when some gap is given by the user between the eye blinks, the system considers the eyeblink count as a digit and appends it to the password string. For example, if the password is 123, at first user should blink an eye one time and wait for some time without any additional blinks and after that user need to give the 2-eye blinks without any gap.

2.5 Feasibility Analysis

The feasibility of the project is analyzed in this phase and the 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
- Technical Feasibility
- Social Feasibility

Economical feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system is 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.

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 for 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.

Social feasibility:

The aspect of the 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.

CHAPTER 3

ANALYSIS

It is essential to understand the requirements of the project which plays a crucial role in developing and achieving or meeting the desired goal. The Hardware Requirements and the Software Requirements of the project for developing and reproducing the results of the projects are discussed in detail.

3.1 Hardware Requirements

Develop Deep Learning is a hard task without proper hardware support. To overcome this issue, the development of a deep learning model is done in the Google collaboratory platform. Which provides high-end hardware support to develop machine learning and deep learning models. The configuration of the CPU and GPU based on the Google collaboratory are:

- Graphical Processing Unit (GPU) used is 1X TeslaK80 with 2496 CUDA cores, 12GB GDDR5 VRAM
- Central Processing Unit (CPU) used if 1X single core hyper threaded Xeon Processors, 45MB Cache with 12.6 GB RAM and 320GB disk.

3.2 Software Requirements

These are the essential software requirements that are necessary to build and deploy the Deep Learning model and the web application. The windows10 operation system with Intel i5 7^{th} processor having 8GB RAM is used and the installation of the programming language and the important libraries that are used to develop the project are addressed below:

3.2.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python supports modules and packages, which encourages program modularity and code. Python 3 is used to develop the project. A detailed explanation of the Python 3 installation procedure can be accessed through the official documentation of Python at (https://docs.python.org/3/using/index.html).

3.2.2 TensorFlow

TensorFlow is an open-source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning models and algorithms and makes them useful by way of a common metaphor. It uses Python to provide a convenient front-end API for building applications with the framework while executing those applications in high-performance C++.

The TensorFlow 2.2.0 version is used throughout the project. Here is the official site for installation instructions (https://www.tensorflow.org/install/).

3.2.3 Keras

Keras is one of the leading high-level neural network APIs. It is written in Python and supports multiple back-end neural network computation engines. The Keras 2.2.5 version is used throughout the project to develop deep learning models.

In the first step of this tutorial, we'll use a pre-trained MTCNN model in Keras to detect faces in images. Once we've extracted the faces from an image, we'll compute a similarity score between these faces to find if they belong to the same person.

The installation instructions can be accessed using the link (https://keras.io/#installation).

3.2.4 SciKit - Learn

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction. The command pip installs sklearn can be used to install the libraries.

3.2.5 Pandas

Pandas is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool, built on top of the Python programming language. The pandas 1.0.3 is used in the entire project and the command to install the package is pip install pandas == 1.0.3.

3.2.6 Matplotlib

Matplotlib is a collection of command-style functions that make matplotlib work like MATLAB. Each plot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot

with labels, etc. The 3.2.1 version is used in the project and the command to install the package is pip install matplotlib == 3.2.1.

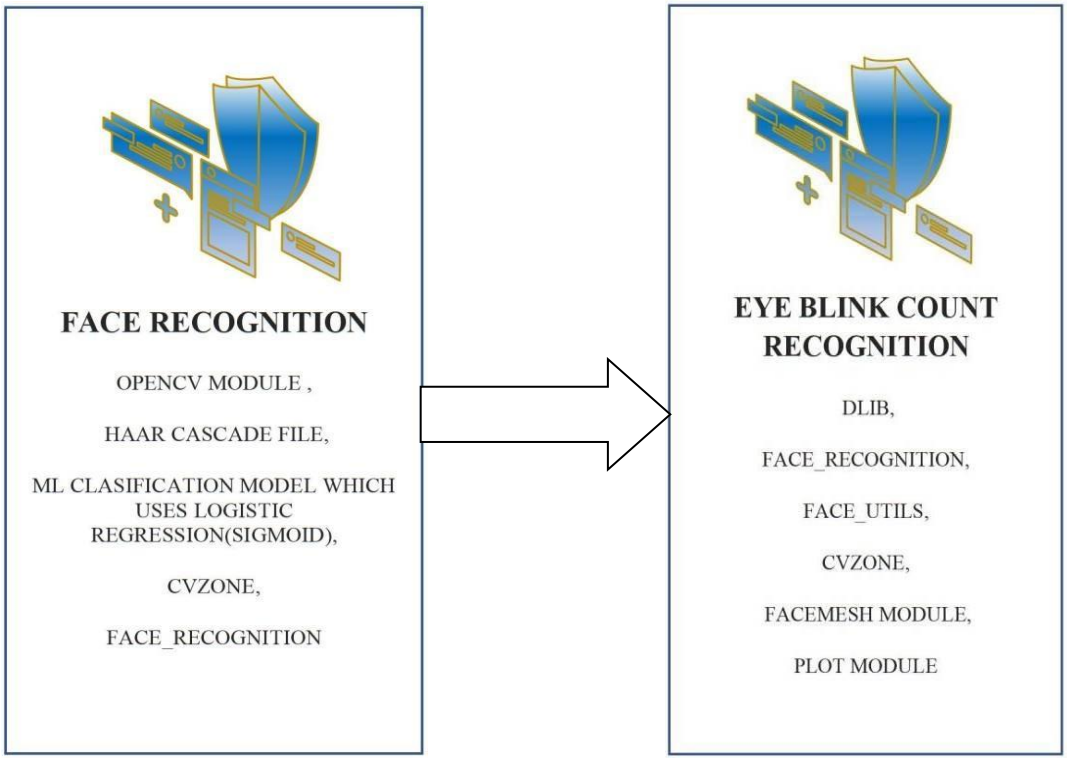
3.2.7 Jupyter

The Jupyter Notebook is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Version 1.0.0 is used in the project and the command to install the package is pip install jupyter == 1.0.0.

3.2.8 Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Version 1.0.2 is used in the project and the command to install the package is pip install flask == 1.0.2.

3.3 SYSTEM ARCHITECTURE



CHAPTER 4

DESIGN

UML diagrams are a way of visualizing a software program using a collection of diagrams. The notation has evolved from the work of Grady Booch, James Rumbaugh, Ivar Jacobson, and the Rational Software Corporation to be used for object-oriented design, but it has since been extended to cover a wider variety of software engineering projects.

4.1 Purpose of UML Diagrams

With the use of UML, an appropriate UML development tool, and an application process or methodology, the design and refining of the application are shifted from the development phase to the analysis and design phase. This reduces risk and provides a vehicle for testing the architecture of the system before coding begins. The analysis and design overhead will eventually pay dividends as the system has been used driven and documented and when it's time to start developing, many UML tools will generate skeleton code that will be efficient, object-oriented, and promote reuse.

Furthermore, the use of UML will help:

- The communication of the desired structure and behavior of a system between analysts, architects, developers, stakeholders, and users.
- The visualization and control of system architecture
- Promote a deeper understanding of the system, exposing opportunities for simplification and re-use
- Manage risk

4.2 Diagrams

There are eight most widely used UML diagrams. They are:

- Class Diagram

- Use Case Diagram
- Sequence Diagram
- Activity Diagram
- Collaboration Diagram
- Deployment Diagram
- State Chart Diagram
- Component Diagram

1. Class Diagram:

The class diagram is a static diagram. It represents the static view of an application. The class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

A class diagram describes the attributes and operations of a class and also the constraints imposed on the system. Class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

2. Use Case Diagram:

Use case diagrams consists of actors, use cases, and their relationships. The diagram is used to model the system/subsystem of an application. A single-use case diagram captures a particular functionality of a system.

It is defined and created from the 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. Hence to model the entire system, a number of use case diagrams are used.

