IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Face recognition application using python

Zainab Ansari,Ujjawal Kanoujia, Mr.Lokesh Kumar, Manvi Sundli

(za800306@gmail.com) (ujjawalkanoujia789@gmail.com)(cse.hod@tulas.edu.in)(manvisundli98@gmail.com)

Software Engineer, Software Engineer, Assistant Professor, Software Engineer

Computer Science,
Tula's Institute, Dehradun, India

ABSTRACT

The history of facial recognition and its introduction is provided in below. The advantages and drawbacks of the facial recognition system are discussed in this section. Section 3 summarized the functional and non-functional requirements of the project and the feasibility of the project. The development method we adopted to build the application and tools and technologies used to build the application are discussed. The structural designs of our application and how the application functions in various layers is discussed. **Various** diagrams and **figures explain step-by-step** how the **whole** application works.

Keywords:

Face **Recognition,** Local Binary Pattern Histogram (LBPH) Eigenface, AdaBoost, **Cascade** Haar Classifier, Principal Component Analysis (PCA), Fast PCA, Linear Discriminant Analysis (LDA).

1. Introduction

The human face has **unique** physical **shapes** and **features** that are used to identify or **confirm identity. Face Recognition** records this **facial biometric. Various** face recognition methods measure **facial** biometric **parameters.** Facial recognition has become a very important topic in recent years. Facial recognition is effectively applied **to** various applications **such as** security systems, authentication, **access** control, surveillance **systems, smartphone** unlocking and social networking systems, etc.

Face recognition is not **used** as the **primary** form of **badge in most practices.** However, **advances** in technology and **algorithms allow** facial recognition **systems** to replace standard **password** and fingerprint scanners.

Facial recognition systems are employed throughout the world today by governments and private companies. Their effectiveness varies, and some systems have previously been scrapped because of their ineffectiveness. The use of facial recognition systems has also raised controversy, with claims that the systems violate citizens' privacy, commonly make incorrect identifications, encourage gender norms and racial profiling, and do not protect important biometric data. These claims have led to the ban of facial recognition systems in several cities in the United States. As a result of growing societal concerns, Meta announced that it plans to shut down Facebook facial recognition system, deleting the face scan data of more than one billion user. This change will **be** one of the **biggest changes** in **the use of** facial recognition in the **history of technology.**

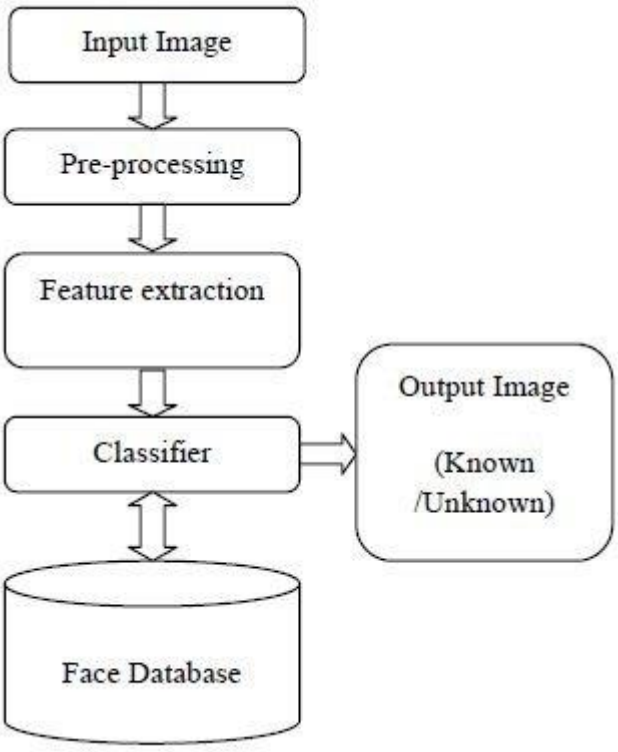

Fig 1. Basic **block diagram** of **face recognition.**

Different approaches to face recognition in still images can be divided into three main groups of stages.

