Developing a Face Recognition System for Indoor Security

By

Selen Coşkun
Yücel Özdemir
Nisa Pınar Rüzgar

TERM REPORT of
CENG416 Thesis and Seminar 2

A Thesis Report Submitted to the
Faculty of Engineering in Partial Fulfilment of the
Requirements for the Degree of

BACHELOR OF SCIENCE

Department: Computer Engineering

İzmir Institute of Technology
İzmir, Turkey

June 2019

We approve the thesis of **Selen Coşkun, Yücel Özdemir, Nisa Pınar Rüzgar**

Date of Signature

.....

Assoc. Prof. Yalın Baştanlar

Supervisor

Department of Computer Engineering

Table of Contents

1. THE DESCRIPTION OF WORK 4
2. THE WORK PLAN 5
2.1 Overall Strategy 5
2.2 Process Model 5
2.3 Iterative Model 6
2.4 Work Description and Summary Effort 7
3. ANALYSIS & DESIGN 11
3.1 Requirements Specification 11
3.1.1 Functional Requirements 11
3.1.2 Non-Functional Requirements: 12
3.2 Analysis Model 13
3.2.1 Use Case Diagram 13
3.2.2 Design Class Diagram 14
3.2.3 Sequence Diagram 15
4. SOLUTION/PRODUCT & RESULTS 16
4.1 Flow for Use Cases 16
4.2 Experimental Results 21
4.2.1 Face Detection Part: 21
4.2.2 Face Recognition Part: 24
4.3 Installation Requirements 25
4.4 Satisfying Non-functional Requirements 26
5. RELATED WORK/SIMILAR SOLUTIONS 27
6. CONCLUSION & IMPACT 28
7. REFERENCES 29
8. APPENDIX 30

1. THE DESCRIPTION OF WORK

Providing the security of home has been always a necessity. Emerging technologies offer us new methods for this. Security camera systems are one of those technologies which could be found in most of the places.

In our project, we aimed to ensure the safety of indoors by using a smart camera system. Most of the security systems have motion detection technology. However, we want to implement a security system which works with face recognition so that security system warns the owner(s) of the house only by sending a picture via android application when a foreign person is detected. Since pets do not create a danger, the security system does not have to classify them. Our system components are a camera, a computer application, and an android application. We have implemented a computer application which learns, detects and sends pictures, and an android application which turns the computer application on/off and receives pictures.

Firstly, faces of house owners should be introduced to the security system. The average number of family members are 3 or 4 and we are 3 people, so we think the number of project members is enough to success this project. This security system is active/inactive when the owner(s) turn it on/off via on/off button of an android application. When the security system detects a face, it compares with the pre-introduced faces, and if the detected face does not match with a face belongs to homeowners, then it warns the homeowner(s) by sending a picture of the unmatched face to the android application via the internet. In addition, the security system should be able to work in the soft light.

During the development of face recognition part of the project, we have followed the 4 main steps below:

1. Finding all the faces in the image with Histogram of Oriented Gradients (HOG)
2. Posing and projecting faces with face landmark estimation
3. Encoding faces with algorithms of Convolutional Neural Network
4. Recognizing the faces with SVM classifier

During the project, we are planning to use Python, Java, Android Studio, OpenCV, OpenFace, Dlib and Torch on Linux.

2. THE WORK PLAN

2.1 Overall Strategy

Naturally, we have begun our project with some research about existing applications and the development kits. We have learnt some SDKs and APIs including Android Studio, Dlib and OpenFace. Android Studio enables the house owners to display an image sent by the security system to an Android application. Dlib detects faces and creates a dataset whereas OpenFace aligns faces, and Scikit learn classifıes faces. Our schedule have continued with the design of user interface and system components. We have created use-case, component, class, etc. diagrams for easier implementation. These diagrams have made our development easier.

2.2 Process Model

Among the software process models, the most suitable one for our project is the iterative process model because this model have modules and these modules are being developed. When all modules are developed, the project is completed.
In our project, we have a face recognition system at the end. In order to develop such a system, the face must first be identified, and the face must be recognized. Then, to interact with the user, the system interface and an Android application should be developed.
Also, it is easy to manage each iteration and more flexible to changing needs of the project. Thus, iterative model is a good choice for the software process model for our project.

The steps which were used in the project:

Planning: We have determined the tools and technologies to carry out the project.

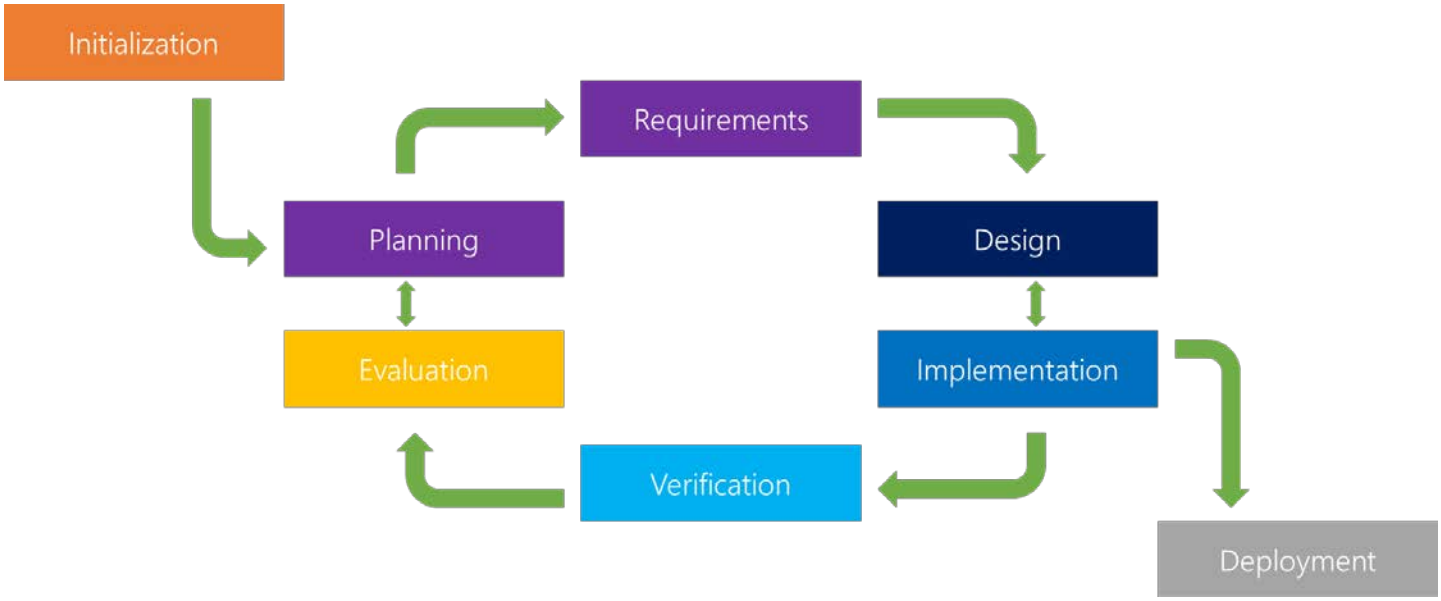
Analysis & System Design: Defining overall system architecture by building some diagrams.

