© SEP 2025 | IRE Journals | Volume 9 Issue 3 | ISSN: 2456-8880

Towards Seamless Attendance Monitoring: A ML-Driven Real-Time Facial Recognition System for Automated Class Attendance

UGBANA C. KELVIN[1], TAMUNO-OMIE J. ALALIBO[2], NKOLIKA O. NWAZOR[3]
[1] Centre for Information and Telecommunication Engineering, University of Port Harcourt, Nigeria
[2] Computer Engineering Department, Rivers State University, Port Harcourt, Nigeria
[3] Electrical/ Electronic Engineering Department, University of Port Harcourt, Nigeria

Abstract- Conventional attendance recording methods, are susceptible to delays, impersonation, and hygiene concerns. This paper presents a real-time attendance monitoring system tailored for educational environments where accuracy, and record integrity are essential. The proposed system integrated a CNN-SVM-based facial recognition model with a pan/tilt servo-controlled camera to enable dynamic coverage and continuous subject tracking. The hardware setup comprises a Raspberry Pi 5 as the main processor, an IR night-vision camera for reliable capture in varying light, an Arduino Nano for servo actuation, and a dual-axis servo platform for camera mobility. On the software side, Python, Dlib, OpenCV, and OpenFace support facial detection and embedding. The model, trained on multiple samples per student, was optimized for offline execution to ensure privacy, speed, and secure local storage. The system achieved a recognition accuracy of 96.8% with an average inference time of 0.79 milliseconds per instance. Field validation in real classroom environments confirmed consistent performance under motion and low-light scenarios, with automated attendance sheets generated by associating recognized identities with corresponding timestamps. Leveraging deep learning and embedded processing, the system provides a portable, hygienic, and efficient alternative to traditional attendance methods, enhancing classroom management, data integrity, and scalability.

Index Terms- Automated attendance, facial recognition, real-time tracking, classroom monitoring, Raspberry Pi.

I. INTRODUCTION

Attendance management is a vital administrative function in both educational and organizational settings. Yet, many institutions still depend on manual methods such as roll calls and signature registers, which are often time-consuming, prone to errors, and susceptible to manipulation through practices like proxy attendance [1]. In large lecture halls, such as those at the University of Port Harcourt where student numbers frequently exceed 100 per sitting, maintaining accurate records becomes a challenge, resulting in inefficiencies that compromise performance tracking and discipline.

To overcome these limitations, institutions have adopted technologies such as RFID cards, fingerprint scanners, and electronic sign-in systems [2]. While these approaches improve data collection, they are not without drawbacks, including high implementation costs, limited scalability, and hygiene concerns [3]. Fingerprint systems are vulnerable to degraded prints, RFID cards may be misplaced or misused, and iris scanners remain prohibitively expensive. These shortcomings reinforce the need for a non-contact, cost-effective, and adaptable alternative.

Facial recognition, powered by Convolutional Neural Networks (CNNs), has gained attention as a viable solution owing to its accuracy, efficiency, and contactless operation [4]. Unlike fixed-camera setups, which are costly and restricted to specific locations, mobile CNN-enabled systems with pan-tilt mechanisms provide dynamic classroom coverage. This reduces roll call duration, lowers infrastructure requirements, and enhances accessibility [5]. Such innovations are particularly relevant in Sub-Saharan

Africa, where financial and infrastructural constraints hinder the widespread adoption of conventional biometric technologies.

This study presents the development of a portable, offline-capable CNN-SVM-based facial recognition attendance system tailored for resource-limited environments. By combining real-time detection, on-device processing, and mobility, the system addresses the weaknesses of manual and contact-dependent methods, offering a scalable and low-cost solution for contemporary attendance management.

II. LITERATURE REVIEW

In the domain of computer vision, both Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) have been applied to facial recognition, though they differ in how they handle features and classification. SVMs function purely as margin-based classifiers and do not perform feature extraction themselves. Instead, they depend on handcrafted or pre-computed descriptors such as Histogram of Oriented Gradients (HOG) or Local Binary Patterns (LBP), which makes them effective for smaller datasets and relatively straightforward to interpret [6][7]. By contrast, CNNs integrate feature extraction and classification in a single architecture. They learn hierarchical feature representations directly from raw images; progressing from low-level edges and textures in the early layers to abstract facial patterns in the deeper layers [8][9]. Through convolutional filters, pooling operations, and nonlinear activations, CNNs automatically extract meaningful features from raw pixels, eliminating the need for pre-computed descriptors. Although CNNs typically require larger datasets and higher computational resources, they have consistently achieved state-of-the-art accuracy in facial recognition tasks [10][11]. Owing to their complementary strengths, CNNs and SVMs can be effectively combined within facial recognition pipelines [6][12].

