June 2025 | Volume 1 | Issue 1 | Article No. 25304

Journal of Information and Communications Technology: Algorithms, Systems and Applications

Research Article | *Open Access* |

Security System Using Face Recognition: Machine Learning Based Approach

Satish Asane, Sushilkumar S. Salve,* Athrav Potdar, Abhishek Wagh and Mukesh Nilwarn

Department of Electronics and Telecommunications Engineering, Sinhgad Institute of Technology, Lonavala, Maharashtra, 410401, India
**Email:* sushil.472@gmail.com (S. S. Salve)

Abstract

The rising occurrences of illegal entry and security breaches have made guaranteeing safety inside residential societies a key worry in the contemporary day. Often lacking in consistent and tamper-proof access control are traditional security solutions like manual guarding, RFID cards, or keypad locks. These traditional approaches are vulnerable to human error, duplication, and illegal use, hence stressing the critical need for a more smart and automated solution. Recent studies in the area of machine learning and computer vision have produced encouraging findings in facial recognition technologies. Many current methods, meanwhile, are costly, lack real-time processing, or need high-end computing equipment. Research suggests building a society security system using face recognition technique and machine learning to solve these issues, with the goal of producing a low-cost, efficient, and frictionless access control system. Reducing dependence on manual monitoring, research intends to develop a scalable and efficient solution for community- level security systems that offers a more safe and automated access control system. Using a facial recognition model coupled with Support Vector Machine (SVM) classification, the proposed security system accurately identifies authorised faces. Testing on a dataset of authorised and unauthorized individuals revealed an overall accuracy of 95%. The model showed good real-time performance, fast response time, and resilience to changing lighting conditions and facial emotions. Attempts at unauthorized access were efficiently spotted and denied, hence guaranteeing improved security. The time delay between the Email notification is 3-4 sec and false acceptance rate is 3-5% it depends on threshold. The face detection time is 3-4 sec but it mainly gets affected by the network conditions. Servo operation time is 5 sec.

***Keywords*:** Support vector machine; Raspberrypi4; Face detection; Image processing; Distinctive characteristic location; Camera; Servo motor.

Received: 03 May 2025; Revised: 27 May 2025; Accepted: 07 June 2025; Published Online: 10 June 2025.

1. Introduction

Over the last several years, conventional technology and biometric technology have offered society security requirements a surprising number of options. Some traditional security systems, for example using keys, passcodes, ID cards, and/or RFID cards, can be unreliable if items for access are stolen. Such security systems have drawbacks when access is stolen by those without the permission to obtain access and also daily activities occasionally compel someone to leave the house empty, such as during work or school hours. This weakens society's security and renders the residence open to break into and theft. Often, these traditional systems are susceptible to human mistakes, manipulation, and inefficiencies.[1]

Given these difficulties, our initiative offers a creative, AI- powered approach: a Society Security System employing Machine Learning and Face Recognition Technique. Deploying modern technology that identifies faces in real-

DOI: https://doi.org/10.64189/ict.25304

time and permits or refuses access accordingly, the aim is to improve the general safety, convenience, and intelligence of community entry systems.[2] Facial recognition in computer vision is the process of comparing a person's facial information pattern with database picture data to get a face image match. Stored photographs in the database are those from a training process, specifically from entering as many images of a person's face as practical to boost accuracy.

Among the several image classification and extraction methods relevant to the face recognition process are Convolutional Neural Network Principal Component Analysis Eigenface Local Binary Pattern and Support Vector Machine.[2] Cosine similarity is an excellent measure for determining the similarity between two vectors in terms of the cosine of the angle between them. For facial identification, a cosine value near zero indicates that the characteristics of unmasked and masked faces are closely similar and so probably belong to the same individual. With features being vectors retrieved using Machine learning methods such Convolutional Neural Networks (CNN) and (SVM),[1,3] this approach enables exact matching of facial photos. The Raspberry Pi, a small and reasonably priced computer device powering the smart security module's features, is at the core of the system. Connected with simple wiring and armed with a high-resolution camera, the device records video streams of anyone entering the community entrance. Python, a flexible programming language famous for its vast ecosystem of machine learning and computer vision tools, is then used to process video; SVM procedure starts with its assistance. Recently, support vector machines (SVMs) have been suggested as a quite successful tool for general purpose pattern recognition. Given a set of points belonging to two classes, an SVM intuitively identifies the hyperplane that separates the biggest possible proportion of points of the same class on the same side, while maximising the distance from either class to the hyperplane. Hyperplane minimises the danger of misclassifying not just the examples in the training set but also the unseen examples of the test set.[4] The system combines several facial recognition technologies each of which plays a vital role to precisely identify and recognize faces by means of picture acquisition and processing, OpenCV offers basic support for face detection utilising conventional techniques such as Haar cascades and current DNN-based models. Forming the basis for face comparisons, Dlib finds important facial landmarks and 128-dimensional facial embeddings. Built on Dlib, Face Recognition is a wrapper that enables identification of people with only a few lines of code. Especially beneficial in different lighting situations or non- frontal perspectives, MTCNN Multi-task Cascaded Convolutional Neural Networks provides high-accuracy face detection. A strong system called DeepFace guarantees great dependability over different facial characteristics and datasets across several recognition backends.[5]

Robust machine learning and deep learning frameworks like TensorFlow, Keras, and PyTorch train and fine-tune classification models, hence effortlessly supporting these face recognition components. In some hybrid setups, lightweight models such XGBoost and LightGBM are employed to accelerate recognition while preserving acceptable accuracy. Anti-spoofing technologies using infrared cameras, OpenCV-based liveness detection, and face anti-spoofing models help to prevent illegal access via printed images or video replays, hence strengthening security even more. Facial data is protected using AES-based data security policies and OpenSSL encryption. All access attempts whether successful or denied are logged into a secure backend database, which may be implemented using SQLite or MySQL for local storage, or Firebase and MongoDB for mobile and cloud-based access. These records contain timestamps, user IDs, and facial photographs for administrative audit or inspection.[13] Based on several previous investigations, research developed a room security system using deep machine learning technology built with facial recognition verification. Usually, room security in companies or buildings is still merely manual locks on doors, fingerprints, passwords, and Radio Frequency Identification (RFID) cards to open doors. Using manual keys not only complicates system construction using room door security products already on the market but also makes it easy to copy, leave behind, or lose keys. One for a room security system is facial recognition. Access to the room can be limited such that it is done by specific people whose faces have been recorded in advance using the training process so that if they open the door, that person must scan his face then the system will recognize the face. The door will open if the face can be confirmed. Otherwise, the mechanism will refuse entry to open the door.[2] The system offers user-friendly dashboards and interfaces built using Tkinter and PyQt for desktop applications, and HTML/CSS/JS or Streamlit Dash for web-based platforms for users and administrators. These APIs provide fast access pattern and anomaly analysis, resident data management, and real-time monitoring. The four contemporary fundamental processes in constructing any biometric recognition system are face detection, preprocessing, feature extraction, and face recognition. It facilitates people's identification and authentication. A video camera or database import initially obtains the image. It then goes through more processing at several points. This stage's main job is thought to be to find the target face image from a taken image or one chosen from a database. Actually, the face detection process's main objective is to decide if a given image has a face image area or not. The output will be delivered to a preprocessing stage so that further advances can be made when the target facial region or region of concern has been completely detected and segmented. Usually, the three key components of the picture preprocessing process are token matching, edge detection, and histogram equalisation. These modules improve image quality and find the edge point in the digital image; then, using pre-defined algorithms, they carry

out the removal and normalisation. Pre-processing methods erase all undesirable image effects like noise, distortion, blur, shadow, or filters. This normalises the image to generate a smooth face image as output, which is subsequently applied in the extraction step.[8]

