Turkish Journal of Computer and Mathematics Education *Vol.12 No.13(2021), 353-362*

Research Article

Implementation of Low Cost IoT Based Intruder Detection System by Face Recognition using Machine Learning

G.Mallikharjuna Rao[a], Haseena Palle[b], Pragna Dasari[c] , Shivani Jannaikode[d]

[a]Department of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, India, mallikharjunag@gmail.com
[b]Department of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, India, haseenapalle10@gmail.com
[c]Department of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, India, d.pragna1999@gmail.com
[d]Department of ECE, Chaitanya Bharathi Institute of Technology, Hyderabad, India, shivanijannaikode0198@gmail.com

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

Abstract: The intruder may enter the premises without the owner's knowledge. To identify the motion of a person who tries to enter the house will be detected by motion detection sensor. A PIR sensor placed on the door frame, it triggers the USB Camera to capture the person's image. The captured image is processed to detect face and recognize the image using Machine Learning algorithms and OpenCV. During face recognition, Raspberry pi compares the detected face with the approved pictures kept in the database. Raspberry pi captured 28 images processed per second sends an email to the owner weather authorized or unauthorized. An authentication can be verified by the user via the Internet of Things.

Keywords: Machine Learning Algorithms, face detection, face recognition, OpenCV, Internet of Things.

1. Introduction

A home is a place where we keep our valuables and our wealth. But we can never be sure about these items at times when we are away from our house. The increase in the possibility of potential break-ins by burglars is one of the inevitable problems in the world. Just locking the door may not be secure enough, and there is an urgency to protect our house in our absence. Existing systems for theft detection that have been in practice are expensive, complicated, require more space for recording and frequent user action. IoT security solutions to avoid theft [1], there is a requirement for an intelligent security [2,3] system convenient in use and requires minimum human effort. In this paper, we aim to advance a detection system that detects an intruder using face recognition [4,5] of Haar-cascade classifier.

The "Haar-Cascade algorithm" to identify faces of human beings, which is organized in Open CV by Python language and "Local binary pattern algorithm (LBP)" [6,7] for recognition of faces. Compared with other prevailing algorithms, this classifier produces a high recognition rate even with different expressions, efficient feature selection, and a low collection of false-positive features.

[8,9] Mainly focused on design and developing effective and expedient motion detection, an anti-theft surveillance device. The system captures images as soon as the motion exceeds a specific threshold level. Thus, the reduced data volume is reviewed. Data space is saved without detecting static images, object is considered based on Region of Interest (RoI). This system uses convolution neural networks to detect the motion.

The [10] study revealed that the anti-theft motion detection device protects from moving objects such as human beings and animals. The output is accurate in detecting moving objects with body temperature during day and night times. To determine the performance of an evaluation instrument was framed by the researchers. The device was found to have a good understanding and was acceptable in terms of functionality.

2. Workflow

2.1 Use Case Diagram

The Use case diagrams depict the system flow concerning the user perspective, as shown in Fig. 1. The behavioural chart made from use-case examination. It gives us information about how the user framework. Different steps necessary to be performed by the user in this system are present in the chart. The arrow marks represent the connection of the user to the actions needed to accomplish.

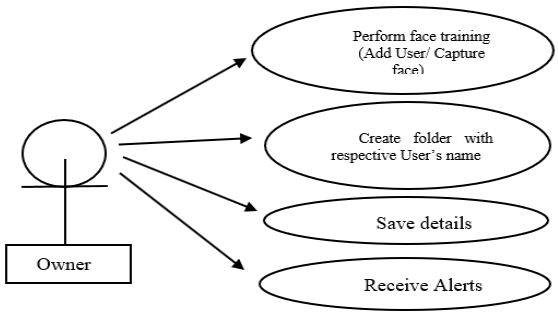

Fig 1. Use Case Diagram

2.2. Flow Chart

The below flow chart represents the workflow of this system as shown in Fig. 2. It reflects a step-by-step approach in building the system, starting from the PIR sensor motion detection to the owner receiving alert emails. The decision boxes used in the flowchart help understand the flow of different paths and the system functional approach. There are three such decisions to be made by the system and act based on the results.

