Your Name: Venkatarao Rebba

Professor Name: Tejaswi Linge Gowda

Subject Name: AME 494/598: Programming for IoT

05 December 2022

Home Security - Alert System

I. Abstract:

In recent years, security has become the most onerous task. Our house is where we keep our valuables and capital. We can never be confident, however, that the asset behind us is secure. We normally lock the doors when we leave the house. However, simply closing the door is insufficient; a system must be in place that maintains track of activities, reports to the owner, and works in accordance with the owner's reaction. Thus, the primary goal of this project is to create a home security system that automatically alerts the homeowner with a picture of an unfamiliar individual who enters the home territory. This project is implemented on an RPi3 board with a camera system that recognizes the person and sends an automatic alert in the Discord application to the registered people.

II. Methodology:

In this project, the RPi3 board with a camera is fixed in the required locations of a house, in front of the main door, for instance. The camera perceives the seen and continuously sends the feed to the alert system module. The alert system module is facilitated with face

recognition technology and identifies the people in the image. For face detection and recognition, computer vision and machine learning techniques are exploited. It checks whether the faces in the image are registered or not. For that, it gets all the registered users from the MongoDB table and compares them. Subsequently, the system sends a Discord message to the homeowners using Google Cloud services such as Cloud Function, Pub/Sub Message, and Cloud Storage if it detects faces that are not registered in the database.

Additionally, in order to prevent continuous alerts, a time window of 1 minute is taken. That is if an unknown person appears in a frame, then for the next 60 seconds, the system does not send a notification of that person.

Face Detection and Recognition:

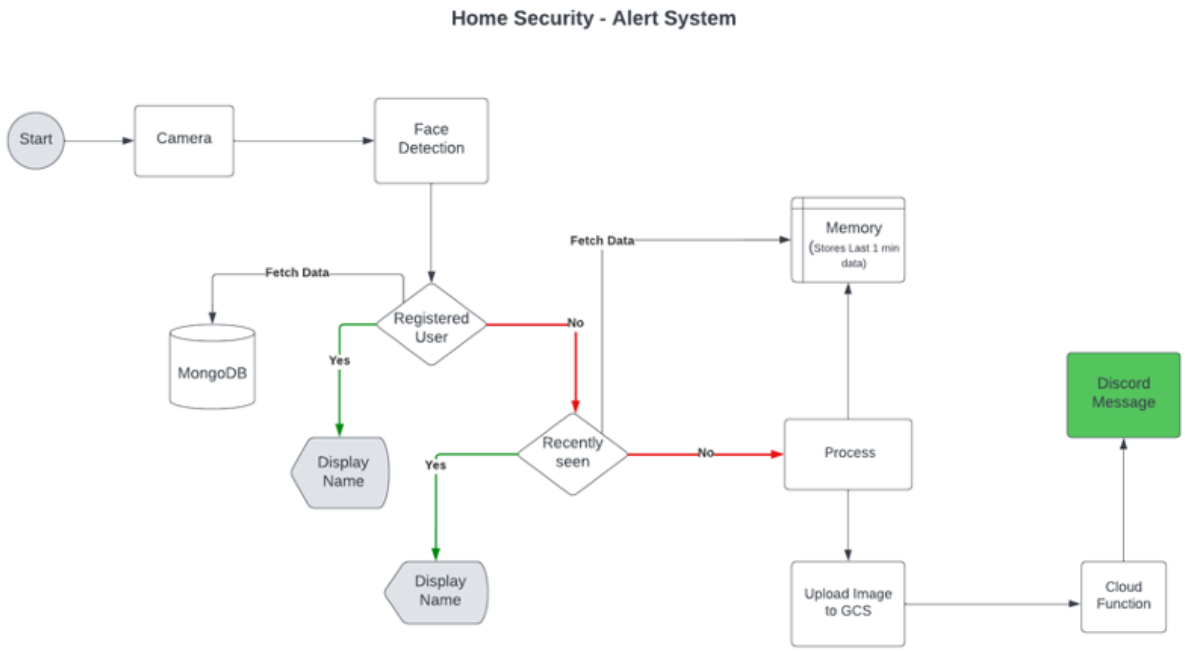
Initially, the interest is to just detect the faces in an image and sends a notification for any change in the number of face detections. This is accomplished using Haar Cascade Classifier using Python OpenCV module. However, this system is not robust for face detection and sends notifications for any person that appears in the frame.

In the second version of the system, face recognition is also integrated. This not only made the system robust but also alleviated the problem of sending alerts for known or registered people.

Face detection is achieved by leveraging the face embedding technique from face_recognition module [1] that internally uses dlib models [2] in python. Essentially, this is implemented using a deep neural network, ResNet-34, that is trained on 3 million faces. By leveraging those models, face_recognition extracts face encodings in an image that are distinct for different faces. Therefore, face recognition is achieved using those face encodings.

Alert System:

Once the system identifies faces that are not registered in the system, it uploads the current frame of the stream to Google Cloud Storage (GCS). A put trigger is set on the GCS that invokes a cloud function whenever a new file is uploaded to the GCS. The cloud function job is to take the image and send it to the Discord group.



System Overview

III. Results:

The system exhibits significant results. The face_recognition is able to detect faces in different angles, and lighting conditions. On a high level, there are three different scenarios for this system.

1. Known person appears in the frame
2. Unknow person appears for the first time in the frame
3. Unknow person appears after once detected

1. Registered user

All the registered user's data is stored in MongoDB. So, it is effortless to add or remove a user. The system detects the user and displays the name of the user as follows. In this case, the system does not send any notifications.

