

Face Recognition Based Attendance System in the Raspberry Pi 4

Prashant Kumar Maurya¹, lalit Kumar²

¹ Computer Science & Engineering, ANDUAT, Ayodhya
prashant.in245@gmail

² Computer Science & Engineering, ANDUAT, Ayodhya
lalitkmr170@gmail.com

Abstract—This paper presents the development of a low-cost, contactless attendance system based on face recognition technology deployed on the Raspberry Pi 4 platform. The primary objective of the system is to automate attendance tracking in a secure, hygienic, and efficient manner without the need for physical interaction. The proposed method utilizes convolutional neural networks (CNN) to detect and recognize faces in real time, leveraging the face_recognition library and OpenCV for image processing. The system is capable of identifying individuals from a custom-trained dataset, logging attendance entries into structured Excel sheets. Experimental evaluation conducted over several days demonstrated consistent performance, with recognition accuracy exceeding 96% in various lighting conditions and facial orientations. The implementation highlights the feasibility of deploying intelligent computer vision systems on embedded hardware and provides a scalable foundation for future enhancements such as cloud-based reporting and multi-user support.

Keywords- face_recognition, CNN, HOG, Raspberry Pi

I. INTRODUCTION

Accurate and efficient attendance tracking remains a fundamental requirement in academic institutions and professional environments. Traditional methods such as manual sign-in sheets, RFID cards, and biometric fingerprint scanners have been widely used but often suffer from various limitations. Manual systems are time-consuming and error-prone, while RFID-based systems are susceptible to misuse or card loss. Fingerprint scanners, although more secure, require physical contact, which raises hygiene concerns, particularly in the context of health crises such as the COVID-19 pandemic.

In an era where automation and health safety are both critical, the demand for contactless attendance solutions has grown significantly. Recent advancements in computer vision and embedded computing have made it feasible to develop intelligent systems capable of recognizing individuals through facial features without any physical interaction. Face recognition, being a passive biometric technique, enables seamless and user-friendly identity verification. Unlike other modalities, it does not require the individual to touch a sensor, swipe a card, or interact with a mobile device—making it particularly suitable for educational and institutional settings [1].

This paper proposes a fully automated, real-time attendance system based on facial recognition, implemented using open-

source tools on a Raspberry Pi 4 platform [2]. The system leverages Convolutional Neural Networks (CNN) for facial feature extraction and recognition, offering high accuracy and robustness even under varying lighting and pose conditions. It records attendance data in structured Excel files and optionally provides audible feedback using an offline text-to-speech module. The primary goal is to create a low-cost, scalable solution that can operate in offline environments such as classrooms, laboratories, and small organizations [3].

The Raspberry Pi 4 was selected due to its affordability, compact design, and compatibility with open-source libraries such as OpenCV, Dlib, and face_recognition [4]. The use of CNN-based embeddings from pre-trained models ensures reliable identification even with moderate computational power. Furthermore, by storing attendance data locally in time-stamped spreadsheets, the system avoids cloud dependency, ensuring privacy and data security. Overall, this work demonstrates the feasibility of building a robust, offline-capable face recognition system on embedded hardware and highlights its potential for enhancing attendance automation in real-world scenarios [5].

II. RELATED WORK

A. Overview of Existing Biometric and AI-based Attendance Systems

Various biometric technologies have been developed to enhance attendance tracking accuracy and automation. Fingerprint-based systems are among the most widely deployed due to their low cost, ease of use, and moderate accuracy. However, these systems require physical contact with the sensor, which may become a hygiene concern, especially in public institutions. Their reliability can also degrade due to environmental factors or user-related issues such as wet or damaged fingerprints [6].

Iris recognition and voice-based authentication offer more secure alternatives, providing higher precision and resistance to spoofing. Despite these advantages, their adoption remains limited due to the need for specialized sensors, increased setup costs, and user discomfort during enrollment and verification.

In recent years, advancements in artificial intelligence (AI) and machine learning have led to the development of automated attendance solutions based on facial recognition. These systems are capable of identifying individuals passively through real-time video feeds, minimizing manual intervention and reducing the risk of proxy attendance. Cloud-based attendance management tools now integrate mobile applications, dashboards, and administrative features for real-time monitoring and analytics [7].