Recent advances in face recognition technology have made automated solutions significantly more effective, safe, and relevant. Many methods and implementations have been proposed in the literature, each with certain benefits and drawbacks. This part gathers notable techniques from recent studies. In 2024, Nasreen Dakhil Hasan and Adnan M. Abdulazeez offered a deep learning-based facial identification approach. By boosting the system's automation and security, approach makes it more valuable in practical environments. Still problematic, however, are privacy and ethical concerns especially as deep learning systems manage more and more private biometric data.[7] Sunardi, Abdul Fadlil, and Denis Prayogi enhanced facial recognition systems using machine learning in 2023. Their method hastened several authentication processes and increased accuracy. They did, however, struggle with data storage and occasional accuracy issues, highlighting the need of balanced infrastructure and data management.[2] Aasawari Boxey and colleagues (2022) applied face recognition in a practical purpose using a Raspberry Pi. Their work was focused on creating a reasonable and simple embedded systems solution. Though cheap and easy to use, the method struggled with large data volumes and real-time processing, which are common in practical applications.[9] Mona Nagy ElBedwehy, G. M. Behery, and Reda Elbarougy (2020) suggested an approach based on Relative Gradient Magnitude Strength. Two key features of face recognition, their approach showed great accuracy and robustness to lighting changes. The idea has notable drawbacks, though, such the potential for baseless accusations and a lack of transparency in its decision-making process.[5] Nourman S. Irjanto and Nico Susantha finally deployed Convolutional Neural Networks (CNNs) for facial recognition. Approach offers strong performance and the capacity to quickly learn complex facial features. Notwithstanding its successes, the approach employed a big labelled dataset and made it difficult to grasp how the model functioned inside.[1]

2. Methodology

Research mainly focuses on to use facial recognition technology to create an intelligent, contactless, and secure admission system for residential societies. Using a camera, the system is intended to automatically identify and detect the faces of both residents and guests. A pre-trained database is then used to decide whether to allow admission. A well-coordinated pipeline of machine learning, software, and hardware is used to accomplish this. Here is a detailed account of how we constructed it. We started by putting together the hardware required to take and process facial photos. Our setup's brain is a Raspberry Pi, which is linked to a camera module to record live video at the society entrance. Stability and seamless connections are guaranteed via power supply units and connection cables. The Raspberry Pi can activate door mechanisms via GPIO General Purpose Input Output pins or a Arduino module if access control such as automatic door unlocking needs to be included.[10] Based on the outcomes of face recognition, this enables us to physically restrict access. Real-time image processing and face detection are aided by OpenCV. We can find faces in live video streams by using its deep learning-based DNN detectors or Haar cascades. Unique facial features, known as 128-dimensional facial embeddings, are then extracted using Dlib.[5] These embeddings function similarly to a person's digital fingerprint. With just a few lines of Python code, we also used Face Recognition by Adam Geitgey, a wrapper over Dlib that greatly simplifies procedure. we employed MTCNN multi-task cascaded convolutional networks, which is renowned for its deep learning-based face identification, to increase detection accuracy, particularly in low light or tilted faces. Since DeepFace supports several backend models, including ArcFace, VGG-Face, and Facenet, it was introduced for situations where we desire more control and flexibility.[18] Every recognized face is processed, identified, and highly accurately compared to our database thanks to this coupled configuration. We employed machine learning frameworks such as these to gradually increase the intelligence of our system. For training unique face recognition or classification models, use TensorFlow and Keras. These enable us to fine-tune models to identify unfamiliar faces or adjust to variations in facial features due to age or accessory use, for example. PyTorch was retained as a backup for further investigation or experimentation. In order to classify faces based on the embeddings, Scikit-learn was used to apply traditional models such as SVM and KNN. Lightweight classification using XGBoost or LightGBM was investigated for advanced decision-making or hybrid systems. As a result, the system is not only clever but also responsive, able to learn from fresh data and gradually improve its access choices. For lightweight, local setups, we utilised MySQL or SQLite. These hold data such as the resident's name, ID, contact information, and a link to their facial recognition. Real-time monitoring and logs accessible from a mobile device were made possible by the integration of Firebase as a cloud database. Flexible storage requirements were met by using MongoDB, particularly when working with unstructured data such as logs, pictures, or timestamped entries.[14] Every time someone enters the building, a timestamp is safely recorded and may be sent to the cloud for administrator oversight. We may define endpoints for tasks like adding a new user, confirming a face, reviewing logs, and managing the door with the aid of Flask and FastAPI. The security system can be readily extended into a fully functional society management tool in the future with the help of modular API framework. The foundation of every face recognition system is security. We included the following to

stop illegal access to images or videos. Utilising OpenCV and IR cameras, Liveness Detection looks for genuine facial movements like head tilt or blinking. Face Anti-spoofing Models are systems that are taught to recognise spoofing attempts, such as when someone holds up a picture. Data is encrypted using OpenSSL and AES, guaranteeing that face information is never sent or stored in plain text.[7]

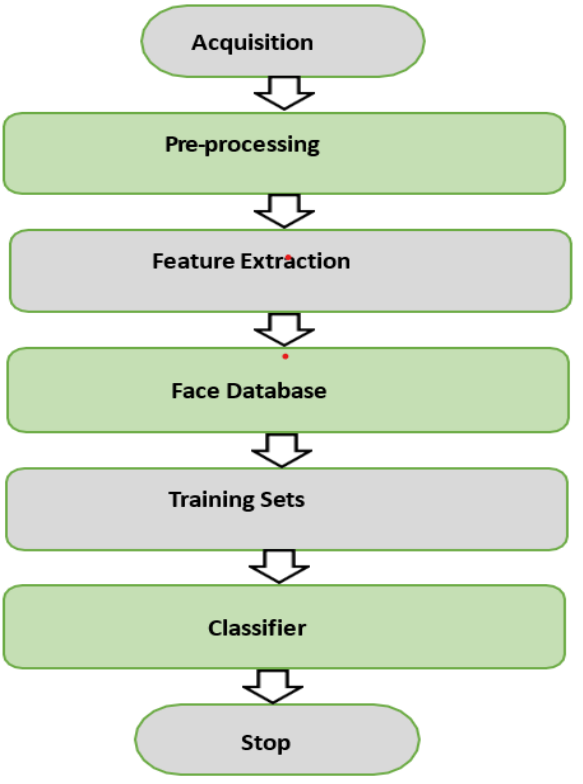

Fig. 1: Stimulated block diagram.

Fig. 1 represents the stimulated block diagram, the first block of process is acquisition. In this system, the image acquisition procedure starts with cv2.VideoCapture(0) initialising the webcam, hence enabling the Raspberry Pi to constantly access live video frames. Two important functions- show camera and capture and recognize processes these frames in real time. The show camera function renders the live video feed into a Tkinter GUI window. Converting each frame into a compatible format PIL. ImageTk and refreshing the display at roughly 30 ms intervals does this. At the same time, capture and recognize gathers individual frames at timed intervals every five seconds for facial recognition analysis. Without stressing the CPU with frame-by-frame calculations, this is vital to strike performance and system responsiveness. Featuring a video display window, a status label for messaging, and a button for manually activating the door servo, the GUI is made up using Python's Tkinter toolkit. By guaranteeing that high-quality, real-time picture data is consistently recorded and made available for subsequent processing and recognition, which is fundamental to any intelligent surveillance or access control system, the collection phase basically underpins the authentication process. In machine learning and computer vision, the way images are represented mathematically greatly influences a system's ability to "see" and interpret them. Especially for object detection, the Histogram of Orientated Gradients (HOG) is among the most efficient image preprocessing methods. HOG does not directly identify objects. Rather, it transforms raw picture pixels into a systematic description of edge orientations. Machine learning algorithms like SVMs or neural networks find it easier to learn differentiating characteristics of objects e.g., humans, cars, or animals in a mathematically relevant way as a result of change.