A. HOG–SVM Based Face Detection in Dlib

In the study, Dlib was employed for implementation of the facial recognition attendance system. Dlib comes with a face detector based on on the combination of HOG and SVM. This approach applies five orientation-specific HOG filters corresponding to frontal, left, right, forward-left, and forward-right face poses. The method is optimized for CPU execution, making it computationally efficient while delivering reliable detection for frontal and slightly angled faces, even under minor occlusion The HOG–SVM detector in Dlib was trained with a minimum detectable face size of 80×80 pixels, which limits its ability to recognize smaller faces below this threshold. Despite the constraints, the HOG–SVM model remains a practical choice for real-world face detection tasks where speed and moderate pose tolerance are required [11][13].

B. CNN-SVM Pipeline for Face Recognition

For facial recognition, this study adopted a CNN–SVM hybrid model. In this approach, the CNN serves as a feature extractor, generating embeddings that represent compact numerical signatures of face images. These embeddings are then classified using SVM, leveraging its strong generalization ability to achieve robust recognition performance.

In a typical hybrid CNN–SVM pipeline for facial recognition, the process begins with an input image $I \in \mathbb{R}^{(H \times W \times C)}$, where H, W, and C denote the height, width, and number of channels respectively (e.g., $I \in \mathbb{R}^{(128 \times 128 \times 3)}$ for a color face image). The CNN functions as the feature extractor, progressively transforming the raw input image into a compact and discriminative feature representation. Within the convolutional layers, the feature extraction process is defined in Equation 1, Non-linear activations, ReLU, are then applied as in Equation 2, followed by pooling operation (see Equation 3) to reduce spatial dimensions while preserving discriminative features. After several convolutional and pooling stages, the high-level feature maps are passed through a fully connected layer, mathematically defined by Equation 4. The output of this stage is a dense feature vector, commonly referred to as the embedding or face signature (see Equation 5), typically of dimension $d \in \{128, 512\}$. The embedding f serves as a compact descriptor uniquely representing the input face. Instead of employing the CNN's native softmax classifier, an SVM is trained on these embeddings to perform identity classification. The feature vector f is supplied as input to the SVM, which learns decision boundaries between different identities. In the linear case, the SVM decision function is expressed in Equation 6, where w and b represent the weight and bias respectively. For more complex boundaries, a non-linear kernel-based SVM are employed, as defined in Equation 7. The final identity prediction is made by selecting the class associated with the maximum decision score, as expressed by Equation 8. Thus, the overall hybrid CNN–SVM pipeline can be summarized compactly as a two-stage process: deep feature extraction via CNN, followed by robust classification using SVM, as shown below:

$$I \implies CNN \Rightarrow f \implies SVM \implies \hat{y}\,.$$

$$S^l = I * W^l + b^l \quad (1)$$

$$A^l = \max(0, S^l) \quad (2)$$

$$P^l = \text{pool}(A^l) \quad (3)$$

$$z = W_{fc} \cdot P + b_{fc} \quad (4)$$

$$f = g(z) \;\; ; \;\; f \in \mathbb{R}^d \quad (5)$$

$$y = w \cdot f + b \quad (6)$$

$$y = \textstyle\sum_{i=1}^{N} \propto_i y_i K(f, f_i) + b \quad (7)$$

$$\hat{y} = argmax\ (y) \quad (8)$$

where W^l are learned filters; b^l is the bias; S^l represents the resulting feature maps; f_i are support vector features, y_i are their corresponding labels, $\propto_i$ are the learned coefficients, and $K(f, f_i)$ is the kernel function.

This hybrid approach exploits the representational capacity of CNNs in learning features, while leveraging the robust separation ability of SVMs, often resulting in improved recognition accuracy, particularly in scenarios with limited training data [7][12].

C. OpenFace: A GoogleNet-Based Approach

Open Face employs CNNs as the backbone of its facial recognition pipeline, enabling automatic extraction of discriminative features from visual input. Its architecture is derived from FaceNet [10] and incorporates a modified GoogleNet (Inception) network, which uses multiscale convolutional kernels (1×1, 3×3, 5×5) to capture both fine-grained local details and broader global facial images [14][15]. Unlike conventional classifiers that output discrete labels, OpenFace generates 128-dimensional embeddings trained using triplet loss, which minimizes intra-class distance while maximizing inter-class separation [10]. This embedding-based representation supports identity verification through vector similarity measures rather than traditional classification.

In this study, OpenFace is adopted as the core recognition engine. It processes live video input by detecting facial landmarks, extracting embeddings, and comparing them against stored datasets for identity confirmation. Its open-source implementation, computational efficiency, and robustness in unconstrained environments make it a practical and scalable solution for real-world facial recognition tasks [16].

III. MATERIALS AND METHODS

The proposed facial recognition attendance system is developed as a modular embedded platform that integrates computer vision, deep learning, and hardware automation for autonomous operation. The hardware is designed with a modular architecture to ensure flexibility, scalability, and ease of maintenance, while the software framework is built around Python and the OpenFace library for facial recognition. The subsystem incorporates three major components:

i Software intelligence (face detection, encoding, and classification).
ii Data handling (real-time logging and attendance record storage).
iii Mechanical scanning (servo-controlled camera panning and tilting).