Implementation: Implementing system interface. By using OpenFace, Dlib, Scikit Learn libraries carrying out the face detection and recognition.

Testing: With our team and several other people, making some testing to verify whether the system recognizes our team and sends a notification to our team members cellphones when it detects the other people's faces.

Evaluation: Examining the status of the project with our advisor and decide whether it suits the requirements.

2.3 Iterative Model





2.4 Work Description and Summary Effort

Table 2.1: Work package list

Work package No[1]	Work package title	Type of activity[2]	Lead participant No[3]	Lead participant short name	Person-months [4]	Start month [5]	End month
1	Research and Design	MGT	1,2,3	Nisa Yücel Selen	1.5	0	0.5
2	Face Detection	MGT	1,2,3	Nisa Yücel Selen	3	0.5	1.5
3	Dataset Creation with Sample Images	MGT	1,3	Nisa Yücel	2	1.5	2.5
4	Face Recognition	MGT	1,2,3	Nisa Yücel Selen	6	2.5	4.5
5	System Interface	MGT	1,2	Yücel Selen	2	4.5	5.5
6	Android Application	MGT	2,3	Nisa Selen	3	5.5	7
		TOTAL			17.5		

1 Workpackage number: WP 1 – WP n.

2 Please indicate one activity per work package:

SUPP = Support activities; MGT = Management of the consortium; OTHER = Other specific activities, if applicable.

3 Number of the participant leading the work in this work package.

4 The total number of person-months allocated to each work package.

5 Measured in months from the project start date (month 1).

Table 2.2: Work package description

Work package 1: Research and Design

Objectives
Investigating possible development kits for our project

Description of work
This work package includes the research and learning part of our project. Task 1: Drawing Use cases and determining functional and non-functional requirements This task was done in order to get a strong base for our application. This strong base was obtained by observing the interaction between user and system and by stating all possible cases in this interaction. Task 2: Researching about libraries. Learning for face detection: Nisa Pınar Rüzgar, Yücel Özdemir, Selen Coşkun

Work package 2: Face Detection

Objectives
Detecting the human faces in an image

Description of work
This work package includes the face detection method called Histogram of Oriented Gradients in Dlib. Thanks to this method, we have come up with 68 specific points (called landmarks) that exist on every face, the top of the chin, the outside edge of each eye, the inner edge of each eyebrow (Appendix Figure 1). Each of us have participated in this work package for providing the group motivation for the next work packages. Tasks: Learning OpenCV and Dlib for face detection: Nisa Pınar Rüzgar, Yücel Özdemir, Selen Coşkun

Work package 3: Dataset Creation with Sample Images

Objectives
Saving detected faces in JPG format in order to create dataset

Description of work
Name of a person is taken, and a folder is created with this name. Then, by using the Histogram of Oriented Gradients algorithm (Appendix Figure 2), detected faces from security system are saved to the folder by aligning images in JPG format (Appendix Figure 3). Each person has own folder tagged with his/her name. All these stuffs was made by Nisa Pınar Rüzgar and Yücel Özdemir.

Work package 4: Face Recognition

Objectives
Recognizing each human face that saved in dataset by using OpenFace

Description of work
Our system detects the human faces and classifies them. We have used Convolutional Neural Network of Torch for generating embeddings belongs to each face and save them. Then Torch builds a classifier using neural network algorithms. With SVM (Support Vector Machine), we have classified the faces. This task was done by all team members.

Work package 5: System Interface

Objectives
Developing an interface for desktop application

Description of work
In this work package, we have created a simple interface for the user. We are planning to implement it on Java. You can see UI we expect to build. This task was made by Yücel Özdemir and Selen Coşkun.

Work package 6: Android Application

Objectives
Developing an android application

Description of work
Users can turn on/off the security system. They also can receive notification which displays unknown person's image that is downloaded from the webpage. This work was made by Nisa Pınar Rüzgar and Selen Coşkun.

Deliverables WP 6 was delivered in 7 months.

Table 2.3: Summary of staff effort

Participant no./short name	**WP1**	**WP2**	**WP3**	**WP4**	**WP5**	**WP6**	**Total person months**
Selen	0.16	0.33	0	0.66	0.5	0.75	2.33
Nisa	0.17	0.33	0.5	0.66	0	0.75	2.33
Yücel	0.17	0.33	0.5	0.66	0.5	0	2.33
Total	0.5	1	1	2	1	1.5	7

3. ANALYSIS & DESIGN

3.1 Requirements Specification

3.1.1 Functional Requirements

F.R.1 Turn on/off: The users should be able to activate or disactivate the security system by using their android application.
Actors: User, Android Application, Computer
Precondition: User must download the Android application. Also, both the security system and the android application need to be connected to the internet.
Flow:

- For turning on or off the security system, user must press the 'Turn On' or 'Turn Off' button on their android application.

F.R.2 Get Notification: If a face not in the dataset is detected, a notification including a picture of the detected face must be sent to the android application.
Actors: Computer, Android Application
Precondition: The system must be turned on, connected to the internet. Also, the android application must be connected to the internet.
Flow:

- The security system detects a face while it is open.
- Compares that face with other faces in the dataset.
- If it doesn't match with any face in the dataset then, it sends the picture to the android application through the server.

F.R.3 Save Image: Users can enlarge unknown images and save them to storage of their mobile phones.
Actors: User, Android Application
Precondition: The system should be connected to the server and some images should exist in the server.
Flow:

- Detected unknown image is sent to Android application.
- The user saves the image to his mobile phone after touching the image.

F.R.4 Add new user: The users of the system must be introduced to the system.
Actors: User, Computer
Precondition: System is running, video camera is turned on.
Flow:

- Registration page is opened, and user adds new user to the system by entering new user's name and password.
- User should press 'Recognize' button to start video recording, and he/she should press 'Q' button on the keyboard to terminate capturing.

Postcondition: User information saved in the system. System runs re-training.

3.1.2 Non-Functional Requirements:

- Classifying the person must be faster than 2 seconds.