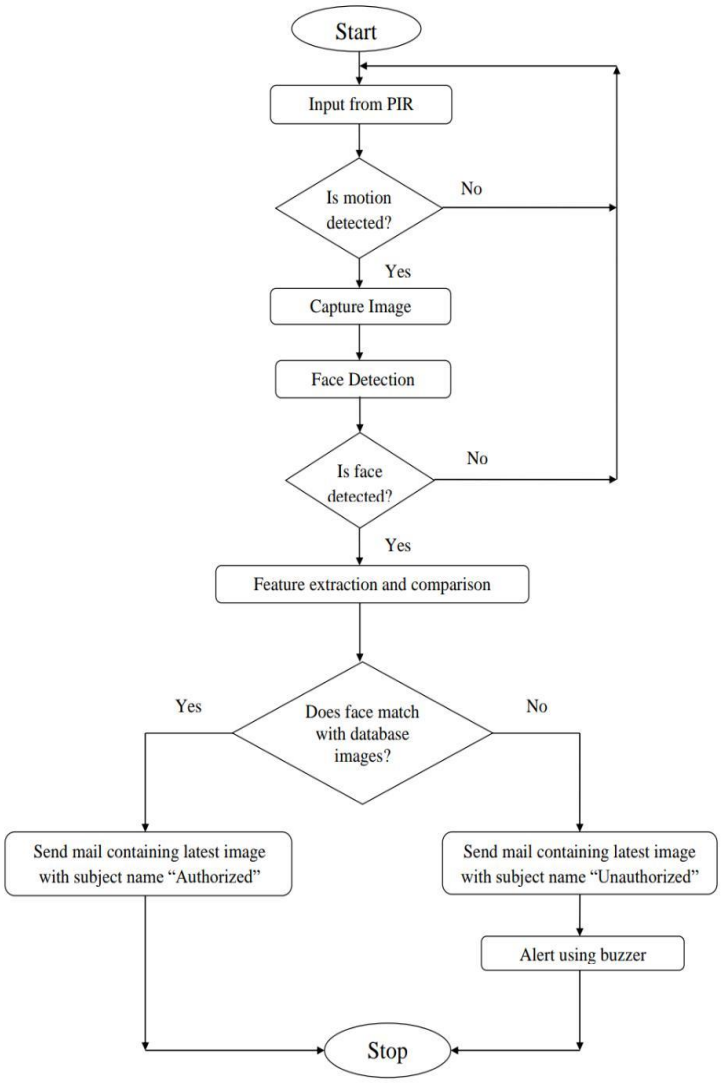

Fig 2. Flow Chart for complete hardware and software procedure

3. Approach

The system consists of four main modules as follows as shown in Fig. 3:

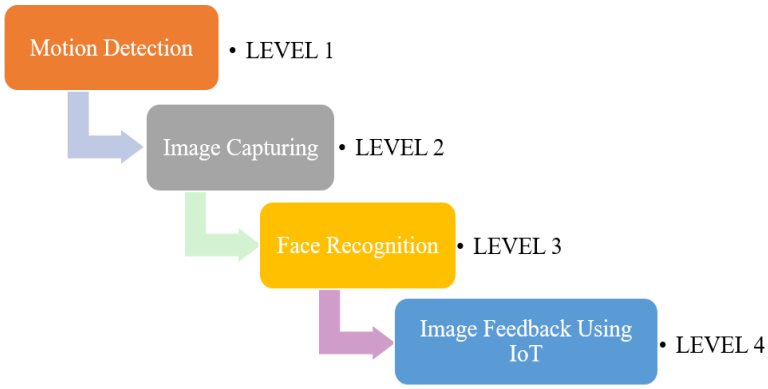

Fig 3. Various system modules

3.1. Motion detection

Motion sensor, i.e., PIR sensor, is used to detect the motion. It is placed in the doorframe or windows of the building to monitor the area around it. Once it detects the movement, it notifies the Raspberry Pi about it.

3.2. Image capturing

Image capturing or image acquisition is obtaining a digital image from a vision sensor, such as a camera, installed in the door's frame. The camera monitors the moments happening around and initiated by the Raspberry Pi to capture the image of the person standing at the door. In this system, a USB camera with a night vision source will capture the images even in the dark.

