Development of Real Time Face Recognition System Using OpenCV

D. Mary Prasanna [1], Ch. Ganapathy Reddy [2]

[1] PG Student, Dept of ECE (DECS), GNarayanamma Institute of Technology & Science for women, Hyderabad, TS, India
[2] Professor, Dept of ECE , GNarayanamma Institute of Technology & Science for women, Hyderabad, TS, India

---***---

Abstract - *A real-time, GUI based automatic Face detection and recognition system is developed in this project. It can be used as access control system by registering the staff or students of an organization with their faces, and later it will recognize the people by capturing their images with faces, when they are entering or leaving the premises. The system is implemented on a desktop with a Graphical User Interface, Initially it detects the faces in the images that are grabbed from a web camera. All the tools and operating, used to develop this system like Ubuntu, open Face, Python ..., are open source tools.*

This real time GUI based *face recognition system is developed using Open source tool Open face. Open Face is the face recognition tool developed by Carnegie Mellon University, using OpenCV. Open Face, consists in a broader Prospective, three phases: Detection, Feature extraction, and Recognition. The dimensionality of face image is reduced by the Histogram of Oriented Gradients (HOG) and this algorithm is developed to detect frontal views of faces. After detecting the face part of image, extract the 128 face features for the given image by using a Deep Neural Network algorithm and the recognition is done by the Support Vector machine (SVM) classifier. HOG is one of the most popular representation methods for a face image. It not only* reduces *the dimensionality of the image, but also extracting the facial features of the given images, and retains some of the variations in the image data. So dimensionality of face image reduced by HOG using deep learning algorithm and recognition is done by SVM approach.*

Key Words*: *Face Recognition, HOG, DNN, SVM, Open Face, OpenCV, GUI

1. INTRODUCTION

Face Recognition is used to recognize a person by using some features of that particular person's face, by matching with stored models of each individual face in a group of people. Face recognition is the natural way of identification and authenticating a person. Face Recognition plays an important role in human's day-to-day communication and daily lives. Security and authentication of a person is crucial in any industry or establishment. So, in today's environment, there is a great deal of interest in automatic face recognition using computers or devices for verification of identities round the clock and sometimes remotely. Face recognition has become one of the most challenging and interesting problems in the pattern recognition and image processing.

1.1 Motivation

Face biometrics are useful for a person's authentication, which is a simple and non-intrusive method. Face recognition have substantial potential in two areas: a) it can help the users to caught criminals and terrorists. b) It can be used in controlling access to areas where security risks are especially high c) It can also be used for automatic access control. High security is required for accumulation of data and information. Government can use it for Law enforcement, Voter Verification, Security, Immigration, Legislature, prisons, Benefit Payments and to find out Missing Children and robbers. It can also be used in the Commercial applications like Banking, Gaming Industry, Residential Security, Internet, E-commerce, and Health Care.

Therefore it expected to have lot of demand for real time face recognition systems for access control and not only for investigation of unlawful activities, it will be also useful to stop the of unlawful or terrorist activities. However, if it is not a real system, we can detect the activity after the incident is occurred but we cannot stop/prevent it before it happens. If the Face recognition systems to be used to stop any unlawful or terrorism activity or use it for access control, an user friendly real time face recognition system is required.

1.2 Problem Statement

The Aim of the Project is to develop a real time GUI based face recognition system that would use a camera to view the outside world and then detect and recognize the individual faces in the scene by using Open Face. Open Face is the face recognition tool developed using OpenCV, an open source library of modules for Computer Vision and Machine Learning written in C++.

2. LITERATURE SURVEY

This Research work is concentrates on face recognition problem as a part of biometric attendance system.

Face recognition has been an active research area over last 30 years. This Research spans several disciplines such as image processing, pattern recognition, computer vision, and neural networks. Face recognition has applications mainly in the fields of biometrics, access control, law enforcement, and security and surveillance systems.

The problem of face recognition can be stated as follows: Given still images or video of a scene, identifying one or more persons in the scene by using a stored database of faces. The problem is mainly a classification problem. Training the face recognition system with images from the known individuals and classifying the newly coming test images into one of the classes is the main aspect of the face recognition systems.

This Problem seems to be easily solved by humans where limited memory can be the main problem; whereas the problems for a machine face recognition system are:

- Facial Expression Change
- Illumination Change
- Aging
- Pose change
- Scaling factor
- Presence and absence of spectacles, beard, mustache etc.
- Occlusion due to scarf, mask or obstacles in front.

The problem of automatic face recognition (AFR) is a composite task that involves detection of faces from a cluttered background, facial feature extraction, and face identification. A complete face recognition system has to solve all sub problems, where each one is a separate research problem. This research work concentrates on the problem of facial feature extraction and face identification.