- Face Recognition
- Face Extraction
- Face Recognition

2. Face Recognition Method

2.1. Face Detection

Face detection is a function that identifies a person's face in a digital image. The system identifies the face of a person present in an image or video. To determine which photo or 4 videos contain a face (or several), you need to see the full structure of the face. Human faces share similar features such as eyes, nose, forehead, mouth, and chin. So, the purpose of face detection is to find the position and size of a face in an image. The detected face is used in a face recognition algorithm.

One of the important problems we are trying to solve in computer vision is the automatic detection of objects in images without human intervention. Face detection can be thought of as such a problem when it comes to detecting human faces in images. Human faces may have some differences, but in general it's safe to say that there are certain characteristics associated with everyone's face. Although there are many different face recognition algorithms, the Viola-Jones algorithm is one of the oldest methods still in use today and we will use it later in this article. After finishing this article, you can take a look at the Viola-Jones algorithm. We will provide a link to this at the end of this article. Face detection is usually the first step in many face-related technologies such as face recognition or face authentication. However, facial recognition can have very useful applications. Perhaps the most successful application of facial recognition is photography. When you take a picture of your friend, the digital camera's built-in facial recognition algorithm detects the position of your face and adjusts focus accordingly.

2.2 Feature Extraction:

In this step, features are extracted from the detected face. In LBPH, the first local binary image of the image is computed and a histogram for face recognition is generated. This will create a template. A template is a data set that represents the unique and unique characteristics of a detected face.

Now that we have cropped the face **from** the image, we extract **the** features from **the image. Here,** we **will** use face **embeddings** to extract **facial features. The** neural network takes an image of **a human** face as input and outputs a vector **representing** the most important **facial features.** In machine learning, this vector is called **an embedding, so** this vector **is called a** face embedding.

2.3. Face Recognition:

Face Recognition **allows you** to uniquely identify and verify a **person's** face by comparing and analysing **biometric data.** A **facial** recognition system is an application used **to identify** or **verify an identity in** a digital image. **At** this **stage,** the algorithm **has** already **been** trained. Each histogram **generated** is used to represent each image **in** the training **data set.** So, **if you have** an input image, **you step back on** this new image and **create** a histogram **representing** the image.
So, to find **an** image that matches the input **image,** we just need to compare **the** two histograms and return the image with the closest histogram.

Different approaches can **be used** to compare histograms **(compute** distance between two **histograms). Example:** Euclidean distance, **chi-square,** absolute value, **etc.** In this example, we can use Euclidean distance based on following formula:
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.

Tools and Technologies:

Tools and techniques used in the project are described in this section. This project focused was mainly focused on Python Programming and its libraries.

PYTHON:

Python is a very popular programming language and can be used for various purposes.
It was created by Guido van Rossum in 1991 as Python 0.9.0. It was created as the successor of the ABC programming language. Python 2.0 was released on 16 October 2000 and added many features like list comprehension and garbage collecting system. On 3 December 2008, Python 3.0 was released. Python is a multipurpose programming language that works on different platforms like Windows,

Linux, Mac, Raspberry Pie, etc. Python is **more** popular than other programming languages **because of its simpler** syntax than other programming languages. Its syntax allows programs to write **easy-to-understand** code **with** fewer lines. It runs **on the** interpreter **machine. So you can run your** code as soon as **you write it.**

This project uses Python for web **development.**This project **showed** how **to use** Python for **efficient** and reliable web **applications. This project uses various** Python **frameworks and libraries.**

FLASK:

Flask was created by Armin Ronacher of Pocoo, an international group of Python enthusiasts formed in 2004. According to Ronacher, the idea was originally an April Fool's **prank, which became so popular** that **it turned** into a serious **app.** The **title** is a **game** on the **old** Bottle **platform.** Flask is a **microweb** framework written in Python. It is classified as a microframework because it does not require **any special** tools or libraries. **There is** no database abstraction layer, form validation, or other components **that existing third-party** libraries provide **in common.** However, Flask supports extensions that can add application **functionality** as implemented in Flask itself. **There are extensions to object-relational** mappers, form validation, upload handling, various **public** authentication **techniques,** and several common **framework-related** tools.