3. Sequence Diagram:

A sequence diagram simply depicts the interaction between objects in sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

4. Activity Diagram:

Activity Diagrams are generally used to illustrate the flow of control in a system. We can also use an activity diagram to refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. It focuses on the condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.

5. Collaboration Diagram:

Collaboration diagrams which are also known as Communication Diagrams are used to show how objects interact to perform the behavior of a particular use case or a part of a use case. Along with sequence diagrams, collaboration is used by designers to define and clarify the roles of the objects that perform a particular flow of events in a use case. They are the primary source of information used to determine class responsibilities and interfaces.

6. Deployment Diagram:

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams can be used to model the hardware topology of a system, model the embedded system, model the hardware details for a client/server system, model the hardware details of a distributed application, and even in forward, reverse engineering.

7. State Chart Diagram:

A State Chart diagram describes a state machine. A state machine can be defined as a machine that defines different states of an object and these states are controlled by external or internal events.

State Chart diagrams are useful to model reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State Chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of a State Chart diagram is to model the lifetime of an object from creation to termination.

8. Component Diagram:

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. It does not describe the functionality of the system but it describes the components used to make those functionalities.

Component diagrams are very important from an implementation perspective. It can be mostly used in modeling the components of a system, modeling the database schema, the executables of an application, and even the system's source code.

4.3 Sequence Diagram

In this sequence diagram, we have four objects namely the user, web application, CNN model, and database.

- Initially the user runs the web application and the web application gives access and opens a camera to capture the images.
- Then the camera extracts the faces and gives them to the CNN model.
- Then the CNN model pre-processes the captured images and predicts the class of the image.

- Based on the prediction of the class, the attendance is posted into the database for the designated periods based on the timings of the college hours, and the details and status of attendance are displayed on the output screen.

Figure 4.1 represents the process flow in the form of a Sequence Diagram.

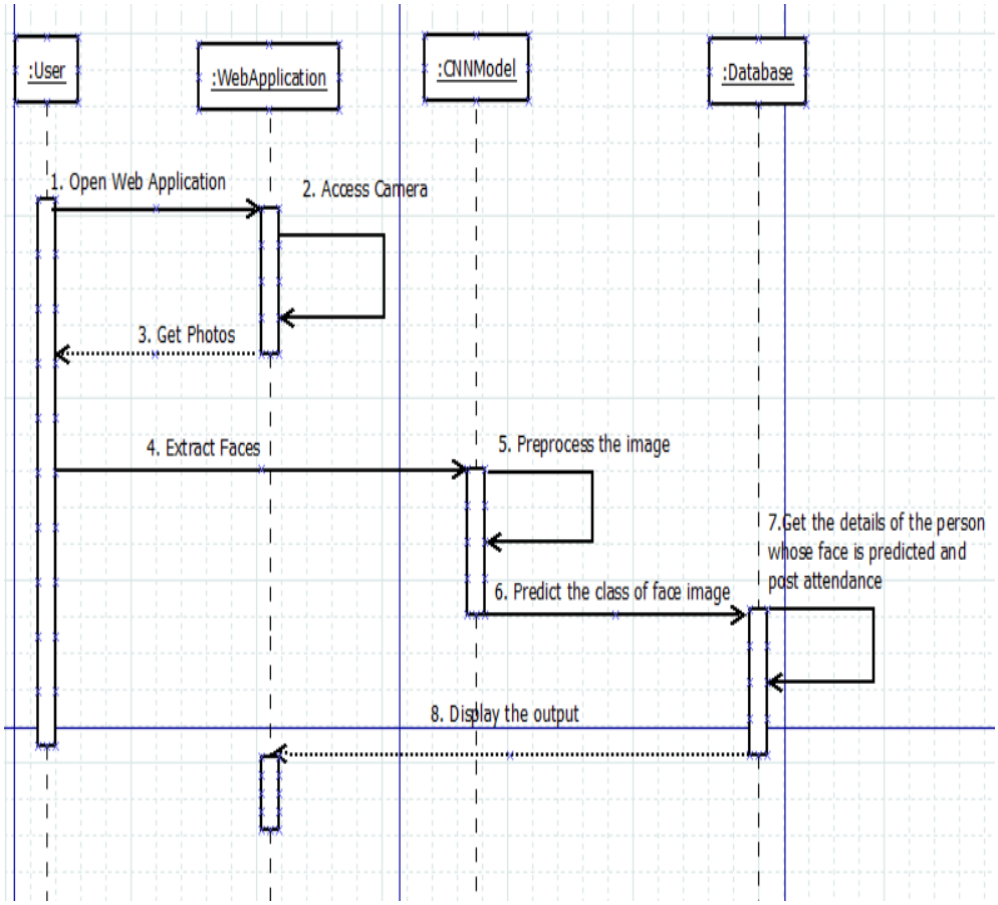


Fig. 4.1 Sequence Diagram

CHAPTER 5

DATA COLLECTION AND ANALYSIS

The data for Deep Learning is a key input to the model that comprehends such data and learns the features for future prediction. Although, various aspects come during the deep learning model development, without which various crucial tasks cannot be accomplished. In other words, data is the backbone of the entire model development without that it is not possible to train a machine that learns from humans and predicts for humans.

So, the project also focuses on creating a data set of human faces for face recognition. The data is collected through an automated program that takes the faces of the humans, stores and transforms them into a dataset. The dataset contains human faces at different lighting conditions and angles.

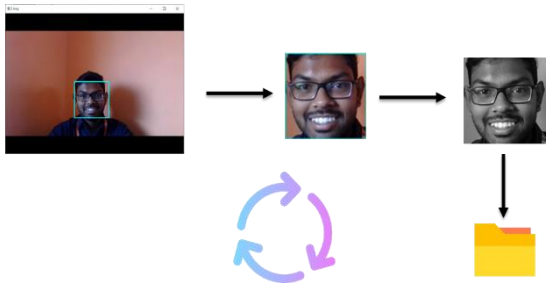


Fig. 5.1 Data Collection

Figure 5.1 illustrates the process flow of the data collection as follows:

- Identify the location of the face in the video frame.
- Extract the face image and convert it into a grayscale image.
- Attach the label w.r.t to the class of the image and write it into a CSV file.

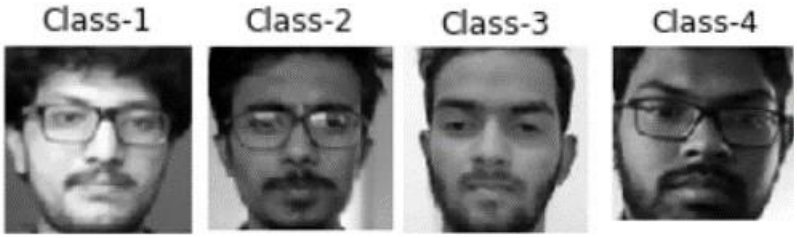


Fig. 5.2 Image w.r.t Classes

Figure 5.2 illustrates the samples of the data based on the classes.

5.1 Data Augmentation

Data Augmentation is a strategy that enables practitioners to significantly increase the diversity of data available for training models, without actually collecting huge data. It can be done by using techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks. In order to achieve high accuracy, a large volume of training data is required. Hence, the data augmentation technique is used to generate new samples by manipulating the existing data. In the current work, synthetic images are being generated randomly, by applying the following operations:

- Zoom
- Shear
- Height shift
- Rotation

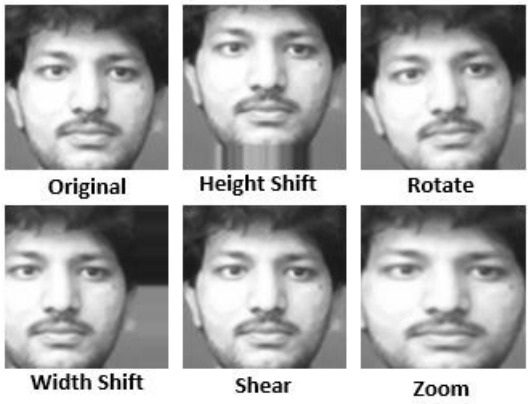


Fig. 5.3 Data Augmentation

5.2 Data Visualization and Analysis

Data visualization is a technique that uses an array of static and interactive visuals within a specific context to help people understand and make sense of large amounts of data. The data is often displayed in a story format that visualizes patterns, trends, and correlations that may otherwise go unnoticed. It often helps to understand the patterns in the data. Principal Component Analysis (PCA) is used in order to visualize the data in different forms. Here are some of the different visualizations to understand the data.

