International Journal of Research in Engineering, Science and Management
Volume 7, Issue 5, May 2024
https://www.ijresm.com | ISSN (Online): 2581-5792


# Face Recognition Attendance System

Anuj Golasangi[1], Manjunath Choudri[2*], Pragati Bulla[3], Vinutana Devaraddi[4], P. K. Deshpande[5]
[1,2,3,4]Student, Department of Information Science and Engineering, Basaveshwar Engineering College, Bagalkote, India
[5]Professor, Department of Information Science and Engineering, Basaveshwar Engineering College, Bagalkote, India

***Abstract*: Face recognition attendance systems have gained significant attention due to their ability to automate attendance tracking while ensuring accuracy and security. This abstract presents a face recognition attendance system developed using Python Flask Framework and various libraries including OpenCV and face_recognition. The system consists of modules for student and faculty authentication, registration, attendance marking, viewing, exporting, and statistical analysis. For students, the system provides a secure login interface and allows registration with unique identifiers. Students can view their attendance records and logout securely. Faculty members have access to a dashboard where they can mark attendance by uploading images, view attendance records, and export them for further analysis. The system also provides statistical insights into attendance patterns.**

***Keywords*: face recognition, face detection, attendance, OpenCV, image processing.**

## 1. Introduction

In today's technologically advanced world, there is widespread usage of digital devices, including smartphones and other computing devices. These devices have become an integral part of our daily lives, offering various functionalities such as communication, organization, and entertainment. The system should be capable of accurately identifying individuals and recording their attendance based on predefined criteria. It should display the attendance status in real-time, providing instant feedback to users.

Additionally, the system should have the ability to generate detailed attendance reports, including timestamps and attendance records of individuals. It should also offer features such as notification alerts for late arrivals or absences.

This system aims to streamline the attendance management process for various organizations, eliminating the need for manual attendance tracking methods. The attendance data generated by this system will serve as a reliable and efficient means of monitoring attendance patterns and enhancing organizational efficiency. This system represents a paradigm shift in the way attendance is managed, offering a seamless and secure alternative to conventional methods. The Face Recognition Attendance System leverages the capabilities of advanced computer vision and machine learning technologies to automate the process of attendance tracking. Unlike traditional methods that rely on manual data entry, cards, or biometric scans, FRAS identifies and records attendance through the analysis of facial features unique to each individual. This contactless and non- intrusive approach not only enhances security but also provides a more convenient and user-friendly experience.

## 2. Related Work

In [1], NFC (Near Field Communication) Technology with a camera incorporated in a mobile device," according to a study publication, NFC technology and a mobile application are used to improve the attendance system. At the time of enrolment in the faculty, each student is issued an NFC tag with a unique ID, according to the research article. The travelling instructor will then take attendance at each lesson by touching or distributing these tags. The integrated camera will then take a picture of the student's face before sending all of the data to the college server for verification. The benefits of this technology include the ease of use of NFC and the fast connection speed. It greatly facilitates the process of being in the present moment. However, if the NFC tag was not tagged by the user, the system would not be able to identify infringement automatically. Aside from that, the usage of a mobile app was necessary since the NFC student was interrupting the teacher. Would it be a support system to record everyone present if a pastor failed to bring his mobile phone to work? Furthermore, because of a confidential topic, most professors would not want their iPhones to be used in this manner. As a result, instead of the NFC marker, unique student information such as biometrics or face recognition, guanine to the student should be employed. This ensures that a specific student will be the first to take attendance. Whereas [2] involves taking images of the employee using a camera in order to capture their faces and visions. When the result is located on the face website, the taken image is compared individually with the face mask to display the employee's face, where presence is noted. The key benefit of this method is that the presence is recorded on a highly secure server that no one else can access. Furthermore, the face detection algorithm in this suggested system is built employing a skin-splitting approach to improve the accuracy of the detection process. Despite ongoing efforts to improve the accuracy of the face detection algorithm, the system remains unaffected at this time.

This system [3] has been implemented with the help of three basic steps: A. detect and extract face image and save the face information in an xml file for future references. B. Learn and train the face image and calculate eigen value and eigen vector

*Corresponding author: manjunathchoudri297@gmail.com

of that image. C. Recognise and match face images with existing face images information stored in xml file. At first, openCAM_CB() is called to open the camera for image capture. Next the frontal face [2] is extracted from the video frame by calling the function ExtractFace(). The ExtractFace() function uses the OpenCv HaarCascade method to load the haarcascade_ frontalface_alt_tree.xml as the classifier. The classifier outputs a "1" if the region is likely to show the object (i.e., face), and "0" otherwise. The classifier is designed [4] such a manner that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure is done several times at different scales. After the face is detected it is clipped into a gray scale image of 50x50 pixels. An image in which each pixel contains the average value for that pixel across all face images in the training set. The dataset is centred by subtracting the average face's pixel values from each training image. It happens inside cvCalcEigenObjects(),.Recognize() function, which implements the recognition phase of the Eigenface program [5]. It has just three steps. Two of them loading the face images and projecting them onto the subspace are already familiar. The call to loadFaceImgArray() loads the face images, listed in the train.txt, into the faceImgArr and stores the ground truth for person ID number in personNumTruthMat. Here, the number of face images is stored in the local variable, n TestFaces.

