

CS6301 MACHINE LEARNING LAB
B.E CSE VI Q - BATCH

TEAM MEMBERS: 4

TEAM NO: 1

S. No.	REG. NO.	NAME
1	2019103026	Karthika T
2	2019103035	Monica S
3	2019103506	Anika A S
4	2019103533	Kanimozhi K

Project Title: Student Attendance System using face recognition.

Student Attendance System using face recognition

Problem statement:

The existing system faces the issue of wasting time and it becomes complicated when the strength is more. It is very tedious job to carry out the attendance in log books and to maintain the records. Face recognition is a difficult issue in computer vision. Some of the problems to deal with include lighting issues, posing issues, scale variability, low image capture accuracy, and partially occluded faces are all issues that need to be addressed. As a result, face recognition algorithms must be resistant to changes in the above parameters.

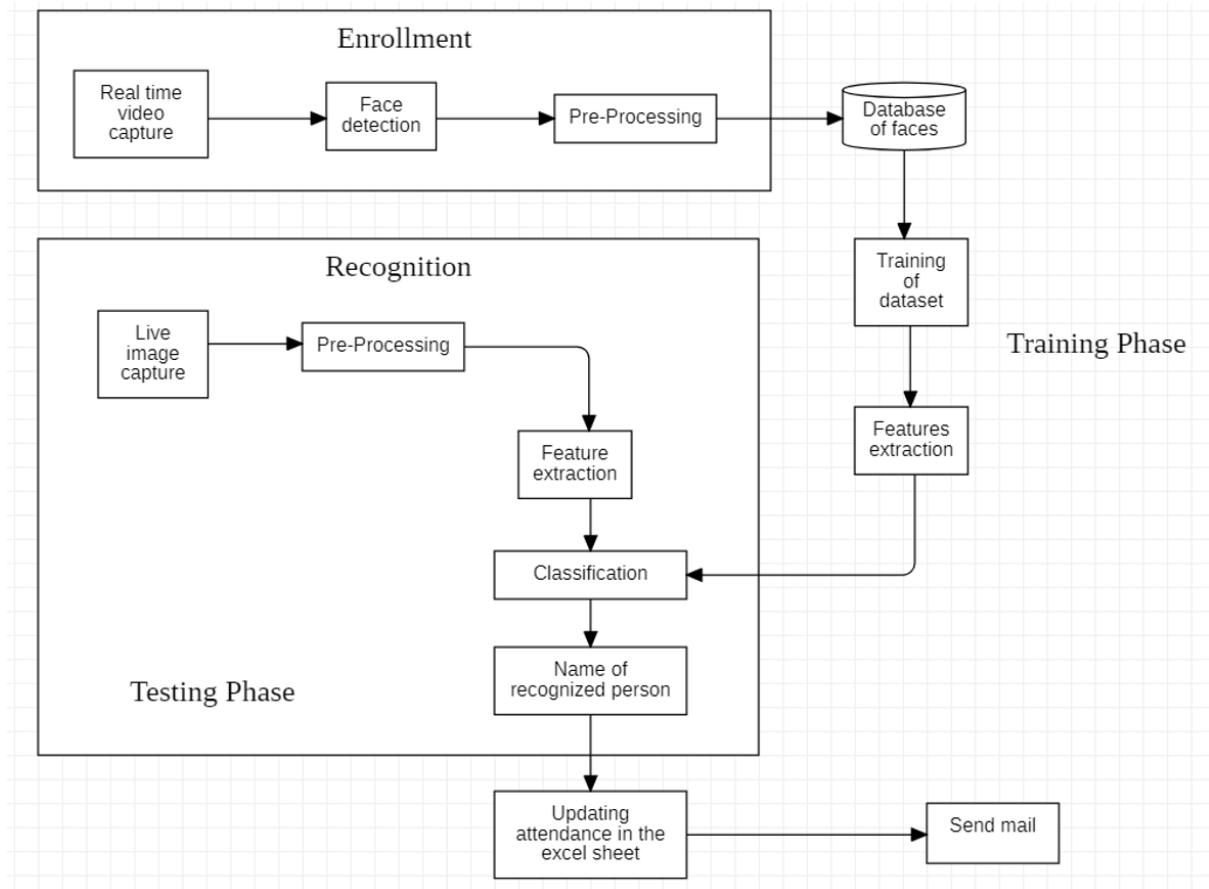
The training process for the face recognition student attendance system is slow and time-consuming in the PCA eigenface algorithm. To deal with this we have used Viola-Jones algorithm is used to detect the face. Facial recognition is done using LBPH. LBPH extracts the histogram of the image and concatenates it to form the face descriptor by segmenting the image into the local region. The distance between the biometrics of the probe image and the trained image is calculated. For calculation, we can use KNN, CNN and SVM algorithm. But CNN achieved low accuracy and SVM achieved low time complexity. To improve accuracy and time complexity, KNN algorithm should be used.