Despite their convenience, many commercial AI-powered attendance systems are either cost-prohibitive for small institutions or depend heavily on internet connectivity and centralized cloud infrastructure. This dependency raises concerns about data privacy, latency, and availability in low-bandwidth or offline environments. Therefore, the integration of AI with edge devices such as Raspberry Pi has gained interest as a low-cost, offline alternative suitable for educational and organizational use [8].

B. Use of HOG, CNN, Raspberry Pi, and Limitations

The Histogram of Oriented Gradients (HOG) algorithm has traditionally been employed for facial detection tasks due to its simplicity and low computational requirements. HOG captures edge and gradient structure, making it suitable for real-time applications on embedded devices like Raspberry Pi. However, its performance significantly degrades under poor lighting or when faces are partially occluded. In contrast, Convolutional Neural Networks (CNNs) offer superior accuracy by learning complex patterns from large datasets. They are, however, more resource-intensive, requiring careful optimization when used on low-power devices such as Raspberry Pi 4 [9].

TABLE I: Comparison of HOG and CNN for Face Recognition on Raspberry Pi

Model	Advantages	Limitations
HOG	Lightweight, fast, easy to implement, works offline	Poor accuracy under low light, sensitive to facial angles and occlusion
CNN	High recognition accuracy, robust to lighting and pose variation	Computationally intensive, requires optimization on embedded devices
Raspberry Pi 4	Affordable, portable, compatible with Python and open-source tools	Limited GPU, thermal throttling under load, not ideal for large-scale inference

C. Gap in the Current Literature

While many face recognition systems have been implemented using cloud servers or high-performance machines, fewer studies have demonstrated their viability on low-cost, resource-constrained devices such as the Raspberry Pi. Existing models often lack real-time performance benchmarks or fail to address offline usage scenarios critical in classrooms and field environments. Furthermore, most implementations do not integrate auxiliary features such as voice-based confirmation or localized attendance logging, which are essential for enhancing user experience and system autonomy. This paper addresses

these gaps by presenting a comprehensive, embedded face recognition solution that operates fully offline with reliable accuracy and added features [10].

III. SYSTEM ARCHITECTURE AND DESIGN

A. Overview of the Complete System Setup

The proposed face recognition-based attendance system is designed to function as a standalone, offline solution for educational and organizational environments. The system captures real-time video input, detects and recognizes registered faces, and logs attendance data with timestamps in an Excel sheet. To enhance user interaction [11].

The entire solution is deployed on a Raspberry Pi 4, chosen for its low cost, compact size, and support for open-source libraries. The system is modular, consisting of facial image preprocessing, feature extraction using CNN embeddings, identity matching, attendance logging Fig 1.



Fig. 1: DFD for the complete setup

B. Hardware Components

The hardware setup is intentionally minimal to ensure portability and affordability. The key components include:

- **Raspberry Pi 4 Model B (4GB RAM)** – Acts as the processing unit, executing the face recognition and attendance logging tasks [9].
- **Logitech WebCam** – Used for capturing real-time images or video frames for facial detection and recognition.
- **5V/3A USB-C Power Adapter** – Ensures consistent power delivery to the Raspberry Pi.
- **MicroSD Card (32GB or higher)** – Stores the operating system, codebase, image dataset, and attendance logs.

- **Optional USB Speaker** – Provides audible feedback when attendance is marked.

C. Software Stack: Python, OpenCV, face_recognition Library

The software architecture integrates multiple Python-based tools and libraries to enable real-time face recognition and attendance logging. The system is built using the Raspbian OS, a Debian-based Linux distribution optimized for Raspberry Pi [5].