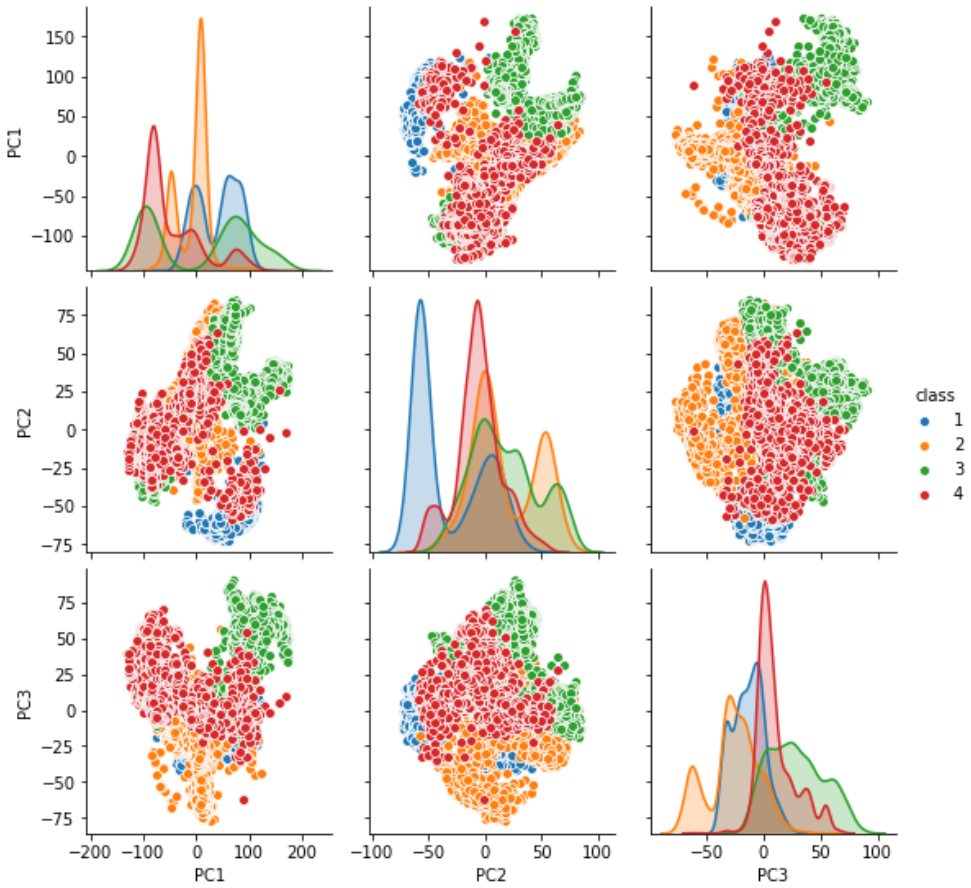


Fig. 1 Pair Plot of the data

A **pair plot** is used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or making linear separations in our dataset. A Simple 2D plot is used to understand the relationship or pattern between two variables or dimensions in the dataset as shown in Figure 5.4. Where PC means Principle Component.

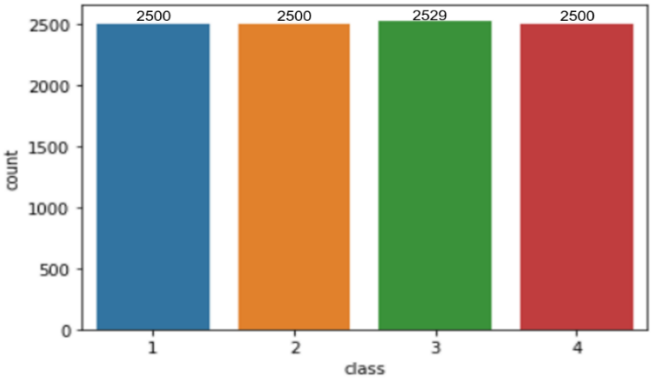


Fig. 5.6 Count Plot

Scatter plots are important in statistics because they can show the extent of correlation, if any, between the values of observed quantities or phenomena (called variables). If no correlation exists between the variables, the points appear randomly scattered on the coordinate plane. If a large correlation exists, the points concentrate near a straight line. Figure 5.5 shows the three-dimensional scatter plot using the PCA.

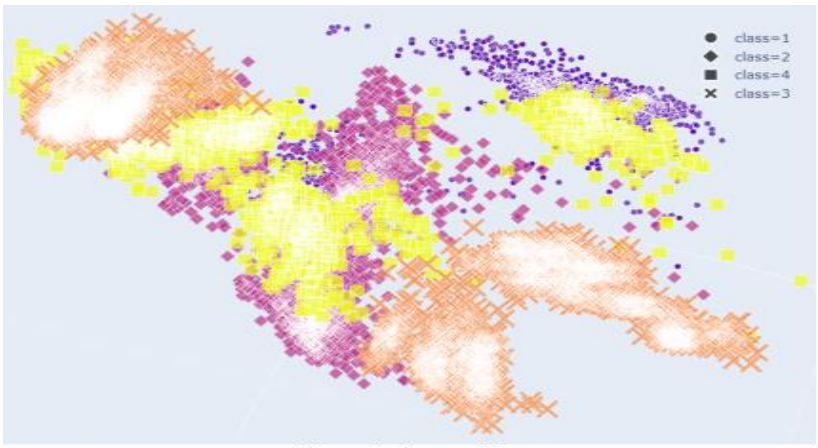


Figure 1: Scatter Plot

Fig. 5.5 Scatter Plot

Count Plot is used to understand the samples in the data with respect to class and the visualization is shown in Figure 5.6.

Correlation is usually defined as a measure of the linear relationship between two quantitative variables (e.g., height and weight). Often a slightly looser definition is used, whereby correlation simply means that there is some type of relationship between two variables. This post will define positive and negative correlation, provide some examples of correlation, explain how to measure correlation, and discuss some pitfalls regarding correlation.

When the values of one variable increase as the values of the other increase, this is known as a positive correlation (see the image below). When the values of one variable

decrease as the values of another increase to form an inverse relationship, this is known as a negative correlation. Figure 5.7 represents the correlation between the three principal components (PC).