The system's primary function is to automate attendance recording by detecting and recognizing student faces as the camera systematically scans the classroom, thereby minimizing manual intervention. A prototyping methodology is employed, allowing iterative development through early model creation, functionality testing, feedback collection, and subsequent refinement of the system. Figure 1 presents the dataset of student images used for training and validating the facial recognition model, while Figure 2 shows the hardware and software block diagram of the proposed system.

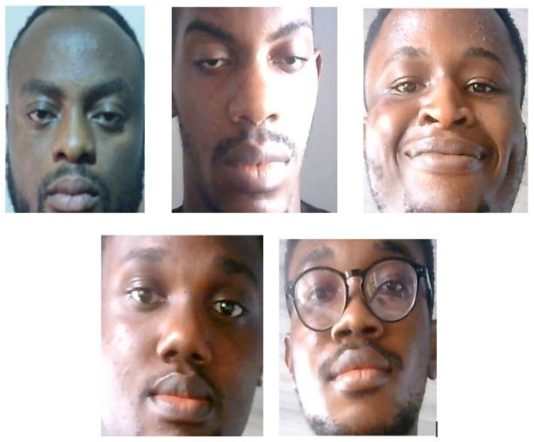


Figure 1: Face picture collected from the class.

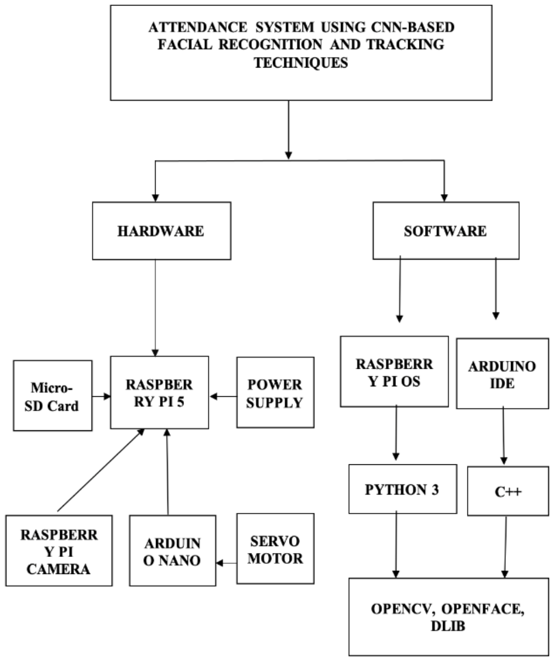


Figure 2: System Design Block Diagram

A. Software Intelligence
The software intelligence of the facial recognition system operates through the following steps:

i Facial Detection and Face Positioning
Face detection constitutes the foundational step of the system. At this stage, the detection algorithm analyzes the entire image to determine the presence and location of a face. Once a face is identified, the process advances to the localization of facial landmarks, which serve as the structural basis for alignment and subsequent recognition.

For accurate landmark localization, this study adopts the Face Landmark Estimation algorithm introduced in [17]. The method employs an ensemble of regression trees to detect a predefined set of 68 facial landmarks on each identified face, as illustrated in Figure 3. These landmarks include critical points such as the chin, inner and outer eye corners, eyebrow contours, nose ridge and base, and lip boundaries [16][17]. Following landmark detection, geometric normalization is performed to align faces into a uniform orientation. This is achieved using affine transformations, including rotation, scaling, and shearing, to ensure that essential regions such as the eyes and mouth are mapped to consistent positions. By standardizing facial orientation, the system minimizes the effects of pose variations and head tilts, thereby enhancing the robustness and reliability of subsequent feature extraction [11].

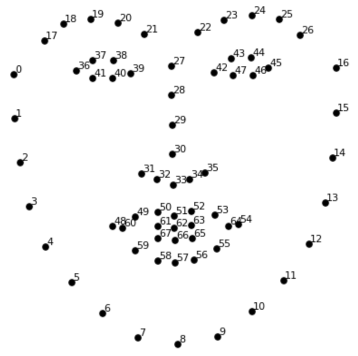


Figure 3: The 68 Landmarks located on every face.

ii Face Encoding
Once a face has been detected and geometrically aligned, the next stage in the recognition pipeline is face encoding, where distinctive features of each individual are extracted. The objective is to represent every face as a fixed-length numerical vector, commonly referred to as an embedding. These embeddings capture the essential discriminative attributes of the face, allowing mathematical comparison of identities even under variations in pose, orientation, or illumination. In principle, one might attempt to define features manually—for

example, by measuring the distance between the eyes, the length of the nose, or the shape of the jawline. However, this study employs a Deep Convolutional Neural Network (DCNN), which automatically learns the most relevant and discriminative features directly from data. The encoding model maps each face into a 128-dimensional feature vector:

$f = [f_1, f_2, \dots, f_{128}]$, where each component f_i represents a learned facial feature. The network is trained using a triplet loss function, which processed three images three images simultaneously:

i. Anchor: an image of a known face for training.
ii. Positive: another image of the same individual.
iii. Negative: an image of a different individual.