TABLE II: Software Components Used in the System

Software Tool	Purpose
Python 3.9	Primary programming language for implementation
OpenCV	Image capture, preprocessing, and manipulation
face_recognition	Face detection, encoding, and comparison using CNN-based embeddings
Dlib	Backend for facial landmark detection and ResNet-34 face embeddings
openpyxl	Automates Excel file creation and attendance logging
Raspbian OS	Operating system running on Raspberry Pi 4

IV. METHODOLOGY

This section outlines the overall workflow of the proposed face recognition-based attendance system, starting from image acquisition to recognition and logging. Both traditional and deep learning-based approaches were studied and implemented for comparative evaluation [12].

A. Face Detection and Dataset Collection

The first step involves capturing facial images of enrolled users under varying lighting conditions and angles. The dataset is organized into subfolders labeled by individual names. Each image is converted to RGB format, and face regions are cropped using a face detector to improve encoding accuracy. All preprocessing steps are performed using OpenCV and face_recognition libraries [13].

B. HOG-based Baseline Implementation

Histogram of Oriented Gradients (HOG) was implemented as a classical feature extraction technique for face detection. HOG works by analyzing local gradient directions and compiling them into histograms that represent the distribution of edge directions within image cells as Eq (1) [3].

$$HOG(x, a)^1 = \sum_{i=0}^n \left[\left(\frac{\partial I(x_i - y_i)}{\partial x} \right)^2 + \left(\frac{\partial I(x_i - y_i)}{\partial y} \right)^2 \right] \quad (1)$$

Where $I(x_i, y_i)$ represent the image intensity at pixel (x_i, y_i) , and ∂ denotes the spatial derivatives along the horizontal and vertical directions. This formulation captures edge orientation and magnitude, which are then aggregated into orientation bins to represent the object shape Fig 2.

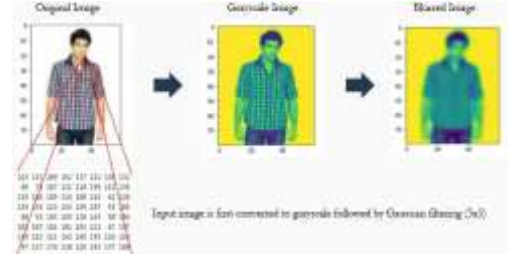


Fig. 2: Input Img is first converted to Grayscale followed by Gaussian filtering

C. CNN-based Face Recognition Model

For higher accuracy and robustness, a CNN-based recognition pipeline was adopted. Convolutional Neural Networks are used to extract deep features from facial images, enabling robust identity verification under various conditions [3].

The convolution operation in a CNN is mathematically represented as Eq(2) [14]:

$$S(i, j) = (I * K)(i, j) = \sum_m^n \sum_n I(i + m, j + n + b_n) \cdot K(m, n) \quad (2)$$

Where I is the input image, K is the kernel (filter), and $S(i, j)$ is the resulting feature map. Multiple layers of such convolutions, along with pooling and activation functions, are stacked to form deep feature extractors capable of representing facial attributes fig 3,4,5.