Objectives :

This case study is called Student Attendance System. In this case study we will be dealing with the task of recognising face for student's attendance.

- To reduce the time that is consumed when attendance is taken manually. Unlike the manual process, an online system easily helps management to analyze student's attendance details as per requirement.
- Various reports such as Student-wise attendance, Day-wise attendance, Class-wise attendance, Month-wise attendance, and much more can be easily generated quickly. Attendance list can be printed out easily when required as the data is ready to be obtained from the system with the format based on the manual attendance sheet
- Saving attendance records into the system will be more secured as compared to paper-based records

Architecture diagram:



Modules :

1. Database creation
2. Image Capturing and Detection Phase
3. Image Recognition phase
4. Send mail to parents

List of modules :

Module 1: Database creation

The original database containing the images of the students is created by taking a live real time video of the students, and splitting the video into sixty frames. Converts them to grey scale and storing only the faces of the students as images. The face can be detected by using Haar cascade classifier. The pixel of the image is reduced according to scale factor. The rectangles should be put around the detected face. The time of the video to end is 500ms and it captures until it goes to 60 frames.

Input : Live real time video of the students.

Output : Creation of student database

Module 2 : Training images

We will be training the respective images using the LBPH (Local Binary Pattern Histogram) algorithm and store their respective histogram values in trainer.xml file. LBPH has four parameters radius,neighbours, gridx and grid y to calculate the histogram values. Comparing the stored and trained images against the captured images to mark the attendance.

Input : captured images of students

Output : Trained images of students

Module 3 : Recognition phase

In Image processing we capture the image using a camera. This camera will capture image of its nearby surrounding and only detect the facial part of that particular image and it starts pre-processing the images to overcome the problems of illumination and pose variations. Comparing captured image against the stored images in the database, this method is done by making use of the LBPH algorithm (Local Binary Pattern Histogram). Local Binary Pattern (LBP) 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. Each image stored in the database has histogram value calculated and is cross checked against the calculated Histogram value of the images extracted from the captured video feed. The distance between the

biometrics of the captured image and the trained image is calculated. If the calculated distance is less than the threshold, then the captured image is recognized.

Input : Trained features database

Output : Images of present students are recognised for particular lecture

Module 4 : Send mail to parents

Attendance is marked in an excel sheet, if the uploaded image matches the image stored in the database, then the attendance is marked present for that lecture and saved, For remaining students they are marked as absent in the particular lecture. The csv file will be send to respective students.

Input : Details of recognised students

Output : sent mail

Implementation details with snapshots :

Module 1: Database Creation

The database containing the images of the students is created by taking a live real time video of the students.

Code:

 Checking the camera to capture the video.

```
def camer():
    import cv2
    # Load the cascade
    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    # To capture video from webcam.
    cap = cv2.VideoCapture(0)
```

Defining function for checking the camera and we load the downloaded Haarcascade xml file to detect the faces.