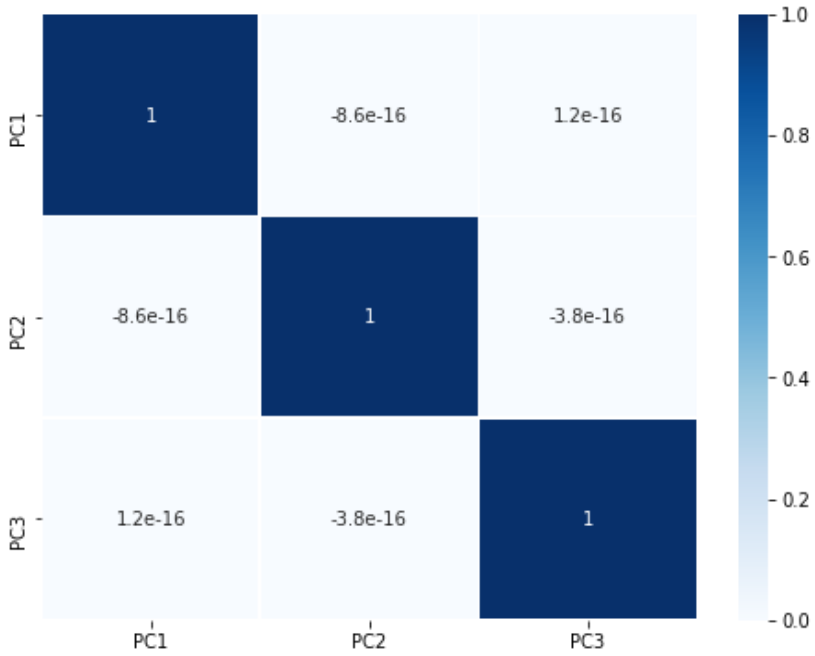


Fig. 5.7 Correlation Between the Principal Components

CHAPTER 6

DEVELOPING CNN MODEL

The advancements in Computer Vision with Deep Learning have been constructed and perfected with time, primarily over one particular algorithm a Convolutional Neural Network. A Convolutional Neural Network (CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other.

6.1 Role of CNN in Computer Vision

Computer Vision is an interdisciplinary field of science that aims to make computers process, analyze images and videos and extract details in the same way a human mind does. Earlier computer vision was meant only to mimic human visual systems until we realized how AI can augment its applications and vice versa.

Convolutional Neural Networks have been playing a significant role in many applications including surveillance, object detection, object tracking, healthcare, retail, banking, automobile, financial services, etc. Extensive research is recorded for face recognition using CNNs, which is a key aspect of surveillance applications. The rise of deep learning has enabled Deep Neural Networks (DNN) to achieve greater results in pattern recognition and Convolutional Neural Networks (CNN) is a class of DNN which is most commonly applied to analyzing visual imagery. It is used not only in Computer Vision but also for text classification in Natural Language Processing (NLP).

CNNs perform well when compared to the feed-forward neural networks as it uses less number of parameters for computation and also it uses different layers to learn the patterns. CNNs are used for a wide range of image-related tasks such as image classification, object detection/localization, image generation, visual question answering, image segmentation, etc. The availability of computation power at a lower cost has enabled a path to solve a lot of problems that have been considered virtually a hard task for computers for the past few years.

Many architectures are created to solve computer vision tasks and some of the widely popular CNN architectures are AlexNet (2012), VGG-16 (2014), Inception-v1 (2014),

Inception-v3 (2015), ResNet-50 (2015), Xception (2016), Inception-v4 (2016), Inception-ResNet-v2 (2016), ResNeXt-50 (2017).

6.2 Essential Layers to Develop CNN

The Convolutional Neural Networks CNNs have several different filters/kernels consisting of trainable parameters which can convolve on a given image spatially to detect features like edges and shapes. These high numbers of filters essentially learn to capture spatial features from the image based on the learned weights through backpropagation and stacked layers of filters can be used to detect complex spatial shapes from the spatial features at every subsequent level. Hence they can successfully transform the given image into a highly abstract representation that is easy for predicting.

- Convolutional Layer
- Batch Normalization Layer
- Max Pooling Layer
- Dropout Layer
- Fully Connected Layer

Let's see the detailed explanation of the above-mentioned layers that are used in order to build an efficient CNN model.

6.2.1 Convolutional Layer

A convolutional layer is the main building block of a CNN. A convolutional layer contains a set of filters whose parameters need to be learned. The height and weight of the filters are smaller than those of the input volume. Each filter is convolved with the input volume to compute an activation map made of neurons. In other words, the filter is slid across the width and height of the input and the dot products between the input and filter are computed at every spatial position. The output volume of the convolutional layer is obtained by stacking the activation maps of all filters along the depth dimension. Since the width and height of each filter is designed to be smaller than the input, each neuron in the activation map is only connected to a small local region of the input volume. In other words, the receptive field size of each neuron is small, and is equal to the filter size. The local connectivity is motivated by the architecture of the animal visual cortex where the receptive fields of the cells are small. The local connectivity of the convolutional layer allows the network to learn filters

which maximally respond to a local region of the input, thus exploiting the spatial local correlation of the input (for an input image, a pixel is more correlated to the nearby pixels than to the distant pixels). In addition, as the activation map is obtained by performing convolution between the filter and the input, the filter parameters are shared for all local positions. The weight sharing reduces the number of parameters for efficiency of expression, efficiency of learning, and good generalization.

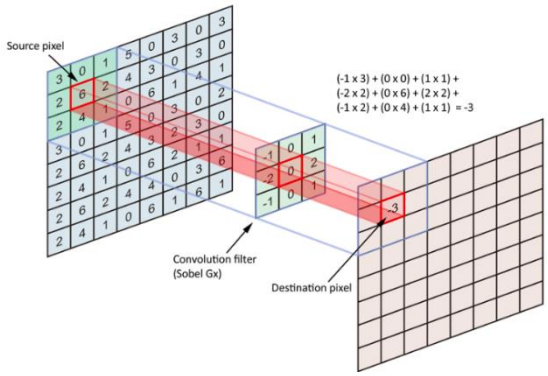


Fig. 6.1 Operation of Convolution Layer

The above example illustrated the convolutional operation in 2D, but in reality, convolutions are performed in 3D. Each image is namely represented as a 3D matrix with a dimension for width, height, and depth. Depth is a dimension because of the color channels used in an image (RGB). Generally, numerous convolutions on the input are performed, where each operation uses a different filter. This results in different feature maps. In the end, we take all of these feature maps and put them together as the final output of the convolution layer.

6.2.2Batch Normalization Layer

Training deep neural networks with tens of layers is challenging as they can be sensitive to the initial random weights and configuration of the learning algorithm. One possible reason for this difficulty is the distribution of the inputs to layers deep in the network may change after each mini-batch when the weights are updated. This can cause the learning algorithm to forever chase a moving target. This change in the distribution of inputs to layers in the network is referred to by the technical name “internal covariate shift”.

Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing

the learning process and dramatically reducing the number of training epochs required to train deep networks.

6.2.3. Max pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the convolved feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effective training of the model.

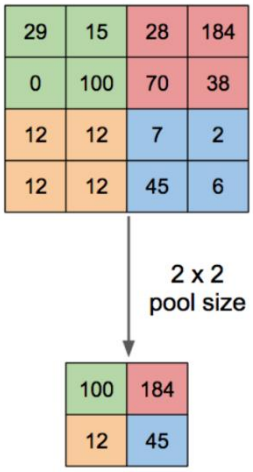


Fig. 6.2 Max Pooling Operation

Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. Figure 6.2 illustrates the max pooling operation.

6.2.4. Dropout Layer

In machine learning, regularization is a way to prevent over-fitting. Regularization reduces over-fitting by adding a penalty to the loss function. By adding this penalty, the model is trained such that it does not learn an interdependent set of feature weights. A dropout is an approach to regularization in neural networks which helps reduce interdependent learning amongst the neurons. It forces a neural network to learn more robust features that are useful in conjunction with many different random subsets of the other neurons. roughly doubles the number of iterations required to converge. However, the training time for each epoch is less. Figure 6.3 illustrates the operation of dropout.

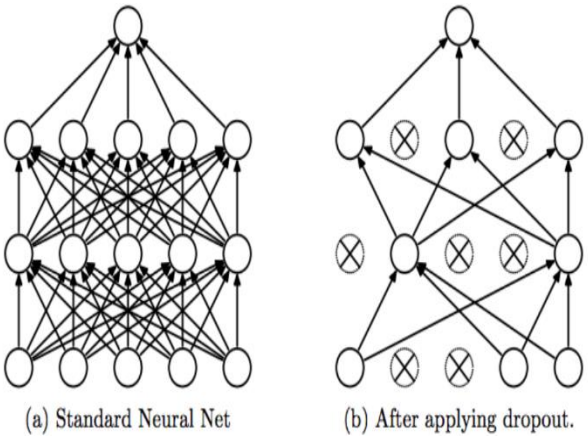


Fig. 6.3 Operation of Dropout

.6.2.5 Fully Connected Layer

Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space. Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector.