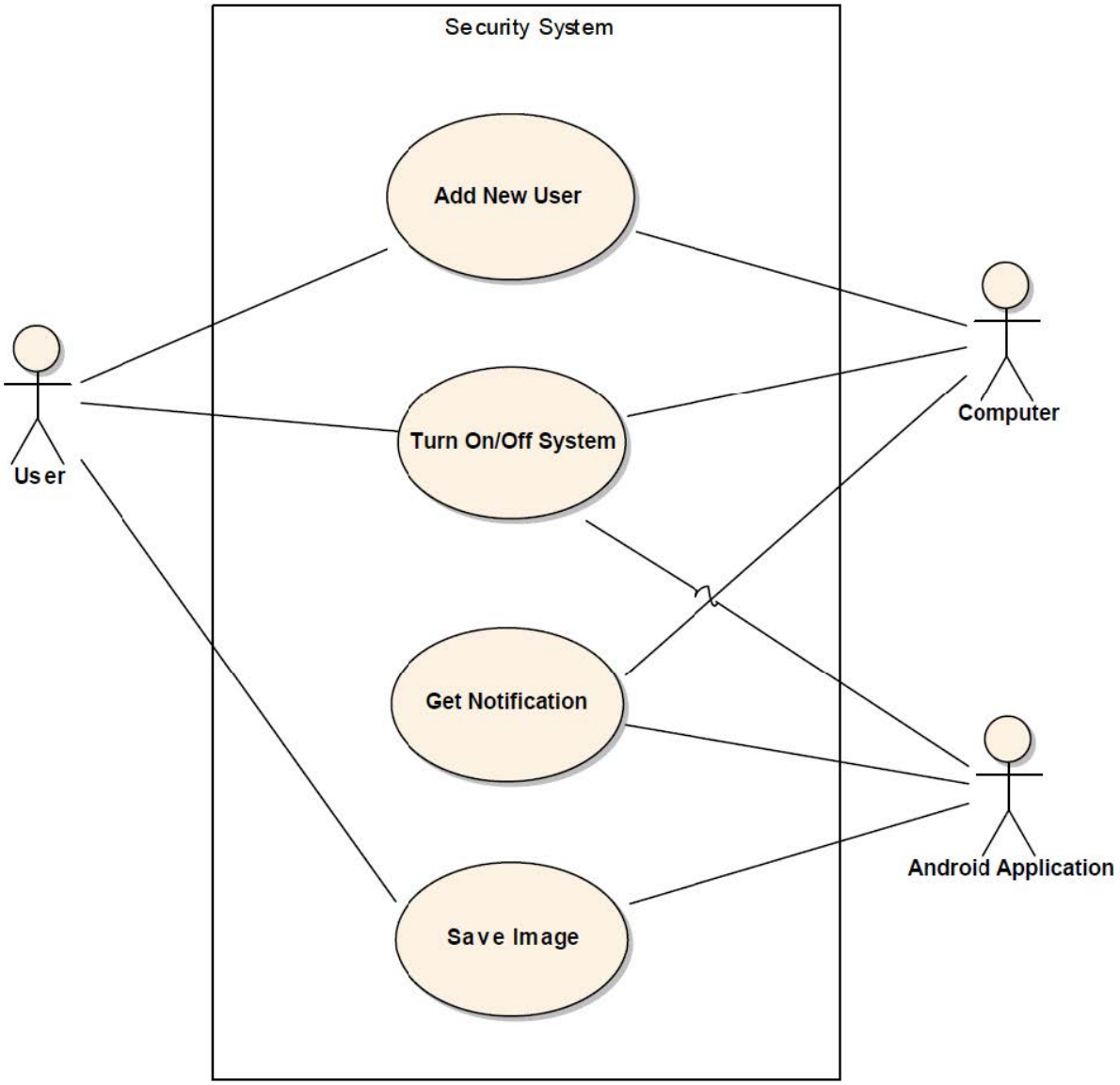
- Accuracy of recognizing a face which belongs to users of the system must be more than 98% whereas 90% for other people.

- Sending the image from the camera system to the Android application mustn’t take longer than 5 seconds.

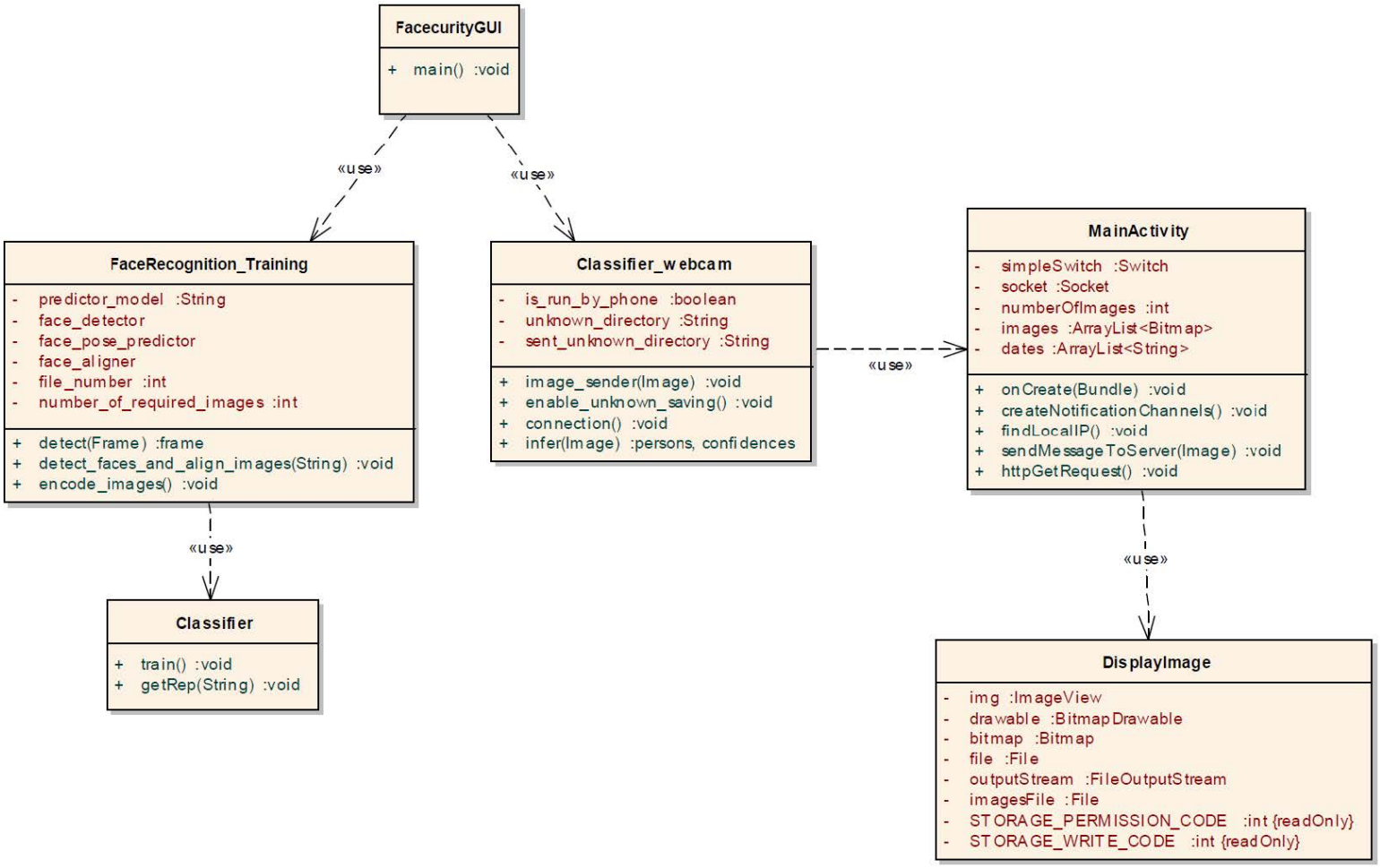
- This application should be able to work on Android 7.1.1 or greater.

