

Facial Recognition Project

SAIT Information System Security program.

Carl Vincent Saban | Monte Kang | Jun Wang

AGENDA

- Introduction
- Facial Recognition Technology
- Detail of the Project
- Problems and Solutions
- Ways to improve the project
- DEMO TIME
- Q&A.

Introduction

Introduction

What is our project?

- **Build a facial recognition system with IFF functionality (system is able to differentiate between friends and foes), it scans faces of all users who access the system and add them to a database (only new faces should be added to the database), should have logging system (who accesses the system, times, new records in the database), and finally it should alert the user when a “foe” is detected.**
- **Build a two way authentication system: using username/password as the first way of authentication and biometric facial feature as the second way of authentication.**

Introduction

Goal for the project

- The main goal of the project was to make a facial recognition system that can be run in a restricted room or area for surveillance and recording the information of accessed personnel.
- Develop a 2 way authentication system with functions adding new users and verify users' credentials.
- Accuracy | Timing | Security

Introduction

Achievement from the project

- **Developed a face recognition system, with the function of automatically creating database tables and saving face features into the database.**
- **accuracy rate of 99.38%.**
- **2 way authentication system with password and facial recognition**

Facial Recognition Technology

Facial Recognition Technology

Facial Recognition Technology

Facial recognition technology (FRT) has emerged as an attractive solution to address many contemporary needs for identification and the verification of identity claims.

Facial recognition technology (FRT) allows the automatic identification of an individual by matching two or more faces from digital images.

Facial recognition refers to a multitude of technologies that can perform different tasks for different purposes. In this regard, a key distinction is whether facial recognition is used for verification or identification.

Facial Recognition Technology

Identification

Identification means that the template of a person's facial image is compared to many other templates stored in a database to find out if his or her image is stored there. Facial recognition can help verify personal identity.

In this project, our real-time face recognition system (`face_surveillance.py`) is an example of utilizing identification technology. Identification is used based on facial images obtained from video cameras.

Facial Recognition Technology

Verification

Verification technology forms the base of the second authentication method of our 2 way authentication system. Facial Authentication is a form of biometric authentication that relies on the unique biological characteristics of an individual to verify that she is who she claims to be. The users are authenticated by verifying their faces as one credential to securely access their accounts.

Facial Recognition Technology

OpenCV

OpenCV (Open Source Computer Vision) is a library of programming functions for real-time computer vision. The face detection part of the project was made using an OpenCV Library and python model face-recognition.

The OpenCV library proved to be flexible enough for the project as it can accurately detect a face in real time and highlight it by drawing a rectangle around the faces passing by. While faces are being detected, the application analyzes every video frame for face recognition.

Facial Recognition Technology

OpenCV

Advantages:

- works almost real-time on CPU.
- Simple Architecture.
- Detects faces at different scales.

Disadvantages:

- The major drawback of this method is that it gives a lot of False predictions.
- Doesn't work on non-frontal images.
- Doesn't work under occlusion.

Facial Recognition Technology

HoG Face Detector in Dlib (face_recognition)

The face_recognition module was built using dlib's state-of-the-art face recognition built with deep learning.

This model is based on pre-trained models, meaning that the software develops rules for identification of faces based on a database of facial images.

This was possible through the increase in the availability of facial images at higher quality and the increase in computing power to process large amounts of data.

Facial Recognition Technology

HoG Face Detector in Dlib (face_recognition)

Advantages:

Works very well for frontal and slightly non-frontal faces

Works under small occlusion

Easy to implement

Disadvantages:

It does not detect small faces as it is trained for a minimum face size of 80×80.

The bounding box often excludes part of forehead and even part of chin.

Does not work for side face and extreme non-frontal faces, like looking down or up.