Sample of haarcascade_frontalface_default.xml File:

```

<_>
  <maxWeakCount>9</maxWeakCount>
  <stageThreshold>-5.0425500869750977e+00</stageThreshold>
  <weakClassifiers>
    <_>
      <internalNodes>
        0 -1 0 -3.1511999666690826e-02</internalNodes>
      <leafValues>
        2.0875380039215088e+00 -2.2172100543975830e+00</leafValues></_>
    <_>
      <internalNodes>
        0 -1 1 1.2396000325679779e-02</internalNodes>
      <leafValues>
        -1.8633940219879150e+00 1.3272049427032471e+00</leafValues></_>
    <_>
      <internalNodes>
        0 -1 2 2.1927999332547188e-02</internalNodes>
      <leafValues>
        -1.5105249881744385e+00 1.0625729560852051e+00</leafValues></_>
    <_>
      <internalNodes>
        0 -1 3 5.7529998011887074e-03</internalNodes>
      <leafValues>
        -8.7463897466659546e-01 1.1760339736938477e+00</leafValues></_>
    <_>
      <internalNodes>
        0 -1 4 1.5014000236988068e-02</internalNodes>
      <leafValues>
        -7.7945697307586670e-01 1.2608419656753540e+00</leafValues></_>

```

```

    <_>
      <internalNodes>
        0 -1 5 9.9371001124382019e-02</internalNodes>
      <leafValues>
        5.5751299858093262e-01 -1.8743000030517578e+00</leafValues></_>
    <_>
      <internalNodes>
        0 -1 6 2.7340000960975885e-03</internalNodes>
      <leafValues>
        -1.6911929845809937e+00 4.4009700417518616e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 7 -1.8859000876545906e-02</internalNodes>
      <leafValues>
        -1.4769539833068848e+00 4.4350099563598633e-01</leafValues></_>
    <_>
      <internalNodes>
        0 -1 8 5.9739998541772366e-03</internalNodes>
      <leafValues>
        -8.5909199714660645e-01 8.5255599021911621e-01</leafValues></_></weakClassifiers></_>
<_>

```

By drawing the rectangle around faces, the camera captures the faces and if we press 'q' the camera stops checking.

```
while True:
    # Read the frame
    _, img = cap.read()
    # Convert to grayscale
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

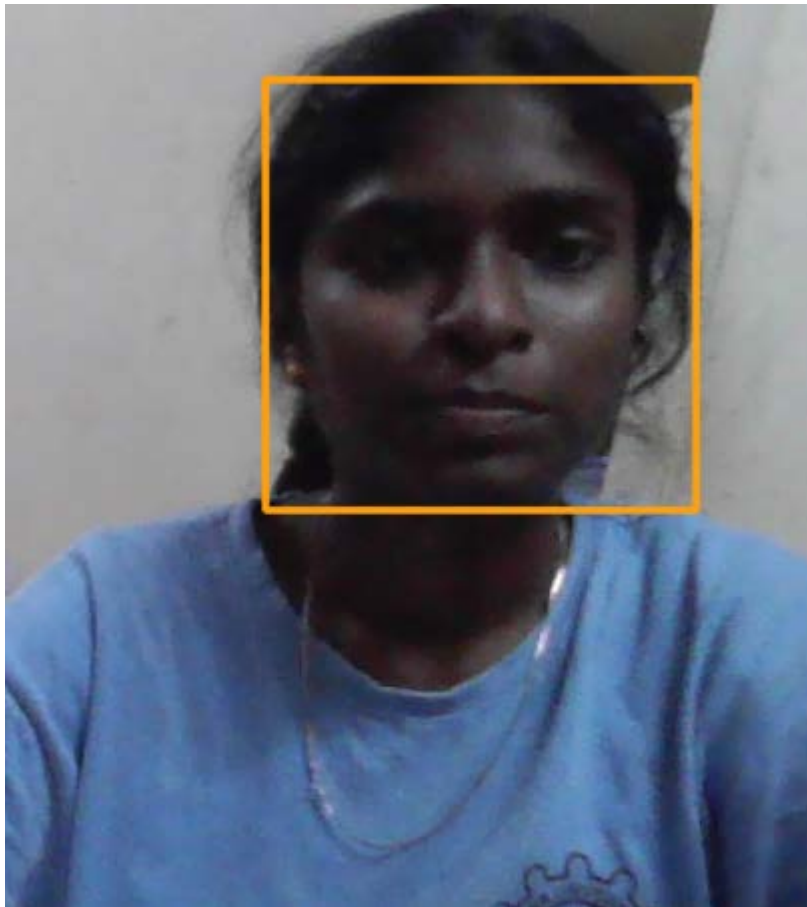
    # Detect the faces
    faces = face_cascade.detectMultiScale(gray, 1.3, 5, minSize=(30, 30), flags=cv2.CASCADE_SCALE_IMAGE)

    # Draw the rectangle around each face
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
    # Display
    cv2.imshow('Webcam Check', img)

    # Stop if escape key is pressed
    if cv2.waitKey(0) & 0xFF == ord('q'):
        break

# Release the VideoCapture object
cap.release()
cv2.destroyAllWindows()
```