3.2 Analysis Model

3.2.1 Use Case Diagram

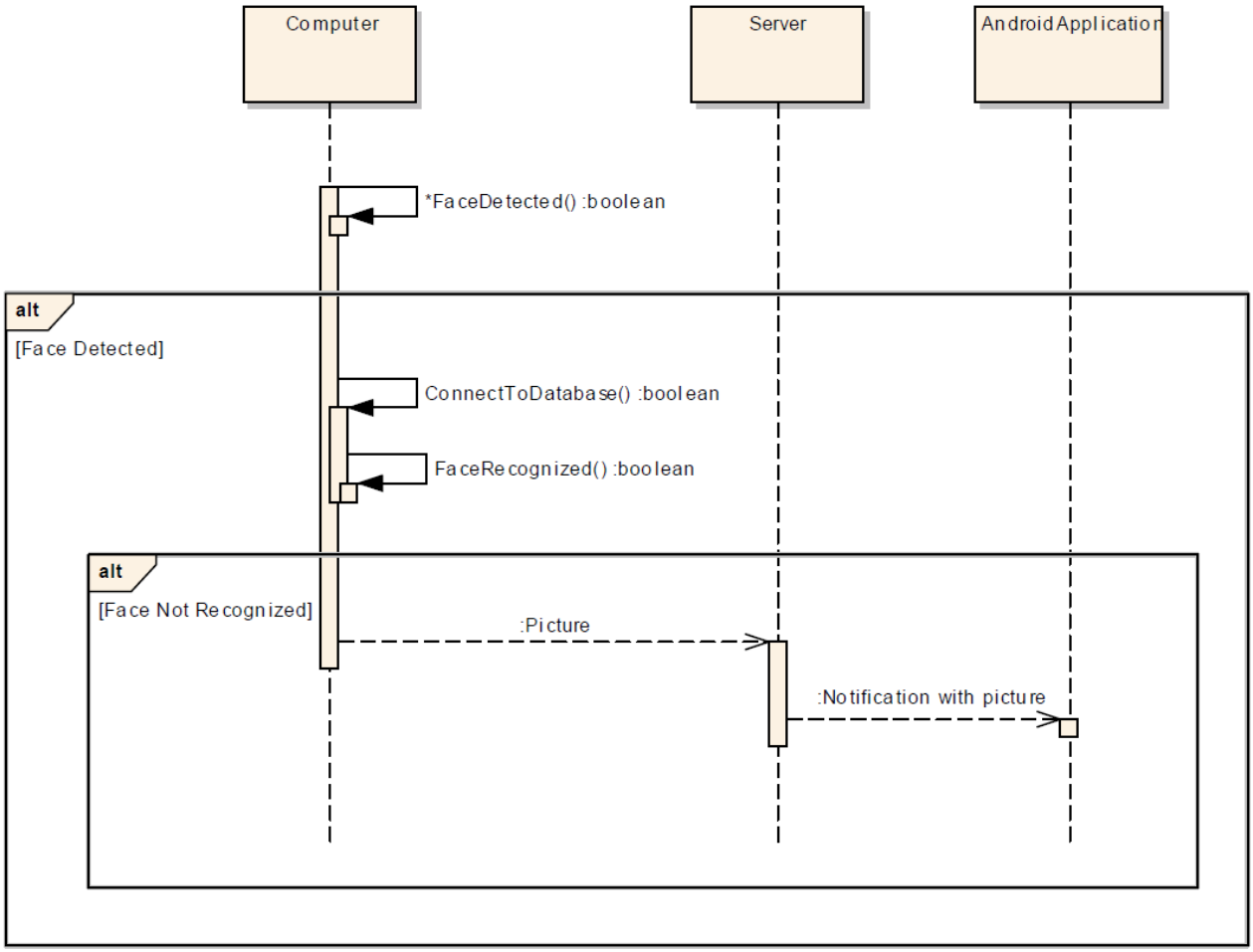


3.2.2 Design Class Diagram



3.2.3 Sequence Diagram

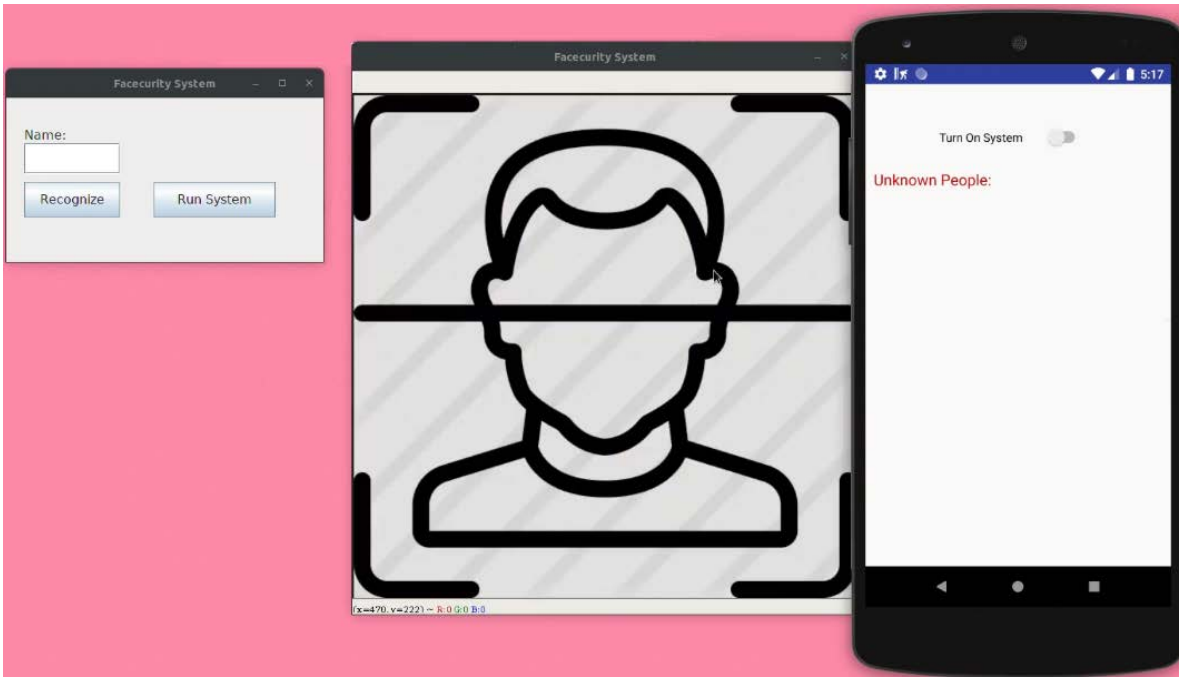
Notification Sequence Diagram

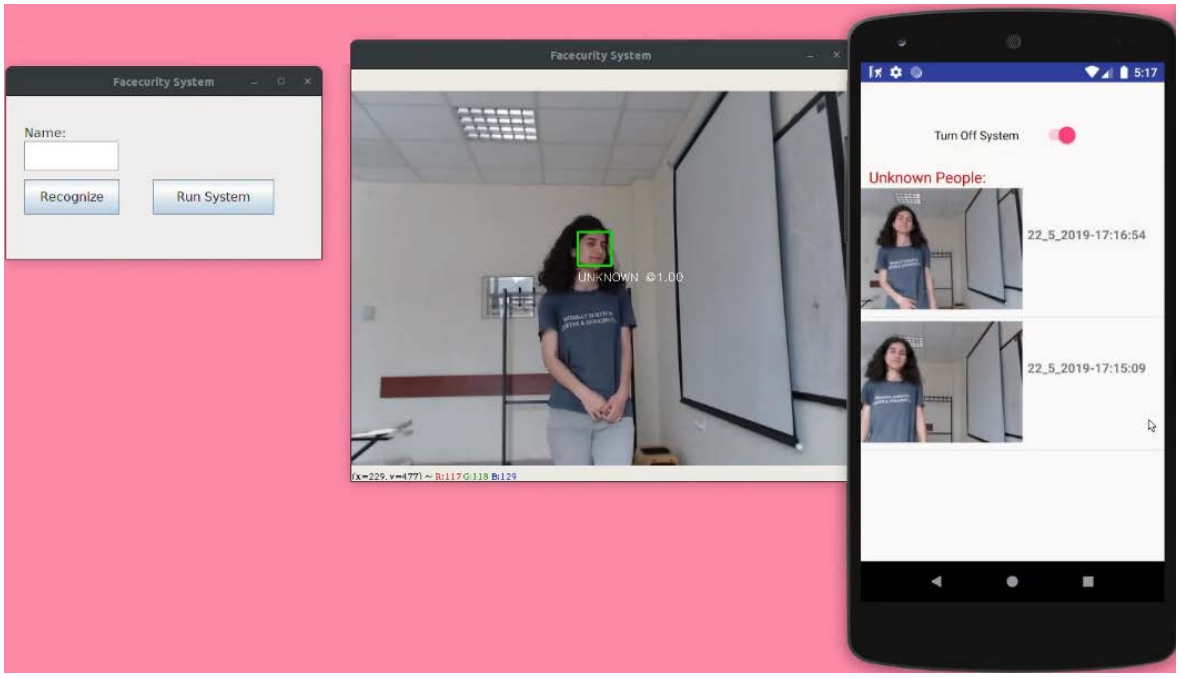


4. SOLUTION/PRODUCT & RESULTS

4.1 Flow for Use Cases

Turn On/Off System:
