Gulnaz Khabibullina - 1001555830
Nishad Aherrao - 1001351291
CSE 4340-001

Final Project Report

Security Camera: Facial Recognition

Team Collaboration Tools and Documentation -

Github Repository (Public) GNU GPL License -
https://github.com/nishad10/faceRecognitionSecurity

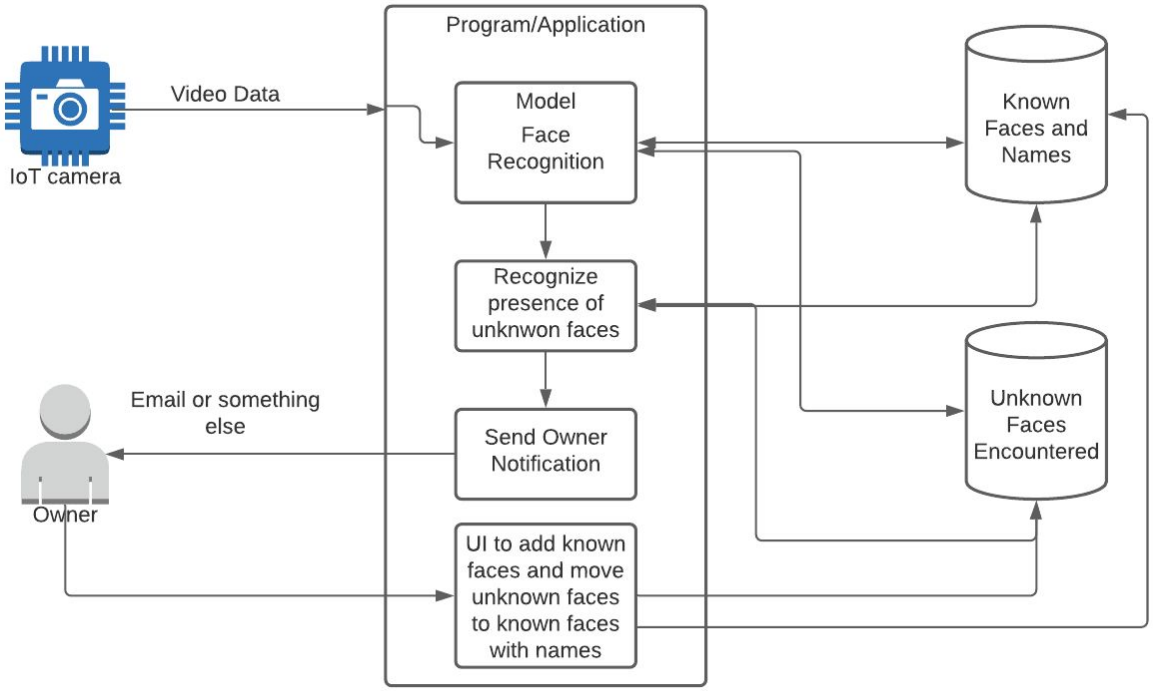
Youtube Link for a demo presentation -
https://youtu.be/ikXQcRfVvYs

System requirements -

Task	Due Date	Progress
Requirements Planning	October 25, 2020	Completed
Designed Application (High Level)	October 27, 2020	Completed
Decided on exact model to use along with programming language and UI/UX specifics	October 29, 2020	Completed
Program can now identify faces	November 5, 2020	Completed
~~Webcam/Camera can transfer feed to our program~~ Use webcam instead of IOT camera	November 12, 2020	Skipped/Modified
Integrated the program so that it can recognize faces from a video feed from camera	November 23, 2020	Completed

UI/UX minimal or console based commands to let user add faces to known list or move faces from unknown to known list	December 1, 2020	Completed
Identify presence of unknown faces and notify user	December 3, 2020	Completed

Workflow and System Design -



Overview -

Our program will serve the list of requirements defined above. The program has a basic UI that can be used to control it. On running the program you are given a startup window where you can choose to set up your email settings, add your/friends pictures to the list of known faces etc. Once that is done you can run the program itself so that it can start monitoring. If it sees an unknown face in the feed it will send you an alert. We have set it up so that it sends you an alert every 30 seconds. This can be changed by changing the *timethreshold* value.

Overall we are very happy with our program. We were able to complete all the outlined requirements in time. We were also able to fast track the progress on the project as to be able to present early. We have been able to take an established machine learning model and use its face

recognition capabilities to set up a basic security system for your home. The only parts we had to skip/modify and would like to have worked on more is integrating the system with a live video feed form IOT camera that you can hook up to the front of your door. Instead we have used a webcam. This would have required a webserver setup to stream the video and further work on security so we decided to skip it so as to be able to complete the project in time. So we would say the project has been a success overall.

Accuracy and Performance -

Built using dlib's state-of-the-art face recognition built with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark.

As we are using a prebuilt module for face recognition we achieve a high accuracy of 99.38% as shown above. Please take a look at Caveats/Issues on their repository to understand this accuracy further.

We are also using a tolerance level of 0.6 as we want to be able to end up with false positives instead of missing out on an actual alert however this can be changed if wanted.

The module we used for face recognition has been tested on a Raspberry Pi so we expect similar performance levels given that we do not use any substantial amount of extra resources/computation power. However we have not tried running this on a Raspberry Pi. Given that we have a standardized code it should work without modifications on any linux distribution but we haven't tried it.

Setup / Installation -

You can refer to our github page for detailed look at the installation and setup of the program here: (Installation)
https://github.com/nishad10/faceRecognitionSecurity#installation

Or you can skip to runtime instructions for the program here: (How to use program)
https://github.com/nishad10/faceRecognitionSecurity#program-instructions-runtime

Distinguishing the FaceRecognition Library from our program -

The face recognition library that we used helps us recognize faces in a picture and helps us label them based on a list of known pictures. This is the only part of the program that the library helps us do. We have used this to integrate it with a program that can do much more.

What we built:

1. A UI to control the program easily.
2. Setup upload options in UI so you do not have to modify code or copy paste images to code repository.
3. Use the library/module on a live video feed.
4. Process the video feed, by splitting it into frames (images) and process every other frame for faces.
5. Setup email notifications, so that the program can notify user when an unrecognized face is in front of the camera for more than a timethreshold (30 seconds).
6. Setup Email options in UI so that you don't have to change any code and can change/setup with any email you want.

Technology/ Solutions Stack Details -

In order to implement our version of face recognition environment we have used the existing sources that are available on GitHub.We used the following repository with implementation of our own customized features and functions.

Python - https://github.com/ageitgey/face_recognition

We are using this built in standard module in python to help us send email notifications.

https://docs.python.org/3/library/email.html#module-email to send the user any notifications.

For our UI we are using another standard module from python called tkinter.

https://docs.python.org/3/library/tkinter.html

As for processing the video and frames and displaying a modified version of the video with names under faces we are using OpenCV.

https://pypi.org/project/opencv-python/