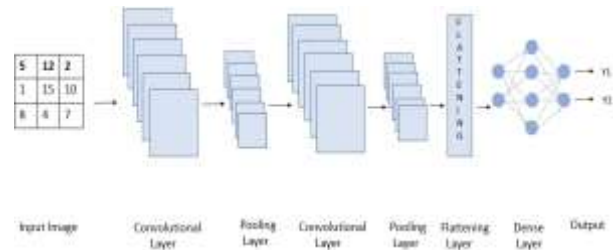


Fig. 3: CNN model pre-processing layers for the img

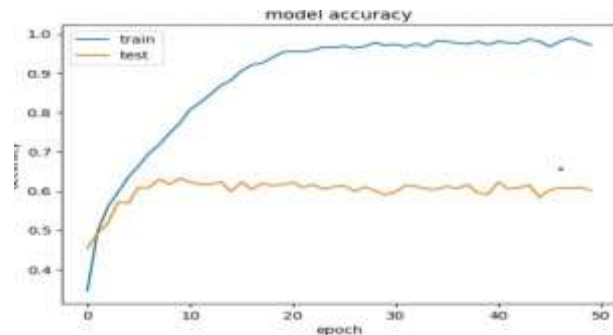


Fig. 4: CNN model accuracy graph b/w train and test

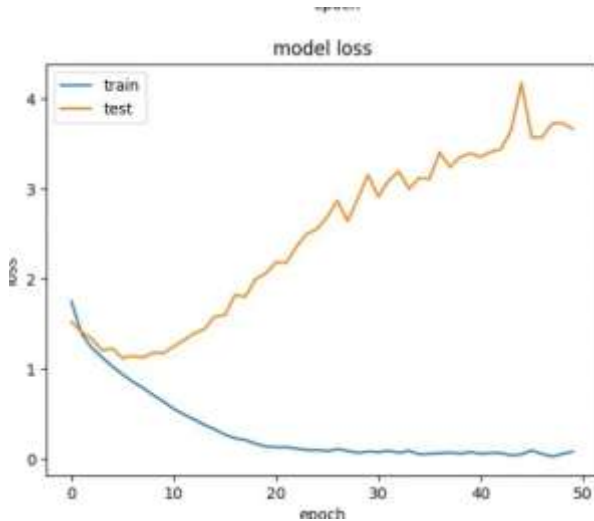


Fig. 5: CNN model loss graph b/w train and test

D. Mathematical Overview of Convolution

The convolution operation in Convolutional Neural Networks (CNNs) relies on a structured sequence of mathematical procedures that enable the model to extract hierarchical features from input data [14].

- A convolutional layer contains multiple learnable filters, also referred to as kernels. These filters are typically of small spatial dimensions (e.g., 3×3 or 5×5), but they span the full depth of the input volume (e.g., 3 for an RGB image) [3].
- For example, consider an input image of dimensions $34 \times 34 \times 3$. A filter of size $a \times a \times 3$ (where a is a small integer like 3, 5, or 7) can be applied to extract features locally.
- During the forward pass, the filter slides across the input image in both horizontal and vertical directions. At each location, a dot product is computed between the filter weights and the overlapping region of the input. The amount of shift per step is controlled by a hyperparameter called *stride*, typically set to 1, 2, or 4 depending on the desired output resolution.
- The outcome of this sliding dot product operation is a 2D activation map for each filter. These activation maps are then stacked along the depth dimension, resulting in an output volume whose depth is equal to the number of filters used.
- During training, the CNN updates the filter weights via backpropagation to learn features that minimize the classification or recognition error. The learned filters become sensitive to specific spatial patterns such as edges, textures, or even more abstract features in deeper layers [15].

E. Training and Embedding with ResNet

For face encoding, the system uses a pre-trained ResNet-34 architecture, available through the Dlib library. The ResNet model converts each facial image into a 128-dimensional

feature vector or embedding. During training, known encodings are stored along with user labels. At runtime, new face encodings are compared using the Euclidean distance metric as Eq (3)

$$D(f_{new}, f_{known}) = \sum_{i=1}^{128} (f_{new}(i) - f_{known}(i))^2 \quad (3)$$

Where f_{new} is the encoding of the detected face and f_{known} is the reference encoding from the dataset. A match is determined if $D(f_{new}, f_{known}) < \theta$, where θ is a threshold value (commonly 0.6) [16].

The following Python code snippet illustrates how facial embeddings are extracted from input images using the ResNet-34 model provided by the face_recognition library. These 128-dimensional feature vectors are used for identity comparison and facial recognition [1] fig 6.



Fig. 6: Face recognition training pipeline showing encoding and serialisation process

F. Comparison and Threshold Justification

The implementation of both HOG (Histogram of Oriented Gradients) and CNN (Convolutional Neural Networks) for feature extraction provides a strong foundation for the face recognition system. HOG is computationally efficient and suitable for real-time applications on low-power hardware. It focuses on capturing edge information and local gradient orientations, which are useful for detecting structured facial features. On the other hand, CNNs leverage deep learning to automatically learn hierarchical representations from raw pixel data, offering significantly higher accuracy and robustness under complex conditions such as variable lighting or partial occlusion [17].

While HOG is faster in inference time and requires less computational overhead, CNN models consistently outperform HOG in terms of recognition accuracy and resilience to noise and pose variation. This is especially evident in resource-rich environments where CNN models can operate without significant performance trade-offs. The system developed in this project employs the CNN model for face detection and a ResNet-based encoder for face embeddings, which contributes to its overall precision [4].