Another system [7] where it involves the steps. The first step of the face recognition process is face detection. Face detection presents the well-studied field in the computer vision domain. As a result of decades of research, nowadays there are numerous machine learning algorithms applicable for this task. In recent years, CNNs achieved advanced results in image classification and object detection. Due to its runtime performance, for this step, a state-of-the art CNN cascade is used for a face detection task, introduced by Haoxiang Li et al in [8]. The cascade consists of 6 CNNs, 3 CNNs for binary classification (face and non-face) and 3 CNNs for bounding box calibration. The final step of developing the face recognition model for tracking employees' attendance consists of training the classifier based on the previously generated embedding from employees' dataset by the deep CNN. Due to the fact that this system is based on smaller dataset, linear Support Vector Machine (SVM) was applied for this classification task.

## 3. Proposed Work

The proposed system for face recognition attendance marks a departure from traditional physical attendance tracking methods by leveraging the power of image uploading and facial recognition technology. In this system, rather than relying on dedicated cameras at entry points, individuals can upload their images through a designated platform or application. These images are then processed using sophisticated facial recognition algorithms that analyze facial features to identify individuals accurately. Once the system successfully recognizes a person's face, it automatically marks their attendance, eliminating the need for manual entry or physical presence at a specific location.

### *A. System Design*

#### *1) Architecture*

Face Recognition Module: This is the core component responsible for detecting and recognizing faces in images or video streams. It employs algorithms such as deep learning-based convolutional neural networks (CNNs) to extract facial features and match them against known identities.

Database: The database stores information about registered users, including their biometric data (such as facial templates) and associated metadata (e.g., user ID, name, role). It also maintains attendance records, storing timestamps and user identifiers for each attendance event.

User Interface: The user interface provides an interface for users, administrators, faculty, and other stakeholders to interact with the system.

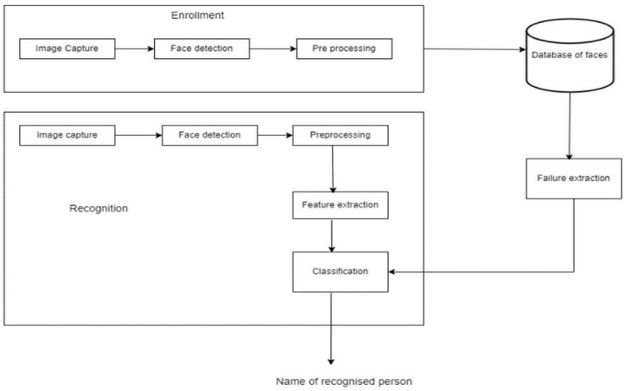


Fig. 1. Architecture

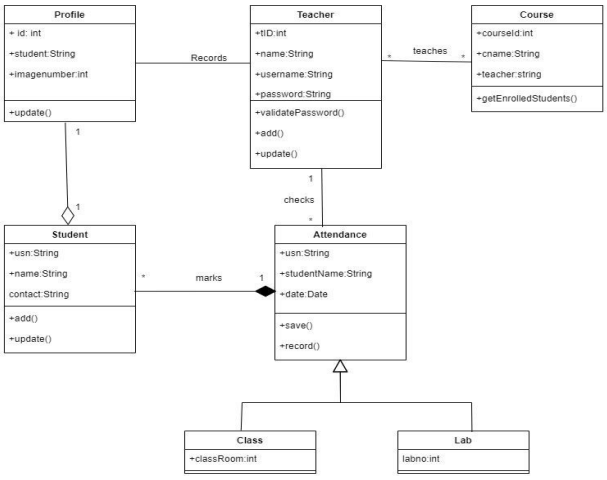


Fig. 2. Class diagram

- User: Initiates the attendance process by presenting their face to the system.
- Admin: Manages system settings, user accounts, and access privileges.
- Student: Represents individuals whose attendance is being recorded.
- Faculty: Responsible for overseeing attendance and accessing attendance records.
- Database: Stores information about registered users, attendance records, and system configurations.