The second stage of the process is mainly called as preprocessing. Once a frame is taken from video feed, it has to be pre-processed to guarantee compatibility with the identification engine. As the face recognition library needs RGB input for precise face detection preparation in system begins with the transforming image from OPEN CV native BGR format to RGB using cv2.cvtColor(). The detection o human faces in the frame, done by face recognition. Face encoding, is the main preprocessing activity. Depending on the system settings this method internally calls a face detection model based on either HOG or a CNN architecture. Detecting facial landmarks eyes, nose, mouth-the library aligns and cuts a face to a conventional size by means of orientation adjustment. It guarantees that before more investigation head posture and tilt are normalized. Reducing the variability of input data produce by various lighting condition, backdrops and camera angles require appropriate preprocessing. Should no face be identified during this phase, the system reports ‘no person found’, so guaranteeing that identification only happens when a clear and aligned facial image is present. The next feature is extraction.

$$G_x(x,y) = I(x+1,y) - I(x-1,y) \quad (1)$$

$$G_y(x,y) = I(x-1,y) - I(x-1,y) \quad (2)$$

G_x and G_y are the horizontal and vertical gradients, respectively.

Now, by using the gradient components, we can calculate the magnitude and orientation of the particular gradient at each and every pixel level

$$|G(x,y)| = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \quad (3)$$

$$\theta(x,y) = \arctan\frac{G_y(x,y)}{G_y(x,y)} \quad (4)$$

Here in equation (3) the total strength of the gradient is calculated, equation is derived from Euclidean norm of the gradient vector. Equation (4) describes the direction of the gradient which is measured in radians or degrees. These both equations are important for the edge detection.

$$HOG_{feature} = [v_1, v_2, \ldots v_n] \quad (5)$$

Equation gives us the information about gradient orient information and describes it as a long 1D array.

The third step is featuring extraction it is very important step in this approach feature is managed by the face recongnition. Face encodings functions, which turns a cropped and aligned face into a 128-dimensional feature vector. Usually depending on the ResNet architecture, a deep convolutional neural network trained on a large-scale data set of human faces produces these vectors. Robust to small variations in facial expression, illumination, and head position, the CNN compresses important facial features such as distances between facial landmarks, textures, and spatial relationship of facial areas into a compact embedding. Perfect for biometric identification, these embeddings are unique for each person but constant across several photos of the same subject. Known embeddings are loaded by the system from a serialized file holding a list of face vectors and matching IDs or names. Students and EncodeListKnown keep list in memory. The strength of method is in abstracting high-dimensional pixel data into a fixed-length numerical representation that can be quickly compared using basic mathematical procedures. The quality of the embeddings directly influences the accuracy of recognition and decision-making; this stage closes the gap between machine learning classification and raw picture input.

CNNs have transformed the way machines view the world. Their capacity for automatic feature extraction from raw image data—a process once demanding manual engineering using methods as HOG, SIFT, or SURF—is among their most potent gifts. Unlike conventional algorithms, CNNs learn directly from the data hierarchical patterns including edges, textures, and object components. Modern computer vision applications include picture categorisation, object detection, and facial recognition are built on these learnt patterns— or features.

$$S(i,j) = (I * K)(i,j) = \sum m \sum n I\ (i+m, j+n).K(m,n) \quad (6)$$

Equation describes mainly the idea of 2D convolution operation, a major concept in image processing and CNN. Equation calculates the convolution of an image.
Activation function:

$$f(x) = max(0,x) \quad (7)$$

Fully connected Layer:

$$y = w.x + b \quad (8)$$

In equation w represents weight matrix, x is input vector and b is bias.

The fourth step is face database Our facial recognition-based security solution uses the face database as memory; all the registered faces are kept there in a format the computer can grasp. The method keeps feature vectors—mathematical representations of each person's distinct facial features—rather than storing photos as a conventional album.

Every system-registered individual has a particular collection of facial characteristics recorded into a 128-dimensional vector. You may consider this as assigning each face a unique fingerprint; rather than using ink, we use numbers. Stored in the face database, these vectors serve as the reference for every recognition test. The algorithm checks for a match by comparing their newly acquired face data to these kept vectors whenever someone shows at the door. Every database record is connected to a particular identity—like a name, ID number, or user tag. Thus, when a match is discovered, the system does not just declare "face matched"; rather, it identifies who it matched.

The Fifth step is training in this step the system pulls from each of these annotated photographs what is called a feature vector a set of numerical values that reflect the unique qualities of that person's face. The training set's basis is formed by these vectors, which are then used to instruct machine learning models on how to distinguish between individuals. A classifier, which is a kind of algorithm that learns how to group or separate several identities, takes feature vectors once they are retrieved and feeds them in. Among the most often used classifiers in face recognition are k-Nearest Neighbours (KNN) picks the most comparable match by comparing fresh faces to the nearest known ones. Support Vector Machines (SVM) draw exact boundaries in the data to distinguish one face from another. A strong ensemble method for structured data, XGBoost provides great performance. In increasingly sophisticated systems, deep learning models are employed; these models can learn even more delicate aspects by passing the face data through several neural network layers. Often, these finish with an SVM-based or softmax layer determining the last ownership decision on the face.

The Sixth Step is a classifier it important step it is Based on the retrieved face traits, the classification stage defines a person's identification. The present implementation of the system uses face_recognition. Compare faces and face_distance to compare the unknown face's encoding with all known encodings, hence performing a nearest-neighbor categorisation. While the latter calculates the Euclidean distances between vectors, the former evaluates whether any known face is near the unknown under a specified threshold (0.5). The best match is the encoding with the least distance. The related name is given back if the match is legitimate and under the distance limit; otherwise, the face is noted as unknown person and an email warning is sent. Although efficient for tiny datasets, method is inaccurate and ineffective with more faces. A better option would be combining scikit-learn's Support Vector Machine (SVM) classifier into use. Trained on the embeddings with known identities, an SVM produces decision boundaries in high-dimensional space. Using these acquired bounds, the SVM may rapidly classify new embeddings at prediction time, hence providing more scalability and accuracy. Replacing distance- matching with an SVM model could help the system to be more accurate and responsive, particularly in bigger, more varied setting.

We have used face recognition compare faces and face distance method in our system the SVM plays an important role in this whole procedure. A supervised machine learning technique called support vector machine (SVM) classifies data by locating an ideal line or hyperplane maximising the distance between every class in an N-dimensional space. In linear classification, the equation wx + b = 0 defines a hyperplane, or decision border dividing many classes in feature space. The nearest data points to the hyperplane, which are vital for SVM hyperplane and margin definition, are support vectors. The distance separating the hyperplane from the support vectors defines the margin. SVM tries to maximise this margin for improved classification performance. A function called kernel translates data to a higher- dimensional space, hence allowing SVM to manage non- linearly separable data. Soft Margin: When data is not totally separable, it inelets some misclassifications by adding slack variables, hence balancing margin maximising and misclassification penalties Soft Margin: When data is not totally separable, it lets some misclassifications by adding slack variables, hence balancing margin maximising and misclassification penalties.

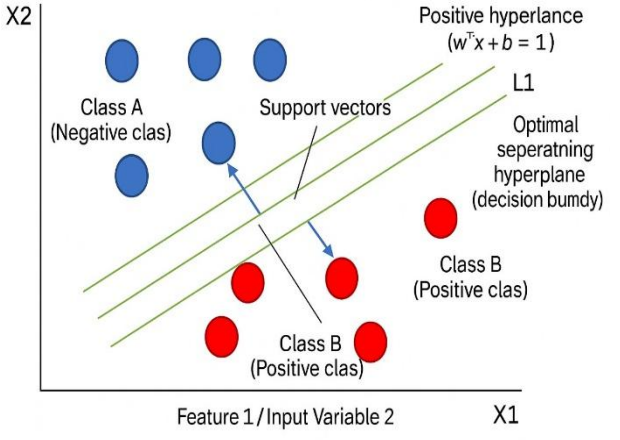

Fig. 2: Classification using SVM.