The system is executed on PC, and PC waits for a trigger from Android application to start classifying.



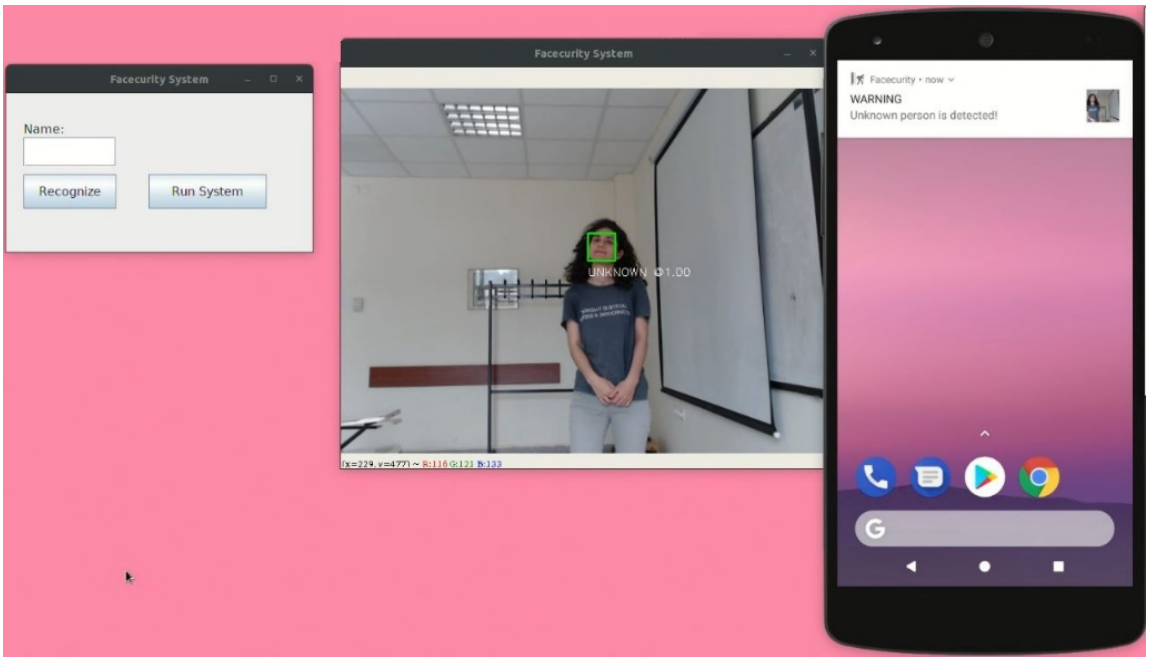


Get Notification:

First of all, the system has 3 classes.



Nisa is an unknown person and her image is sent to mobile phone with a notification.



Save image:

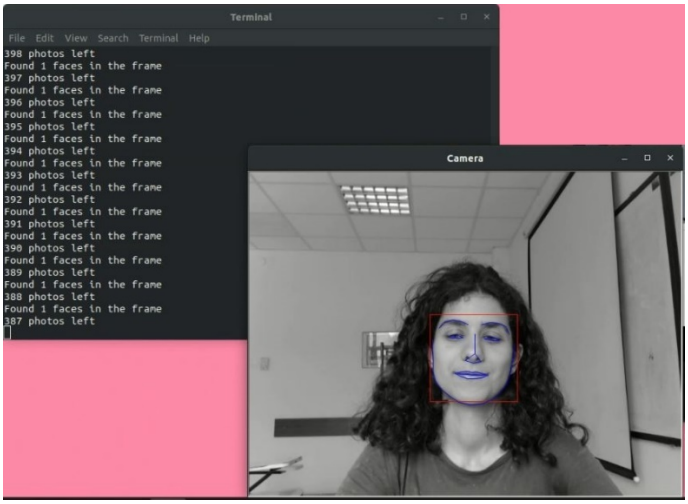
If user wants, the user can save the image to his mobile phone.