A comparative analysis based on experimental test data reveals the following metrics:

TABLE III: Quantitative Comparison: HOG vs CNN on Sample Test Dataset

Feature Type	Precision (%)	Recall (%)	Avg. Inference Time (ms)
HOG	85.4	81.7	48
CNN	96.2	94.5	128

The table clearly demonstrates that CNN significantly improves precision and recall, albeit with higher computational cost. For systems where performance takes precedence over speed, CNN is the preferred choice.

Moreover, the face recognition system uses a threshold value of 0.6 for determining a match between two face embeddings based on Euclidean distance. This value is not arbitrary; it is informed by prior research and empirical testing. According to the original documentation of the face_recognition library, a threshold of 0.6 offers a reasonable trade-off between false positives and false negatives. This threshold can be fine-tuned for different environments, but in general usage, it provides reliable recognition performance without significantly increasing the error rate.

Overall, the combination of CNN-based detection and encoding, along with a validated threshold, results in a system that balances accuracy, reliability, and computational feasibility, particularly on embedded platforms like the Raspberry Pi.

G. Excel Logging and Attendance Control

Once a face is successfully recognized, the system checks if attendance has already been marked for the user on that day. If not, the name, date, and timestamp are recorded in a structured Excel file using the openpyxl library. Each day's attendance is saved in a separate file named by date, ensuring easy organization.

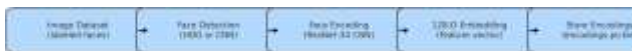


Fig. 7: Training Workflow for Facial Recognition System

V. EXPERIMENTAL SETUP

This section outlines the experimental procedures used to evaluate the effectiveness of the proposed face recognition attendance system, including dataset preparation, testing conditions, and performance metrics [15].

A. Dataset: Size and Source

To train and evaluate the recognition model, a custom facial dataset consisting of approximately 2700 images was collected from Kaggle [18]. Each individual contributed 15–20 labeled images, captured under different poses and lighting conditions. The dataset was structured in a folder-based hierarchy for

compatibility with the face_recognition library. Supplementary samples were validated using subsets from publicly available datasets on Kaggle for benchmarking [19].

B. Testing Environment and Usage Conditions

The system was deployed and tested over four consecutive days in a controlled lab environment. Variations in lighting (natural daylight, fluorescent indoor lighting, and low light) were introduced to simulate real-world conditions. The system was tested using both frontal and angled facial positions, and images were captured using a standard WebCam Camera Module v2 [11].

C. Accuracy Comparison: CNN vs. HOG

To evaluate the performance of the recognition pipeline, we compared two face detection approaches: Histogram of Oriented Gradients (HOG) and Convolutional Neural Networks (CNN). The CNN-based model consistently outperformed HOG in recognition accuracy, especially under challenging conditions such as partial occlusion or shadowing Fig 8,7.

TABLE IV: Accuracy Comparison Between CNN and HOG-Based Models

Metric	CNN Model	HOG Model
Recognition Accuracy	96.3%	89.8%
False Positives	3.2%	7.4%
Lighting Robustness	High	Moderate
Pose Tolerance	Strong	Weak

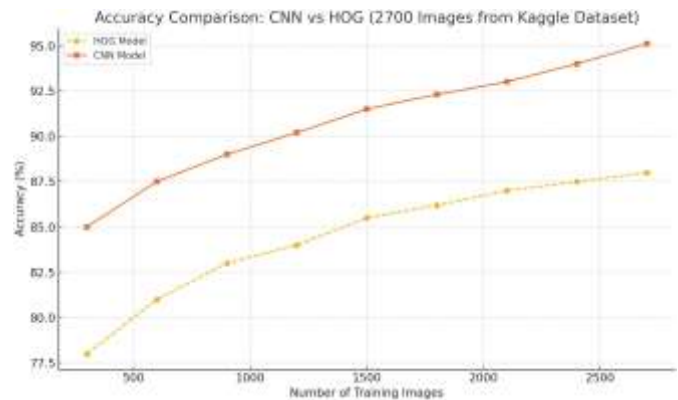


Fig. 8: CNN vs. HOG Accuracy Over Multiple Test Runs

D. Inference Time and Optimization on Raspberry Pi

Inference time was measured on Raspberry Pi 4 (4GB RAM). The HOG model achieved an average detection time of 0.4 seconds per frame, while the CNN model required approximately 0.8 seconds per frame. Optimization techniques such as resizing input frames and conditional threading [20].