The goal is to minimize the distance between the anchor and positive embeddings while maximizing the distance between anchor and negative embeddings. This ensures that embeddings from the same identity cluster closely together, while embeddings from different identities remain well separated

iii. Face Matching

The final stage of the recognition pipeline is face matching, where the system determines whether a test face corresponds to a known individual. Once an encoding is generated for the test image, it is compared against stored embeddings of enrolled individuals.

In this paper, a Linear SVM classifier is employed for identity determination. Given a test embedding f, the classifier evaluates the decision function using Equation 6. The predicted identity corresponds to the class associated with the highest score (see Equations 7 and 8). The SVM implementation in the Python face recognition library is computationally efficient, capable of classifying embeddings within milliseconds. This real-time performance makes it particularly well suited for deployment in attendance management systems, where both accuracy and speed are critical.

B. Database for Student Faces and Attendance Records

The system maintains a lightweight yet efficient database that supports both student image storage and attendance record management:

i. Dlib Enrolment Process for Image Storage

Enrolment represents a critical phase in the operation of the facial recognition–based attendance system. The process, shown in Figure 4, ensures that each student's face is accurately captured and securely stored for future recognition. During enrolment, facial images are acquired using Python in conjunction with the Dlib-based face recognition library. Only valid and encodable faces are admitted into the database, thereby minimizing errors in later recognition stages. Each successfully encoded face is stored as a numerical template, serving as a reference signature for subsequent comparisons. To maintain structure, a dedicated folder organizes labeled headshot images captured during enrolment, with each label corresponding to the student's unique identifier. The diversity and quality of these initial samples are fundamental, as they directly influence the reliability and accuracy of the recognition process during attendance marking.

ii. Attendance Marking and Record

During system operation, whenever a face is detected and encoded, the resulting embedding is compared against the stored database of enrolled student encodings. Upon a successful match, the system retrieves the associated student's identification number or name. Using Python's datetime module, the precise timestamp of recognition is recorded. This attendance event, comprising the student's name (or ID) and timestamp. is automatically appended to a structured CSV file using Python's csv module. The CSV file serves as the primary repository for attendance data and offers several advantages:

i Real-time marking of attendance with minimal manual intervention.
ii Ease of export and compatibility, as CSV log files can be directly integrated into spreadsheet applications such as Microsoft Excel for analysis, and record management.

This streamlined yet effective storage format ensures both reliability and interoperability, making it well suited for deployment in academic administrative environments.

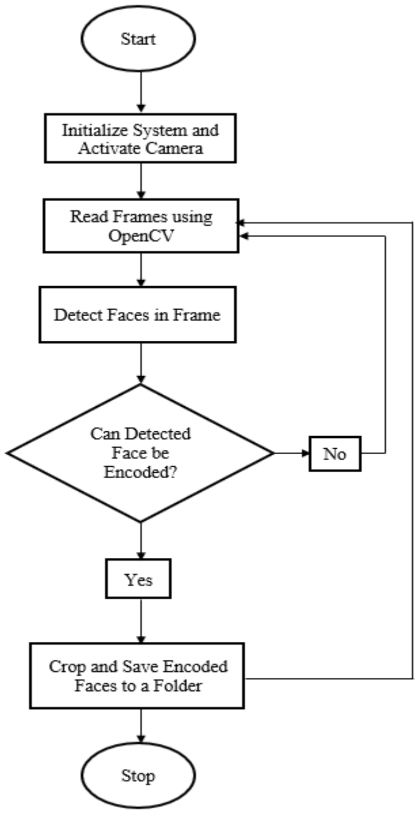

Start
Initialize System and Activate Camera
Read Frames using OpenCV
Detect Faces in Frame
Can Detected Face be Encoded?
No
Yes
Crop and Save Encoded Faces to a Folder
Stop

Figure 4: Flow Chart for User Enrolment into the facial recognition system

C. Hardware Design

The implementation of the attendance system integrates both computational and electromechanical components to achieve efficient real-time operation. Figure 5 illustrates the hardware operational flowchart, while Figure 6 presents the corresponding circuit diagram. The core hardware modules incorporated in the design are as follows:

i Raspberry Pi 5: This Serves as the primary processing unit, responsible for executing the face recognition pipeline and coordinating peripherals.

ii Night-Vision Enabled Camera: it captures live video input, ensuring usability even in low-light classroom environments.

iii OpenFace + Dlib Libraries: They handle face detection, landmark localization, and deep CNN-based embeddings (face signatures).

iv LCD Display and 4x3 Keypad: They facilitate user interaction, feedback, and manual control options.

v SD Card / Spreadsheet Storage: It stores labeled student images and student attendance logs for future retrieval and reporting.

vi Power Supply (Battery Pack/AC Adapter): This provides portable, flexible power options suitable for operation in diverse environments.

vii Pan/Tilt Servo Motors: They ensure controlled camera movement for scanning by rotating camera horizontally and vertically, thereby expanding coverage.