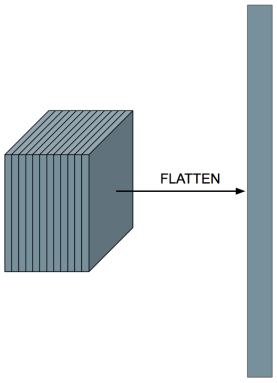


Fig. 6.4 Fully
Connected Layer

The flattened output is fed to a feed-forward neural network and backpropagation is applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the softmax classification technique.

6.3 Architecture of Proposed CNN

The proposed CNN model consists of 20 layers. These are Two-Dimensional Convolutional Layer (Conv2D), Batch Normalization Layer, Max Pooling Layer, and Dense Layer. The architecture takes a grayscale image with shape (100,100, 1) and results in the class label of the image as the prediction. CNN is a deep learning neural network sketched for processing structured arrays of data such as portrayals.

Layer	Output Shape	#Parameters
Conv2D	(None, 98, 98, 64)	640
BatchNormalization	(None, 98, 98, 64)	256
Conv2D_1	(None, 96, 96, 64)	36928
BatchNormalization_1	(None, 96, 96, 64)	256
Conv2D_2	(None, 96, 96, 64)	102464
BatchNormalization_2	(None, 96, 96, 64)	256
MaxPooling2D	(None, 96, 96, 64)	0
Dropout	(None, 46, 46, 128)	0
Conv2D_3	(None, 46, 46, 128)	73856
BatchNormalization_3	(None, 46, 46, 128)	512
Conv2D_4	(None, 44, 44, 128)	147584
BatchNormalization_4	(None, 44, 44, 128)	512
Conv2D_5	(None, 44, 44, 128)	409728
BatchNormalization_5	(None, 44, 44, 128)	512
MaxPooling2D_1	(None, 22, 22, 128)	0
Dropout_1	(None, 22, 22, 128)	0
Conv2D_6	(None, 20, 20, 256)	295168
BatchNormalization_6	(None, 20, 20, 256)	512
Max Pooling2D_2	(None, 10,10, 256)	0
Dropout_2	(None, 10, 10, 256)	0
Flatten	(None, 25600)	0
Dense	(None, 256)	6553856
BatchNormalization_7	(None, 256)	1024
Dense_1	(None, 128)	32896

BatchNormalization_8	(None, 128)	512
Dense_2	(None, 5)	645

Table 6.1: Architecture of Proposed CNN Model

The total number of parameters of the CNN is 7,658,629 of which 7,656,197 are trainable and 2,432 are non-trainable. A detailed description of the architecture is shown in Table-6.1.

The two-dimensional convolutional layer is used to extract the features from the previous input, the batch normalization layer is used in order to normalize the input and also overcome the problem of vanishing gradient and exploding gradient. The max pooling layer is used to reduce the dimensionality of the input and the dropout layer is used to avoid the overfitting problem.

6.4 Training the CNN

The proposed Convolutional Neural Network (CNN) is trained on the data containing 4 classes. Which is divided in the ratio of 80:20 into the train with 8524 samples and the

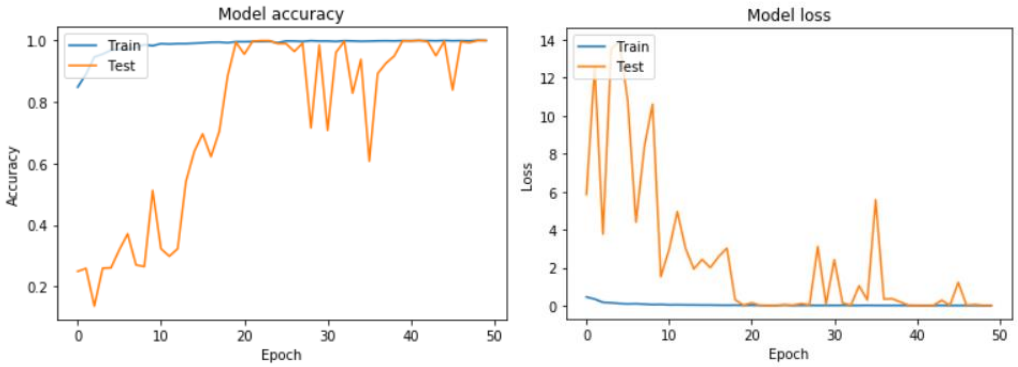


Fig. 6.5 Accuracy and Loss of the CNN Model

test with 1505 samples.

The following parameters are also used in the training process:

- Optimize – RMSProp
- Learning Rate – 0.001
- Loss Metric – Categorical Cross Entropy
- Batch Size – 256
- Epochs – 50

The accuracy and the loss of the CNN model during the training w.r.t number of epochs on train and test data are represented in Figure 6.5. It is evident that with an increase in the number of epochs the accuracy is increasing and the loss is decreasing and at a certain number of epochs, both the loss and accuracy do not change further.

6.5 Visualizing Output of Intermediate Convolutional Layers

Training the CNN involves feeding forward your training data, generating predictions, and computing a loss score, which is used for optimization purposes. However, it may be the optimizer gets stuck after some time and it is needed to know why this occurs and, more importantly, the convolutional layers, which learn features from the image, that can be used by densely connected layers for classification purposes.

Especially with problems that are less straightforward, CNNs can be tough to train. In some cases, it does not even converge. Visualizing layer outputs gets important in those cases. As convolutional layers, together with additional layers such as pooling layers down sample the image in the sense that it gets smaller and more abstract. When this happens, a neural network might no longer be able to discriminate between the classes, and hence show inadequate performance.

Visualizing the features learned by the network helps to tune since one can see an error made by the network and able to point out the cause directly. Furtherly, if the model performance is not improved, then extending and improving the overall design of the model can be helpful. Based on the knowledge of the current design, including its strengths and weaknesses of it. The intermediate layer visualizations of convolutional layers Conv2D_1, Conv2D_2, Conv2D_3, Conv2D _4, and Conv2D_5 of the CNN model corresponding to an input image are shown in Fig-6.6.