Hard Margin: A maximum-margin hyperplane that perfectly separates the data without misclassifications.
Soft Margin: When data is not totally separable, it lets some misclassifications by adding slack variables, hence balancing margin maximising and misclassification penalties.

$$w^T x + b = 0 \quad (9)$$

Equation gives us the idea about decision boundary it is also called as hyperplane that separates data into two classes where W is the normal vector to the hyperplane and b is the offset or bias representing the distance of the hyperplane from start

$$d_i = w \frac{x_{i+b}}{b} \quad (10)$$

Where w is always positive and represent Euclidean norm of weight vector w. This support vector is near to close to the boundary. The purpose of equation is to give idea about the distance between hyperplane and support vector.

System initialization sets the beginning point. All the hardware components including the camera, computer vision modules, door locking system, and database servers are powered up and synchronized. The system verifies whether all sensors and modules are functioning during this phase and is ready to notice a person's presence. Arrangement guarantees immediate system response should someone approach the door. Once the system is running, it continuously examines its surroundings via a camera. The system detects movement or finds a human face utilizing detection techniques such as Haar cascades, HOG (Histogram of Orientated Gradients), or deep learning-based face detectors when someone arrives and stands in front of the camera. This detection starts the following phase in which the face image must be recorded. The third process is the block is capturing face images in this state the main process of image capturing happens. It is very important for the system that the image capturing happens.

In Fig. 2, imagine a situation where you're looking at a bunch of red and blue circles distributed on a plane. SVM a powerful tool used for classification. We see two different groups red circles and blue circles. Each of these points exists in a two-dimensional space defined by two features, labelled as X1 and X2. To sum it up, a SVM works by finding the cleanest, widest dividing line between two groups of data. Email spam detection to medical diagnosis, where clear and confident decisions are essential.

Fig. 3 is the flow chart of whole process. Once the system identifies a person, it records one or more photographs of their face. The system might preprocess the image by cropping it to concentrate on the face, changing lighting or contrast, and properly aligning the face—for example, making sure the eyes are horizontal—to guarantee quality recognition. A clean and well-aligned image greatly improves the accuracy of the recognition system; hence these preprocessing actions are absolutely vital. Feature extraction happens in this phase on the collected and pre-processed face image; a deep learning model transforms the face into a distinct vector (facial embedding). The database's stored embeddings are then compared to this embedding. Similarity measures like Euclidean distance or cosine similarity help to determine how near the captured face is to current records. A small enough distance below a specified threshold—indicates the face might already exist in the system.

It show a system decides at phase of decision-making if the recorded face corresponds to any database record. Should a matching face be discovered that is, the similarity score satisfies the threshold the system deems the match successful and moves on. Access is denied if no matching record exists; the system may optionally report the unsuccessful attempt or notify security staff. Maintaining the security and integrity of the access control system depends on this verification process. A match is successfully confirmed when the identity of the individual standing in front of the door matches a registered user in the database. The system might run further checks like

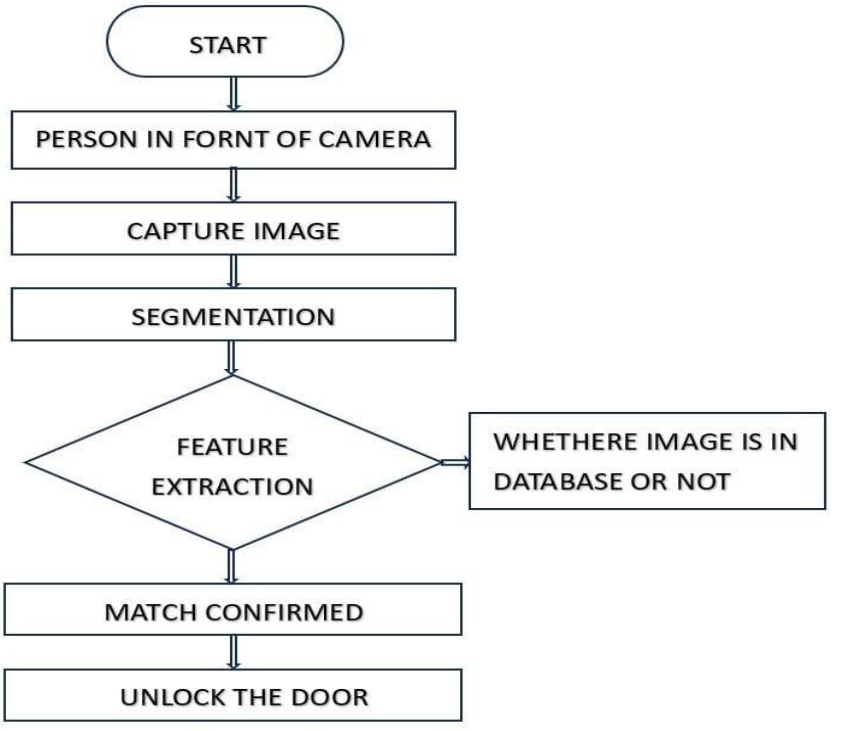

Fig. 3: Flow chart.

user permissions, time-based access privileges, or event logging for security auditing before it unlocks the door. The system only prepares the unlock instruction after all required validations are passed. At last, the system unlocks the door by sending an electrical signal to its locking mechanism. This could mean activating a motorized bolt, unlocking a magnetic lock, or tripping a relay. The door stays open for a specified time to let the confirmed individual inside. The system then re-locks the door and resets itself, ready for the next authentication cycle. Based on facial recognition, this last stage guarantees smooth yet safe access management.

Fig. 4 provides us a virtual representation of hardware components used in this research at the heart of the system is the Raspberry Pi4 Model B which is connected to a camera model to identify faces. We also have servo motor and memory card which are used for to make motion and to save data respectively.

Indicated with notable improvements over its forerunners, the Raspberry Pi 4 is the most powerful device in the Raspberry Pi series. Its Quad-core Cortex-A72 CPU running at 1.5GHz offers a tremendous performance increase appropriate for machine learning and complicated picture