For Pan/Tilt coverage, the scanning mechanism is designed to mimic the natural head movements of a human observer scanning through the classroom. The servo motors rotate the camera through predefined angular positions, as expressed in Equation 9. At each position, the system captures frames and processes them in real time. This dynamic scanning strategy enhances recognition coverage without the need for multiple fixed cameras. Figure 7 show partial implementation of the system hardware.

$$\theta_{pan} = \{0°, 30°, 60°, \dots, 180°\}$$

$$\theta_{tilt} = \{-30°, 0°, +30°\} \qquad (9)$$

Figure 5 shows the implementing of the system hardware setup.

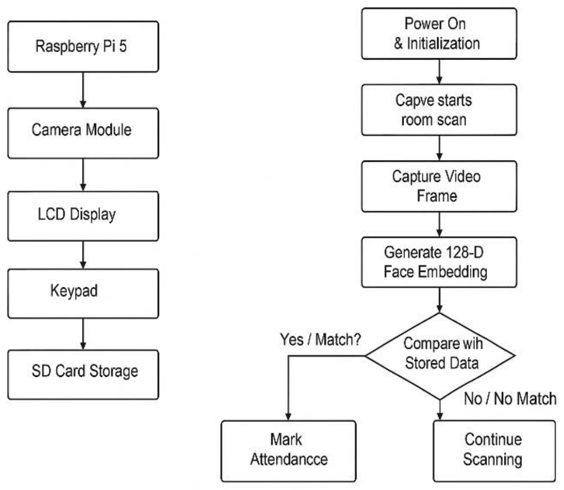


Figure 5: Hardware Operation block diagram (left) and flowchart (right)