3.3. Face Recognition

The captured image is sent from the USB camera to the raspberry pi for further processing. Then the image is processed for face recognition. Three main processes in the face recognition system are as follows:

i Face Detection

ii Feature extraction

iii Comparison

Grey Scale: Greyscale images contain only shades of grey and no color. The input image from the camera is converted to grayscale to increase the accuracy to locate the face features and reduce the effect of illumination variance on the image. Using OpenCV library results in the new image pixel value from the original image pixel, and R, G, B represent Red, Green, and Blue.

3.4. Face Detection

Face detection is to locate and mark a face within an image eliminating backgrounds and hairstyles. The technique increases the precision of feature findings. The face detection processed using Haar feature-based cascade classifier from OpenCV.

4. Haar Cascade algorithm

This algorithm is to identify faces in an image or a real-time video. Though it is an outdated framework, it is exceptional in real-time face detection. This algorithm takes time to train, but it will detect the real-time faces with great speed. Many positive images containing faces and many negative images that do not include the faces are given to train the algorithm. All these images are stored in XML files and can be read with OpenCV methods. The algorithm explained in 4 stages: Selecting Haar features, creating an integral image, Running AdaBoost training and Creating classifier cascades.

4.1. Selecting Haar features

A Haar feature is essential for calculations performed on adjacent rectangular regions at a specific location in a detection window. These calculations involve adding the pixel intensity in each region and calculating the differences between the sums. A primary method to discover what area is lighter or more obscure is to summarize the pixel estimations of the two areas and compare them. The amount of pixel esteems in the dark area will be smaller than the lighter portions of pixels. On the off chance that one side is lighter than the other, it very well might be an edge of an eyebrow, or once in a while, the centre segment might be shinier than the adjacent boxes,

interpreted as a nose. There are three types of Haar-like features, as shown in Fig. 4. The features are Edge features, Line features and Four-sided features or centre-surround features

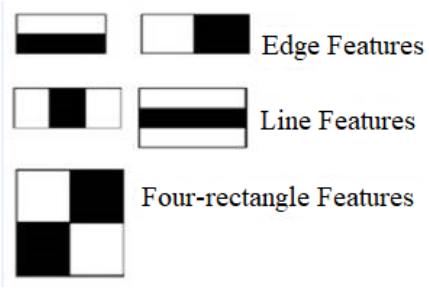

Fig 4. showing Haar-like features

Edge features and Line features are valuable for identifying edges and lines separately—the four-sided features used for discovering corner to corner (diagonal) elements. The estimation of the component is determined as an individual value: the summation of the pixel values in the dark region subtracted by the sum of pixel values in the white area.

Value = Σ (pixels in black area) - Σ (pixels in white area) - (1)

Using all the above results, we obtain a piece of valid information out of the face.

4.2. Creating an integral image

Haar features can be challenging to determine for a large image. The integral image formed when dealing with extensive features. The selected image is a quick and efficient way to calculate the sum of pixel values in an image. Each point value of an image is the sum of all pixels above and to the left, including the target pixel s(x,y) as shown in Fig. 5 and 6.

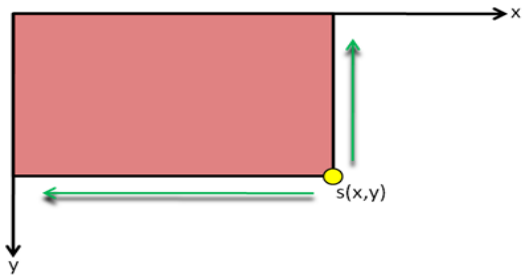

Fig 5: Summing up the pixels to get individual value

$$s(x,y) = i(x,y) + s(x,y-1) - s(x-1, y-1) \quad - (2)$$

These are then used to compute the Haar features.