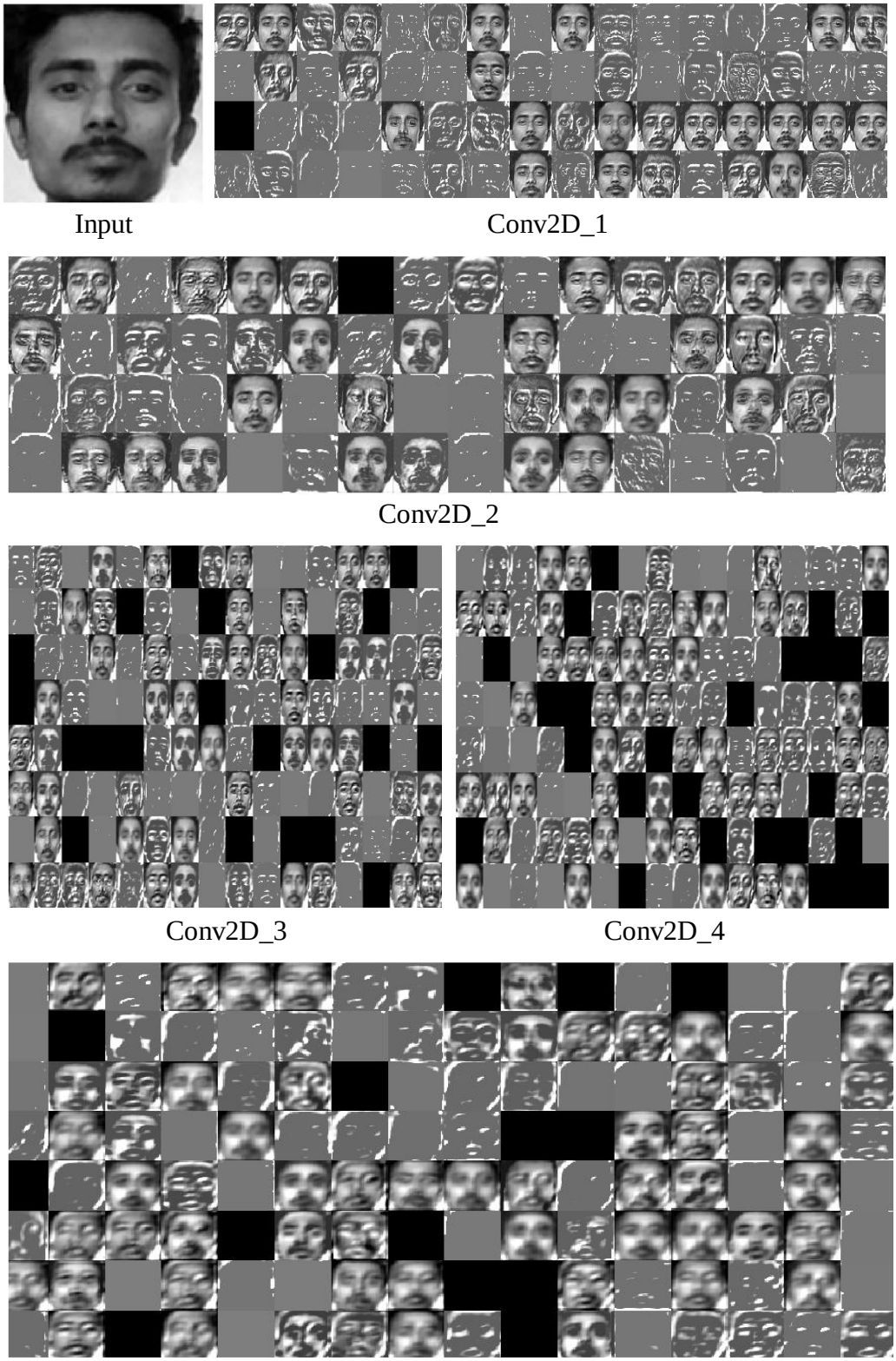


Fig. 6.6 Intermediate Layer Visualisation of CNN

6.6 Evaluating CNN Model

Model evaluation metrics are required to quantify model performance. The choice of evaluation metrics depends on a given task (such as classification, regression, ranking, clustering, and topic modeling, among others). Some metrics, such as precision-recall, are useful for multiple tasks. Supervised learning tasks such as classification and regression constitute a majority of applications. Let's see some of the popular metrics to evaluate the model performance.

1. Confusion Matrix:

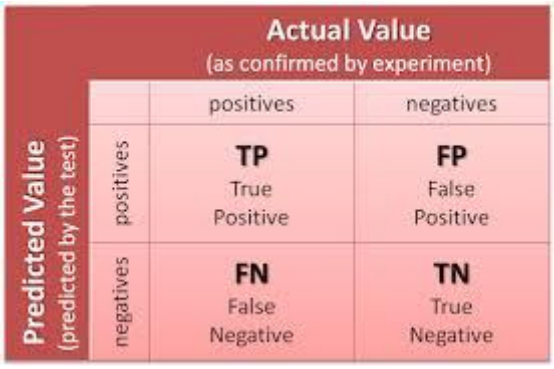


Fig. 6.7 Explanation of Confusion Matrix

A confusion matrix is a technique for summarizing the performance of a classification model. The number of correct and incorrect predictions is summarized with count values and broken down by each class. This is the key to the confusion matrix. Figure 6.7 shows the interpretation of the confusion matrix.

- True positives (TP): Predicted positive and are actually positive.
- False positives (FP): Predicted positive and are actually negative.
- True negatives (TN): Predicted negative and are actually negative.
- False negatives (FN): Predicted negative and are actually positive.

2. Classification Accuracy:

Accuracy in classification problems is the number of correct predictions made by the model over all kinds of predictions made. In the Numerator, are our correct predictions (True positives and True Negatives), and in the denominator, is the kind of all predictions made by the algorithm (Right as well as wrong ones).

$$\text{Classification Accuracy} = (TP + TN)/(TP + FN + TN)$$

Formula 6.1: Accuracy

3. Precision:

Precision is a valid choice of evaluation metric when we want to be very sure of our prediction.

$$\text{Precision} = (TP)/(TP + FP)$$

Formula 6.2: Precision

4. Recall:

Another very useful measure is recalled, which answers a different question: what proportion of actual Positives is correctly classified?

$$\text{Recall} = (TP)/(TP + FN)$$

Formula 6.3: Recall

5. F1 Score:

The F1 score is a number between 0 and 1 and is the harmonic mean of precision and recall.

$$\text{F1 Score} = (2*\text{Precision}*\text{Recall})/(\text{Precision} + \text{Recall})$$

Formula 6.4 F1 Score

6. Kappa Score:

Kappa Score is a statistic that measures inter-annotator agreement. This is a function that computes Cohen's kappa, a score that expresses the level of agreement between two annotators on a classification problem. It is defined as

$$K = Po – Pe/1 - Pe$$

Where Po is the empirical probability of agreement on the label assigned to any sample (the observed agreement ratio), and is the expected agreement when both annotators assign labels randomly? Pe is estimated using a per-annotator empirical prior over the class labels.

6.7 Performance analysis

The average execution time of face recognition authentication under our method is 4.92 seconds, and the average execution time of the eye blink authentication under our method is 14.92 seconds.

In this section, "user self", "color photo" and "face mask" are evaluated. During the experiment of user identity authentication in-person self, each participant personally uses each user identity authentication on the mobile devices. However, each algorithm

may be failed when the faces have different angles, out of detection distance, and rich expression. These cases degrade the accuracy of each algorithm. Fig. 8 shows that our

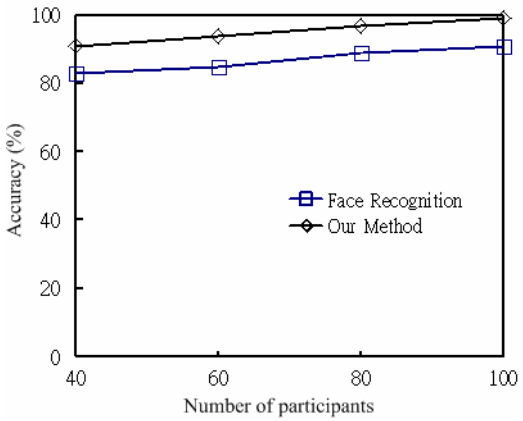


Fig. 6.8 Accuracy of face recognition and eye-blink detectionunder the user's self

The method is slightly superior to face recognition under user- self-identification.

In the "color photo" recognition, we print the color photos of the user in an A4 size. Fig. 9 shows that the color photos of the user can easily cheat face recognition. Eye blink detection can perform better than face recognition in the color photos of the user. However, the color photos of the user under the handshake deteriorate the performance of our method due to the false alerts of theeye blinking.

In the face mask, we make the masks of the users from the color photos of the users. Fig. 10 shows that face recognition is still cheated by using the mask of the user's color photo. Fig. 10 also demonstrates that our method can easily recognize the fake user identity of the mask since our method uses the horizontal and vertical projections of the eye region to detect the dark gaps around the eye region of the mask.