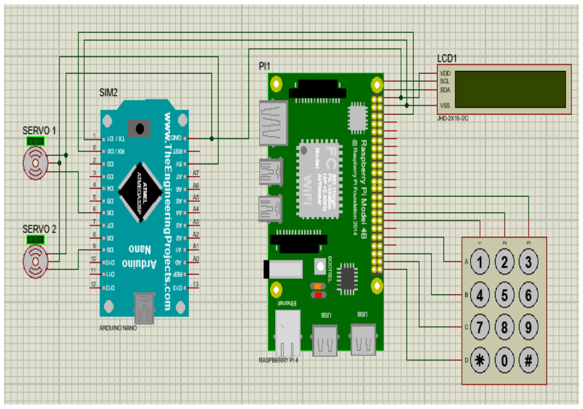


Figure 6: Circuit diagram for facial recognition class attendance system

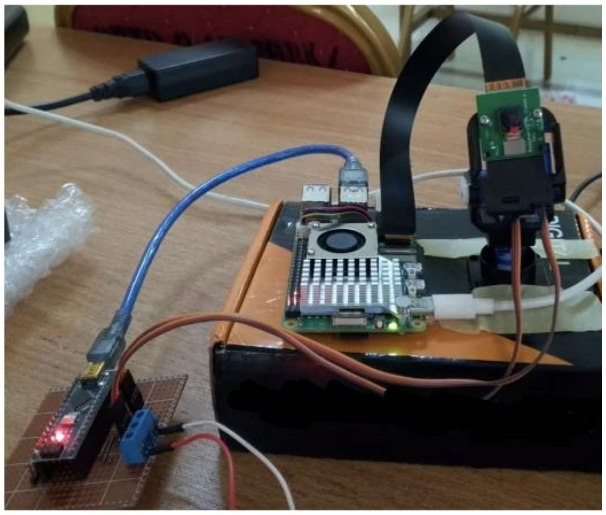


Figure 7: System hardware prototype setup

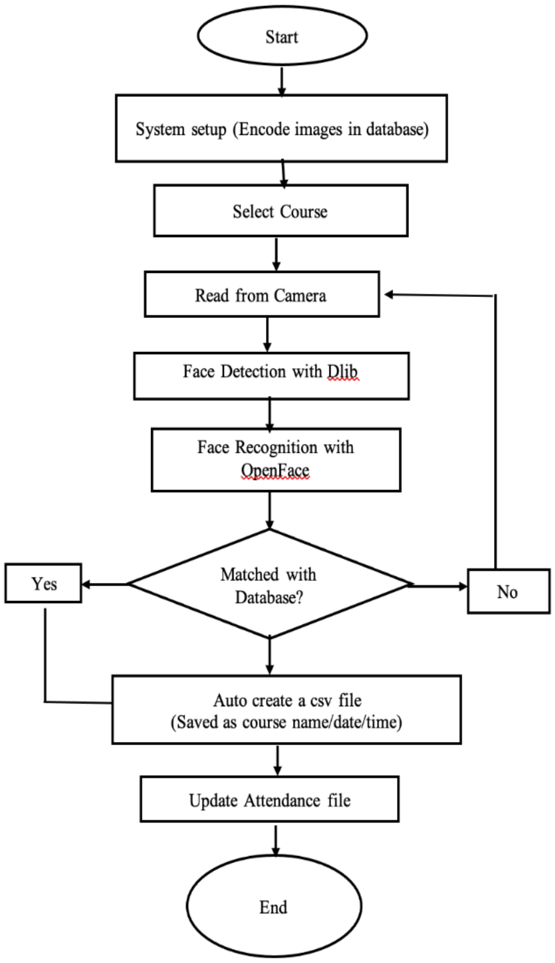

Start
System setup (Encode images in database)
Select Course
Read from Camera
Face Detection with Dlib
Face Recognition with OpenFace
Matched with Database?
Yes
No
Auto create a csv file (Saved as course name/date/time)
Update Attendance file
End

Figure 8: System Flow Chart

D. Overall System Operation

The proposed face recognition–based attendance system developed in this work employed structured process. For recognition, the system combines Dlib's facial landmark prediction with OpenFace's CNN-based embedding model [16], transforming real-time video input into 128-dimensional feature vectors. These embeddings are compared against a pre-enrolled database of student identities. This design makes the solution particularly practical for institutions with limited infrastructure, as it supports offline operation while reliably maintaining attendance records.

Figure 8 presents the flowchart of the overall system operation.

Workflow Overview

1. Classroom Scanning: Servo motors pan/tilt the camera to cover classroom region.
2. Face Detection: Dlib and OpenFace algorithms detect student faces in each captured frame.
3. Feature Extraction: A CNN generates 128-dimensional embeddings for each detected face
4. Face Recognition: Each embedding is compared with stored encodings in the student database using Linear SVM classifier
5. Identity Confirmation: if the classifier returns a positive match, the identity is confirmed.
6. Attendance Logging: The recognized identity is stored in a CSV file with a timestamp:
   $A = \{(ID_i, t_i) \mid i \in Recognized\ Students\}$

where ID_i is the student identifier and t_i is the loggedtime.

The continuous loop of scanning, recognition, and logging is implemented as outlined in the pseudocode that follows.

```
BEGIN
  Initialize camera
  Initialize servo (pan/tilt)
  Initialize face recognition model (OpenFace)
  Load stored facial embeddings from local database

  WHILE system is running DO
    FOR each angle in predefined_pan_angles DO
      Rotate servo to angle
      frame <= capture_frame_from_camera()

      IF face_detected (frame) THEN
        embedding <= extract_embedding(frame)
        match <= compare_embedding_to_database(embedding)

        IF match IS found THEN
          IF NOT already_marked_today(match.person_id) THEN
            log_attendance (match.person_id, current_time)
            Display ("Attendance marked for " + match.person_id)
          ELSE
            Display ("Already marked today")
          END IF
        ELSE
          Display ("Unrecognized face")
        END IF
      ELSE
        Display ("No face detected")
      END IF
      WAIT 1 second
    END FOR
  END WHILE
END
```

IV. DISCUSSUION OF RESULTS

The system was evaluated on key parameters such as recognition accuracy, inference speed, and adaptability in real-time classroom environments. The results indicate strong performance, with the system achieving a recognition accuracy of 96.8% and an average inference time of 0.79 milliseconds per recognition, thereby demonstrating both precision and responsiveness for small- to medium-sized classrooms.

The implementation validated the system's end-to-end functionality across multiple operational stages. As shown in Figure 10, the system initializes seamlessly, confirming hardware readiness and establishing the software environment. Once active, the interface prompts course selection (Figure 11), ensuring proper categorization of attendance records. During classroom execution, the system continuously scans students enrolled in the chosen course (Figure 12), detecting faces in real time and matching them against stored embeddings. Successful recognition is depicted in Figure 13, where identified students are displayed alongside their names. Figure 14 highlights the robustness of the system under varying facial orientations, confirming tolerance to non-frontal poses and natural classroom dynamics. Finally, Figure 15 shows the automated logging of attendance into a file containing both student names and timestamps, validating the accuracy and reliability of the attendance record-keeping process. attendance logs were maintained in CSV format, viewed in Microsoft Excel, and stored locally for privacy and operational continuity in low-resource environments where internet connectivity is limited. This local-first approach strengthens both security and reliability, making the system particularly suited for institutions in rural or infrastructurally disadvantaged settings. The packaged device (Figure 9) further underscores portability and suitability for classroom deployment. Collectively, these findings confirm the system's robustness, scalability, and adaptability under practical classroom conditions.

