

Automated Facial Recognition based Attendance System using OpenCV in Python

Avoy Sain¹, Samrat Dutta², Rimpi Saha³, Unmesh Mandal*⁴

^{1,2,3,*4}Department of Computer Science, Bidhan Chandra College, Rishra, India
31, G. T. Road (East), Rishra, Hooghly, 712248, West Bengal, India

Corresponding Author's E-mail : unmesh.mandal@gmail.com*⁴

ARTICLE INFO

Article History:

Accepted: 05 Nov 2023

Published: 25 Nov 2023

Publication Issue

Volume 9, Issue 6

November-December-2023

Page Number

105-111

ABSTRACT

Face detection and recognition systems work by detecting faces present in an image or in a video frame and identifying the person in the image. In this work, we are interested in face detection to achieve an automatic attendance system. This work is implemented using Python and can be operated from any standalone device. This automated system stores the attendance records of students/employees with proper timestamps in the local Secondary Memory. All these records are stored date-wise. The implementation of an Automated Facial Recognition based Attendance System can help in identifying and verifying a person's identity from a digital source in real-time. Accurate attendance records are very important for classroom evaluation in schools and colleges. It also helps to keep track of the attendance of the employees in different organizations. The traditional system of manual attendance tracking in institutes can result in errors and missed or duplicated entries. The adoption of the Face Recognition-based attendance system could help eliminate these problems and shortcomings of the classical manual attendance system. The proposed work is tested with different persons with different age groups and genders in real time. The system successfully identifies the proper persons with significant accuracy and records their attendance.

Keywords : Face Detection, Face Recognition, Automated Attendance System, Open CV, Python.

I. INTRODUCTION

We have built a face recognition & verification-based attendance system using Python. We have tried to go one step further to face detection through a face recognition system. It's a system that can identify

human faces with proper authentication. There are a lot of projects and research works that have been done to achieve a proper and efficient face recognition system. The aim is to select the appropriate approach of face recognition so that we can achieve a system

with increased accuracy and speed up the entire process.

TH. Hasan et al. [9] proposed a system that can efficiently detect faces & objects in images or live video. The work uses LPB (Local Binary Pattern), EigenFaces, FisherFaces, YOLO etc & Faster R-CNN which can detect object boundaries. Naveen raj M. et al. [10] proposed a system to record attendance data using an Automatic face recognition algorithm to reduce human effort in managing records. The main goal of this project was to reduce manipulations of data & human efforts through Computer vision & database management. Balachandran et al. [11] proposed a system to identify faces using a neural network. VGGFace framework is used for this purpose. The process is done in two phases, Training and identification.

Gupta et al. [12] proposed a system to improve offline attendance system's flaws and avoid time wasted in counting traditional attendance based on image processing methods. The Student Attendance system's primary function is to perform, incorporate, and manage attendance notes, perform an automatic estimate on the amount of present and absent depending on the topic and affability of the class, and then produce an automated document or spreadsheet. They used the OpenCV library, Haar-Cascade for face detection, and LBPH for face recognition; after that, individual student training occurred, and eventually, the device produced a spreadsheet that provided the number of students present in the classroom with a picture or video captured live. Apoorva et al. [16] proposed a technique for efficient face recognition in real-time using Haar-cascade. On the OpenCV website, they tracked faces using Haar classifiers. Face detection has a good level of precision. Since the computation period is concise, the proposed method will successfully identify more than one face which is helpful for rapidly looking for suspects.

The main purpose of this project is to change our old attendance system to an automatic one. Here comes Python's face recognition module. In short, we are using this module to make an attendance system which can use the system database to make entries of the person and use theft detection techniques to filter non-verified users. Every time a person is recognized a Timestamp is stored in the database. The proposed system is completely portable because we can make an embedded system for this and databases can be remotely accessed via this device.

We are using Python's '*face_recognition*' module to achieve face recognition. This module is designed with an accuracy of 99.38%. Every time it detects a face from a frame it crops the face from the image & based on the facial structures it generates 128-d encoding which is further used to match another set of images to check whether the image matches with another or not. The deep learning module is trained to make vector representations from the facial structure, it is also called '*face embedding*'.

1. The '*face_recognition.face_locations*' detects multiple faces present on the image.
2. The '*face_recognition.face_landmarks*' detects facial structure, like eyes, nose, and lips location.