Output:



✚ Creating Database of faces for training.

```
def takeImages():  
    Id = input("Enter Your Id: ")  
    name = input("Enter Your Name: ")  
  
    if(is_number(Id) and name.isalpha()):  
        cam = cv2.VideoCapture(0)  
        harcascadePath = "haarcascade_frontalface_default.xml"  
        detector = cv2.CascadeClassifier(harcascadePath)  
        sampleNum = 0
```

For creating database of students we define function for entering their **Id** and **Name**. Then we load Haarcascade xml file to the path.

```
while(True):  
    ret, img = cam.read()  
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
    faces = detector.detectMultiScale(gray, 1.3, 5, minSize=(30,30), flags=cv2.CASCADE_SCALE_IMAGE)  
    for(x,y,w,h) in faces:  
        cv2.rectangle(img, (x, y), (x+w, y+h), (10, 159, 255), 2)  
        #incrementing sample number  
        sampleNum = sampleNum+1  
        #saving the captured face in the dataset folder TrainingImage  
        cv2.imwrite("TrainingImage" + os.sep + name + "." + Id + '.' +  
                    str(sampleNum) + ".jpg", gray[y:y+h, x:x+w])  
        #display the frame  
        cv2.imshow('frame', img)  
        #wait for 200 milliseconds  
        if cv2.waitKey(200) & 0xFF == ord('q'):  
            break  
        # break if the sample number is more than 100  
        elif sampleNum > 10:  
            break  
    cam.release()  
    cv2.destroyAllWindows()
```

Detecting the faces of students using Haarcascade by giving **scaleFactor=1.3** and **neighbors=5**. Then we save the captured faces in TrainingImage folder. If we press 'q' or the image sample is greater than 60 then the camera stops capturing.

Output:



```
res = "Images Saved for ID : " + Id + " Name : " + name
header=["Id", "Name"]
row = [Id, name]
if(os.path.isfile("StudentDetails"+os.sep+"StudentDetails.csv")):
    with open("StudentDetails"+os.sep+"StudentDetails.csv", 'a+') as csvFile:
        writer = csv.writer(csvFile)
        writer.writerow(j for j in row)
        csvFile.close()
else:
    with open("StudentDetails"+os.sep+"StudentDetails.csv", 'a+') as csvFile:
        writer = csv.writer(csvFile)
        writer.writerow(i for i in header)
        writer.writerow(j for j in row)
        csvFile.close()
else:
    if(is_number(Id)):
        print("Enter Alphabetical Name")
    if(name.isalpha()):
        print("Enter Numeric ID")
```

Saving the details of captured faces in **StudentDetails.csv** file.

Student 1:

```
Enter Choice: 1
Enter any key to return main menu: [ WARN:0@15.348] global D:\a\opencv-python\opencv-python
0
-----
**** Student Attendance System using Face Recognition ****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 2
Enter Your Id: 1
Enter Your Name: Preethi
Enter any key to return main menu: [ WARN:0@32.334] global D:\a\opencv-python\opencv-python
0
```

Student 2:

```
-----
**** Student Attendance System using Face Recognition ****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 2
Enter Your Id: 2
Enter Your Name: Sandhiya
Enter any key to return main menu: [ WARN:0@105.934] global D:\a\opencv-python\opencv-py
0
```

Student 3:

```
***** Student Attendance System using Face Recognition *****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 2
Enter Your Id: 3
Enter Your Name: kirupa
Enter any key to return main menu: [ WARN:0@640.025] global D:\a\opencv-python\open
0
█ -----
```

Student 4:

```
█ -----
***** Student Attendance System using Face Recognition *****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 2
Enter Your Id: 4
Enter Your Name: Karthika
Enter any key to return main menu: [ WARN:0@674.639] global D:\a\opencv-python\
0
```