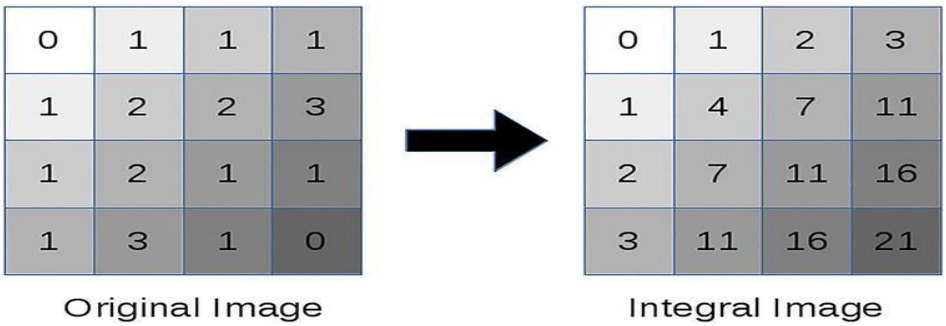

Fig 6: Integral image from the original image

4.3. Running AdaBoost Training

Haar Cascade algorithm uses a 24x24 base window size for evaluating haar features in any captured images. The window consists of nearly 160,000 features, but only a few of these features are important to identify a face. So, the AdaBoost algorithm is to identify the best features among the 160,000 features. Haar-like feature algorithm is a weak learner to select the features, sort, and size in a final classifier. AdaBoost is used to check the performance of classifiers by evaluating subregions of the images supplied to it. Few subregions will yield strong

responses indicating that it has a human face classifier, whereas some produce weak responses indicating that the classifier does not have a face. The final result is a strong classifier (equation 4) containing the best-performing weak classifiers. Below is the AdaBoost Algorithm. First, the weak classifier with the lowest weighted classification error should be selected by fitting the weak classifiers to the data set. Then the weight for t_{th} weak classifier should be calculated as shown in equation 2 and weights have to be updated as shown in equation 3.

1. *Input:* Give sample set S = (x1, y1), ...(xn, yn) xiϵ X, yiϵ Y = {-1, +1}, number of iterations T.
2. *Initialize:* $W_{i,j} = 1/N$ i = 1,…. N
3. For t = 1, 2,…,T,
   i. Train weak classifier using distribution Wt.
   ii. Calculate the weight (W_i) training error for each hypothesis.

$$h_n\varepsilon_t = \sum_{i=1}^{N} w_{t,i} \mid k_i - y_i \mid \quad \text{- (3)}$$

iii. Set: $a_t = \frac{1}{2}\log\frac{1-st}{st}$ - (4)

iv. Update the weights. $W_{t+1,i} = 1 + \frac{W_{t,i}}{Z_t} * \begin{cases} e^{-a}t \\ e^{a}t \end{cases} = \frac{w_{t,i} * \exp(-\,a_t\, y_i\, h_t\,(x_i))}{z_t}$

- (5)

4. *Output:* The final hypothesis, also the more robust classifier.

$$H(x) = sign\,(\sum_{t=1}^{T} a_t h_t(x)) \quad \text{- (6)}$$

Where,

T stands for number of iterations,

Wi,j stands for weight of each data point,

ε_t stands for classification error,

a_t is weight of t_{th} classifier,

$H(x)$ represents final hypothesis ie., the strong classifier,

z_t stands for normalization factor,

h_t designates the t^{th} weak classifier and

t represents its corresponding weight.

4.4. Creating classifier Cascades

The AdaBoost will finally select the best features around 2500, but it still takes a lot of time for computations. The cascade quickly discards non-faces and avoids wasting time and calculations. Thus, achieve the necessary speed for real-time face detection. The way towards recognizing a face is partitioned into various stages. In the first stage, the subregion passes through the best features, such as identifying the nose bridge or identifying the eyes. In the following steps, all the remaining components are present. When a sub-region of an image enters the cascade, assessed in the first stage. If that stage considers the subregion as positive, meaning that it thinks it's a face, the output of the stage is "maybe." A subregion set "maybe," is sent to the next step of the cascade, and the process is recurring till the last stage. If classifiers approve the image, it is finally classified as a human face and is accessible to the user detection. If the first stage gives negatives, then the image is immediately discarded as the human face is not detected. If it passes the first stage but fails the second stage, the image discarded as shown in Fig. 7.

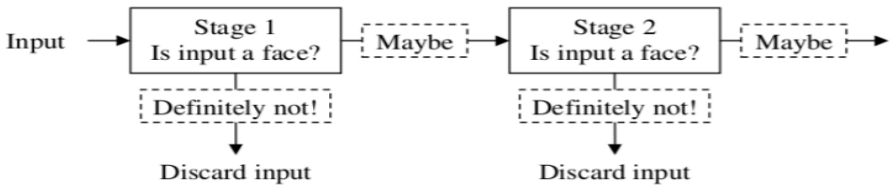

Fig 7. Creating Classifier Cascades

Feature Extraction

After the face detection, the face features are extracted from the image. The next step is to extract the features from it using the face embedding model. A neural network inputs the image of the person's face and outputs a vector representing the essential features. A face embedding is a vector representing the features extracted from the face and is used to recognize faces. The face embeddings for the same face may be close in the vector space, whereas the face embeddings of two different faces may be far away. The image datasets are given to a pre-trained network to get the respective embeddings and save these embeddings in a file for the next step.