After cropping the image is processed using a deep neural network, which has three classifications-

1. Anchor: Known image containing a person's face in the database.
2. Positive image: Frames which contain a face. Usually, frames are taken from live video, but it can be a single image.
3. Negative image: Any frame which does not contain faces. Any other objects other than those that have facial properties.

After any known person is found on the frame the program creates a csv file which contains the person's name & timestamp.

II. METHODS AND MATERIAL

2.1. Python Libraries used

2.1.1 NumPy:

It is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

2.1.2 OpenCV:

OpenCV(Open-Source Computer Vision Library) [1] is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

2.1.3 Dlib:

It is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real-world problems. We are using the 'face_recognition' module to perform the face Detection. We must install the *dlib* library before installing this module.

2.1.4 HAAR cascade:

Haar Cascade classifiers are an effective way for object detection. This method was proposed by Paul Viola and Michael Jones in their paper Rapid Object Detection using a Boosted Cascade of Simple Features. Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier. Here we've used a cascade classifier to detect the facial properties of the face in the image.

2.2. Algorithms used in Dlib

2.2.1 HOG & Linear SVM:

Dlib's HOG + Linear SVM (Support vector Machine) face detector is fast and efficient. By nature of how the Histogram of Oriented Gradients (HOG) descriptor works, it is *not* invariant to changes in rotation and viewing angle.

2.2.2 MMOD CNN:

Max-Margin (MMOD) Convolutional Neural Network (CNN) is a deep learning algorithm that uses HOG and linear SVM algorithms to detect faces. It is more accurate & robust.

Table 1. Pros & Cons of used Library Modules

Module or Library	Advantages	Disadvantages
Dlib CNN	Works on any angle.	Slow processing in weak CPU. Doesn't work for real-time detection.
Dlib HOG	Ideal for frontal face detection.	Doesn't work for side faces.
Cascade classifier	Simple to implement.	Lots of false positives can be detected.

2.3. Proposed System

The proposed system works on the 'face_recognition' module. The system follows the following steps for Automated Facial Detection and Recognition for Attendance System:

2.3.1 Capture live video:

The live video is captured through a webcam. In the testing phase, the used video dimensions are 640x480 and 1920x1080. However, the system can work properly with different resolution settings. The flexibility in terms of the selection of resolutions is achieved because only facial points are being used by the recognition system. That can be done if the face is clear in the image in good lighting condition.

2.3.2 Filesystem:

Every image containing the face from an authorised user is kept in a folder. Names of the image file are set by the name of the person followed by the '.jpeg' file extension. For example, the picture of a person named Avoy is considered as 'Avoy.jpeg'. The naming convention is developed for users, so anyone can understand and use the system.

2.3.3 Face matching:

Every frame of the real-time video from the webcam is cropped and transformed according to face dimensions. The 'face_recognition' module's 'face_locations' function is used to get the locations of the face region in the image. Then according to facial properties, a 128-d encoding named 'face encoding' is achieved. It is a vector whose values reflect the facial properties. Every image present in the database goes through the above process. The program matches every face encoding calculated from the live video captured by the webcam with the present encodings in the database. When a best Match occurs, that frame is recognised.

2.3.4 Timestamp:

Once a frame is recognised the program puts the person's name & current time in a csv file. The CSV file is auto-generated according to the date. For each new day, a new CSV will be created.

2.3.5 Frame data:

The frame captured from the live video feed is resized to 0.25 times its actual size. The actual reason for this is to show a smaller output of frames on the screen. Let, any webcam's resolution be 1920x1080, so after resizing the window size in the screen will be 480x270.

After running the Python file, the program checks for a webcam to be online. If it's not, then the program tries for some time then prints an error message as request timeout & terminates the program. If the webcam is live, then the program follows the steps

mentioned in Fig. 1 and continues until an interrupt is placed. In our program, the 'q' button on the keyboard is used as an interrupter. When the 'q' button is pressed the program terminates.

2.4. Dataset Preparation

During the experiments, all the images containing facial information of the authorised persons are stored in a folder. As stated earlier, image names are referred to the format 'Xyz.jpeg', where 'Xyz' is a person's name. Fig 2. represents the snapshot of the sample dataset stored in the secondary memory. Though the sample size is small but contains real images of people and can be used for real-time verification during the experiments. The proposed system is supposed to work for larger databases with more image samples with the same accuracy.