Components of Flask:

Flask is interconnected to two main parts.

• Werkzeug is a utility library meant for usage with the Python language. Mostly, it is a Web Server Gateway Interface or WSGI app that can create software items for request, response, or utility functions.
• Jinja 2 is a template engine for Python programming purposes, and it resembles the Django web frameworks templates.

OpenCV:

OpenCV is an opensource machine learning and computer vision library. OpenCV is a **free, cross-platform library.** It was **released** in 1999. Intel **released** OpenCV to **promote** **CPU-intensive** applications.

Developed in C++. **Provides** bindings for **the** Java and Python programming **languages .**It runs **on various** operating systems such as Linux, Windows, OSx, etc. It focuses on video **capture,** image **processing** and analysis. It has face detection and **object detection.**

OpenCV can be used to read and write **images,** capture and save **video. You** can perform feature detection **such as** faces, cars, images, **and more. Libraries are used by** Yahoo, Google, Microsoft, Intel, and many **other well-known companies.**

A person had to manually **and accurately determine** the coordinates of facial features such as the **center of** the **pupil, the inner** and **outer corners** of **the eye,** and the **widow** peak **of** the hairline. The coordinates were used to calculate 20 distances, including the width of the mouth and eyes. **So a** human **can** process about 40 **images per hour, making** a database of **calculated** distances. **The** computer then automatically **compared** the distances **in** each **photo, calculated** the difference **in distances,** and **returned records** closed **with** possible **matches. The** algorithm uses edge or line detection proposed by Viola and Jones in their **2001** research paper **"Fast** Object Detection **Using** Boosted Cascade of Simple **Functions".** The algorithm is given **many** positive **face** images and **many** negative **non-face** images to train **on.** The model created **in** this **tutorial** is available **in** the OpenCV GitHub repository. **You** can read **it using** the OpenCV **method. This includes** models **such as** face **recognition,** eye detection, upper and lower body detection, **and** license plate **detection.**