Most of the current face recognition algorithms can be categorized into two classes, image template based and geometry feature-based. The template based methods compute the correlation between a face and one or more model templates to estimate the face identity. Whereas feature based methods extract local facial features and use their geometric and appearance properties.

The face is our primary focus of attention in social intercourse, playing a major role in conveying identity and emotion. We can recognize thousands of faces learned throughout our lifetime and identify familiar faces at a glance after years of separation. Computational models of face recognition, in particular, are interesting because they can contribute not only to theoretical insights but also to practical applications.

Computers that recognize faces could be applied to a wide variety of problems, including criminal identification, security systems, image and film processing, and human computer interaction. Unfortunately, developing a computational model of face recognition is quite difficult, because faces are complex, multidimensional, and meaningful visual stimuli. Eigen face is a face recognition approach that can locate and track a subject's head, and then recognize the person by comparing characteristics of the face to those of known individuals.

3. DEMOS IN OPENFACE

Open Face is core provides a feature extraction method to obtain a low-dimensional representation of any face. Demos/classifier.py shows a demo of how these representations can be used to create a face classifier.

In Open Face, to measure the system performance we are having four real time demos. They are

- Real time web demo
- Comparison demo
- Training a classifier
- Real time Sphere Visualization

3.1 Real Time web demo

Real time web demo shows the full face recognition pipeline .In Practice, object tracking like dlib's should be used once the face recognizer has predicted a face.

In the edge case when a single person is trained, the classifier has no knowledge of other people and labels anybody with the name of the trained person.

The web demo does not predict unknown users and the saved faces are only available for the browser session. If you're interested in predicting unknown people, one idea is to use a probabilistic classifier to predict confidence scores and then call the prediction unknown if the confidence is too low.

The drawback of this real time web demo is:

i. The data base is stored in Google Server.
ii. Once the browser is closed the complete data will be erased.

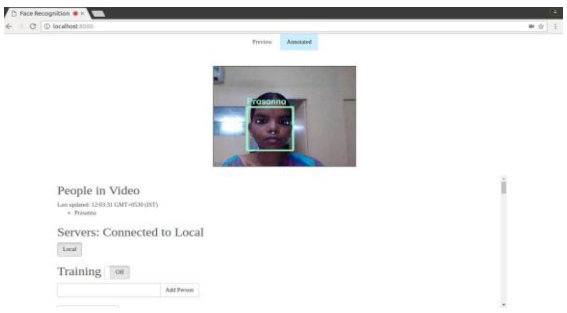


Fig 1: Real time web demo

3.2 Comparison demo

Here in comparison demo, we are comparing the images of two persons with two images of each. To compare the images of two persons we are calculating the Euclidian Distance (L2 Distance). If the Squared L2 distance between representations is less than one then it will be identified as same person, Otherwise it will recognized as different Person.

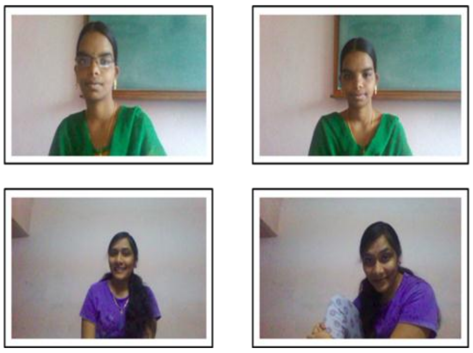


Fig 2: Comparing of two images

The following distance between images of Prasanna and Trishutha were generated with ./demos/compare images/examples/{Prasanna*, Trishutha*} in openface.

mage 1	Image 2	Distance
Prasanna1	Prasanna2	0.474
Prasanna1	Trishutha1	1.045
Prasanna1	Trishutha2	1.370
Prasanna2	Trishutha1	1.352
Prasanna2	Trishutha2	1.470
Trishutha1	Trishutha2	0.152

Table 1: Comparison demo of two images

3.3 Training a classifier

Open face is core provides a feature extraction method to obtain a low-dimensional representations of any face. Demos/classifier.py shows a demo of how these representations can be use to create a face classifier. There is a distinction between training the DNN model for feature representation and training a model for classifying people with the DNN model.