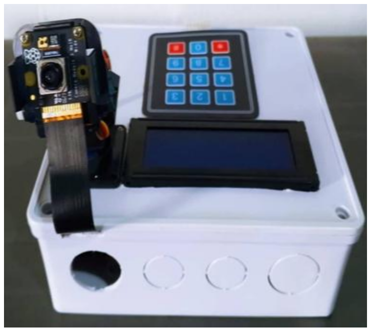


Figure 9: System Casing and Packaging

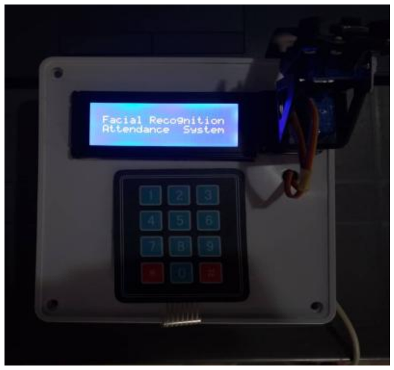


Figure 10: The attendance system is initializing

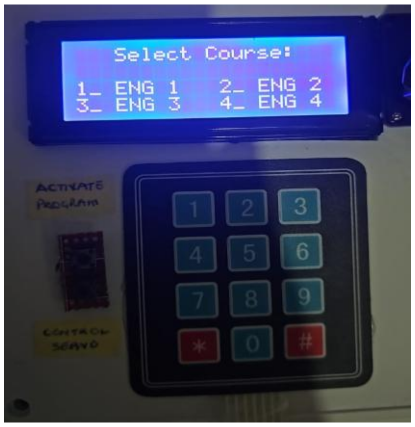


Figure11: System prompts the user to select a course.

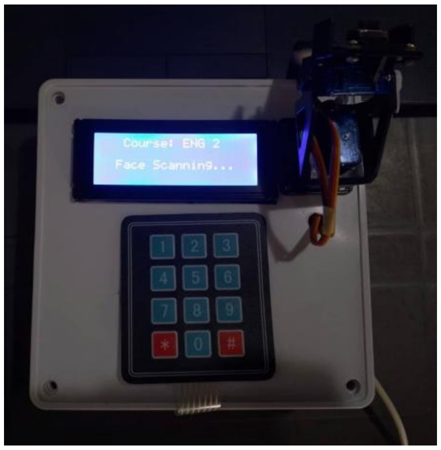


Figure 12: The system scanning class for a course

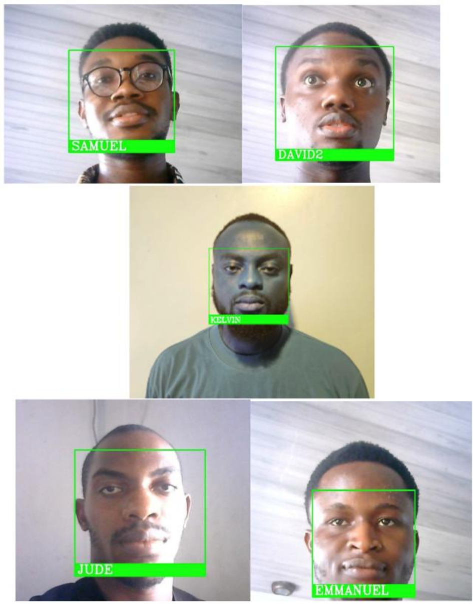


Figure 13: Recognized students alongside their corresponding names.

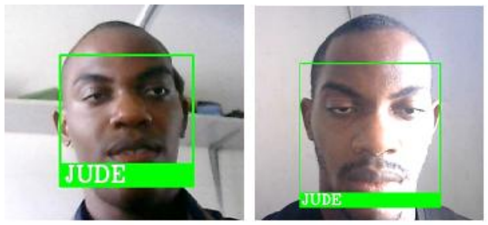
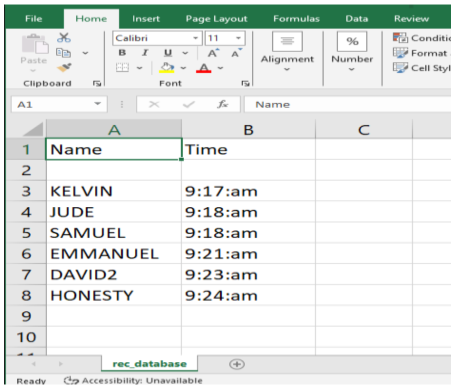


Figure 14: Recognition of the same individual at different angles.



	A	B	C
1	Name	Time	
2			
3	KELVIN	9:17:am	
4	JUDE	9:18:am	
5	SAMUEL	9:18:am	
6	EMMANUEL	9:21:am	
7	DAVID2	9:23:am	
8	HONESTY	9:24:am	
9			
10			

Figure 15: Attendance file showing recognized students with detection time.