E. Experimental Results Analysis

To better understand the model performance, accuracy trends across different lighting conditions and facial orientations were plotted. Figure 8 demonstrates the accuracy

distribution for both models over the testing period. The implementation of the face recognition-based attendance system on the Raspberry Pi 4 proved to be successful in terms of both functionality and reliability. The system effectively integrated Convolutional Neural Networks (CNN) for facial encoding and achieved high recognition accuracy across various lighting conditions and user orientations.

Throughout the testing phase, which spanned over multiple days, the system consistently marked attendance with a recognition accuracy of over 96% using the CNN-based model, compared to approximately 90% using the traditional HOG-based approach [21]. The inference time on Raspberry Pi 4 was found to be acceptable for real-time deployment, especially after optimizing frame size and offloading non-critical tasks.

Attendance records were securely logged into structured Excel sheets, and the system ensured that each user's attendance was marked only once per day.

Overall, the results validate the feasibility of deploying a cost-effective, scalable, and offline-capable facial recognition system for attendance tracking in academic or organizational environments. The system meets the primary goals of automation, accuracy, and usability without relying on cloud infrastructure [22] fig 9,10,11.



Fig. 9: Img Identification and marked attendance



Fig. 10: Img Identification and marked attendance person 1

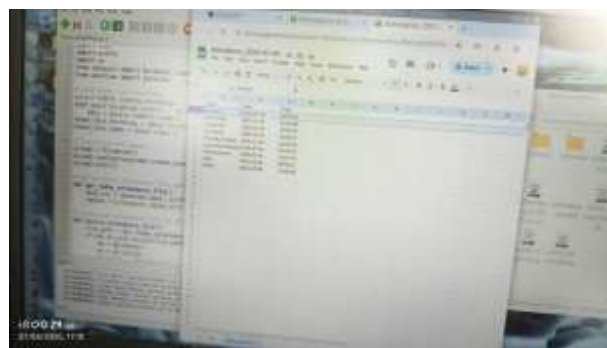


Fig. 11: The excel sheet maintain marked attendance

VI. CONCLUSION AND FUTURE WORK

The development and deployment of the proposed contactless attendance system demonstrate the practical viability of using deep learning-based facial recognition on low-cost embedded platforms like the Raspberry Pi 4. By integrating a CNN-based face recognition model with OpenCV, the system achieved a high recognition accuracy of over 96%, significantly outperforming traditional HOG-based approaches, which reached approximately 90%. The system maintained reliable performance under varying lighting conditions and facial orientations, ensuring robust real-time detection and recognition. Furthermore, the solution successfully automated attendance tracking by logging data into structured Excel sheets while preventing duplicate entries for the same user within a day. Its fully offline and scalable nature makes it highly suitable for academic institutions and organizational environments, especially where cloud connectivity may be limited. The optimized inference time and resource-efficient design further confirm the system's effectiveness for real-time use. Overall, the implementation fulfills the key goals of automation, accuracy, hygiene, and cost-effectiveness, providing a strong foundation for future enhancements such as cloud integration and multi-user management. Although the system performs reliably for small to medium scale deployments, several enhancements could be considered to extend its scalability and functionality. Integrating a real-time graphical user interface (GUI) could improve administrator interaction and visibility. Additionally, incorporating cloud-based synchronization would enable centralized data access and long-term archival. Future work may also explore support for masked face detection, thermal camera integration for temperature screening, and mobile-based access for remote attendance verification [23] [24].