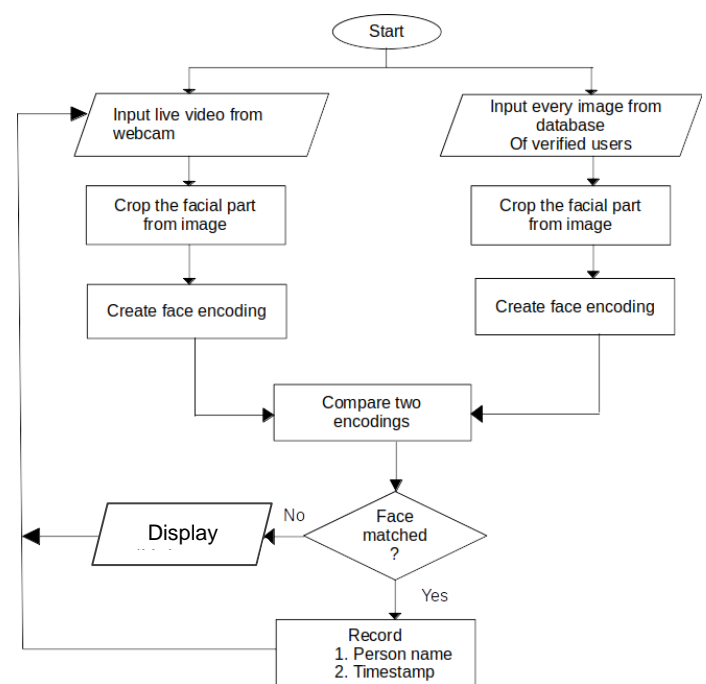


Figure 1. Flowchart of the Automated Attendance System using Facial

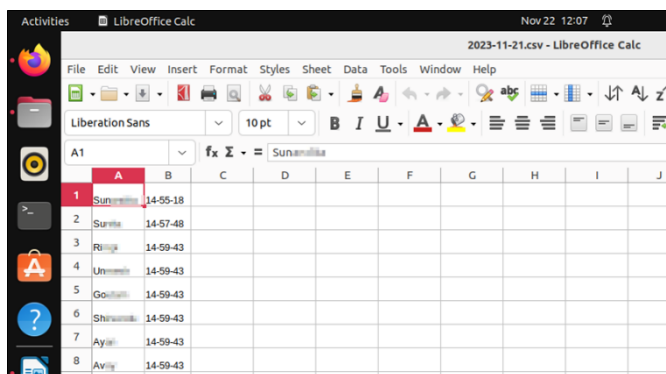


Figure 2. Snapshot of the stored images of people with different Age and Gender groups

III. RESULTS AND DISCUSSION

The Experimental results are presented in Fig. 4 by the collection of the snapshots of the outputs taken during the live testing of the system. The proposed work is tested with multiple persons in real-time from multiple age and gender groups. The system has shown similar performance for all types of samples.

During the testing, the timings of recordings of the attendances are recorded in the CSV files. The CSV files are named by the corresponding dates of the day. A snapshot of such a CSV file is depicted in Fig. 3. The name of the CSV file is the date of the corresponding day. The names of the persons with the exact timestamp of attendance are recorded.



	A	B	C	D	E	F	G	H	I	J
1	Sunanda	14-55-18								
2	Sarita	14-57-48								
3	Rishi	14-59-43								
4	Unmesh	14-59-43								
5	Govind	14-59-43								
6	Shrawan	14-59-43								
7	Ayaz	14-59-43								
8	Avin	14-59-43								

Figure 3. Snapshot of the CSV file, recording person's name and Attendance time stamps

After capturing images and verifying those images, two types of outputs are generated by the automated attendance system. One for known persons and the other for Unknown persons. During recognition using a webcam if the person's face completely matches with the image from the database, then the name of that person will be visible on the screen. If the face doesn't match with the image from the database, then our system will declare that person as an 'Unknown person'. The high accuracy rate of our proposed system makes it more reliable and trustworthy.

However, while using the system we have faced some issues worth discussing. *False Positive* is the condition when the face matches with the database, but it is an incorrect result. It can be misleading, because if a false positive result is generated wrong person can get the attendance even if he/she is not present. This may happen in poor lighting conditions. A good lighting condition solves the issue with ease.

Another issue is with the colour space. The RGB colour space does not work perfectly in low lighting Conditions, because RGB cannot detect light intensity and the colour Code changes according to light conditions. In that case, HSV(Hue Saturation Value) or LAB(Lightness, channel A, and channel B) colour spaces should work fine. These colour spaces can represent colours with light intensity, so the actual colour will be kept.