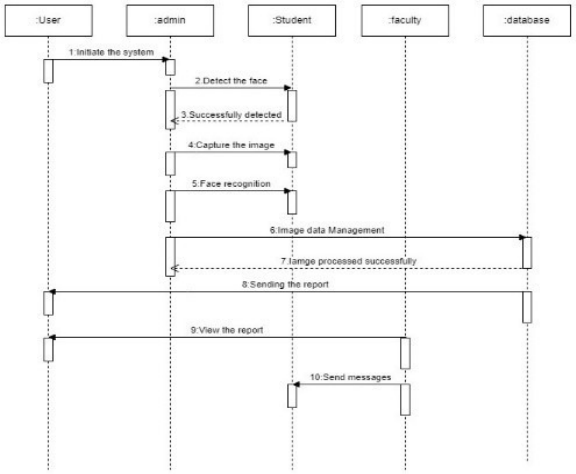


Fig. 3. Sequence diagram

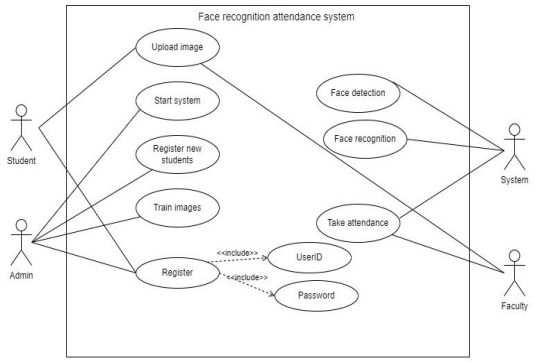


Fig. 4. Use case diagram

*Use cases:*

Upload Image: Users uploads the images of individuals which is stored in the local drive of the system. This step is crucial for enrolling new students or updating existing records.

Start System: The system is initiated, either manually by an administrator or automatically at scheduled intervals, to begin processing attendance tasks.

Register New Students: Administrators or authorized users input relevant information for new students, such as name, student ID, and any other required details, into the system. The captured images are associated with these records for future recognition.

Train Images: The system processes the captured images to extract facial features and trains its recognition algorithms. This step is essential for improving accuracy in identifying individuals during the recognition process. Face Recognition: When individuals arrive, the system compares their faces with the enrolled images in its database to find matches. If a match is found, the system retrieves the associated student information.

## B. Methodology

Flask handles user requests based on URLs. Separate routes are defined for student and faculty login, registration, dashboards, and functionalities like viewing attendance or uploading attendance images. HTML templates are used to render user interfaces for forms, data displays, and informative messages.

A connection is established to the MySQL database using the MySQL Connector/Python library.Separate tables store student information, faculty information, and attendance records.The application interacts with the database using queries.

The faculty member initiates the attendance marking process by uploading an image containing students. This image could be captured by a camera or uploaded from a file.

The system uses the face_recognition library to detect faces within the uploaded image. This detection is performed using the face_recognition.face_locations() function, which identifies the bounding boxes around faces in the image.

Once the faces are detected, the system extracts facial features, or "face encodings", from each detected face using the face_recognition.face_encodings() function. These face encodings represent unique numerical representations of facial features.

The system has pre-trained data consisting of face encodings and corresponding student identifiers (e.g., USNs).

Using this pre-trained data, the system employs a machine learning classifier, typically a Support Vector Machine (SVM), trained with student face encodings and their corresponding identifiers.

The classifier predicts the identities (USNs) of the students present in the uploaded image based on the extracted face encodings. This prediction is made using the predict() function of the classifier.

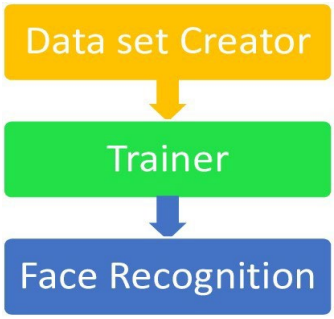


Fig. 5. Face recognition

After predicting the identities of students in the image, the system potentially looks up each predicted USN in the database to verify the student's enrolment.

If a match is found between the predicted USN and a student's USN in the database, the system marks the student's attendance as "Present" for the corresponding class.

The attendance marking process involves updating the attendance records in the database, typically by adding new entries indicating the student's presence in the class.The system may also record additional information such as the date and time of the attendance marking, the faculty member responsible, and the subject of the class.

## C. Algorithm for Face Recognition

Input: The uploaded image file.

Output: Recognition of faces in the uploaded image.

1. Check if an image file is present in the HTTP request.
2. If an image file is present: Retrieve the image.
3. Check if the filename of the image is not empty.