Student 5:

```
-----
**** Student Attendance System using Face Recognition ****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 2
Enter Your Id: 5
Enter Your Name: Monika
Enter any key to return main menu: [ WARN:0@719.488] global D:\a\opencv-python\opencv-pyt
0
```

Module 2: Training of images:

Training Database of faces and extracting their histogram values

```
def getImagesAndLabels(path):
    # get the path of all the files in the folder
    imagePath = [os.path.join(path, f) for f in os.listdir(path)]
    print(imagePath)

    # create empty face list
    faces = []
    # create empty ID list
    Ids = []
    # now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePath:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now we are converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        Id = int(os.path.splitext(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces, Ids
```

Defining function for extracting Images and Labels of particular student and putting them in a array.


```
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, Id = getImagesAndLabels("TrainingImage")
    Thread(target = __recognizer.train(faces, np.array(Id))).start()
    # Below line is optional for a visual counter effect
    Thread(target = __counter_img("TrainingImage")).start()
    recognizer.save("TrainingImageLabel"+os.sep+"Trainer.yml")
    print("All Images are trained!")

# Optional, adds a counter for images trained (You can remove it)
def counter_img(path):
    imgcounter = 1
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    for imagePath in imagePaths:
        print(str(imgcounter) + " Images Trained", end="\r")
        time.sleep(0.008)
        imgcounter += 1
```

Here we Train the images using LBPHFaceRecognizer and we store the resulted histogram values in Trainer.yml file. It stores the values of all trained faces.

Output:

```
%YAML:1.0
---
opencv_lbphfaces:
  threshold: 1.7976931348623157e+308
  radius: 1
  neighbors: 8
  grid_x: 8
  grid_y: 8
  histograms:
    - !!opencv-matrix
      rows: 1
      cols: 16384
      dt: f
      data: [ 4.49999981e-02, 4.99999989e-03, 2.49999994e-03, 0.,
        1.75000001e-02, 2.49999994e-03, 0., 9.99999978e-03, 0., 0.,
        0., 0., 2.24999990e-02, 2.49999994e-03, 7.49999983e-03,
        7.49999983e-03, 2.24999990e-02, 2.49999994e-03, 0., 0.,
        1.24999993e-02, 0., 2.49999994e-03, 4.99999989e-03,
        1.75000001e-02, 4.99999989e-03, 0., 0., 4.49999981e-02,
        2.49999994e-03, 9.99999978e-03, 2.49999985e-02,
        2.49999994e-03, 2.49999994e-03, 0., 0., 4.99999989e-03, 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 2.49999994e-03,
        0., 0., 0., 2.49999994e-03, 0., 0., 1.24999993e-02, 0., 0.,
        0., 1.40000001e-01, 4.99999989e-03, 4.25000004e-02,
        3.99999991e-02, 1.49999997e-02, 4.99999989e-03, 0., 0.,
        4.99999989e-03, 0., 0., 0., 2.49999994e-03, 0.,
        2.49999994e-03, 0., 4.99999989e-03, 0., 4.99999989e-03, 0.,
        2.49999994e-03, 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        2.49999994e-03, 0., 2.49999994e-03, 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 2.49999994e-03, 0.,
```

```

1.49999997e-02, 0., 0., 2.49999994e-03, 4.99999989e-03, 0.,
2.49999994e-03, 1.24999993e-02, 2.49999994e-03, 0., 0., 0.,
0., 0., 2.49999994e-03, 0., 9.99999978e-03, 0., 0., 0., 0.,
0., 0., 0., 2.49999994e-03, 0., 0., 0., 2.49999994e-03, 0.,
2.49999994e-03, 2.49999994e-03, 0., 4.99999989e-03, 0., 0.,
0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
7.49999983e-03, 0., 0., 2.49999994e-03, 4.99999989e-03, 0.,
4.99999989e-03, 2.49999994e-03, 4.99999989e-03, 0., 0., 0.,
9.99999978e-03, 0., 4.99999989e-03, 1.24999993e-02, 0., 0.,
0., 7.49999983e-03, 0., 0., 0., 2.49999985e-02, 0., 0., 0.,
0., 0., 0., 0., 9.99999978e-03, 2.49999994e-03,
4.99999989e-03, 0., 0., 0., 0., 0., 2.49999994e-03, 0.,
2.49999994e-03, 0., 0., 0., 2.49999994e-03, 0.,
7.49999983e-03, 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0., 1.24999993e-02, 2.49999994e-03,
3.50000001e-02, 0., 1.27499998e-01, 4.99999989e-03,
2.49999994e-03, 0., 1.99999996e-02, 0., 0., 0.,
2.49999994e-03, 0., 0., 0., 4.99999989e-03, 0.,
4.99999989e-03, 0., 2.49999994e-03, 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0., 1.75000001e-02, 0., 2.49999994e-03, 0.,
1.24999993e-02, 2.49999994e-03, 0., 2.49999994e-03,
7.49999983e-03, 0., 0., 0., 2.49999994e-03, 0., 0., 0.,
7.49999983e-03, 7.49999983e-03, 2.24999990e-02, 0.,
7.49999983e-03, 2.49999994e-03, 2.49999994e-03, 0.,
2.49999994e-03, 0., 2.49999994e-03, 2.49999994e-03,
1.49999997e-02, 4.99999989e-03, 7.49999983e-03,
7.49999983e-03, 1.42499998e-01 ]