We evaluate the performance of each method by using several metrics such as precision, recall, and accuracy defined in shows that each method can perform well to detect eye blinking as the users do not wear glasses. When the users wear glasses with thin rims and thick frames, Krolak is worst among all methods as shown in Tables 2 and 3. This is because Krolak a traditional eye-

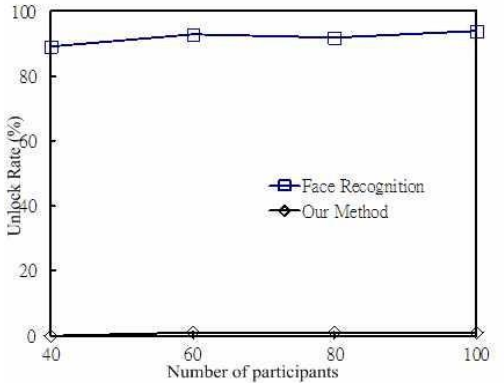


Fig. 6.9 Unlock rate of face recognition and eye-blinkdetection under the color photos of the user's self

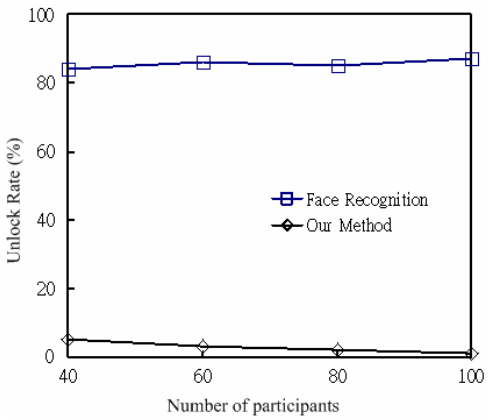


Fig. 6.10 Unlock rate of face recognition and eye-blinkdetection under the mask of the user's self

Method	**Precision**	**Recall**	**Accuracy**
Krolak	92.82	94.17	91.53
REGT	94.03	95.76	92.38
Our method	97.63	99.27	95.88

Table 6.2: Performance of eye-blink detection under no-glasses wearing

Method	Precision	Recall	Accuracy
Krolak	83.18	85.33	80.47
REGT	95.54	96.91	93.72
Our method	96.81	98.36	94.93

Table 6.3: Performance of eye-blink detection under glasseswearing with a thin rim

Method	Precision	Recall	Accuracy
Krolak	76.04	78.28	74.35
REGT	90.55	91.18	87.49
Our method	94.91	96.25	93.74

Table 6.4: Performance of eye-blink detection under glasseswearing with a thick frame

blink detection does not consider bright noise and glasses frame over the location of eyes due to glasses wearing. Although REGT yields fair performance under glasses wearing, it is still inferior to the proposed method as demonstrated in Tables 2 and 3. The reason is that the frame difference in REGT cannot effectively filter out the bright noise and the edges of the glasses frame.

CHAPTER 7

IMPLEMENTATION

7.1 System Requirements

The project proposes a facial recognition system using Convolutional Neural Network (CNN). The CNN implicitly extracts the features from the human faces and predicts the class for a given human face which can be achieved through training and hyperparameter tuning of the proposed novel CNN. The project also aims to develop an end-to-end process to build a custom facial recognition system i.e. to deal with real-time data. This section describes the proposed web-based application for attendance posting using face recognition using CNN, which is developed using Flask framework and python3.

The main components of the application are:

- Front-end-interface (Camera)
- Server
- CNN model
- Database
- End result (Display)

7.1.1 Front-End Interface:

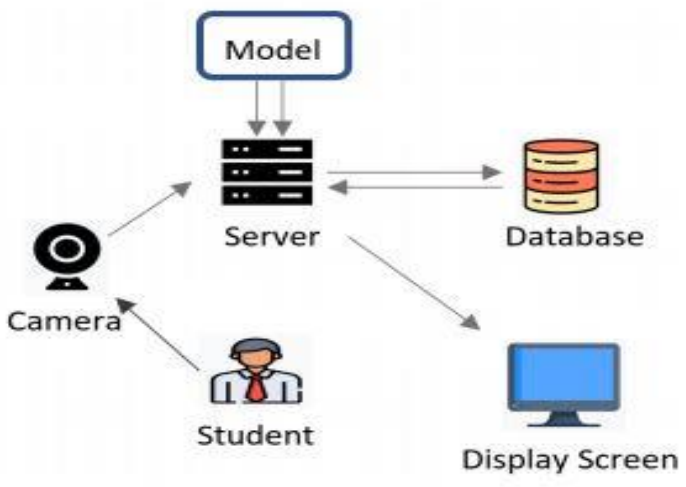


Fig. 7.1 An overview of proposed application

A simple web page that takes the video frame as input through the camera using an open computer vision (OpenCV) library. The video frame that is taken as input is then sent to the server where already the novel CNN model is already dumped. It is to be noted that in the image returned from the front-end interface the location of the face is identified by the automated script and a newly resized image fit for the input size of the model is formed.

7.1.2 Server:

The back end of the application maintains the connection with the front-end interface, model, database, and display screen. The input frame that is taken from the front-end interface is processed to identify the face location and resize it accordingly to the input size for the model for prediction. The script and the model necessary for processing the image and providing the input to the model respectively are previously dumped into the server and a call is made accordingly whenever the application comes to the running stage. Finally, the most important duty of the server is maintaining the database regarding the information on existing faces as well as their posted attendance for a particular period.

7.1.3 CNN model:

The novel CNN model was developed for real-time datasets with high accuracy. It receives a resized image that is obtained from the front-end interface as input and generates a label that is nearer to a face existing in the database.

7.1.4 Database:

A database is created which maintains the details regarding details of students, and their attendance including the timestamps. It becomes handy for the server to update, and select values whenever and wherever necessary.

7.1.5 End-result:

The end result is a web page that includes the details of the student such as his/her name, registered number, period, the status of attendance, and timestamp.

7.2 Working on the application

As a beginner in machine learning, it might be easy for anyone to get enough resources about all the algorithms for machine learning and deep learning but when we start to look for references to deploy ML models to production, a very profitable solution is to use the very light web framework to deploy the successfully trained ML models as well as to create an API,(Application Programmable Interface). The Flask framework is really flexible and handy to develop complete end-to-end web applications for any type of project and is also very suggestible when deploying API for the working of machine learning models.

Firstly, after running the application, the very first screen is the web page (front-end interface) that helps the user to capture the video frame. On the web page, a small button is highlighted, once after clicking it the video frame is processed to identify the location of the face in the frame and then it is resized into those dimensions that are fit for the input size of the developed CNN model. After successfully resizing the input frame, it is loaded into the model for making a prediction.

The Flask framework provides the scope to import any package from Python directly, so with this convenience, we are going to import all the necessary packages for loading the model. Generally, a function namely load model from the package TensorFlow is used to load the model. After loading the model, the resized image is fed to the model for processing to generate a label. The novel architecture of the proposed custom *CNN* model uses the classifier softmax, which is an activation function that turns numbers into probabilities that sum to one.

The CNN model outputs a vector that represents the probability distributions of a list of potential outcomes. Since the desired output is a single label related to only one of the trained faces, the arg max function helps in taking out the maximum of the probability distributions generated by softmax. Since the output from the softmax layer is a single-dimensional vector when it is passed to the arms function the index of the maximum value is returned. From that index value, the server matches the existing label values with that of the generated index value and if there is a match, it posts all the details corresponding to that label in the form of a web page containing the information such as the status of attendance which includes Student Name, Registered Number, Period, Timestamp.