However, in this scenario, we must train the deep learning algorithm to detect faces in these colour spaces too. Because Python's 'face_recognition' module can only identify RGB colour space, that's why we must change the colour space of the webcam's frames from BGR to RGB while using the system.

IV. CONCLUSION

The proposed system is designed to provide an automated attendance system to easily record student/employee attendance. This system saves time and effort. This automated system can improve an institution's attendance mechanism systems, by reducing drawbacks and faults in the traditional manual system.

From the present work, there exists scope for improvements. Firstly, HSV or LAB colour spaces could be considered over RGB colour space, which would improve the efficiency in low-light situations. The performance of the Haar-cascade classifier used in the proposed work could be improved further or other.

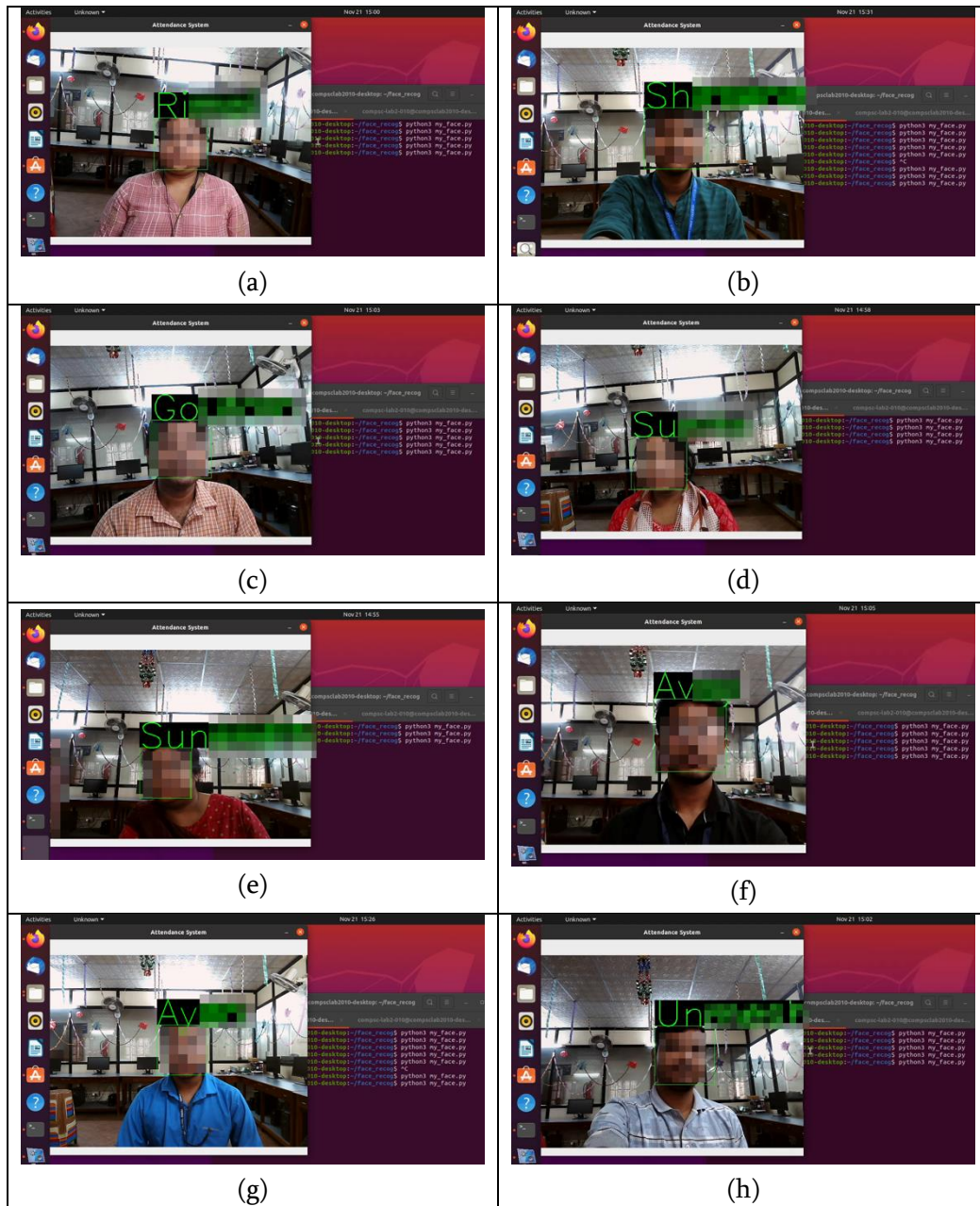


Figure 4. Realtime Facial Recognition of different persons with different Age and Gender (a) – (h)

classifiers could be considered to improve the accuracy of the proposed system. The work is implemented for the detection of a single face in the frame considering one person will be in front of the camera at a time for attendance. The work can be further extended to provide support for the simultaneous detection of multiple faces in a frame. This may allow us to record the attendance of all the students in a class or all the employees in a meeting etc. in a single attempt.