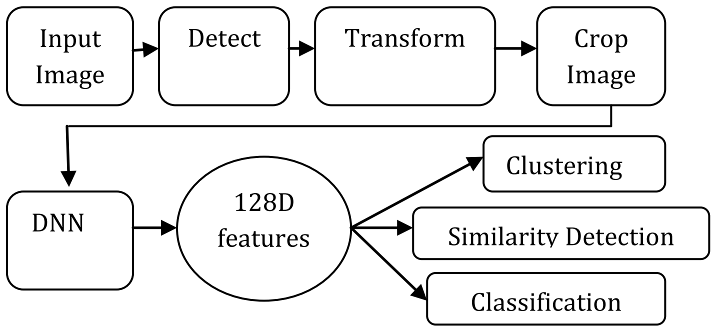


Fig 3: Classification and similarity detection

To Train a classifier we follow the steps given below:

1. Encode a picture using the HOG algorithm to create a simplified version of the image. Using this simplified image, find the face part of the image that most looks like a generic HOG encoding of a face.

2. Figure out the pose of the face by finding the main landmarks in the face. Once we find those landmarks, use them to warp the image so that the eyes and mouth are centered.

3. Pass the centered face image through a neural network that knows how to measure features of the face. Then for every input image it will generate 128 measurements.

4. After generating 128 measurements for the given input image, SVM classifier is used to recognize new face.

Create a classification model:

1. Create raw image directory

Create a directory for your raw images so that images from different people are in different subdirectories. The names of the labels or images do not matter, and each person can have a different amount of images. The images should be formatted as jpg or png.

2. Preprocess the raw images

Here for the given input image , it will detect the face part of the given image and apply the affine transform for the given image. Then extract the features of given image, finally it will aligned (crop) the face part of that particular image.
3. Generate Representations

In this we are generating the 128 facial features of each and every input image. And it will creates the labels and representations of all images.

4. Classification Model

After generating the face features, the system will classify how many people present in the data base. Suppose in my system I am trained 20 persons each of 20 images, then the system will recognized as 20 classifiers are present in the classification model.

After Generating classifier.pkl, The Test images are compared with the database. If the given image is in the training , then it will recognize the particular person name with confidence. Otherwise it will recognize as unable to find a face image.

3.4 Real Time Sphere Visualization

This Real time display is using the Open Face facial recognition system. It is analysing the stream of information coming into the computer from the camera, looking for faces. When a face is found, a rectangle is placed around it on the screen. When more than one person is identified, it differentiates between them with separate colours. The system also represents the face as points on a sphere, as shown in the figure. Similar faces (like identical twins) will be represent nearer to one another on the sphere than the faces that are more visually different. Here the computer uses 128 dimensions to represent the faces instead of the just three physical dimensions represented by the sphere.

S.No	Training-Images (20)	Test-Images	Accuracy (%)
1	Achala	4	100
2	Buvana	4	75
3	Divya	4	75
4	Harika	4	50
5	Majid	4	100
6	Mallesh	4	25
7	Mounika	4	75
8	PLN	4	100
9	Prasanna	4	75
10	Prathap	4	25
11	Ranadheer	4	50
12	Rekha	4	100
13	Renuka	4	75
14	Santhi	4	75
15	Trishutha	4	50

Table 1: Accuracy Table for 15 Training-images each of 4 Test-Images

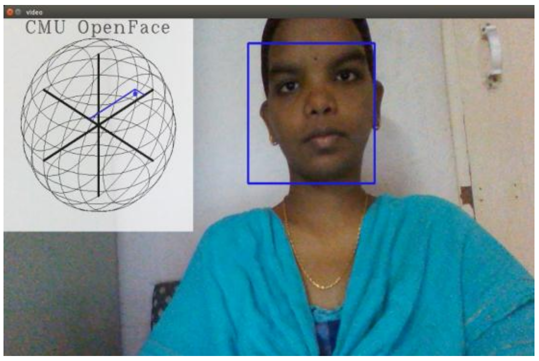


Fig 4: Real time sphere visualization in CMU Open face.

The above fig shows Real time sphere visualization in cmu open face. In this it will detect and recognize the face part of the particular person who ever present in front of the web camera. And whenever the person want to move the positions (like rotation, bending etc.) then the sphere positions will also move based on the Person movements.

If two or more persons present in front of the camera, then it will identify the number of persons. After identifying number of persons; it will detect and recognize the face parts of the given image. Then for each & every person it will detect the face part and appear bounding boxes with different colors. If the Persons want to move the positions then the sphere positions will also move based on the persons movements.

4. PROPOSED SYSTEM

How to build the Open Face is explained in the previous Case. The potential for building Face Recognition systems using Open Face is shown by the examples given at CMU website only. Now how to build a face recognition system, suitable for our institution or for any other application, using open Face will be explained in this chapter with the example of building a face recognition system built for our unit (NERTU, OU).

Once open face is installed and ready to run, the following steps are required to build a face recognition system.

1. Building database by registering the faces of the people to recognized.
2. Detection of Faces from the images of registered people and extraction of facial features from the detected faces.