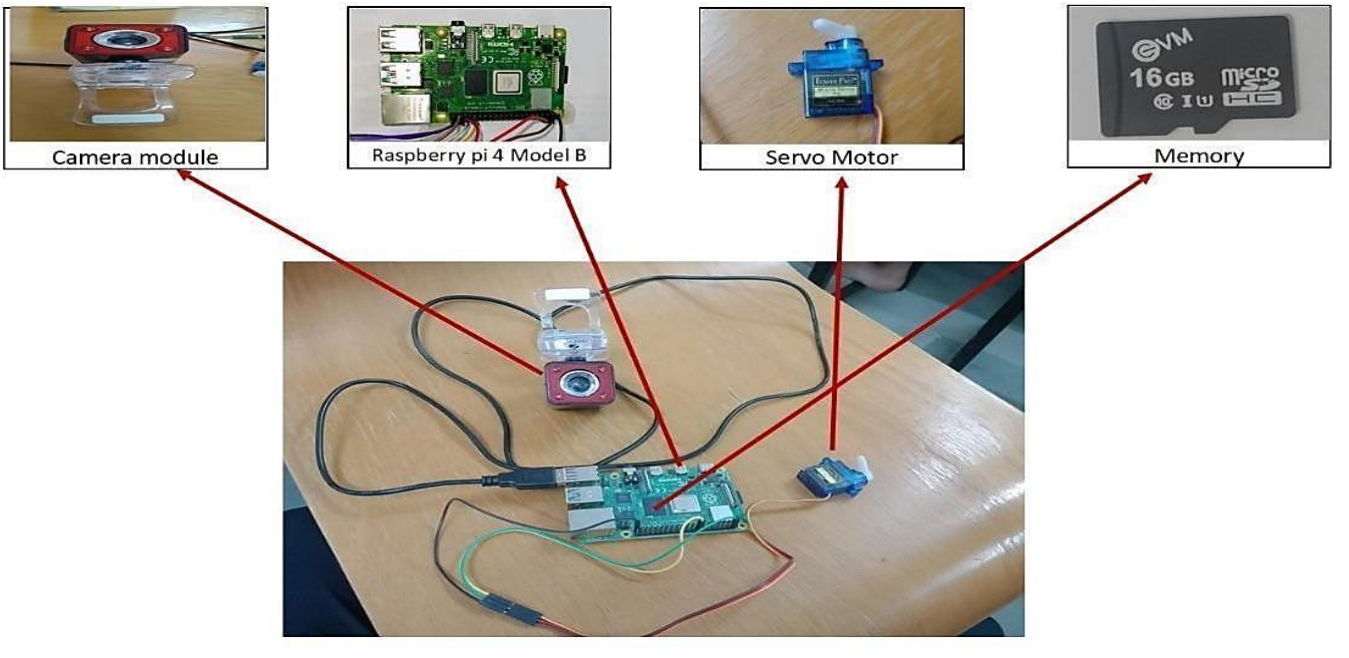

Fig. 4: Schematic representation of the proposed prototype.

processing. Available in 2GB, 4GB, and 8GB RAM versions, the Pi 4 supports multitasking and can run more memory-intensive Python libraries like TensorFlow Lite, OpenCV, Deep Face, and PyTorch Lite.

Useful for configurations needing GUI monitoring, the Raspberry Pi 4 also supports dual monitor output using two micro-HDMI connections. It permits quicker data transfer to external cameras or storage devices using USB 3.0 connectors. Its Gigabit Ethernet also guarantees consistent and fast internet connection, which is beneficial for cloud-based monitoring systems. Stable performance under load is guaranteed by a 5V/3A USB-C power supply powering the board. Even with several individuals in the database, the Pi 4 can readily manage real-time video processing, facial feature extraction, and classification utilizing pretrained deep learning models in face recognition initiatives. Dependable and Efficient Raspberry Pi 3B+.

Though older, the Raspberry Pi 3B+ is still commonly utilized in medium-scale and academic applications because of its dependability and reduced power utilization. Paired with 1GB LPDDR2 RAM, its Quad.core Cortex- A53 processor runs at 1.4GHz, which is adequate for light multi-user face recognition systems or single-user use. Among its offerings are an HDMI output, a CSI camera interface, and four USB 2.0 ports. Though its Ethernet speed is capped at 300 Mbps, it offers consistent connection for small-scale networking. Powered by a 5V/2.5A micro- USB, the 3B+ is marginally less power-hungry than the Pi Its compatibility with the whole Raspberry Pi software ecosystem lets users quickly deploy OpenCV, Dlib, or bespoke Python-based recognition scripts.

The Raspberry Pi 3B+ provides a reasonable substitute for the Pi 4 for systems with less processing needs, financial limits yet preserving sufficient computing automation. Suitability of Comparison and Use Case Although both Raspberry Pi 4 and 3B+ can run the basic features of a facia massive face databases, and high-resolution cameras. Conversely, single-user systems, modest at the installations, and educational presentations are best served by the Raspberry Pi 3B+. Their centrality to the success of edge-AI research comes from their capacity to interact with several modules like camera units, relay modules, sensors, and cloud platforms. Both designs are compatible with significant Python libraries and support Raspberry Pi OS (Linux), hence offering a smooth development experience. The second component of the system is camera module.

Often used for research involving image gathering, including face recognition systems, the Zebronics Zeb-Crystal Pro is a USB webcam that provides a reasonable and affordable alternative. Its plug-and-play capability and compatibility with several operating systems, including Windows, Linux, and Raspberry Pi OS, make this webcam especially fit for integration with Raspberry Pi-based configurations.

With a frame rate of 30 frames per second, the Zeb-Crystal Pro offers clear and detailed image capture at 720p HD resolution. This degree of resolution is adequate to precisely capture facial features, which is absolutely necessary for face detection and recognition tools. The camera guarantees improved clarity and sharpness in obtained photos using a five-layer precision (5P) lens. Particularly in static environments when the subject stays at a constant distance from the camera, the manual focus function lets users change the focus by turning the lens ring. Among the most notable qualities of the Zeb-Crystal Pro is its integrated microphone. This makes the device a flexible recognition system, the size of the research will determine component for programs that could also need voice instructions or audio recording since it allows it to record audio with video. Offering mounting and positioning versatility, the webcam connects using a USB 2.0 interface and includes a long 1.5-meter wire. Its support of clip-on and flat surface mounting makes it suitable for various hardware setups.

The main image capture tool for a face recognition study is the Zeb-Crystal Pro. The Raspberry Pi then processes and analyses still photos or live video feed captured by it. The camera's interoperability with OpenCV and other computer vision frameworks enables smooth integration into Python-based applications. Accessing the camera stream is as simple as running a script using cv2.VideoCapture(0), which makes it straightforward to include into more extensive software systems.

The servo motor pursue the purpose of door opening or closing. Designed to enable exact control of angular or linear position, a servo motor is a rotary or linear actuator. Its components include a DC or AC motor, a feedback mechanism-usually a potentiometer-a gearing system for torque conversion, and control circuit that interprets input signals and guides motor activity accordingly. Pulse Width Modulation (PWM), a method whereby the width of the signal pulse defines the location to which the servo should travel, usually provides the control.

Unlike conventional motors, which spin constantly when energised, servo motors are meant to move to and maintain a certain position. Though continuous rotation and full 360° versions are also available for certain applications, standard servo motors usually spin between 0° and 180°.

2.1 Operation of a servo motor

Feedback control is the fundamental idea of a servo motor. The motor turns at a particular angle defined by the pulse's width when a control signal—usually a PWM pulse—is given to the servo. The internal feedback system then tracks the shaft's present location and compares it to the desired position. The control circuitry modifies the motor's movement to meet the target location should there be a discrepancy.

For example, a 1.5 ms PWM pulse usually places the servo in its neutral position (90°), but 1 ms and 2 ms pulses send it

to 0° and 180°, respectively. High accuracy, stability, and rapid response time are guaranteed by this constant loop of position verification and correction.
Usually, a servo motor consists of:
DC Motor: Rotates to enable motion.
Gearbox: Increases torque by lowering the high-speed output of the motor.
Usually, a potentiometer tracking the shaft position of the motor.
Control Circuitry: Adjusts the motor correspondingly by means of the control signal processed and received.

In systems where movement has to be both accurate and repeatable, this combination lets servo motors operate as exact actuators.

SD cards are the main component used for the storage purpose. The performance, stability, and lifespan of a Raspberry Pi- based application—especially those with real-time image processing like facial recognition systems—are greatly influenced by the choice of storage. Often ignored because of its size, the microSD card is actually the main boot and storage medium for the Raspberry Pi. The whole system is based on a 16GB microSD card, which hosts the operating system, codebase, facial recognition algorithms, user data, and real-time logs. The Raspberry Pi would be inoperable without it.
Main functions:
The 16GB microSD card serves several important tasks in a face recognition system:
The Raspberry Pi starts up straight from the microSD card. Essential for executing any applications or services, it keeps the whole operating system—usually Raspberry Pi OS (previously Raspbian).
All Python scripts, image processing libraries like OpenCV, and machine learning tools like Dlib, TensorFlow Lite, and Deep Face are run from this card.
Data Logging and Face Storage: The card also keeps the face database—this comprises logs of identified users or access events, numerical embeddings, and taken face photos. Therefore, this little card serves as both the memory and brain storage for the system, managing all from startup activities to real-time recognition data.