Comparing faces

As face embeddings for every face stored in the database, the next step is to recognize a new image passed to the system. Firstly, the face embedding for the image is computed and then compared with the other face embeddings stored in the database. The face is recognized if embedding is closer to any different embeddings. If the new face embedding is not comparable to any other embeddings in the database, then the face is not recognized.

5. System Implementation

The system's technique involves two signs of progress the input image and the image captured through live streaming. These procedures trail applied on four known face acquisitions, pre-processing, face detection using Haar-cascade classifier, and feature extraction using face embedding models. The detailed steps involved in the implementation of this project are as follows as shown in Fig. 8:

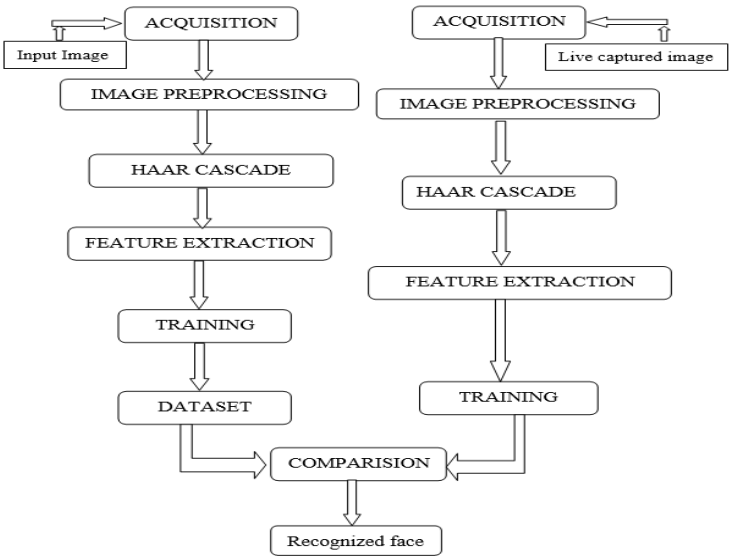

Fig 8. System Implementation Steps

Step1: System setup: The different hardware components such as the PIR sensor, USB camera, monitor, mouse, keyboard, a piezoelectric buzzer is connected to the Raspberry pi board using various cables to the different slots and pins available on it. The status of the PIR sensor goes to '1,' which initiates the camera for acquiring the image. The quantity remains zero if no motion is detected as shown in Fig. 9.

Fig 9. System setup

Step 2: Image acquisition: When a person enters into the door's frame, his/her motion image captured by the PIR sensor, which outputs a 5V signal to the Raspberry pi board, which intern initiates the camera for image capturing.

Step 3: Pre-processing: The image captured by the camera is resized to 500 pixels and converted from BGR to grayscale for face detection and from BGR to RGB for face recognition.

Step 4: Haar Cascade: The face in the processed image is detected using the Haar cascade classifier, and a bounding box is drawn around the detected face. In the next step, features are extracted from it.

Step 5: Feature Extraction: The face embedding model is used to extract the features from the face.

Step 6: Training: First, the prepared dataset to train the algorithm. In the dataset preparation, 30-40 snapshots of the authorized person faces are taken as shown in the figure in different angles, stored in a file with the person's name. The face embeddings of different faces stored in the database are considered, and then the model is trained with these face embeddings as shown in Fig. 10.