Detail of the Project

Detail of the Project

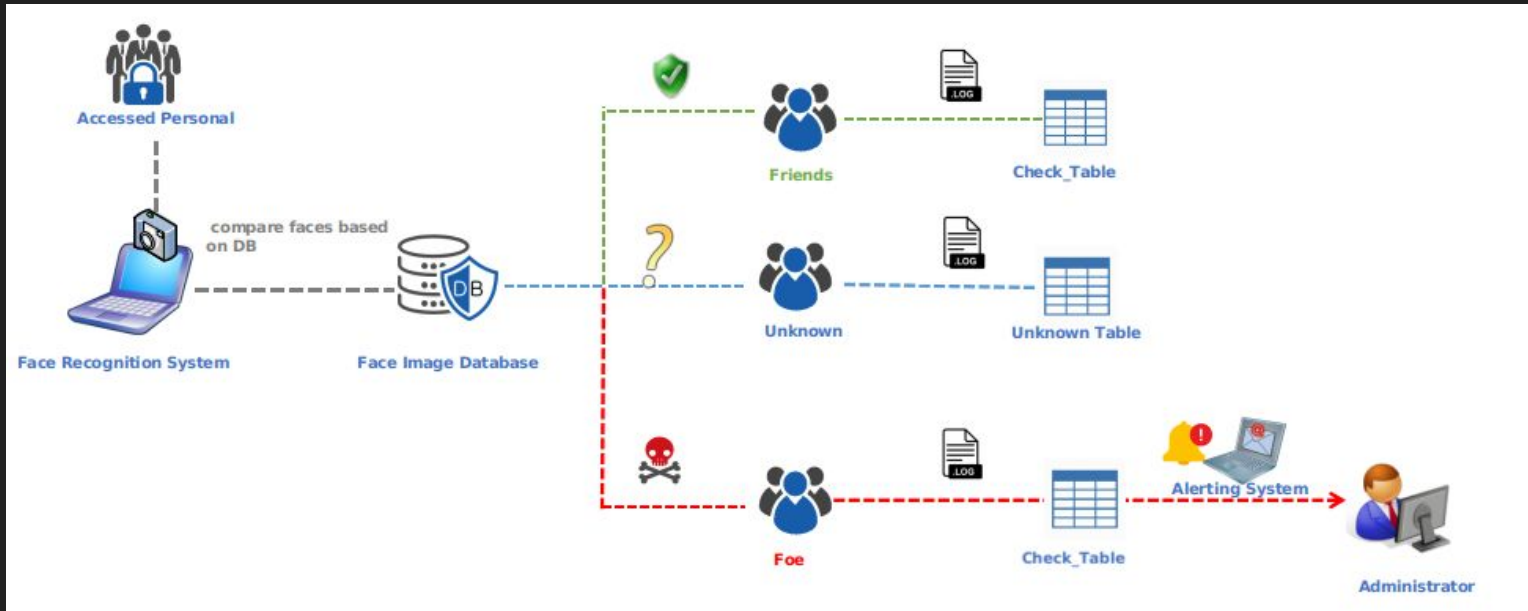
Operational and Environmental Requirements

- **OS:** Ubuntu 19.10
- **Equipment:** Webcam installed on the laptop
- **Language:** Python 3.7
- **Database:** MySQL
- **Python module:** OpenCV, Face_Reognition in Dlib



Detail of the Project

Main Structure



Detail of the Project

Database

We created 5 Mysql tables (face, check_table, unknown, users, usergroups) in iss database. There are three to six fields depending on the purpose of the table.

SAIT_face_recognition2020

- > __pycache__
- ▼ images
 - > Donald_Trump
 - > Jun_Wang
 - > Modea_Kong
 - > Vincent_Saban
- > unknown
- 1.jpg
- add_new.py
- camera_log

face

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	Auto_increment
fname	varchar(40)	NO		NULL	
lname	varchar(40)	NO		NULL	
encoding	blob	NO		NULL	
isfriend	tinyint(1)	YES		0	
isdelete	tinyint(1)	YES		0	