3. Recognition face in the test image with SVM classifier.

Once the open face is installed the required modules or software's will be readily available to complete these three tasks.

To check whether the system is working properly or not, initially a face recognition system for NERTU having 15 people (staff and students) is built and the same procedure is explained below. Then after that we have built a real time system with GUI.

Twenty images having the face of each person, for 15 people are collected using CHEESE software, which is an open source tool developed by Daniel G. Siegel

So, this database contains 300 face images. It consists of each person with 20 different poses like smiling, looking serious, head bend, eyes moving and closing eyes with very small variations in the database. Totally 300 images are used for training and 60 images are used for testing. Here I will show the result of 15 persons with 20 images.

Here I am not taking any real time pictures. So, the accuracies will be different. To improve the accuracy, increase the number of training images of each person. Then after improving data base I will get better results which I will trained manually in my system. Now I Build a real time GUI based Face Recognition System.

REMARKS:

The accuracy of some of the people in the above table is less due the following reasons.

- Some people are having so much of Pose Variations, which is not present in the training-image folder.
- For Poor Illumination & Brightness conditions
- Because of Expressions, the accuracies may vary for each and every individual.

4.1 Need for an automated System

Today there is a rising need for systems that assist in security and surveillance. The authentication of individuals can no longer be done efficiently by manual methods. For example, consider a concert of a famous musician that incurs huge response with, let us say, 10,000 people attending the show. Imagine guards standing at the front doors, manually letting people in, after checking their tickets. They should check the authenticity of the ticket and also verify if it is the same person. This would surely demand a huge manpower. Also, it is extremely tedious to authenticate a crowd as large as 10,000 people leading to delay in schedule and, not to mention, high probability of occurrence of errors resulting from common mistakes by humans. We, thus, need machine-controlled systems that perform the given task efficiently in a short duration of time with minimal human intervention. Face recognition based authentication is very near to natural way of authentication by people. So in this chapter, a real time GUI based face recognition system built by me is explained below. There are several ways in which this automation of monitoring systems can be achieved.

4.2 Graphical User Interface

The **graphical user interface** (**GUI**), is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. GUIs were introduced in reaction to the perceived steep learning curve of command-line interfaces (CLIs), which require commands to be typed on a computer keyboard.

The actions in a GUI are usually performed through direct manipulation of the graphical elements. Beyond computers, GUIs are used in many hand held mobile devices such as MP3 players, portable media players, gaming devices, smart phones and smaller household, office and industrial controls. The term GUI tends not to be applied to other lower-display resolution types of interfaces, such as video games (where heads-up display (HUD) is preferred), or not including flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in the tradition of the computer science research at the Xerox Palo Alto Research Center (PARC).

Designing the visual composition and temporal behavior of a GUI is an important part of software application programming in the area of human–computer interaction. Its goal is to enhance the efficiency and ease of use for the underlying logical design of a stored program, a design discipline named usability. Methods of user-centered design are used to ensure that the visual language introduced in the design is well-tailored to the tasks.

Here **GUI** is developed using ***Gui***. Here I built a **real time** system is developed using **Qt Designer**. In this GUI based Real time Face Recognition system we are mainly focusing on two phases

- Training Phase
- Test Phase

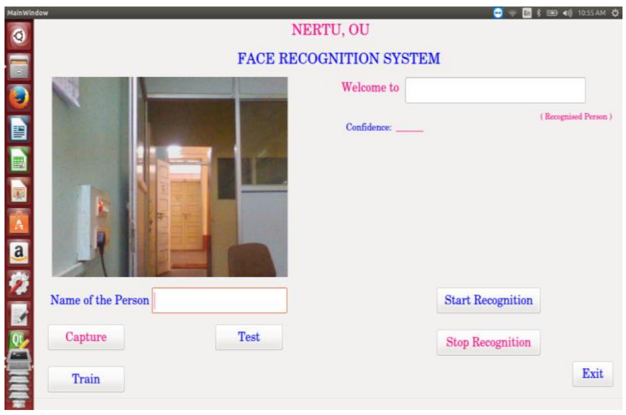


Fig 5: Real time Face Recognition System display

Figure 5 shows the Real time face recognition system display. In Training Phase it is divided in to two parts:

1.Capture
2.Train

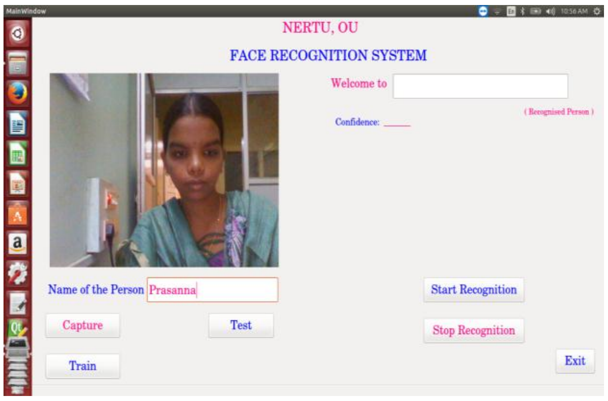
CAPTURE:

For building the training database in real time, to capture the images of a person, first we are enter the person name, then the system will capture the images of that Particular Person standing in front of camers. For Capturing images the following code is used:

```
def capture_Image(self):
i=1
camera=cv2.VideoCapture(0)
while(i<=10):
if not os.path.exists('openface/training-images'):
print 'hello'
os.mkdir('openface/training-images')
name=self.lineEdit.text()
```

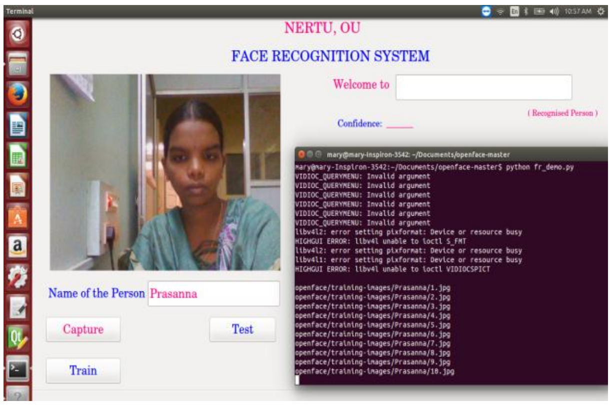
```
if not os.path.exists('openface/training-images/' + name):
os.mkdir('openface/training-images/' + name)
ret, image=self.camera.read()
imagename='openface/training-images/' + name + '/' +
str(i) + '.jpg'
print imagename
```

How to capture the images in a real time based GUI System is given below.



Name of the Person: Prasanna

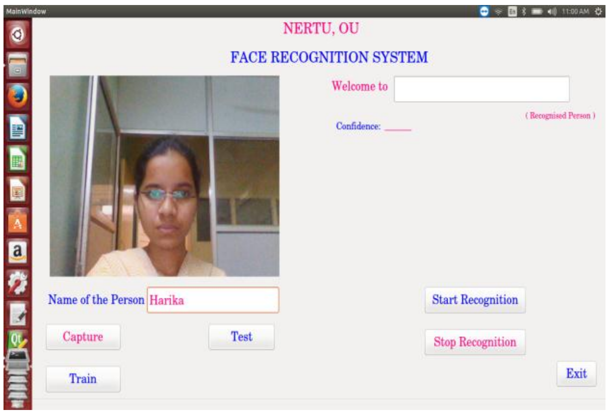
Fig 6: Enter the name of the person as Prasanna



Name of the Person: Prasanna

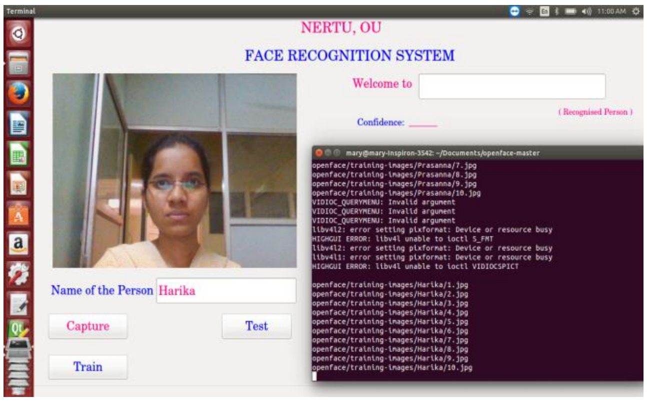
```
mary@mary-Inspiron-3542: ~/Documents/openface-master
mary@mary-Inspiron-3542:~/Documents/openface-master$ python fr_demo.py
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
libv4l2: error setting pixformat: Device or resource busy
HIGHGUI ERROR: libv4l unable to ioctl S_FMT
libv4l2: error setting pixformat: Device or resource busy
libv4l1: error setting pixformat: Device or resource busy
HIGHGUI ERROR: libv4l unable to ioctl VIDIOCSPICT

openface/training-images/Prasanna/1.jpg
openface/training-images/Prasanna/2.jpg
openface/training-images/Prasanna/3.jpg
openface/training-images/Prasanna/4.jpg
openface/training-images/Prasanna/5.jpg
openface/training-images/Prasanna/6.jpg
openface/training-images/Prasanna/7.jpg
openface/training-images/Prasanna/8.jpg
openface/training-images/Prasanna/9.jpg
openface/training-images/Prasanna/10.jpg
```