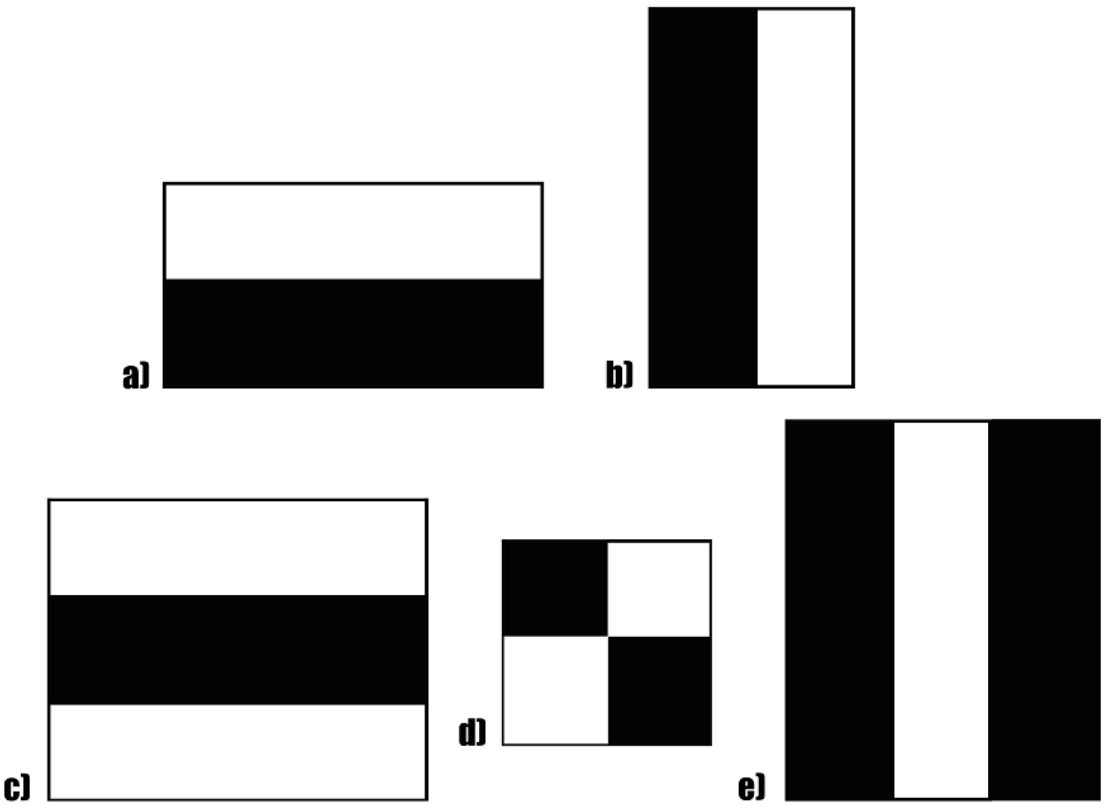

Figure: Example of **a** Haar **function**

Methodology:

This section describes how LBPH is used for face recognition. First, a **data set for the images** is collected and each image is **tagged** with a unique **ID.** The **image is split** into an 8X8 grid and converted **to** grayscale. A 3X3 matrix of each pixel containing **the** intensity **(0-255)** is extracted from the image. The central value **threshold** of this matrix is **taken,** which **determines** the **adjacent values of** the **matrix. Each adjacent** value is compared **to** the central value. **Set to 1 if** the **adjacency** value is greater **than** or equal to the threshold **and** set to **0 if** the **adjacency** value is less than the **threshold. The** matrix **values will then** contain **only** binary **values. Decimal** values **are** calculated using the **following** formula :

$$LBP(xc,yc) = \sum 7\ n=S(ic - in)2n$$

In the above formula, **'n'** is the 8 neighbours of the **center** pixel, **ic** and in are the level **values of** the **gray center** pixel and the **periphery,** respectively. **Pixels. If** x is greater than or equal to the **threshold, then S(x) is equal to 1.** S(x) is 0 if x is less than the threshold.

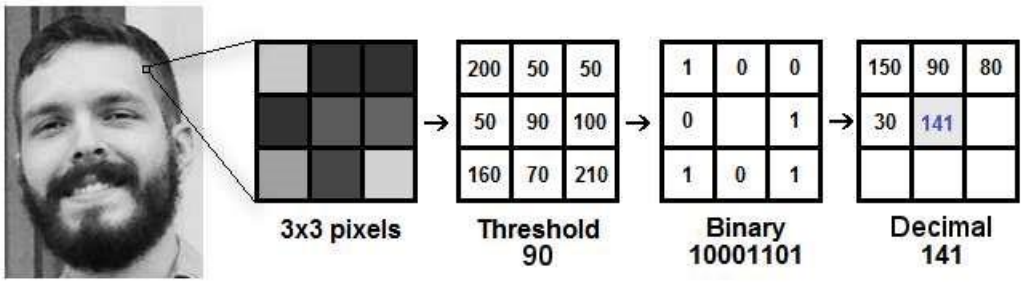

Fig: Extracting 3X3 **Pixel Matrix Extraction**

Replaces the calculated decimal value with the **centroid. So in the new image you get** the characteristics of the original image. **When** all processes are complete, **we extract the histograms** from each grid and **merge them.** This process is repeated for all images and a histogram is generated. **Compare histograms simultaneously to** compare two **images. Comparisons** are **performed using histogram intersections.** The formula **is:**

$$\sum j=1 \min(Ij , Mj)$$

where j is the **cell** number and I and M are histogram 1 and histogram 2. If the intersection value **was** greater than **80%,** the image **was recognized successfully.**

Implementation:

This section describes how the algorithm was implemented **for system** design **and** system **testing.** The application was **built** using **the Python Flask** framework. Both the frontend and backend of the project were **built** using **Flask.**

System Design:

The design follows **a three-tier architecture** described below.