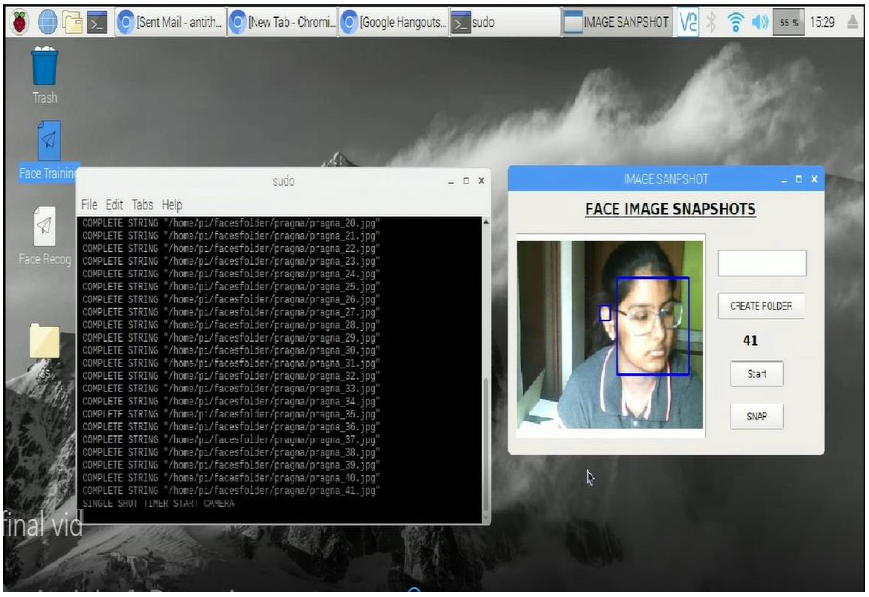

Fig 10. Training the dataset

Step 7: Dataset: The values of face embeddings are stored in the database only in the case of an input image.

Step 8: Comparison: When a new face from the live video is acquired, it undergoes pre-processing, face detection, feature extraction steps. The face embeddings of it are looped over the facial embeddings of all the other faces stored in the database by attempting to match. The recognized face obtained, if its face embeddings are checked carefully with different face embeddings in the database as shown in Fig. 11.

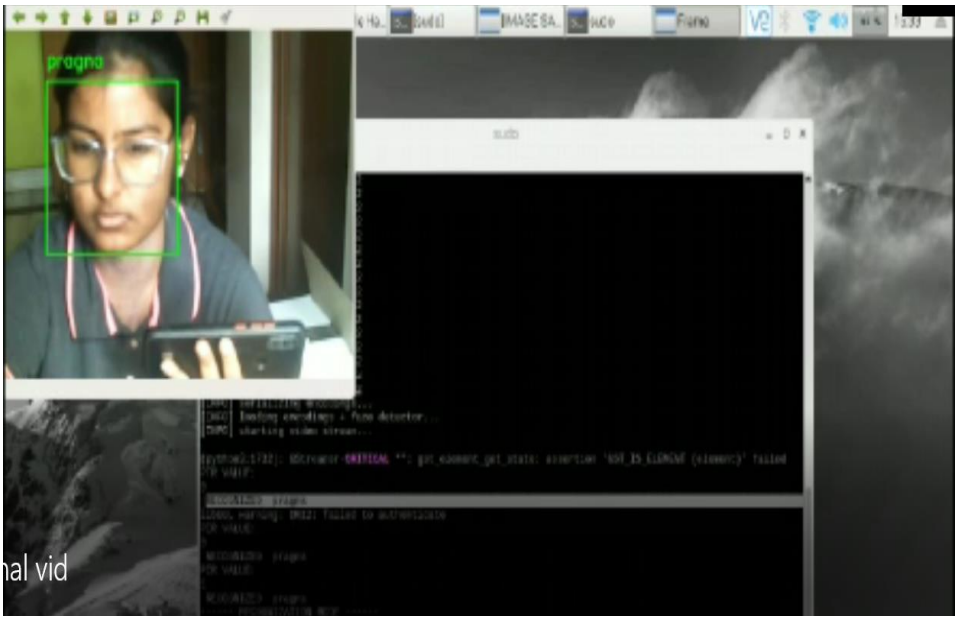

Fig 11. Face recognition

The following as shown in Fig. 12 the recognized faces of 3 different authorized persons with a small rectangle over the face along with their name.