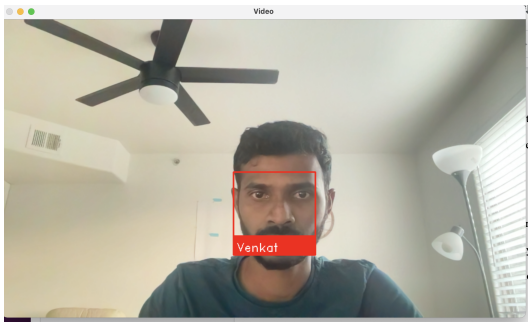


Fig 1: Single Registered User

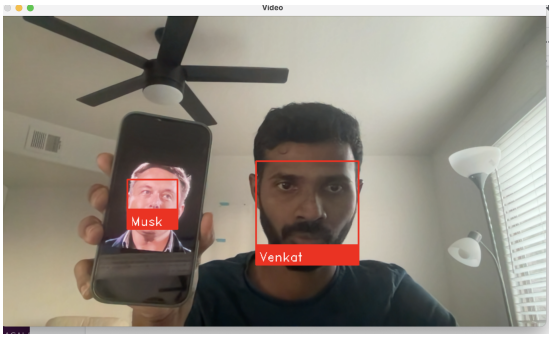


Fig 2: Multiple Registered Users

2. Unregistered user for the first time

In this scenario, an unknown person's face encodings are registered in the system with a temporary name and stored for 60 secs. In addition, the system triggers an alert to send a notification through Discord.

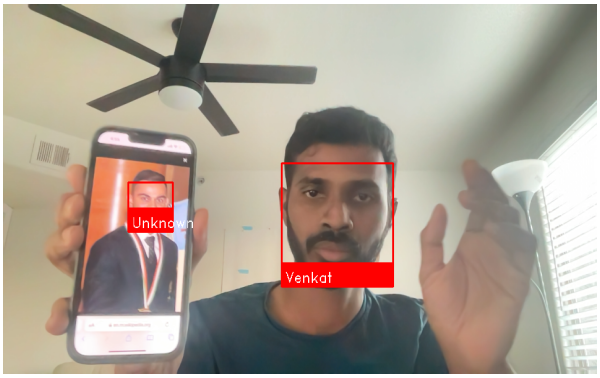


Fig 3: Unknow person detection

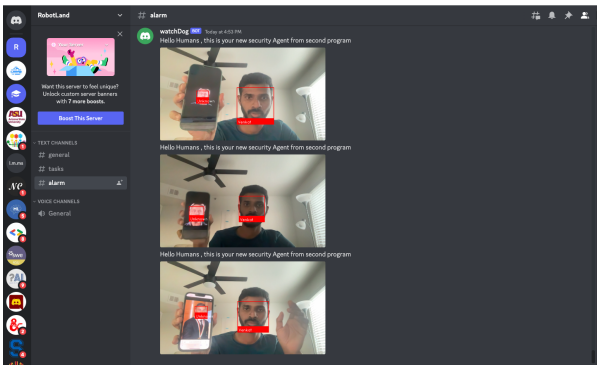


Fig 4: Automatic message in Discord

3. Unregistered user for the second time or within 60 seconds

Though this person is not registered in the database, these face encodings are already stored in the temporary memory. Then, the temporary frame is displayed on the frame. However, it does not trigger any notification.

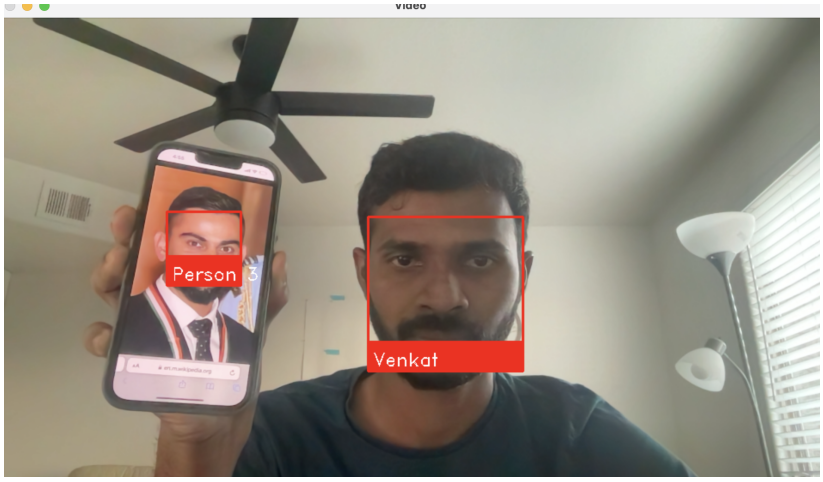


Fig 5. A temporary name is assigned to the unknown person

Performance:

In my laptop, Mac M2 Processor, the average time to process 1 frame is 0.05 sec. It implies that the inference speed is 20 fps. However, due to installation challenges, this system is not tested on the RPi3 board.

IV. Conclusion

In summary, a very cost-effective, easily deployable, scalable, and maintainable system is proposed and proven an alert system for home automation. In order to scale the system, more users can be stored in the MongoDB, increasing the temporary storage time window from 60 secs to 300 secs, and storing all images in the GCS to see the history of the records, for instance. The face recognition module does not give accurate predictions on the blur or low-quality frame. This can be improved by deploying one highly accurate and more weightage model on the GCP as a second level of validation before sending a notification.

References

1. https://face-recognition.readthedocs.io/en/latest/readme.html
2. http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html
3. https://cloud.google.com/
4. https://realpython.com/how-to-make-a-discord-bot-python/