Presentation Layer:

This layer is responsible for the user **interface.**All the components that users see and interact with within the application are in this layer.

Application Layer :

Application layer controls the overall functionality of the system. Functionality such as logging into the system, facial detection, and recognition is all done in this layer.

Data Layer:

In this layer, Data and Information are stored and retrieved in the database. The names, images of students as datasets, teaches are stored in the database. Once the face is matched, marking of attendance in the database.

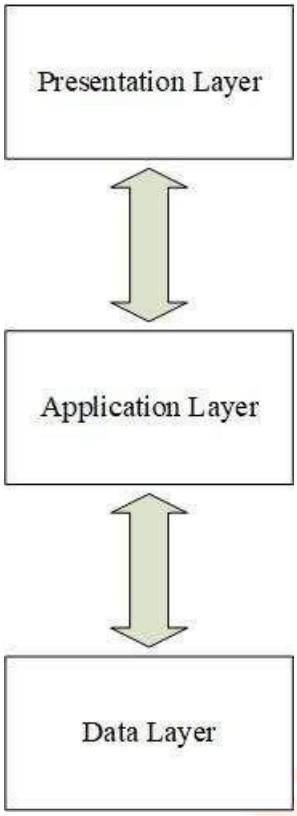

Fig : System Design

Database Design :

For this project, the default flask SQLite was used to create the database. The tables are created by flask's model. It provides an ORM to the underlying database. ORM makes it easy to work with relational databases. The models in flask are Object, which is mapped to the database. When a model is created, flask creates the corresponding table without having to write any SQL code.

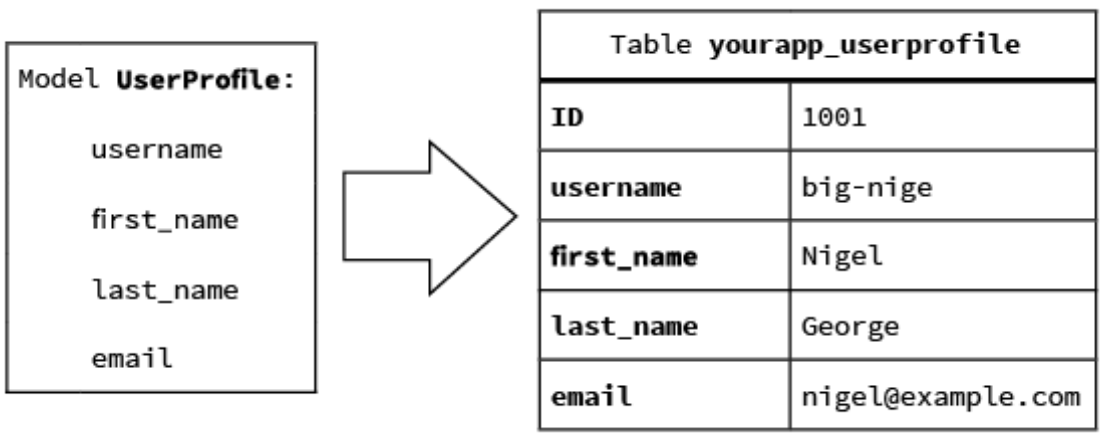

Fig : Flask model and its corresponding table in the database

Creating a model in flask is easy. It contains essential fields needed for our data and specifies the behaviour of the data. Each model is mapped to a single table in the database.

```
def index():
    if request.method=='POST':
        student_id=request.form['ids']
        student_password=request.form['password']
        cursor = mysql.connection.cursor()
        cursor.execute('SELECT * FROM registration WHERE id=%s', (student_id,))
        temp = cursor.fetchone()
        print(temp)
        if temp:
            if temp[7] == student_password:
                return redirect(url_for('dashboard', id=temp[0]))
            else:

                return render_template('Login.html', res='wrong', ids=student_id, pwd=student_password)
        else:
            return render_template('Login.html', sneha='wrong')

    else:
        return render_template('Login.html')
```