check_table

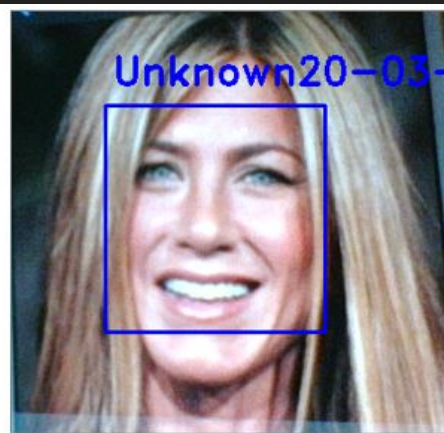
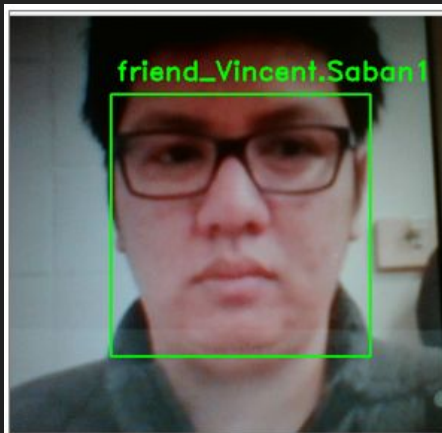
Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	Auto increment
fname	varchar(40)	NO		NULL	
lname	varchar(40)	NO		NULL	
date	date	NO		NULL	
time	time	NO		NULL	

unknown

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	Auto_increment

Detail of the Project

IFF function



Detail of the Project

Alerting functions

- Playing alarm thread

```
logger[("Found a foe!" + str(name[:-1])), 'critical', 'camera']  
#start a new thread for sound the alarm  
talert=threading.Thread(target=alert)  
talert.start()
```

- alerting email

```
#send email for alerting  
tmail=threading.Thread(target=email_sender,args=(name,date,time))  
tmail.start()
```

Detail of the Project

Sign up

Create your account

User Name *

SAIT2020

You can use letters,numbers&periods

Password *

SAITcapstoneFR2020?

Confirm Password *

SAITcapstoneFR2020?

Use 8 to 20 characters with a mix of upper and lower letters, numbers & symbols(?,!_)

☒ Show Password

Upload Image *

/home/user/Downloads/2020winter/capstoneSAIT_face_recognition2020/images/Modea_Kong/modea.jpg

Submit

Register Successful

Return to sign in

Detail of the Project

Sign in

Sign in to your account

Username password correct

User Name


Password

☒ Show Password

[SIGN IN](#)

[SIGN UP](#)

Verifying Face



Submit Capture Cancel

Problems and Solutions

Problems and Solution

Using OpenCv

- **Opencv doesn't work with environments that is written in Qt.**

An example of that is Kde plasma desktop environment.

- **Only solution is to use operating system that uses GTK environment
i.e Ubuntu gnome environment**
- **Opencv has two packages (opencv-python and opencv-contrib-python).**

Problems and Solution

Choosing the face recognition module.

OpenCV with Haar_Cascade vs. Face_Recognition



Problems and Solution

Implementing the alerts.

The face recognition process pauses when sending alerts.

- Before we apply the solution, when a foe is found, the alerting system runs in the major process. The face recognition had to pause and wait for the alerting process to finish. To solve this problem we implemented python multithreading module. By run alert system in separate threads, the major face recognition process runs without interference. This measure makes sure the face recognition system real-time.
- The email alert that we had is getting blocked by the SAIT firewall. we use our own phones hotspot so that it won't use SAIT's network.

Ways to improve the project

Ways to improve the project

MTCNN module

we could improve the accuracy using other machine learning methods. MTCNN which uses tensorflow on its backend, would increase the accuracy of the program and the range of how far it can detect faces.

Advantage:

- Can identify more faces than the other two.
- Can identify faces from a far.

Ways to improve the project

Using better hardware

We can apply knowledge distillation to compress the current model and further reduce the model size using low bit quantization. Better hardware is another option to improve this project, while doing this project we were only using SALT's Toshiba laptop, which only has an i5, an integrated graphics card and the laptop's integrated camera. Doing this project with better hardware would have completely changed on how we approached our project. Instead of finding modules that could barely work on our machines, we could have expanded our search and used some of the GPU extensive modules that use machine learning to detect and recognize faces.

DEMO TIME

DEMO TIME

(1) Facial Recognition

(2) Logging Files

(3) Log-in System with Authentication

(4) Email

Q & A

Facial Recognition Project

Carl Vincent Saban | Monte Kang | Jun Wang

Thanks.

REFERENCE

https://github.com/ageitgey/face_recognition

<http://www.csc.kth.se/~vahidk/papers/KazemiCVPR14.pdf>

<https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>

<https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/>

<https://www.learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/>

<http://blog.dlib.net/2016/10/easily-create-high-quality-object.html>

<https://pypi.org/project/mtcnn/>