V. REFERENCES

- [1]. OpenCV Homepage, <https://opencv.org/> , Accessed: 17-08-2023
- [2]. Face Recognition Homepage, <http://www.face-rec.org/algorithms>, Accessed: 17-08-2023
- [3]. Face recognition and Face detection using the OpenCV, <https://www.javatpoint.com/face-recognition-and-face-detection-using-opencv>, JavaTPoint, Accessed: 28-08-2023

- [4]. Zhao, Wen-Yi & Chellappa, Rama & Phillips, P. Jonathon & Rosenfeld, Azriel. (2003). Face Recognition: A Literature Survey. *ACM Comput. Surv.* 35. 399-458. 10.1145/954339.954342.
- [5]. Kumar Datta, Asit; Datta, Madhura; Kumar Banerjee, Pradipta (2015). Face Detection and Recognition.
- [6]. Wikipedia, Three-dimensional face recognition, https://en.wikipedia.org/wiki/Three-dimensional_face_recognition, Accessed: 15-10-2023
- [7]. Jagtap, A. M., Kangale, V., Unune, K., & Gosavi, P. (2019, February). A Study of LBPH, Eigenface, Fisherface and Haar-like features for Face recognition using OpenCV. In 2019 International Conference on Intelligent Sustainable Systems (ICISS) (pp. 219-224). IEEE
- [8]. Liao, S., Jain, A.K., Li, S. Z. (2016). A fast and accurate unconstrained face detector. *IEEE Transaction of Pattern Analysis and Machine Intelligence*, Vol 38, No 2, pp. 211 – 123.
- [9]. TH. Hasan, R., & Bibo Sallow, A. . (2021). Face Detection and Recognition Using OpenCV, <https://publisher.uthm.edu.my/ojs/index.php/jscdm/article/download/8791/4561>
- [10]. Naveen raj M., & R.Vadivel. (2023). Face recognition based attendance system with location marking. <https://wjarr.com/content/face-recognition-based-attendance-system-using-machine-learning-location-identification>
- [11]. Balachandran, B., Saad, K. F., Patel, K., & Mekhiel, N. (2019, December). Parallel Computer for Face Recognition Using Artificial Intelligence. In 2019 14th International Conference on Computer Engineering and Systems (ICCES) (pp. 158-162). IEEE
- [12]. Gupta, S. (2018, January). Facial emotion recognition in real-time and static images. In 2018 2nd international conference on inventive systems and control (ICISC) (pp. 553-560). IEEE.
- [13]. Sharma, S., & Jain, S. (2019, March). A static hand gesture and face recognition system for blind people. In 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 534-539). IEEE.
- [14]. Srivastava, M., Kumar, A., Dixit, A., & Kumar, A. (2020, February). Real time attendance system using face recognition technique. In 2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC) (pp. 370-373). IEEE.
- [15]. Alcantara, G. K. L., Evangelista, I. D. J., Malinao, J. V. B., Ong, O. B., Rivera, R. S. D., & Ambata, E. L. U. (2018). Head detection and tracking using OpenCV. In 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM) (pp. 1-5). IEEE
- [16]. Apoorva, P., Impana, H. C., Siri, S. L., Varshitha, M. R., & Ramesh, B. (2019, March). Automated criminal identification by face recognition using open computer vision classifiers. In 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC) (pp. 775-778). IEEE

Cite this article as :

Avoy Sain, Samrat Dutta, Rimpi Saha, Unmesh Mandal, "Automated Facial Recognition based Attendance System using OpenCV in Python", *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, ISSN : 2456-3307, Volume 9, Issue 6, pp.105-111, November-December-2023. Available at doi : <https://doi.org/10.32628/CSEIT2390617>
Journal URL : <https://ijsrcseit.com/CSEIT2390617>