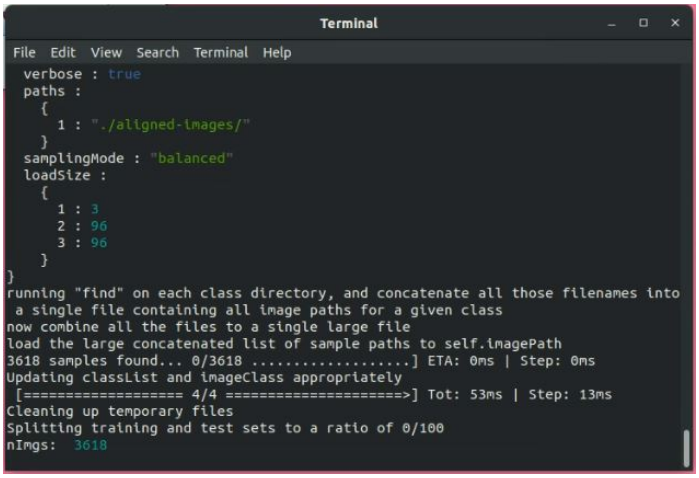
Add New User:



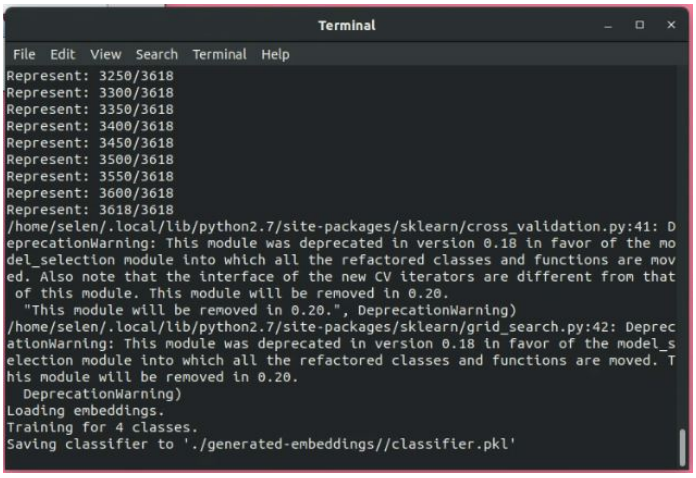
i. Nisa enters her name to text field to introduce herself.



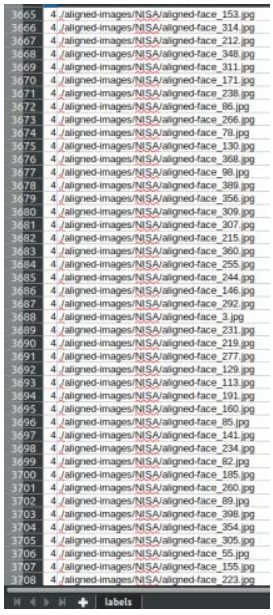
ii. The system captures 400 images (96x96x3) of Nisa by aligning to transfer measurements of her face to the neural network.



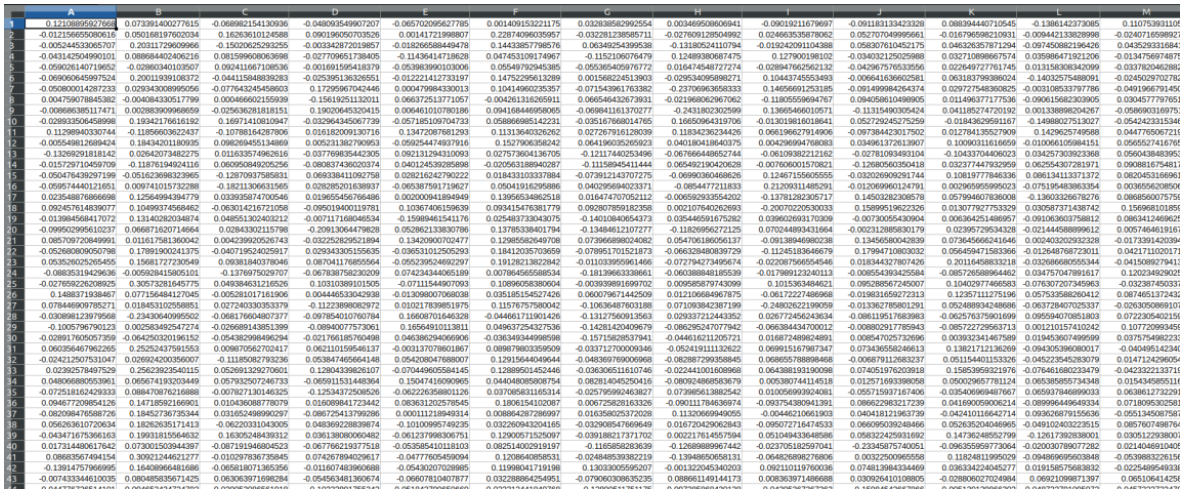
iii. After captured 400 images of Nisa, the neural network starts to generate embeddings (vectors with length of 128).