Fig : Login Model

Login **Model** contains fields **such as Student ID** and Password for **login, so you can see if** students are **currently visible** or total attendance **so far. I** used the flask login module **for** access control. **Provides** user session management **(login, logout** and remembering **sessions) for Flask.**

Enrolment Model:

```
@app.route('/registration',methods = ['POST', 'GET'])
def registration():
    if request.method=='POST':
        student_id=request.form['ids']
        student_name= request.form['name']
        student_branch = request.form['branch']
        student_mob = request.form['mob']
        student_Email = request.form['Email']
        student_address = request.form['add']
        student_password = request.form['pwd']
        print(student_id,student_name,student_branch,student_Email,student_address,student_password)
        dates = datetime.datetime.now()
        cursor = mysql.connection.cursor()
        cursor.execute('SELECT * FROM registration WHERE id=%s', (student_id,))
        temp = cursor.fetchone()
        if temp:
            return render_template('register.html', res='fail')
        else:
            cursor.execute(
            'INSERT INTO registration(id,name,branch,mobile,email,address,password,date) VALUES (%s,%s,%s,%s,%s,%s,%s,%s)',
            (student_id, student_name, student_branch, student_mob, student_Email, student_address,
             student_password, dates))
            mysql.connection.commit()
            cursor.close()
            return redirect(url_for('index'))
    else:
        return render_template('register.html')
```

Enrolment Model is for **enrolling** unregistered students. The **enrolment** model **includes** fields **such as** Student **ID,** Student Name, **and** Student **Department Mobile Number.**, Email, Address, Password.

RESULTS AND DISCUSSIONS:

As this was a small-scale project, data structure and implementation did not have many problems. However, it took the author many efforts with research and study with different technologies needed as these tools and technologies were new to the author. This caused a delay in the development of the project. Despite the delay and difficulties, the author was able to incorporate those tools and technologies and complete the project. However, the success rate of facial recognition was not as expected.

The success rate is depended upon the quality of the camera, lighting, and sufficient dataset in the database. When these factors were to be managed properly, the success rate of face recognition increased. The effort that went to learn and research about LBPH and flask and other tools and technologies was worth it. While the process of researching and implementing was overwhelming, it started to be interesting as the project started to show some results. This project gave the author first hand experience in working on a project using flask and found flask easier and more scalable.

CONCLUSION:

The goal of the project was to build a facial recognition system for student's attendance. Concepts of facial recognition and LBPH thesis. Similarly, web development with flask is also discussed, followed by examples of implementation and explanations.

The result of the project was a successful prototype of a facial recognition system where the admin can create a teacher account and add students and their information to the database. Teachers then can log in to the system and take attendance of the student. The student's face is detected by a camera and attendance is recorded in the database. Teachers and admin could see the attendance report of the students.

Overall, the project was successful in its showcasing how LBPH can be implemented in flask to create a web application. Once implemented, it can be used to take attendance of students and keep track of their attendance records.

REFERENCES:

1.John D. Woodward, Jr., Christopher Horn, Julius Gatune, and Aryn Thomas "Biometrics: A look at facial Recognition," 2003.

URL: https://apps.dtic.mil/sti/pdfs/ADA414520.pdf

2. Li Wang, Ali Akbar Siddique. "Facial recognition system using LBPH face recognizer for anti-theft and surveillance application based on drone technology" , Measurement and Control, 2020

URL: https://journals.sagepub.com/doi/10.1177/0020294020932344

3. A brief history of facial recognition. 2020.

URL: https://www.nec.co.nz/market-leadership/publications-media/abrief-history-of-facial-recognition/

4. Mark Andrejevic & Neil Selwyn (2020) "Facial recognition technology in schools: critical questions and concerns, Learning, Media and Technology,"

URL: https://www.tandfonline.com/doi/full/10.1080/17439884.2020.1686 014

5. Python Documentation [online].

URL: https://www.w3schools.com/python/python_intro.asp

6. LBPH OpenCV: https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#local-binary-patterns-histograms

7. Local Binary Patterns:

http://www.scholarpedia.org/article/Local_Binary_Patterns