7.2.1 Code Description:

```
import cv2
import cvzone
from cvzone.FaceMeshModule import FaceMeshDetector
from cvzone.PlotModule import LivePlot
import csv
cap = cv2.VideoCapture(0)
detector = FaceMeshDetector(maxFaces=1)
plotY = LivePlot(400, 400, [20, 50], invert=True)
idList = [22, 23, 24, 26, 110, 157, 158, 159, 160, 161, 130, 243]
ratioList = []
blinkCounter = 0
counter = 0
color = (255, 0, 255)
```

```
# Open a file in write mode
file = open('blink_data.csv', mode='w', newline='')
# Create a CSV writer object
writer = csv.writer(file)
# Write the header row to the file
writer.writerow(['Blink Count'])
prev = 1
while True:
    success, img = cap.read()
    img, faces = detector.findFaceMesh(img, draw=True)
   if faces:
        face = faces[0]
        leftUp = face[159]
        leftDown = face[23]
        leftLeft = face[130]
        leftRight = face[243]
        lenghtVer, _ = detector.findDistance(leftUp, leftDown)
        lenghtHor, _ = detector.findDistance(leftLeft, leftRight)
        cv2.line(img, leftUp, leftDown, (0, 200, 0), 3)
        ratio = int((lenghtVer / lenghtHor) * 100)
        ratioList.append(ratio)
        if len(ratioList) > 3:
            ratioList.pop(0)
        ratioAvg = sum(ratioList) / len(ratioList)
        if ratioAvg < 35 and counter == 0:
            blinkCounter += 1
            color = (0,200,0)
            counter = 1
        if counter != 0:
```

```
            counter += 1
            if counter > 10:
                counter = 0
                color = (255,0, 255)
            cvzone.putTextRect(img, 'press q to exit', (50, 100),
                               colorR=color)
            if blinkCounter != prev:
            prev = blinkCounter
            print(blinkCounter)
            writer.writerow([blinkCounter])
           imgPlot = plotY.update(ratioAvg, color)
           img = cv2.resize(img, (400, 400))
    else:
        img = cv2.resize(img, (720, 720))
     cv2.imshow("Image", img)
  if cv2.waitKey(1) & 0xFF == ord('q'):
        # If the 'q' key is pressed, break the loop
        break
    file.close()
# Release the VideoCapture object and close all windows
cap.release()
cv2.destroyAllWindows()
```

CHAPTER - 8

RESULTS

The experiment is conducted by using the data which contains 10,029 samples and the data is divided into train and test in order to train and evaluate the CNN model developed. The details of the test and training are shown in Table 8.1. The experimental results of the CNN model and the web application which uses the developed novel CNN model are discussed.

Data	Percentage	Samples
Train	85%	8524
4	15%	1504

Table 8.1: Details of Train and Test Data

8.1 CNN Model

As it is a supervised learning task, Hence the evaluation of the CNN model is carried out using the classification metrics such as Confusion Matrix, Precision, Recall, F1 Score, and Kappa Score.

- The Confusion Matrix is a table that is often used to describe the performance of a classification model on a set of test data.

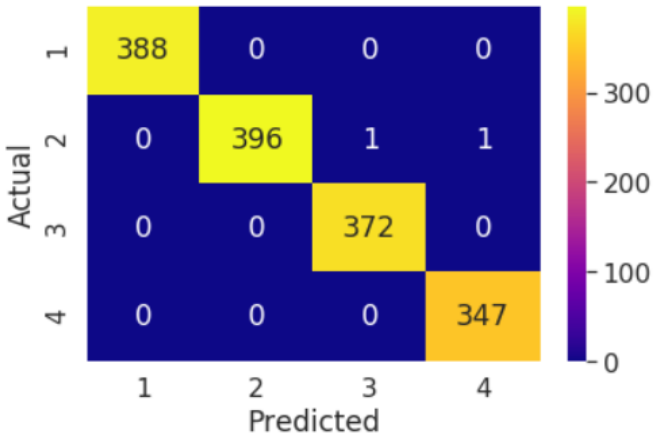


Fig. 8.1 Confusion Matrix

- The class-wise accuracy of the CNN model represents the classification accuracy with respect to each class based on the predictions of the test data and it is represented in Table 8.2.

Class Label	Accuracy
1	100%
2	99.49%
3	100%
4	100%

Table 8.2: Class-wise Accuracy

- The different classification metrics such as Classification Accuracy, Precision, Recall, F1 Score, and Kappa Score are used to evaluate the CNN model performance on the test or unseen data, and the results are illustrated in Table-8.3.

Metric	Value
Classification Accuracy	99.8671%
Precision	99.861%
Recall	99.874%
F1 Score	99.867%
Kappa Score	99.822%

Table 8.3: Classification Metrics

- The Accuracy and the loss of the CNN model are 99.86% and 0.0019 and predictions samples of the images from the test data are shown in Figure 8.2.

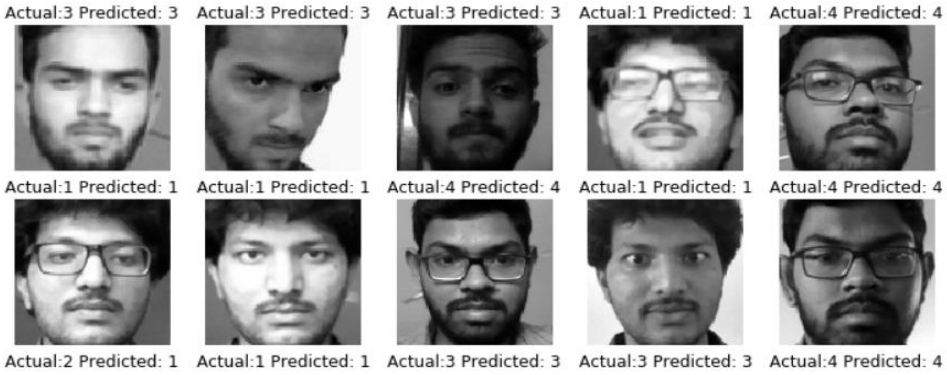


Fig. 8.2 Predictions of Sample Test Images

8.2 Web Application

The web application is designed to post attendance using the developed CNN model for face recognition. The web application posts the attendance into the database for the designated periods based on the timings of the college hours. Figure 8.3 shows the screenshots of the web application which illustrates the capturing of the images and the end result of the application i.e., the status of the attendance is displayed

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENTS

9.1 CONCLUSION

The project formally introduces the role of Convolutional Neural Networks (CNNs) in Face Recognition and the adaption of CNN in attendance posting and also proposes a novel CNN architecture for Face Recognition. The project also provides a detailed explanation of the key components which are essential to building a robust deep learning model, such as collecting real-time data i.e., Human Faces, Data Augmentation and Pre-Processing, Training, and Hyper Parameter Tuning the proposed CNN model. Moreover, the project also provides a web application for attendance posting using Face Recognition by using the developed CNN model.

9.2 FUTURE ENHANCEMENTS

The future scope of the project is to build the dataset with more classes and to train the proposed CNN model as the project uses only four classes. To build a robust web application with different features such as Login Page, Attendance Management, Student Section, Faculty Section, etc.

CHAPTER 10

REFERENCES

[1] S.Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” in 2017 International Conference on Engineering and Technology (ICET). IEEE, 2017, pp. 1–6.

[2] G. Hu, Y. Yang, D. Yi, J. Kittler, W. Christmas, S. Z. Li, and T. Hospedales, “When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition,” in Proceedings of the IEEE International conference on computer vision workshops, 2015, pp. 142–150.

[3] J.Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE Conference on computer vision and pattern recognition, 2016, pp. 779–788.

[4] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” arXiv preprint arXiv:1901.06032, 2019.

[5] A. El-Sawy, E.-B. Hazem, and M. Loey, “Cnn for handwritten Arabic digits recognition based on lenet-5,” in International conference on advanced intelligent systems and informatics. Springer, 2016, pp. 566–575.

[6] Z.-W. Yuan and J. Zhang, “Feature extraction and image retrieval based on alexnet,” in Eighth International Conference on Digital Image Processing (ICDIP 2016), vol. 10033. International Society for Optics and Photonics, 2016, p. 100330E.

[7] H. Qassim, A. Verma, and D. Feinzimer, “Compressed residual-vgg16 cnn model for big data places image recognition,” in 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2018, pp. 169–175.