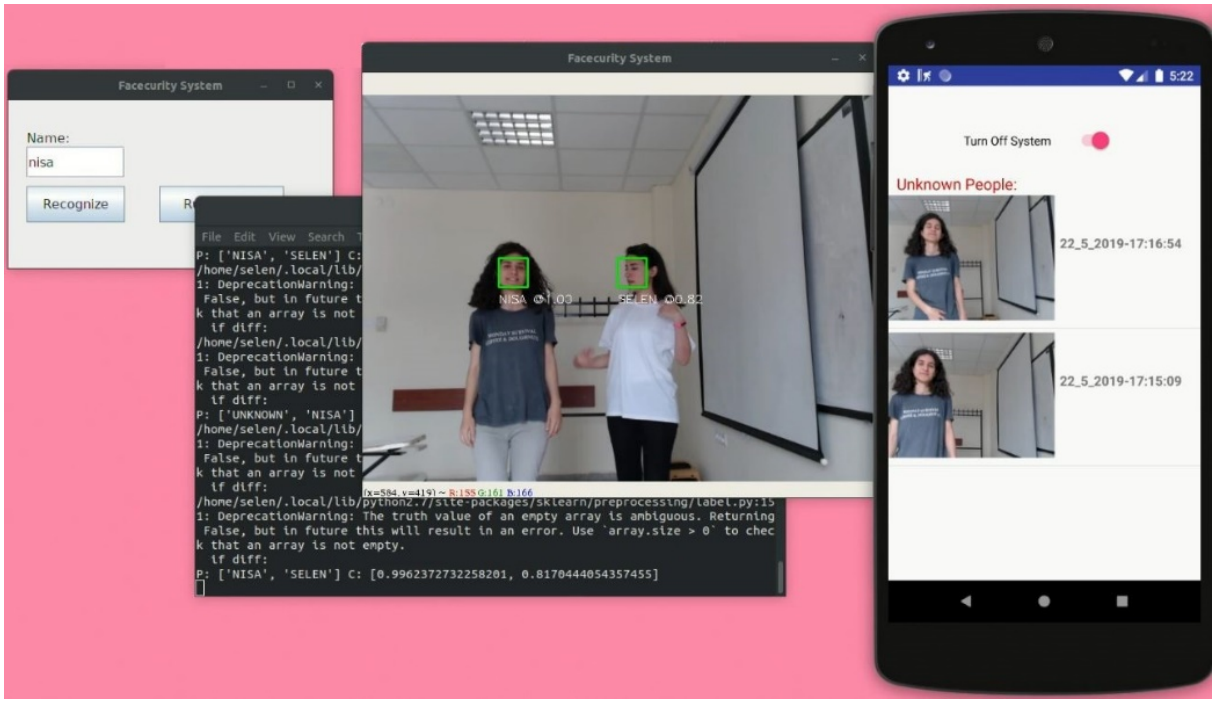
iv. Vectors with length of 128 are generated, and 4 classes are created, classes are Selen, Yücel, Nisa and Unknown. Then trained model is saved as classifier.pkl



Label



Embeddings



v. After the system is activated by turning on in Android application, the system starts to classify faces, and now Nisa is a known person.

4.2 Experimental Results

We have conducted some experiments to select the most accurate face detection and face recognition methods.

4.2.1 Face Detection Part:

We have compared performance of Viola-Jones, MTCNN and HOG algorithms on both well-illuminated and dim environments.

For well-illuminated environment:

- We have collected 900 images, each having 2 faces in it, from a video that lasts 30 seconds in illuminated environment to construct our dataset for illuminated environment.



Viola-Jones

Accuracy = 0.71
Precision = 1, Recall = 0.71
F1-score = 0.83

	Actual positive	Actual negative
Predicted positive	1261	0
Predicted negative	505	0

MTCNN

MTCNN contains 3 different CNN networks (P-Net, R-Net, O-Net). In the first stage, it produces candidate windows through a shallow CNN. Then, it refines the windows by rejecting a large number of non-faces windows through a more complex CNN. Finally, it uses a more powerful CNN to refine the result again and get the final result.

Thresholds for P-Net, R-Net, O-Net = [60, 60, 70] gave the below result.

Accuracy = 0.89
Precision = 0.99, Recall = 0.90
F1-score = 0.94

	Actual positive	Actual negative
Predicted positive	1590	16
Predicted negative	176	0

We couldn't change the threshold values for Viola-Jones and HOG. We wanted to increase the threshold values of MTCNN to get 0 falsely positive images like the HOG and the Viola-Jones to better compare the methods. We have tried different threshold values and finally;

When thresholds for P-Net, R-Net, O-Net = [70, 75, 75] we get False-Positive = 0.

Accuracy = 0.88
Precision = 1, Recall = 0.88
F1-score = 0.93

	Actual positive	Actual negative
Predicted positive	1553	0
Predicted negative	209	0

HOG

Accuracy = 0.84
Precision = 1, Recall = 0.84
F1-score = 0.91

	Actual positive	Actual negative
Predicted positive	1493	0
Predicted negative	273	0

For dim environment:

- We have collected 1400 images, each having 2 faces in it, from a video that lasts 50 seconds in illuminated environment to construct our dataset for dim environment.



Viola-Jones

Accuracy = 0.38
Precision = 1, Recall = 0.38
F1-score = 0.55

	Actual positive	Actual negative
Predicted positive	1080	0
Predicted negative	1736	0

MTCNN

When thresholds for P-Net, R-Net, O-Net = [60, 60, 70]
Accuracy = 0.34
Precision = 0.99999, Recall = 0.35
F1-score = 0.51

	Actual positive	Actual negative
Predicted positive	984	1
Predicted negative	1832	0

When thresholds for P-Net, R-Net, O-Net = [70, 75, 75]
Accuracy = 0.30
Precision = 1, Recall = 0.30
F1-score = 0.46

	Actual positive	Actual negative
Predicted positive	845	0
Predicted negative	1971	0

HOG

Accuracy = 0.56
Precision = 1, Recall = 0.56
F1-score = 0.72

	Actual positive	Actual negative
Predicted positive	1592	0
Predicted negative	1224	0

Result of Face Detection Experiments:

- For our project it's good to be able detect faces in dim environments as well as illuminated environments.
- For illuminated environments HOG and MTCNN are better than Viola-Jones, and there is no big difference in between HOG and MTCNN while for dim environments HOG gives significantly better results compared with the other methods.

Depending on the accuracy and precision-recall metrics we chose to use the HOG method for face detection.

4.2.2 Face Recognition Part:

For the face recognition part, 2 face recognition approaches have been used and tested; an SVM classifier and an MLP classifier. Their performances with respect to their accuracies on the test sets of unknown and known people are compared.

- For training set:
  - 200 images from each known person have been collected from the camera for both SVM and MLP.
  - 200 images of unknown people are collected for MLP and 2500 images of unknown people are collected for SVM.
- For test set: 900 images from known people and 1350 images from 6 different unknown people have been collected. Some samples are below:





- **The accuracy of classifying a defined person** is 0.83 for SVM and 0.85 for MLP.
- **The accuracy of classifying an unknown person** is 0.96 for SVM and 0.90 for MLP.

Result of Face Recognition Experiments:

- Because, correctly classifying an unknown person is more important than correctly classifying a known person for us, we chose SVM for our project.

4.3 Installation Requirements

Programming Languages:

- Python 2.7
- Java 8
- PHP 7.1.4

Libraries:

- Dlib 19.17
- OpenFace 0.2.0
- Torch7
- Scikit Learn 0.20.3
- OpenCV 4.1.0.25

Tools:

- PyCharm
- Android Studio
- Eclipse
- WampServer

Dependencies:

- **Scikit Learn:**
  - Python 2.7
  - NumPy 1.16.2
  - Scipy 1.2.1
  - Joblib 0.13.2
- **Dlib:**
  - NumPy 1.16.2
  - Scipy 1.2.1
  - matplotlib 2.2.4
  - scikit-image 0.14.2
  - Scikit Learn 0.20.3
  - ipython 5.8.0
- **OpenCV:**
  - NumPy 1.16.2
  - Scipy 1.2.1
  - matplotlib 2.2.4
  - scikit-image 0.14.2
  - Scikit Learn 0.20.3
  - ipython 5.8.0
- **Torch:**
  - Lua 5.2
- **OpenFace**
  - OpenCV 4.1.0.25
  - Dlib 19.17
  - Torch7

4.4 Satisfying Non-functional Requirements

- We have tested in different light conditions, the system is classifying for a face in 0.20-0.25 seconds according to light conditions, and classifying is working with 4-5 FPS.