Advantages of the System

i. Offline Operation: Attendance is stored locally (CSV/SD card), ensuring usability in areas with poor internet connectivity.
ii. Scalability: New students can be enrolled by appending additional encodings to the database.
iii. Efficiency: Each recognition and logging operation occurs in milliseconds.
iv. Robustness: Operates reliably under different lighting conditions due to the night-vision camera and CNN-invariant embeddings.

CONCLUSION

This work presented the design and implementation of a facial recognition–based attendance system integrating deep learning, embedded hardware, and automated camera control for classroom deployment. By leveraging OpenFace's CNN-based embeddings with SVM classification, the system achieved a recognition accuracy of 96.8% with an average inference time of 0.79 milliseconds per recognition. Offline data logging into local storage further enhanced privacy, reliability, and usability, particularly in resource-constrained environments.

The results demonstrate that the system performs reliably under varying facial orientations, providing accurate, timestamped attendance records. Its portability, offline capability, and cost-effectiveness make it especially well-suited for rural and resource-limited educational institutions. Future work could focus on integrating a mobile application for greater flexibility and student/teacher notifications, alongside exploring renewable energy solutions and reliable backup systems to ensure sustainability and uninterrupted operation in diverse settings.

ACKNOWLEDGMENT

The authors wish to express their sincere gratitude to all the participants who voluntarily consented to provide their facial data for this research. Their contributions were essential to the development, testing, and validation of the proposed system.

REFERENCES

[1] Arafat, A., Pehlivan, H., & Mohammed, F. (2020). Modern attendance classroom system. Proceedings of the Euro-Asia 7th International Congress on Applied Sciences, Trabzon, Turkey.https://doi.org/10.6084/m9.figshare.13003652.v1

[2] Roushan, G., Polkinghorne, M., & Patel, U. (Eds.). (2024). Teaching and learning with innovative technologies in higher education: Real-world case studies. Routledge. https://doi.org/10.4324/9781032635248

[3] Chowdhury, F. Y. (2023). Implementation of attendance management system utilising fingerprint, QR code, and GPS technology in educational institutions. International Journal of Science and Research (IJSR), 12(3), 2016–2019.

[4] Khalifa, I. A., & Keti, F. (2025). The role of image processing and deep learning in IoT-based systems: A comprehensive review. European Journal of Applied Science and Engineering Technology, 3(1), 165–179. https://doi.org/10.59324/ejaset.2025.3(1).15

[5] Gaur, S., Pandey, M., & Himanshu, N. (2024). Realization of facial recognition technology for attendance monitoring through biometric modalities employing MTCNN integration. SN Computer Science, 5(7), Article 579. https://doi.org/10.1007/s42979-024-03225-1

[6] Guo, S., Chen, S., & Li, Y. (2016). Face recognition based on convolutional neural network and support vector machine. In Proceedings of the IEEE International Conference on Information and Automation (pp. 17871792).IEEE.https://doi.org/10.1109/icinfa.2016.7832107

[7] Kim, J., Kim, M., Suh, H., Naseem, M. T., & Lee, C. (2022). Hybrid approach for facial expression recognition using convolutional neural networks and SVM. Applied Sciences, 12(11),5493.https://doi.org/10.3390/app12115493

[8] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

[9] Alalibo, T. O. J., & Nwazor, N. O. (2023). Comparative Analysis of Convolutional Neural Network Models for Solid Waste Categorization. Iconic Research and Engineering Journals, 6(12), 1396-1405.

[10] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823).

[11] Zhang, S. (2022). Face recognition model design based on convolutional neural networks. Scientific Programming, 2022, 1–11. https://doi.org/10.1155/2022/3922377

[12] Khan, S., Javed, M. H., Ahmed, E., Shah, S. A. A., & Ali, S. U. (2019). Facial recognition using convolutional neural networks and implementation on smart glasses. 2019 International Conference on Information Science and Communication Technology (ICISCT), 1–6. https://doi.org/10.1109/CISCT.2019.8777442

[13] Carcagnì, P., Del Coco, M., Leo, M., & Distante, C. (2015). Facial expression recognition and histograms of oriented gradients: a comprehensive study. SpringerPlus,

4(1). https://doi.org/10.1186/s40064-015-1427-3

[14] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, & Rabinovich A. (2015). Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (pp. 19).IEEE.http://doi.org/10.1109/cvpr.2015.7298594

[15] Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. In Proceedings of the British Machine Vision Conference (BMVC).https://ora.ox.ac.uk/objects/uuid:a5f2e93f-2768-45bb-8508-74747f85cad1

[16] Amos, B., Ludwiczuk, B., & Satyanarayanan, M. (2016). OpenFace: A general-purpose face recognition library with mobile applications (CMU-CS-16-118). Carnegie Mellon University.https://cmusatyalab.github.io/openface

[17] Kazemi, V., & Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1867-1874).