Fig 7: Press the button Capture to Capture the images into a folder Prasanna



Name of the Person: Harika

Fig 8: Enter the name of the person as Harika



Name of the Person: Harika

```
mary@mary-Inspiron-3542: ~/Documents/openface-master
openface/training-images/Prasanna/7.jpg
openface/training-images/Prasanna/8.jpg
openface/training-images/Prasanna/9.jpg
openface/training-images/Prasanna/10.jpg
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
libv4l2: error setting pixformat: Device or resource busy
HIGHGUI ERROR: libv4l unable to ioctl S_FMT
libv4l2: error setting pixformat: Device or resource busy
libv4l1: error setting pixformat: Device or resource busy
HIGHGUI ERROR: libv4l unable to ioctl VIDIOCSPICT

openface/training-images/Harika/1.jpg
openface/training-images/Harika/2.jpg
openface/training-images/Harika/3.jpg
openface/training-images/Harika/4.jpg
openface/training-images/Harika/5.jpg
openface/training-images/Harika/6.jpg
openface/training-images/Harika/7.jpg
openface/training-images/Harika/8.jpg
openface/training-images/Harika/9.jpg
openface/training-images/Harika/10.jpg
```

Fig 9: Press the button Capture to Capture the images into a folder Harika

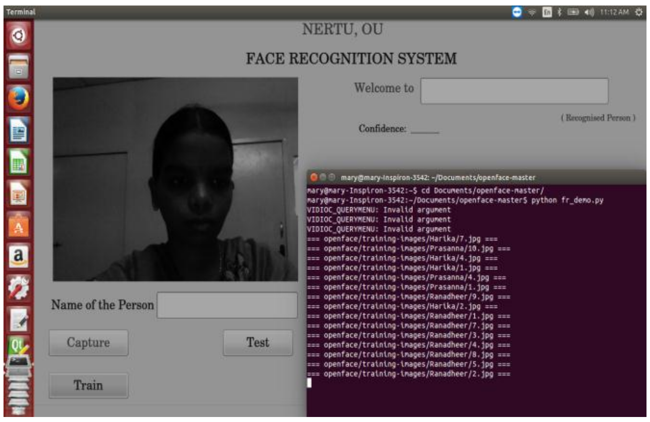
TRAINING PHASE:

After Capturing training images of all the people, we have to train the system to obtain face models for all people. The training procedure is already explained in the previous section.CMU Openface having some models for detecting faces in the images. The shape of the faces is almost similar, throughout the world. So, using those training models, our system is trained which is for detecting faces.

For building the models for recognition of faces, the features should be extracted before obtaining the models for every person. The following script will extract the features and build the models for every person with the face aligned images or faces detected. Here for each and every aligned-image, 128 features are generated.

Shell script for Training (Train_script.sh) :

```
#!/usr/bin/sh
./util/align-dlib.py openface/training-images/ align
outerEyesAndNose ./aligned-images/ --size 96
./batch-represent/main.lua -outDir ./generated-embeddings/
-data ./aligned-images/
./demos/classifier.py train ./generated-embeddings/
```



```
mary@mary-Inspiron-3542: ~/Documents/openface-master
mary@mary-Inspiron-3542:~$ cd Documents/openface-master/
mary@mary-Inspiron-3542:~/Documents/openface-master$ python fr_demo.py
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
VIDIOC_QUERYMENU: Invalid argument
=== openface/training-images/Harika/7.jpg ===
=== openface/training-images/Prasanna/10.jpg ===
=== openface/training-images/Harika/4.jpg ===
=== openface/training-images/Harika/1.jpg ===
=== openface/training-images/Prasanna/4.jpg ===
=== openface/training-images/Prasanna/1.jpg ===
=== openface/training-images/Ranadheer/9.jpg ===
=== openface/training-images/Harika/2.jpg ===
=== openface/training-images/Ranadheer/1.jpg ===
=== openface/training-images/Ranadheer/7.jpg ===
=== openface/training-images/Ranadheer/3.jpg ===
=== openface/training-images/Ranadheer/4.jpg ===
=== openface/training-images/Ranadheer/8.jpg ===
=== openface/training-images/Ranadheer/5.jpg ===
=== openface/training-images/Ranadheer/2.jpg ===
```

Fig 10: Training the database for Face Recognition system for NERTU

Figure 10 shows the training the database **Face Recognition system for NERTU**. After completing the training,