4. Save the uploaded image temporarily to a specified directory.
5. Create a temporary directory named 'temp' if it does not exist already.
6. Save the uploaded image to the temporary directory.
7. Define the directory containing training images.
8. Loop through each person's directory in the training directory: List all images of the person.
9. For each image of the person:
   - Get the image path
   - Load the image file using face_recognition library
   - Detect face bounding boxes in the image.
   - If exactly one face is detected:
   - Encode the face
   - Append the encoding and the person's name to lists
10. Create and train the Support Vector Classifier (SVC) using the encodings and corresponding name.
11. Load the test image file.
12. Detect face locations in the test image.
13. Encode faces found in the test image.
14. If no faces are found in the test image:
    - Flash an error message indicating no face detected in the uploaded image.

*D. Algorithm for storing the recognised faces into the database*

1. Check if the HTTP request method is POST.
2. Retrieve form data:
   Retrieve selected semester from the form (request.form['semester']).
   Retrieve selected branch from the form (request.form['branch']).
   Retrieve selected subject from the form (request.form['subject']).
3. Establish a connection to the database.
4. Create a cursor object to execute SQL queries.
5. Iterate over each predicted USN in predicted_identities: Query the database for student record based on the predicted USN (SELECT * FROM students WHERE usn = %s). Fetch the student record.
6. If a student record is found:
   Construct an SQL query to add attendance record.
   Execute the query to insert attendance record into the database (cursor.execute()).
   Commit the changes to the database (connection.commit()).
7. Flash a success message indicating successful attendance upload (flash("Attendance successfully uploaded.", 'success')).
8. Redirect to the attendance upload page (return redirect(url_for('upload_attendance'))).
9. If an error occurs during the process:
   Flash an error message indicating the error (flash(f"An error occurred during face recognition: {e}", 'error')).
   Redirect to the attendance upload page (return redirect(url_for('upload_attendance'))).
10. Finally, close the database connection (connection.close()).

*E. Results and Discussions*

The code aims to handle the attendance upload process based on the predicted identities of students recognized from uploaded images.

It retrieves form data such as semester, branch, and subject from the HTTP request.

For each predicted identity (USN) obtained from the face recognition process, it queries the database to fetch the student record.

If a student record is found, it adds an attendance record for that student with details such as datetime, faculty id, subject, semester, and status ('Present') to the 'attendance' table in the database.

After successfully adding attendance records for all recognized students, it flashes a success message indicating the successful attendance upload. The code appears to handle the attendance upload process for a single subject and semester. If the application needs to scale to handle multiple subjects, semesters, or even multiple faculties, additional functionalities and database structures may need to be implemented to support this scalability. Flash messages are used to provide feedback to users regarding the outcome of the attendance upload process. This helps in improving the user experience by providing clear feedback on the success or failure of the operation.

Overall, this project provides a foundation for automating the attendance upload process based on face recognition and database integration. Continuous monitoring, testing, and potential enhancements can further improve its robustness and effectiveness in real-world usage.

## 4. Conclusion

This paper presented an overview on face recognition attendance system.

## References

[1] A brief history of Facial Recognition, NEC, New Zealand, 26 May 2020. [Online]. Available: https://www.nec.co.nz/market-leadership/publications-media/a-brief-history-of-facialrecognition/
[2] Face detection, TechTarget Network, Corinne Bernstein, Feb, 2020. [Online]. Available: https://searchenterpriseai.techtarget.com/definition/face-detection
[3] Face Detection with Haar Cascade, Towards Data Science, Dec. 24, 2020. [Online]. Available: https://towardsdatascience.com/face-detectionwith-haar-cascade-727f68dafd0
[4] SenthamilSelvi, Chitrakala, Antony Jenitha, "Face Recognition System Based on Face Recognition," 2014.
[5] Kawaguchi, "Lecture attendance system with continue monitoring," 2011.
[6] Smitha, Pavithra S. Hegde, Afshin, "Automatic attendance management system," 2018.
[7] N. Kar, "Automated attendance management system using face recognition," 2002.
[8] Dhanush Gowda, K Vishal, Keertiraj B, Neha Kumari Dubey, Pooja M. R, 2020, "Automatic attendance using face recognition by MATLAB."
[9] Jyotshana Kanti, "Smart attendance marking system," 2012.
[10] Priyanka Thakare, "Face detection using Eigenface," 2015.
[11] Sharma S, Karthikeyan Shanmugasundaram, Sathees Kumar Ramasamy, "CNN Based Efficient Face Recognition Technique using Dlib," 2016.
[12] Arun Katara et al, "Student attendance using iris recognition system," 2017.