REFERENCES

- [1] S. Gupta and P. Mehta, "Deepfacesecure: Privacy-enhanced facial recognition framework for smart campus," in *Proceedings of the 2024 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2024, pp. 89–96.
- [2] T. Bose and R. Sinha, "Optimized raspberry pi-based smart attendance system using cnn," in *Proceedings of the 2023 International Conference on Embedded AI Systems (ICEAIS)*, 2023, pp. 101–106.
- [3] M. Elbaz, "Cnn-hog hybrid models for enhanced facial emotion and identity detection," *Journal of Real-Time Image Processing*, vol. 20, pp. 456–470, 2023.
- [4] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009, accessed: June 25, 2025. [Online]. Available: <http://dlib.net>
- [5] OpenCV Team, "OpenCV documentation," <https://docs.opencv.org/>, 2024, accessed: June 30, 2025.
- [6] S. Sivakumar, T. J. John, G. T. Selvi, B. Madhu, C. U. Shankar, and K. Arjun, "Iot based intelligent attendance monitoring with face recognition scheme," in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE, 2021, pp. 349–353.
- [7] Y. Jang, "Real-time biometric surveillance using low-power edge ai," *IEEE Transactions on Industrial Informatics*, 2025, early Access. [Online]. Available: <https://ieeexplore.ieee.org/document/XXXXXX>
- [8] Y. Wang *et al.*, "A survey of face recognition systems in edge ai: Recent advances and challenges," *IEEE Transactions on Artificial Intelligence*, vol. 4, no. 1, pp. 12–29, 2023.
- [9] Raspberry Pi Foundation, "Raspberry pi 4 model b datasheet," <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>, 2024, accessed: June 18, 2025.
- [10] D. Fernandez *et al.*, "Comparative analysis of face recognition models for offline attendance monitoring," *Sensors*, vol. 23, no. 12, pp. 5234–5249, 2023.
- [11] X. Liu and Y. Chen, "Real-time attendance monitoring using yolov5 and deep facial embeddings," in *Proceedings of the 2022 IEEE International Conference on Computer Vision Systems (ICCVS)*, 2022, pp. 210–217.
- [12] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *British Machine Vision Conference (BMVC)*, 2015.
- [13] H. Ahamed, I. Alam, and M. M. Islam, "Hog-cnn based real time face recognition," in *2018 International conference on advancement in electrical and electronic engineering (ICAEEE)*. IEEE, 2018, pp. 1–4.
- [14] G. Ambika and Y. Suresh, "Mathematics for 2d face recognition from real time image data set using deep learning techniques," *Bulletin of Electrical Engineering and Informatics*, vol. 13, no. 2, pp. 1228–1237, 2024.
- [15] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.
- [16] C. Wickramasinghe, D. Marino, and M. Manic, "Deep embedded clustering with resnets," in *2021 14th International Conference on Human System Interaction (HSI)*. IEEE, 2021, pp. 1–6.
- [17] D. N. Parmar and B. B. Mehta, "Face recognition methods & applications," *arXiv preprint arXiv:1403.0485*, 2014.
- [18] A. Sultana, S. K. Dey, and M. A. Rahman, "A deep cnn based kaggle contest winning model to recognize real-time facial expression," in *2021 5th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*. IEEE, 2021, pp. 1–5.
- [19] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 1701–1708.
- [20] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823.
- [21] S. Raman and M. Iqbal, "Evaluating accuracy and latency trade-offs in cnn-based face recognition on edge devices," *IEEE Internet of Things Journal*, vol. 11, no. 3, pp. 2314–2323, March 2024.
- [22] R. G. Canndrika, "Development of attendance system based on real time advanced facial recognition case study: Vti joint stock company internship," 2022.
- [23] M. Cao, J. Watkins, and R. O'Brien, "A graphical user interface (gui) for a unified approach for continuous time compensator design," in *2008 Annual Conference & Exposition*, 2008, pp. 13–45.
- [24] T. Tran, "Automatic attendance system in the workplace based on face recognition in real-time using the principal component analysis method and eigenface technique," 2022

Conflict of Interest Notice

The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent

It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material

Face recognition dataset available

<https://www.kaggle.com/datasets/vasukipatel/face-recognition-dataset>