```

Output:

```

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 3
['TrainingImage\\Karthika.4.1.jpg', 'TrainingImage\\Karthika.4.10.jpg', 'TrainingImage\\Karthika.4.11.jpg']
All Images are trained!
Enter any key to return main menu: 0
0
-----
**** Student Attendance System using Face Recognition ****
-----

***** WELCOME MENU *****
[1] Check Camera
[2] Capture Faces
[3] Train Images
[4] Recognize & Attendance
[5] Auto Mail
[6] Quit
Enter Choice: 4
Thank You

Process finished with exit code 0

```

Module 3: Recognition Phase:

By using haarcascade classifier, features will be extracted for the given image and rectangle should be drawn around face. LBPH will calculate the histogram value.

```
import datetime
import os
import time

import cv2
import pandas as pd

#-----
def recognize_attendance():
    recognizer = cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognizer()
    recognizer.read("TrainingImageLabel\Trainer.yml")
    haarcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(haarcascadePath)
    df=pd.read_csv("StudentDetails\StudentDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id','Name','Date','Time','mail']
    attendance = pd.DataFrame(columns=col_names)
```

If the student shows the face in front of camera and if the student face is present in database, the student name will appear on that image.

```
while True:
    ret, im =cam.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=faceCascade.detectMultiScale(gray, 1.2,5)
    for(x,y,w,h) in faces:
        cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)
        Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
        if(conf < 60):
            ts = time.time()
            date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
            timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
            aa=df.loc[df['Id'] == Id]['Name'].values
            tt=str(Id)+"-"+aa
            ab = df.loc[df['Id'] == Id]['mail'].values
            attendance.loc[len(attendance)] = [Id,aa,date,timeStamp,ab]
```

Output:



If the student face is not present in database, then the person is examined as unknown.

```
else:
    Id='Unknown'
    tt=str(Id)
    if(conf > 75):
        noOfFile=len(os.listdir("ImagesUnknown"))+1
        cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg", im[y:y+h,x:x+w])
        cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
    attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
    cv2.imshow('im',im)
    if (cv2.waitKey(1)==ord('q')):
        break

ts = time.time()
```


Output :



If the student is present for that class, the student's id, name ,mail ,date and time should be updated in attendance sheet.

```
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
Hour,Minute,Second=timeStamp.split(":")
fileName="Attendance\Attendance_" + date + "_" + Hour + "-" + Minute + "-" + Second + ".csv"
attendance.to_csv(fileName,index=False)
cam.release()
cv2.destroyAllWindows()
print(attendance)
```

Output :

```
[Id, Name, Date, Time, mail
3, ['Pree'], 2022-04-24, 20:11:09, ['pree@gmail.com']]
```

Module 4: Send mail to parents:

Importing libraries

```
import yagmail
import os
import datetime
import csv
```

Extracting the recently created attendance file and entering the admin mail and password.

```
date = datetime.date.today().strftime("%B %d, %Y")
path = 'Attendance'
os.chdir(path)
files = sorted(os.listdir(os.getcwd()), key=os.path.getmtime)
newest = files[-1]
filename = newest
sub = "Attendance Report for " + str(date)
```

Here attendance report will be sent to receiver's mail id

```
yag = yagmail.SMTP("preethiba192@gmail.com", "Preethi@0512")

with open('Attendance_2022-05-28_14-42-15.csv') as file:
    reader = csv.reader(file)
    next(reader)
    for Id, Name, Date, Time, mail in reader:
        yag.send(
            to=_mail[2:len(list(mail))-2],
            subject=_sub,
            attachments=filename
        )
        print(f'Sent to {Name}')
print("Email sent")
```

Output:

```
0  -----
    ***** Student Attendance System using Face Recognition *****
    -----

    ***** WELCOME MENU *****

    [1] Check Camera
    [2] Capture Faces
    [3] Train Images
    [4] Recognize & Attendance
    [5] Auto Mail
    [6] Quit
    Enter Choice: 5
    Email Sent!

    Process finished with exit code 0
```

Test cases :




Test case	Validation
Frontal profile face	 A frontal profile face of a woman with dark hair, wearing a pink top. A blue bounding box is drawn around her face, and the text ['2-Karti'] is overlaid at the bottom of the box.
Image with spectacles	 A frontal profile face of a woman wearing glasses and a white top. A blue bounding box is drawn around her face, and the text ['1-Moni'] is overlaid at the bottom of the box.
Image with scarf	 A frontal profile face of a woman wearing a dark blue scarf and a pink top. A blue bounding box is drawn around her face, and the text ['2-Karti'] is overlaid at the bottom of the box.

Image with mask



Evaluation metrics:

The most general evaluation metric is the accuracy and loss graph. Various evaluation metrics have been defined below.

- $\text{Accuracy} = (\text{TP} + \text{TN} / \text{TP} + \text{FP} + \text{TN} + \text{FN}) * 100$
- $\text{Precision} = \text{TP} / \text{TP} + \text{FP} * 100$
- $\text{Recall} = \text{TP} / \text{TP} + \text{FN} * 100$
- $\text{F1 score} = 2 / (1 / \text{Recall} + 1 / \text{Precision}) * 100$

True positive (TP): It is actual object of interest that is correctly identified.

False-positives (FP): It is a non-object of interest which is falsely identified as the true object.

False-negatives (FN): It is an actual object of interest falsely identified as negative.

True negative (TN): it is an outcome where the model correctly predicts the negative class.

Accuracy: The ratio of the total number of correctly categorized cases to the total number of occurrences is known as accuracy.

Precision: Precision is defined as the model's ability to effectively determine positive cases.

Recall: It quantifies the number of correct positive predictions made out of all positive predictions that could have been made

F1 score: When both Precision and Recall are important, the F1 score comes in handy. It's the sum of both entities' harmonics.

References:

- M. Sharif, A. Khalid, M. Raza, and S. Mohsin, "Face detection and recognition through hexagonal image processing," Sind University Research Journal, vol. 44, no. 2, pp. 541–548, 2021.C.
- Lin and K.-C. Fan, "Human face detection using geometric triangle relationship," vol. 2, pp. 941–944, in Proceedings of the 15th International Conference on Pattern Recognition. ICPR2000, vol. 2, pp. 941–944, IEEE, Barcelona, Spain, September 2020.
- Bharath Tej Chinimilli, Anjali T, Akhil Kotturi, Vihas Reddy Kaipu, Jathin Varma Mandapati, "Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm," in 4th International Conference on Trends in Electronics and Informatics, 701-704, IEEE-2020.
- Tata Sutabri, Pamungkur, Ade Kurniawan, and Raymond Erz Saragih, "Automatic Attendance System for University Student Using Face Recognition Based on Deep Learning," International Journal of Machine Learning and Computing, Vol. 9, No. 5, October 2019.
- S. Umer, B. Chandra Dhara, and B. Chanda, "Face recognition using fusion of feature learning techniques," Measurement, vol. 146, 2019