- Accuracy of recognizing a face which belongs to users of the system is more than 83% whereas 96% for unknown people.

- We have tested duration of sending images from the camera system to the Android application 10 times, and we have obtained that 13.6 seconds overall. In other words, sending an image from the camera system to the Android application takes 1.36 seconds.

- The Android application works on Android 7.1.1 or greater.

5. RELATED WORK/SIMILAR SOLUTIONS

Smart security systems are used especially in the last a few years, because people give importance to smart devices and people trust them in every field. We usually see news of artificial intelligence, autonomous robots, some cameras powered by artificial intelligence, also we can see a lot of projects about object detection and face recognition on LinkedIn, because people wonder these technologies and care them. There is no a lot of projects about our topic, only some companies produce some productions about smart security systems. Here are some of the best ones:

Nest Cam IQ Indoor: The new indoor security camera from Nest Labs, the Nest Cam IQ ($299), is its first to offer face-recognition technology. It uses a Supersight automatic zooming and tracking feature to identify people in its field of view and match their image to pictures in your library so you can keep track of who is coming and going, and when. [6]

Tend Secure Lynx Indoor: The Tend Secure Lynx is proof that good things do indeed come in small—and affordable—packages. This pint-sized home security camera not only delivers colorful 1080p video and sharp night vision, it is chock-full of features including motion-triggered notifications and recordings, two-way audio, free cloud storage, and facial recognition technology. [7]

Honeywell Smart Home Security: Honeywell's Smart Home Security Starter Kit ($449) comes with everything you need to keep track of what's going on in your home even when you're not there. This DIY security system is easy to install and uses an intuitive mobile app to send alerts when motion is detected or if a door or window has been opened. The base station has a 1080p camera that records motion and stores it in the cloud for free, and it offers a handful of neat features, including built-in Amazon Alexa capabilities, face recognition, and an Enhanced Deterrence option that plays sounds to make it seem like someone is home while you're away. [8]

6. CONCLUSION & IMPACT

In today's world every new technology or product must be carefully examined and analyzed before publishing it. Before even start to project almost everybody thinks its economic value or sustainability in the future. However, it is worth considering the more important constraints than these. In our application here are the what we have considered:

Health: Our project has no threat to human health.

Security: There might be some security problems in our project. One of them is that bad people can hack the security system. Therefore, they can steal pictures of landlords from dataset, and they can introduce own pictures to the security system. Also, bad people can watch inside the place thereby hacking the security system.

Ethics: Sending picture which belongs to unknown people to host is an ethical problem according to Kant.

Economics: On economic side of our application, we do not have money demand from Google Play Store users. Users only responsible for buying the security system.

Sustainability: Almost all security system evolving to biometric security systems and face recognition is gaining importance day by day. That's way we expect our project sustain for a very long time period.

Producibility: It can be produced, because there are some productions like our vision in product line of some technological companies.

7. REFERENCES

[1] www.wikizero.biz/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvQW5kcm9pZF9TdHVkaW8

[2] www.python.org/about

[3] www.medium.com/sololearn/why-is-c-among-the-most-popular-programming-languages-in-the-world-ccf26824ffcb

[4] www.opencv.org

[5] www.cmusatyalab.github.io/openface

[6] www.pcmag.com/review/354592/nest-cam-iq

[7] www.pcmag.com/review/354449/tend-secure-lynx-indoor

[8] www.pcmag.com/review/363772/honeywell-smart-home-security-starter-kit

8. APPENDIX

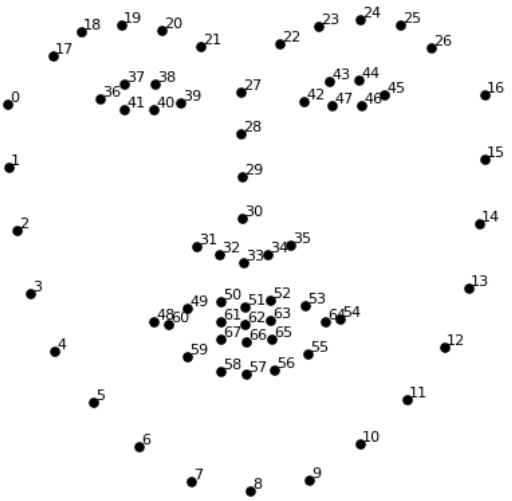


Figure 1: 68 Landmarks on Every Face

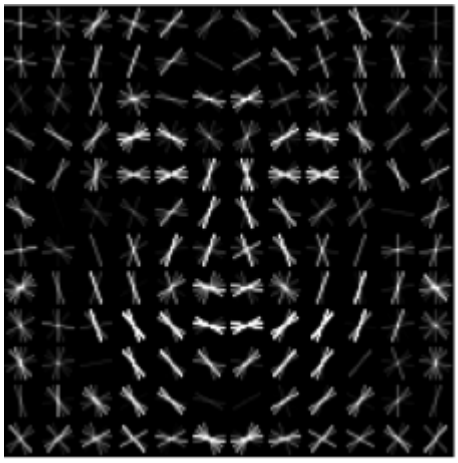


Figure 2: Histogram of Oriented Gradients Concept on Face

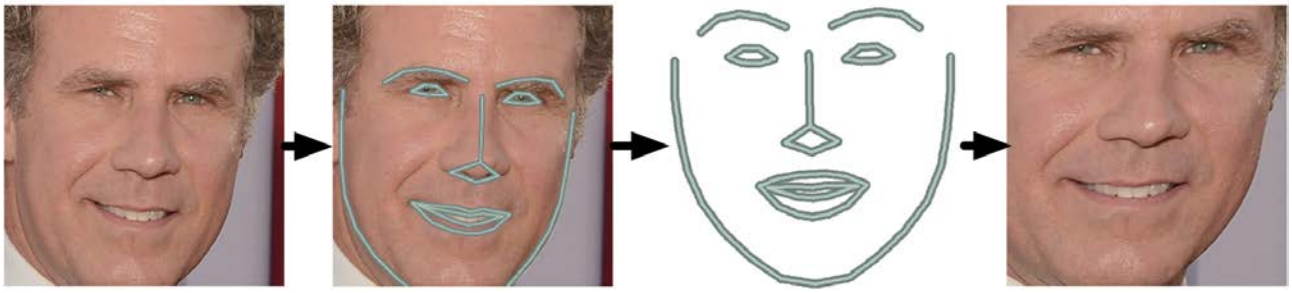


Figure 3: Process from Detecting to Aligning