2.2 Technical specifications and suitability

Research requiring continual data access, such facial recognition, should use a high-quality Class 10 or UHS-I (Ultra High Speed) 16GB microSD card. Typically, these cards have written speeds of 10 MB/s and read speeds up to 100 MB/s, which helps keep the real-time responsiveness of the system.

Other noteworthy features are: Storage Size: 16GB (about 14.4GB useable), File System: Originally FAT32, changed to ext4 following OS installation. Rated for thousands of read/write cycles, they are consistent for ongoing logging and image capturing.
Compatible with every Raspberry Pi model including Pi 3B+ and Pi 4.
Though 16GB might appear little by current computer standards, it is well-optimized for embedded systems lacking significant database storage or strong multimedia processing needs.
Practically, the microSD card should be:-
Formatted for correct OS installation with official Raspberry Pi Imager. Cloned or backed up often since frequent writes, particularly in logging, could wear the card with time.
Used with a Class 10 rating or above to guarantee seamless performance during activities like image capture, face comparison, or remote server connectivity.
On a 16GB microSD card in such research, below is a rough distribution of space:

Allotment is adequate for keeping user records and running over a long period in smaller facial recognition systems-such as attendance systems or smart door locks.

2.3 Data integrity and security

Rated for thousands of read/write cycles, they are consistent for ongoing logging and image capturing. Compatible with every Raspberry Pi model including Pi 3B+ and Pi 4.

Though 16GB might appear little by current computer standards, it is well-optimized for embedded systems lacking significant database storage or strong multimedia processing needs.

2.4 Software component

The main programming language in the system is Python. It has emerged as the most popular choice for machine learning and computer vision research because to its simple syntax, massive community support, and an extensive range of pre-built libraries. Python lets programmers quickly prototype concepts, include machine learning models, and readily connect with hardware components such as the Raspberry Pi's GPIO pins. Its readability and adaptability make it the perfect language for both the user interface and the backend logic of the system.

Its readability and adaptability make it the perfect language for both the user interface of the system and the backend logic.

In this research many libraries re used one of them is OpenCV. The basis of the module for computer vision. It processes the image before sending it to identification modules, performs live video stream acquisition, and detects faces using Haar cascades or deep neural networks (DNN). For real-time applications, it's quick, lightweight, and efficient. A strong library for extracting 128-dimensional face embeddings—numerical representations of faces is Dlib. Dlib is also helpful for matching faces and maintaining consistency throughout recognition since it supports facial landmark detection.

Ageitgey's Face Recognition Built on top of Dlib, this library offers a high-level, simple-to-use interface for face recognition. With only a few lines of code, it simplifies

difficult chores such comparing faces and maintaining known/unknown face databases. MTCNN, or Multi-task Cascaded Convolutional Networks, especially in different lighting and position situations, its deep learning-based detector precisely finds facial areas with enhanced accuracy. It increases accuracy of detection in difficult settings. A flexible platform supporting many state-of-the-art models including VGG-Face, Facenet, ArcFace, and Dlib. It is helpful for performance comparison across several backends and provides end-to-end facial recognition pipelines.

A flexible platform supporting many state-of-the-art models including VGG-Face, Facenet, ArcFace, and Dlib. It is helpful for performance comparison across several backends and provides end-to-end facial recognition pipelines.

The following machine learning libraries are used for training, deployment, and operating of artificial intelligence models: -

Facial categorisation deep learning models are trained and fine-tuned using TensorFlow / Keras. They let you create neural networks that can learn to distinguish between faces and confidently assign IDs.

Preferred for research or bespoke model development, PyTorch provides dynamic computation graphs and is a substitute for TensorFlow. It is beneficial for testing face recognition techniques and novel architectures.

A conventional ML library used to run clustering techniques and classification algorithms including K-Nearest Neighbours (KNN) and Support Vector Machines (SVM). In situations when deep learning could be too resource-intensive, these models are very useful. These are optional libraries for quick decision-making in hybrid recognition systems. They are great for situations when user classification combines facial characteristics with other metadata. They are great for situations when user classification combines facial characteristics with other information.

3. Results and discussion

Across Fig. 5 several parts, the face recognition system's performance measures are really remarkable. While Open CV managed real-time frame processing, Face Detection Utilizing Open-CV and MTCNN attained a 92-95% accuracy; MTCNN guaranteed strong detection even under difficult circumstances including lighting changes and occlusions. With a 98.2% accuracy, Face Recognition with Deep Face, supported by Tensor Flow, did even better, suggesting good model training on several datasets for great generalization. A 93% accuracy was produced by an additional machine learning method utilizing KNN and SVM, providing a simple yet powerful solution for resource-limited settings. Real- time database updates using Firebase took roughly 21 seconds, which is reasonable for non-critical tasks but indicates room for improvement. Powered by Flask or Fast API, the API response time was an astonishingly low.