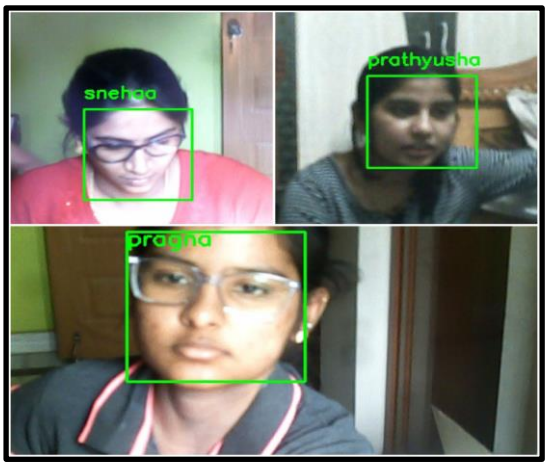

Fig 12. Authorized persons' face recognition

Step 9: Sending alert via email: If the face is recognized, a file containing the entire image with a rectangle box highlighting the face along with the name with which that person's folder was names is attached and sent to the owner via email. If the face is not recognized, a file containing the entire image with a rectangle box highlighting the face and the name "unknown" is attached and sent to the owner via email using the Internet of Things. An alarm is raised through a buzzer on the premises to alert the neighbours.

6. Results and Discussions

Case 1: An authorized person identified: If the captured image has a face that matches with any of the user's face in the database. Then the corresponding person's latest captured image with the label associated with it as per the name of the folder is sent to the owner through an email with the subject name "AUTHORIZED" as shown in Fig. 13.

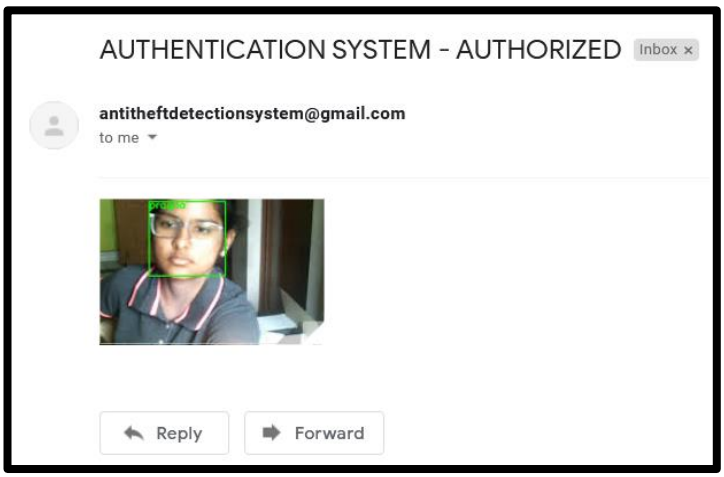

Fig 13. email to owner with the subject "AUTHORIZED."

Case 2: An unauthorized person identified: If the captured image has a face that doesn't match any user's face in the database. Then the corresponding person's latest captured image with the label "unknown" is sent to the owner through an email with the subject name "UNAUTHORIZED" as shown in Fig. 14, along with a buzzer beeping to alert the neighbours about the intruder on the premises.

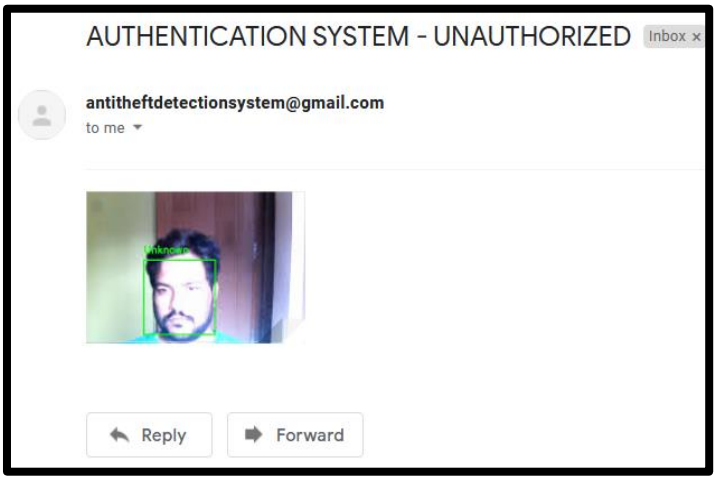

Fig 14. email to owner with subject "UNAUTHORIZED"

*Output Analysis drawn***:** The table below (Table 1) shows the variation in the training period when the number of images increases. It is studied from the above table that increasing the number of images will not cause any drastic increase in the training period.