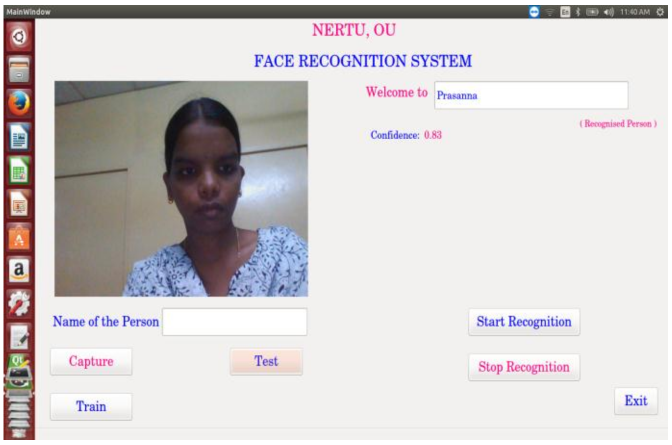
As a separate face model is available for every person of the institute it can used for recognizing those people, who are coming in front of camera.

TEST PHASE:

Here in case of testing, whenever we click on test button it will take the test image by using webcam. Then for the following code will run for recognition of people coming in front of camera:

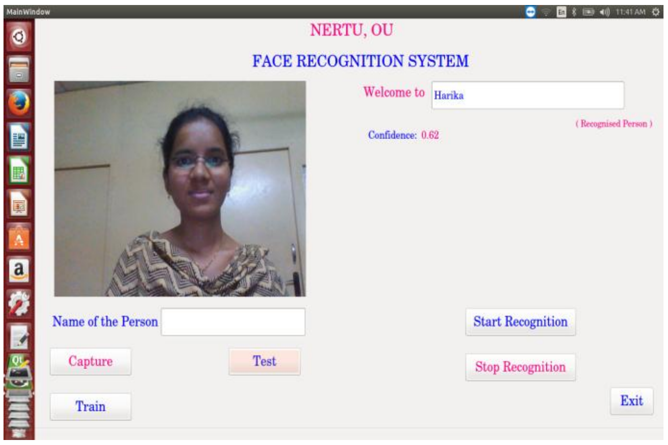
```
def test_cap_Image(self):
i=1
camera=cv2.VideoCapture(0)
while(i<2):
if not os.path.exists('./test-images'):
print 'hello'
os.mkdir('./test-images')
name=self.lineEdit.text()
if not os.path.exists('./test-images/'):
os.mkdir('./test-images/' + name)
ret, image=self.camera.read()
imagename='./test-images/' + str(i) + '.jpg'
print imagename
##  cv2.imshow('test',image)
cv2.imwrite(str(imagename),image)
```

After click on the test button, the system will recognize the person name with confidence. Otherwise displayed as Unknown Person.



Welcome to Prasanna
(Recognised Person)
Confidence: 0.83

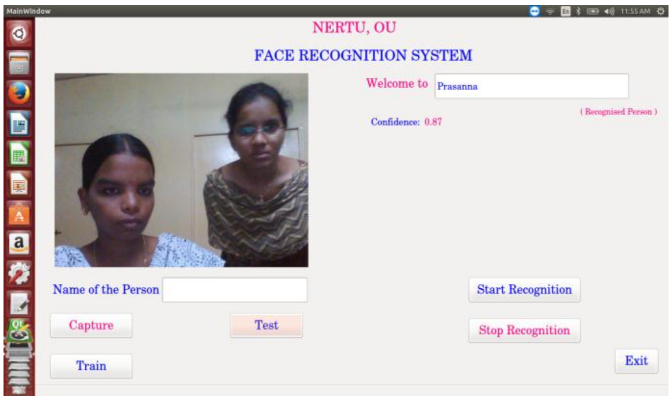
Fig 11: Recognizing Test image as Prasanna



Welcome to Harika
(Recognised Person)
Confidence: 0.62

Fig 12: Recognizing Test image as Harika

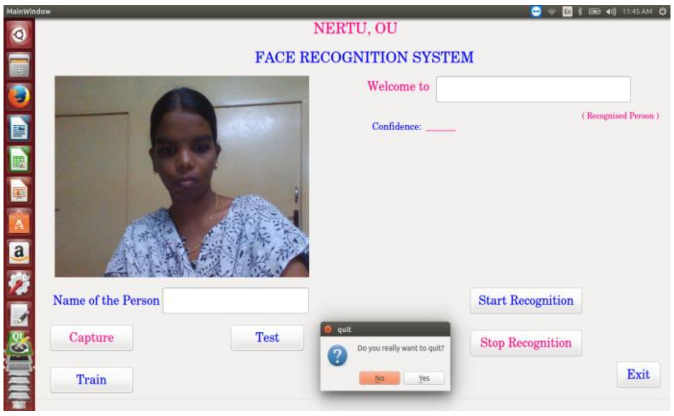
In case of testing two or more persons are present in the system at a time, then the system will recognize the person name as who ever present in the right side.