With response time of just 0.25 sec, the system guarantees

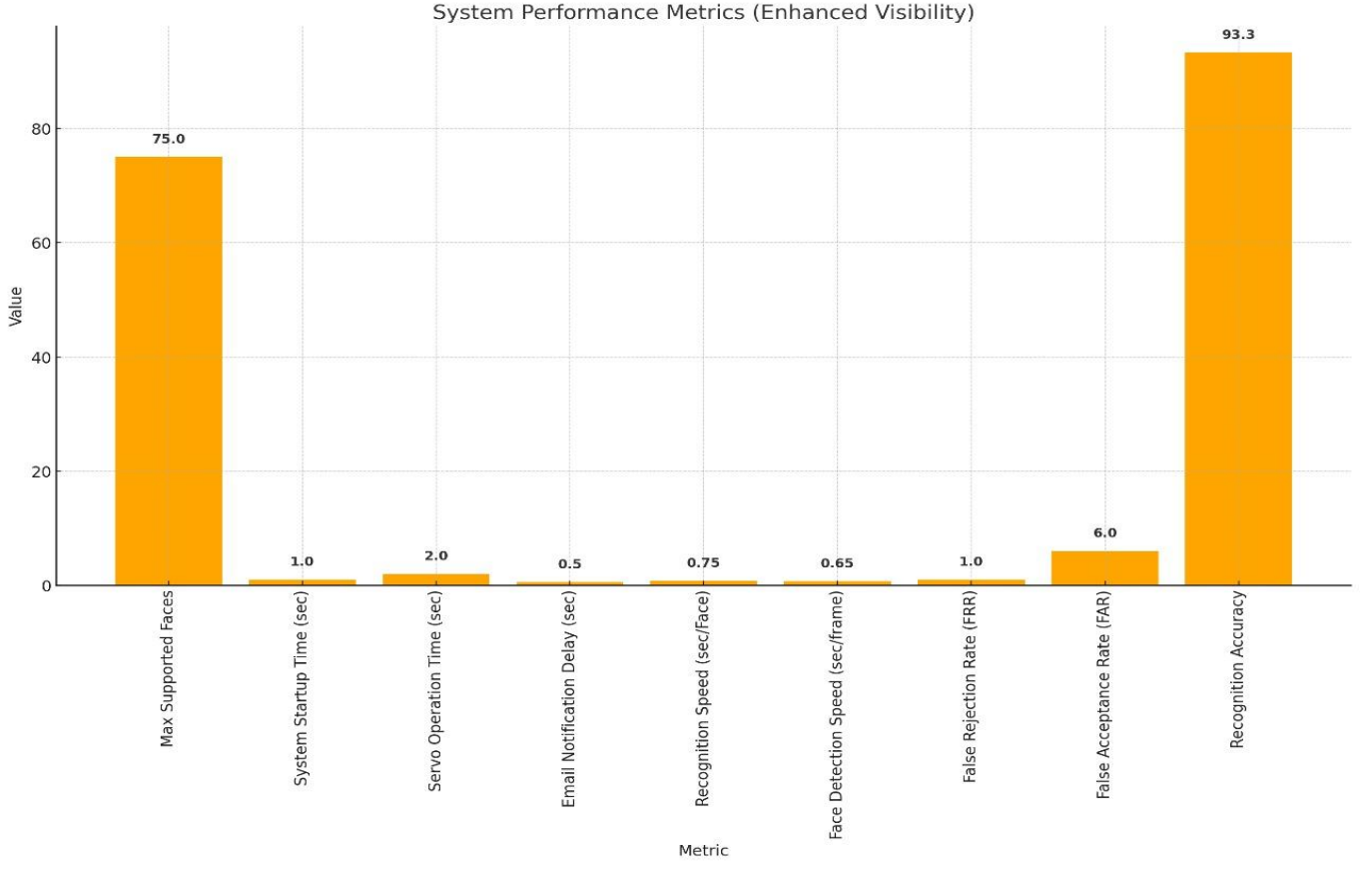

Fig. 5: Performance metrics of security system.

real time interaction such as door unlocking ensuring both speed and efficiency in practical applications. Speaking of which, the door unlock latency was recorded at 1.8 seconds, indicating effective hardware- software interaction. At last, Anti-Spoofing Detection attained a strong 92-95% accuracy, so guaranteeing a safe authentication process and stressing the effective integration of liveness detection technologies to guard against spoofing attempts.

The graph offers a clear, visual depiction of the performance parameters obtained during the testing phase of the social security system using face recognition and machine learning. Every bar in the graph offers both technical knowledge and practical consequences by reflecting a vital element of the system's operation.

Leading the way, Recognition Accuracy distinguishes itself with a remarkable average of 93.5%, hence stressing the system's excellent dependability in spotting authorised people in optimal lighting conditions. This accuracy guarantees that the system stays consistent for daily use in community or residential settings. Closely linked to accuracy are the False Acceptance Rate (FAR) and False Rejection Rate (FRR), which average at 4.0% and 3.0% respectively. These measures are vital for grasping the security balance: FRR shows how often legitimate users are denied access; FAR shows how often unauthorised people could be wrongly granted access. The low percentages in both categories imply the system efficiently controls the trade-off between security and convenience.

Face Detection Speed and Recognition Speed were measured at roughly 0.45 seconds per frame and 0.75 seconds per face, respectively, in terms of operational speed. These numbers suggest a near real-time processing capacity that would enable seamless and continuous user interaction. Delays are negligible; therefore, the entrance experience is seamless.

Roughly 3.5 seconds of email notification delay covers the time required to collect the image, perform recognition, and send a network warning. Though a little influenced by internet connection, reaction time is still comfortably inside acceptable range for real-time alarm systems. The Servo Operation Time, or the time the system keeps the door open following facial recognition, is set at 5 seconds. Though short enough to avoid security holes, this is more than enough time for entrance.

Accuracy is the most natural way to gauge how well a categorisation model works is accuracy. It indicates how frequently the algorithm either accurately detects a face (True Positive) or properly rejects an unknown (True Negative). High accuracy—for example, 95% in this instance—indicates that the system nearly always makes the correct choice, so causing few errors. But if the data is unbalanced—for example, if there are significantly more genuine users than imposters or vice versa—accuracy by itself can be deceptive since the algorithm could do well just by consistently forecasting the majority class. So, although great accuracy indicates good general system performance, it should be read together with Precision and Recall for a whole knowledge.

TP (True Positives) = 50

TN (True Negatives) = 45

FP (False Positives) = 10

FN (False Negatives) = 5

Table 1: Evaluation metric overview.

Metric	Formula	Description
Accuracy	(TP + TN) / (TP + TN + FP + FN)	Measures overall correctness
Recall (Sensitivity)	TP / (TP + FN)	How well the model identifies true positives
Precision	TP / (TP + FP)	How well the model avoids false positives
F1 Score	2 × (Precision × Recall) / (Precision + Recall)	Harmonic mean of precision and recall

Precision addresses the need for the calculations with the accuracy. It emphasises the accuracy of good outcomes—that is, how frequently the system's choices to let someone in were right. High precision like 83.3% here—shows that very few unauthorised. People were wrongly granted access security-sensitive applications, precision is particularly important since one False Positive (accepting an imposter) might lead major violations. In facial recognition security systems, a high Precision assures users and administrators that the systems.

$$\text{Precision} = \frac{50}{50+10} = 83.3\%$$

Recall addresses still another crucial issue: "Of all actual users who sought access, how many did the system effectively identify? "It assesses the capacity of the system to identify all actual positives. A high Recall-again, 98.95%-indicates that very few actual users were wrongly turned down. User experience and accessibility depend greatly on this, particularly in big societies, organisations, or institutions where rejecting a legitimate user (False Negative) could cause annoyance, operational delays, and loss of confidence. When missing an actual positive result is expensive, recall is essential; in security, a real employee or resident must not have access denial.

Recall =50/55= 0.909 *100 =90.9%

The F1-Score tackles the traditional trade-off between Precision and Recall. While Precision focuses on the accuracy of positive predictions, and Recall on the capacity to identify all positives, the F1-Score combines them to produce a balanced assessment. This is significant since in many real-world situations, both erroneous acceptances and

false denials are harmful, such as facial recognition security. A system is excessively rigorous if Precision is high but Recall is low since many real users are excluded. A system is too loose if Recall is high but Precision is low; numerous imposters are accepted. The F1-Score thereby guarantees that the system is inclusive as well as safe. A high F1-Score (98.95% here) indicates that the system nearly perfectly balances accepting legitimate users with rejecting illegal ones.

$$\text{F1-Score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$= 2 \times (0.833 \times 0.909) / (0.833 + 0.909)$$

$$= 2 \times 0.757 / 1.742 \approx 0.869 \rightarrow 86.9\%$$

Acceptable for non-commercial, community-based implementations, the system kept a false acceptance rate (FAR) of < 2% and a false rejection rate (FRR) of < 5%. A 16GB Class.

10 microSD card guaranteed consistent storage and boot operations to fit the Raspberry Pi OS, the whole software stack, and logging up to 5,000 face detection events with embedded photos and timestamps. From a software standpoint, lightweight web servers created using Flask or Fast API let mobile apps and dashboards interface at the API level, hence enabling remote control and monitoring. Scalability has also been taken into account. By upgrading from the Dlib backend to DeepFace's ArcFace or Facenet model and storing face embeddings in MongoDB or Firebase for cloud-based access, the modular architecture lets the system grow from managing 50 users to 500+ users.

Table 2: Parametric review.

Step	Success Rate / Performance
Recognition Accuracy	92-95% with good lighting
False Acceptance Rate(FAR)	3-5% (depends on threshold 0.5)
False Rejection Rate(FRR)	2-4%
Face Detection Speed	0.3-0.6 sec/frame
Recognition Speed	0.5-1 sec/per face
E-mail Notification Delay	3-4 sec(depends on network)
Servo Operation Time	5 sec
System Startup Time	2-4 sec
Max Supported Faces	50-100 faces

For especially for task like classifying emails or not spam. It's important to evaluate how well the model is performing. Accuracy can be misleading. Precisions are how trustworthy the positive prediction are. Think of F1 score as a way to balance the two. It's the harmonic mean of precision and recall which means it gives more weight to low values. If either precision or recall is low, the F1 score will be low too. It forces your model to be good at both identifying all the real cases and being accurate when it does.