Table 1: Time period for various images

S.NO	NO OF IMAGES	TIME PERIOD (SEC)
1	1	0.02
2	5	0.15
3	15	0.56
4	25	0.98
5	30	1.2
6	40	1.5

Image feedback using IoT

The processed image was sent to the owner via email using IoT. An SMTP is a push protocol used for this purpose. If the seeming face matches the faces in the database, the sent mail referred with the subject "AUTHORIZED" and the file attachment containing the recognized face. If the detected face does not match the faces in the database, a sent mail referred with the subject "UNAUTHORIZED" and the file attachment containing the detected face.

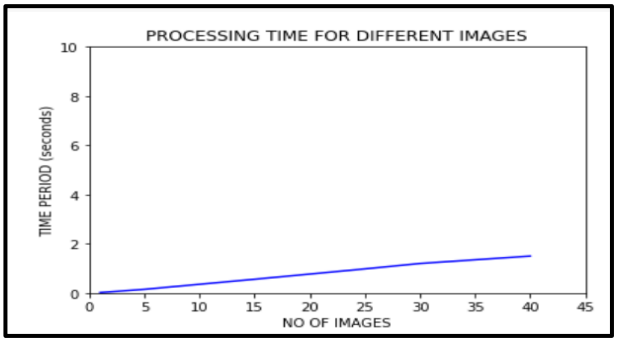

Fig 15: Plot between Numbers of images Vs. Time period (seconds)

Graph plotted between the number of images in the training dataset and the time taken to process those images is illustrated in Fig 15. The comparison on number of images and time period shows within 2 secs it can process 40 images to detect a person, the response time is very much efficient for capturing and identifying the intruder using developed system.

7. Conclusion

The intruder's motion is detected using PIR sensors, and the intruder image is captured. Haar Cascade algorithm provides better computational time and greater accuracy in detecting and recognizing the human face. The image processing rate attained by the system is noted to be around 28 images in 1 sec for the complete process. The alert email is sent to the owner with the latest captured image using IoT. Thus, the developed low-cost system is fast, highly accurate, giving efficient alerts, and serves as a monitoring system. It is convenient to solve security problems which will help to reduce or stop the break-ins.

References

1. Chandrappa, D. R., Pavan, H. R., Sagar, M. V. M., & Dakshayini, M, "IoT Security Solution to Avoid Theft" In 2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 1-3,2018
2. Ge, R., Shan, Z., & Kou, H. "An intelligent surveillance system based on motion detection", In 4th IEEE International Conference on Broadband Network and Multimedia Technology, pp. 306-309, 2017.
3. Murugan, K. S., Jacintha, V., & Shifani, S. A, "Security system using raspberry Pi", Third International Conference on Science Technology Engineering & Management (ICONSTEM), pp. 863-864, 2017.
4. Patil, N., Ambatkar, S., & Kakde, "IoT based smart surveillance security system using raspberry Pi", International Conference on Communication and Signal Processing (ICCSP) pp. 0344-0348, 2017
5. Visakha, K., & Prakash, S. S. (2018, July). "Detection and Tracking of Human Beings in a Video Using Haar Classifier", International Conference on Inventive Research in Computing Applications (ICIRCA) pp. 1-4, 2018.
6. Senthamizh Selvi. R, D. Sivakumar, "Face Recognition Using Haar – Cascade Classifier for Criminal Identification," International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume-7, Issue-6S5, April 2019.
7. Le Tran Huu Phuc, HyeJun Jeon, "Applying the Haar-cascade Algorithm for Detecting Safety Equipment in Safety Management Systems for Multiple Working Environments," September 2019.
8. Ajay Kushwaha, Ankita Mishra, "Theft Detection using Machine Learning," IOSR Journal of Engineering (IOSRJEN), ISSN (e): 2250-3021, ISSN (p): 2278-8719, Volume 8, PP 67-71, March 2018.
9. Dixit SurajVasant, Babar Apeksha Arun, Meher Priya Shivaji," Raspberry-Pi Based Anti-Theft Security System with Image Feedback," Journal of Information, Knowledge, And Research in Electronics and Communication Engineering, October 2017.
10. Rhowel Dellosa, "Development of an Anti-Theft Device using Motion Detection and Body Temperature," Asia Pacific Journal of Multidisciplinary Research, P-ISSN 2350-7756, E-ISSN, Volume 2, No. 6, pp. 2350-8442, 2014.