Welcome to Prasanna
(Recognised Person)
Confidence: 0.87

Fig 13: Recognizing the test-image when two persons are present in front of the camera

For continuous loop of testing images we are using start recognition and stop recognition. Finally, to exit from face recognition application, simply click on exit button in the face recognition system.



Welcome to
(Recognised Person)
Confidence: _____
Do you really want to quit?

Fig 14: Press the exit button to exit from face recognition application

5. CONCLUSION

Face recognition technologies have been associated generally with very costly top secure applications. Today the core technologies have evolved and the cost of equipments is going down dramatically due to the integration and the increasing processing power. Certain applications of face recognition technology are now cost effective, reliable and highly accurate.

The project deals with the implementation of a Face Recognition System Using GUI. Various algorithms for detection and recognition of a face in a given image have been discussed as a part of the Literature . The algorithm that has been used for Face Detection in this project makes use of the detecting the face part by Using HOG algorithm, here in this each and every pixel is compared with neighboring pixels and then it will detect the image with 68 landmarks by using face land mark algorithm; After detecting the face part of the given input image, we are extracting the 128 features/ embeddings by using Deep Neural network concept. Then for recognition we are using SVM Classifier to identify the person. Here in this project we are developing a real time GUI based Face recognition system and further it will be useful for developing a real time face based Biometric attendance system.

REFERENCES

[1] FaceNet: A Unified Embedding for Face Recognition and Clustering. Florian Schrof, Dmitry Kalenichenko, James Philbin. s.l. : IEEE Xplore, 2015. pp. 815-823.

[2] Face recognition: challenges, achievements and future directions. Aly, M. Hassaballah and S. 4, s.l. : IET, DOI: 10.1049/iet-cvi.2014.0084, JULY 30, 2015, Vol. 9, pp. 614-626. ISSN: 1751-9632.

[3] Facial Recognition using OpenCV. Shervin Emami, Valentin Petru. 1, Australia : JMEDS, 2012, Journal of Mobile, Embedded and Distributed Systems, Vol. 4, pp. 38-43. ISSN 2067 – 4074.

[4] Face Detection and Tracking using OpenCV. S.V. Viraktamath, Mukund Katti, Aditya Khatawkar, Pavan Kulkarni. 3, s.l. : SIJ, July-August 2013, The Standard International Journals (The SIJ) , Vol. 1, pp. 45-50. ISSN: 2321 – 2403 .

[5] Implementation of Face Recognition Algorithm for Biometric Time based Attendance System. Adrian Rhesa Septian Siswanto, Anto Satriyo Nugroho, Maulahikmah Galinium. Bandung, Indonesia : IEEE, DOI: 10.1109/ICTSS.2014.7013165, 2014. 2014 International Conference on ICT For Smart Society (ICISS). pp. 149-154. ISBN: 978-1-4799-6321-8.

[6] Security of biometric authentication systems. V.Matyas, Z.Riha. Krackow : IEEE, DOI: 10.1109/CISIM.2010.5643698, 2010. 2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM). pp. 19-28. ISBN: 978-1-4244-7817-0.

[7] Jain, Anil, Ross, Arun A., Nandakumar, Karthik. Introduction to Biometrics. Newyork : Springer, 2011. ISBN 978-0-387-77326-1.

[8] Histograms of oriented gradients for human detection. Triggs, N. Dalal and B. USA : IEEE, DOI: 10.1109/CVPR.2005.177, 2005. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). pp. 886-893. ISSN: 1063-6919.

[9] Prathamesh Timse, Pranav Aggarwal, Prakhar Sinha, Neel Vora. Face Recognition Based Door Lock System Using Opencv and C#. April 2014, Vol. 4, 4, pp. 52-57.

[10] Face Detection in Real Time Based on HOG. N. J. Wang, S. C. Chang and P. J. Chou. Taipei, Taiwan : IEEE, DOI: 10.1109/ISPACS.2012.6473506, 2012. International Symposium on Intelligent Signal Processing and Communications Systems. pp. 333-337. ISBN: 978-1-4673-5081-5.

BIOGRAPHIES

D Mary Prasanna

Post-Graduate Student, Dept. of ECE, G. Narayanamma Institute of Technology & Science, Shaikpet, Hyderabad, India-500008.

Ch Ganapathy Reddy

Professor, Dept. of ECE, G. Narayanamma Institute of Technology & Science, Shaikpet, Hyderabad, India-500008.