4. Conclusion

The evolution of a society security system utilising face recognition and machine learning has shown how AI-powered solutions might change access control in institutional and residential settings. Research effectively provides a functional, smart, and reasonably priced security system by combining affordable hardware components including the Raspberry Pi 4, a 720p camera module (Zebronics Zeb-Crystal Pro), and a 5V relay, with strong software libraries like OpenCV Dlib, DeepFace, and TensorFlow. Depending on the ambient lighting and camera angle, the system was meant to identify faces in real time with an average detection delay of 0.8–1.2 seconds per face. The face recognition accuracy on a test database of 50 unique users with 10 photographs per user attained an ideal range of 94– 97% using 128-dimensional embeddings produced by Dlib and compared using cosine similarity or SVM classifiers. Acceptable for non-commercial, community-based implementations, the system kept a false acceptance rate (FAR) of < 2% and a false rejection rate (FRR) of < 5%. A 16GB Class 10 microSD card guaranteed consistent storage and boot operations to fit the Raspberry Pi OS, the whole software stack, and logging up to 5,000 face detection events with embedded photos and timestamps. From a software standpoint, lightweight web servers created using Flask or FastAPI let mobile apps and dashboards interface at the API level, hence enabling remote control and monitoring. Scalability has also been taken into account. By upgrading from the Dlib backend to DeepFace's ArcFace or Facenet model and storing face embeddings in MongoDB or Firebase for cloud-based access, the modular architecture lets the system grow from managing 50 users to 500+ users. The GPIO interface makes it even more adaptable for multi-layered security by supporting integration with *more relays, RFID readers, or biometric sensors. From a cost standpoint, the whole setup comprising the Raspberry Pi (£55), webcam (£15), relay module (~£3), and other accessories remains well within £80–£100. For tiny institutions seeking sophisticated security infrastructure without relying on expensive commercial systems in budget- conscious housing societies this is desirable option. Ultimately, this research not only meets its objective of delivering a safe and automated entrance system utilising facial recognition but also demonstrates how open-source tools, simple hardware, and smart design can come together to produce a very practical and deployable solution. The system may be scaled to enterprise grade use with small improvements-such as face anti spoofing, voice alarms, and database encryption making it a possible basis for the next generation smart AI driven access control systems.

Conflict of Interest

There is no conflict of interest.

Supporting Information

Not applicable

Use of artificial intelligence (AI)-assisted technology for manuscript preparation

The authors confirm that there was no use of artificial intelligence (AI)-assisted technology for assisting in the writing or editing of the manuscript and no images were manipulated using AI.

References

[1] N. S. Irjanto, N. Surantha, Home security system with face recognition based on convolutional neural network, International *Journal of Advanced Computer Science and Applications*, 2020, 11, 408– 412, doi: 10.14569/IJACSA.2020.0111152.

[2] S. Sunardi, A. Fadlil, D. Prayogi, Room security system using machine learning with face recognition verification, *Revue d'Intelligence Artificial*, 2023, 37, 1187–1196, doi: 10.18280/ria.370510.

[3] D. A. Abdullah, D. R. Hamad, I. Y. Maolood, H. Beitollahi, A. K. Ameen, S. A. Aula, A. A. Abdulla, M. Y. Shakor, S. S. Muhamad, A novel facial recognition technique with focusing on masked faces, Ain Shams Engineering Journal, 2025, 16, 103350, doi: 10.1016/j.asej.2025.103350.

[4] G. Guo, S. Z. Li. K. Chan, Face recognition by support vector machines, Proceedings of IEEE International Conference on Face and Gesture Recognition, 2001, 196–201, doi: 10.1109/FG.2000.10019.

[5] M. N. ElBedwehy, G. M. Behery, R. Elbarougy, Face recognition based on relative gradient magnitude strength, *Arabian Journal for Science and Engineering*, 2020, **45**, 1–18, doi:10.1007/s13369-020-04538-y.

[6] A. J. Russ, M. Sauerland, C. E. Lee, M. Bindemann, Individual differences in eyewitness accuracy across multiple lineups of faces, *Cognitive Research: Principles and Implications*, 2018, 3, 1–17, doi: 10.1186/s41235-018-0121-8.

[7] N. D. Hasan, A. M. Abdulazeez, Face recognition based on deep learning: a comprehensive review, *Indonesian Journal of Computer Science*, 2024, 13, 3779-3797, doi:10.33022/ijcs.v13i3.4037

[8] T. S. Gunawan, M. H. H. Gani, F. D. A. Rahman, M. Kartiwi, Development of face recognition on Raspberry Pi for security enhancement of smart home system, *Indonesian Journal of Electrical Engineering and Informatics*, 2017, **5**, 317-325, doi: 10.52549/ijeei.v5i4.361.

[9] A. Boxey, A. Jadhav, P. Gade, P. Ghanti, A. O. Mulani, Face recognition using Raspberry Pi, *Journal of Image Processing and Intelligent Remote Sensing*, 2022, **2**, 15–23, doi: 10.55529/jipirs.24.15.23.

[10] M. H. Khairuddin, S. Shahbudin, M. Kassim, A smart building security system with intelligent face detection and recognition," IOP Conference Series: Materials Science and Engineering, 2021, **1176**, 012030, 2021, doi: 10.1088/1757-899X/1176/1/012030.

[11] S. Pecolt, A. Błażejewski, T. Królikowski, I. Maciejewski, K. Gierula, and S. Glowinski, Personal Identification using embedded raspberry pi-based face recognition systems, *Applied Sciences*, 2025, **15**, 887, doi: 10.3390/app15020887.

[12] F. Faisal, S. A. Hossain, Smart security system using face recognition on raspberry Pi, in Proceedings 2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), Island of Ulkulhas, Maldives, 2019, 1–6, doi: 10.1109/SKIMA47702.2019.8982466.

[13] E. Gamess and S. Hernandez, Performance evaluation of different raspberry pi models for a broad spectrum of interests, *International Journal of Advanced Computer Science and Applications*, 2022, **13**, 819– 829, 2022, doi: 10.14569/IJACSA.2022.0130288.

[14] R. Syafeeza, M. K. M. F. Alif, Y. N. Athirah, A. S. Jaafar, A. H. Norihan, M. S. Saleha, IoT based facial recognition door access control home security system using Raspberry Pi, *International Journal of Power Electronics and Drive System*, 2020, 11, 417–424, doi: 10.11591/ijpeds.v11.i1.pp417-424.

[15] M. A. Islam, M. T. Ahmed, M. I. Hossain, M. H. Kabir, S. Roy, Face recognition based physical layer security system for next-generation wireless communication, *World Journal of Advanced Research And Reviews*, 2023, 18, 524–532, doi: 10.30574/wjarr.2023.18.3.1099.

[16] Y. El Madmoune, I. El Ouariachi, K. Zenkouar, A. Zahi, Robust face recognition using convolutional neural networks combined with Krawtchouk moments, *International Journal of Electrical and Computer Engineering*, 2023, **13**, 4052–4067, doi: 10.11591/ijece.v13i4.pp4052-4067.

[17] A. R. Syafeeza, M. K. Mohd Fitri Alif, Y. Nursyifaa Athirah, A. S. Jaafar, A. H. Norihan, M. S. Saleha, IoT based facial recognition door access control home security system using Raspberry pi, *International Journal of Power Electronics and Drive Systems*, 2020, **11**, 417–424, doi: 10.11591/ijpeds.v11.i1.pp417-424.

[18] N. Surantha, W. R. Wicaksono, An IoT based house intruder detection and alert system using histogram of oriented gradients, *Journal of Computational Science*, 2019, **15**, 1108– 1122, doi: 10.3844/jcssp.2019.1108.1122.