FACE DETECTION

A PPT on **FACE DETECTION** is submitted to **Department of Computer Science and Engineering, Jalpaiguri Government Engineering College, Jalpaiguri** (A Government Autonomous College).

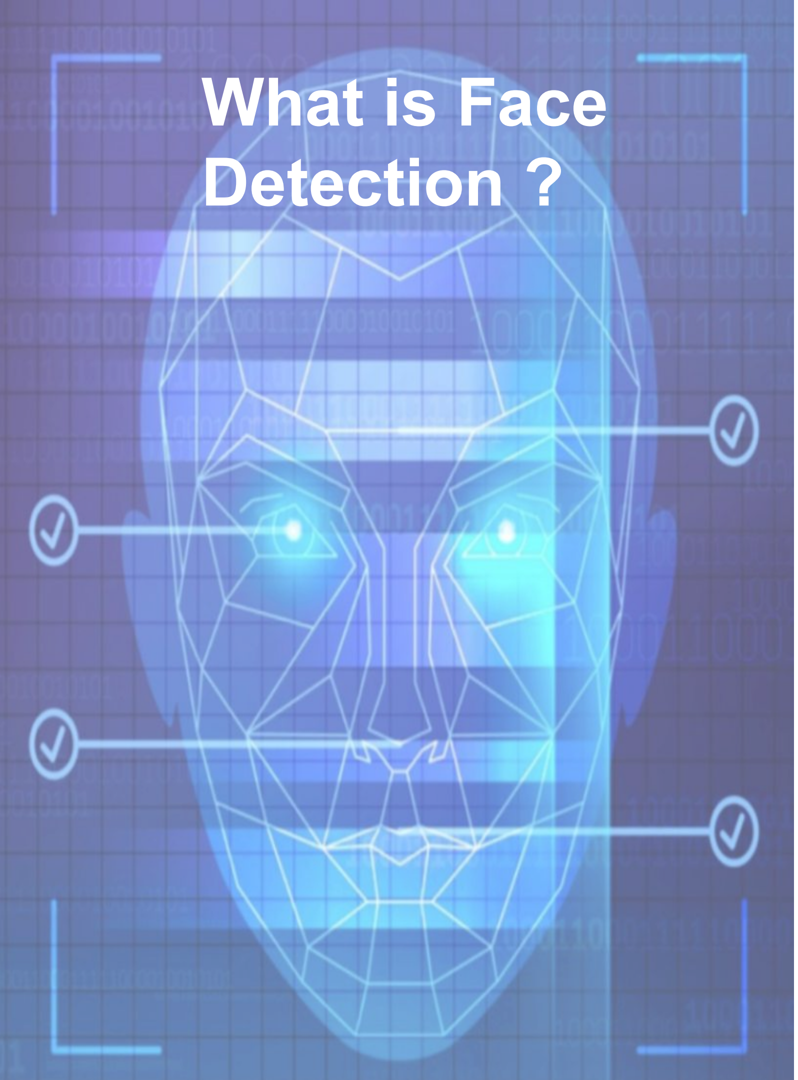
Submitted by Group 10 Members **:**

Arindam Chattopadhyay - 18101104003 (ac2203@cse.jgec.ac.in)
Sandip Kumar Saha -18101104042 (sk2242@cse.jgec.ac.in)
Sagnik Patra – 18101104016 (sp2216@cse.jgec.ac.in)
Subhadip Sardar – 18101104057 (ss2257@cse.jgec.ac.in)

Under the able guidance of **:**

Prof. Swarup Samanta
Dept. of CSE

What is Face Detection ?

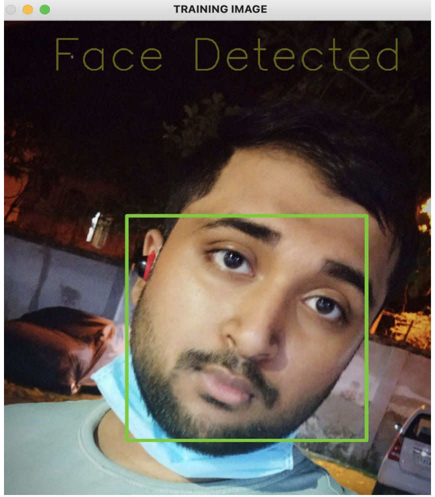


<u>Face detection</u> is a computer technology used for **identifying human faces in digital images**. In this technique, given a face in either a database or in a real time scenario, the face is detected, along with all its features.

With the marvelous increase in video and image database all over the world, there is a continuous need of automatic understanding and examination as manually it is quite hectic. Face Detection is one of the stepping stones of all the Face Analysis systems, and a special class or subclass of object detection, where in **human faces** are detected regardless of their plane of rotation, position, scale, illumination, facial expressions, etc.

SAMPLE FACE :

DETECTED FACE :



APPLICATIONS OF FACE DETECTION

Various applications of the face detection system are --

- **Facial Recognition** : It is a process of identifying or verifying a person from a digital image or a video frame.
- **Document control and access control :** Control can be imposed to document access with face identification system.
- **Biometric Attendance :** It is system of taking attendance of people by their fingerprints or face.
- **Photography** : In recent days, almost all digital cameras use face detection for autofocus.

CHALLENGES FACED IN FACE DETECTION

Challenges in face detection, are the various reasons which reduce the accuracy and power of face detection :

1. **Face occlusion** :
   Face occlusion is hiding face by any object, for example : glasses, mask, hand, hairs, etc.

2. **Orientation** :
   Face orientation is the pose of face with an angle.It is a very big challenge for a face detection system to detect a face accurately in different poses.

3. **Illuminations** :
   Lighting effects may not be uniform in the image, which can give erroneous results.

4. **Interclass Similarity** :
   Different people might look same. They may be either twins or relatives or even strangers (lookalike).

5. **Resolution and Distance** :
   Too much distance from the camera might reduce the resolution of the image, thus it gets blurred and difficult to be identified.

Face Detection in OpenCV

OpenCV :
OpenCV(Open Source Computer Vision) is a library of programming functions for real-time computer vision. While we used OpenCV to facilitate face recognition, OpenCV itself was not responsible for detection of faces. Along with it we required two external libraries, namely :

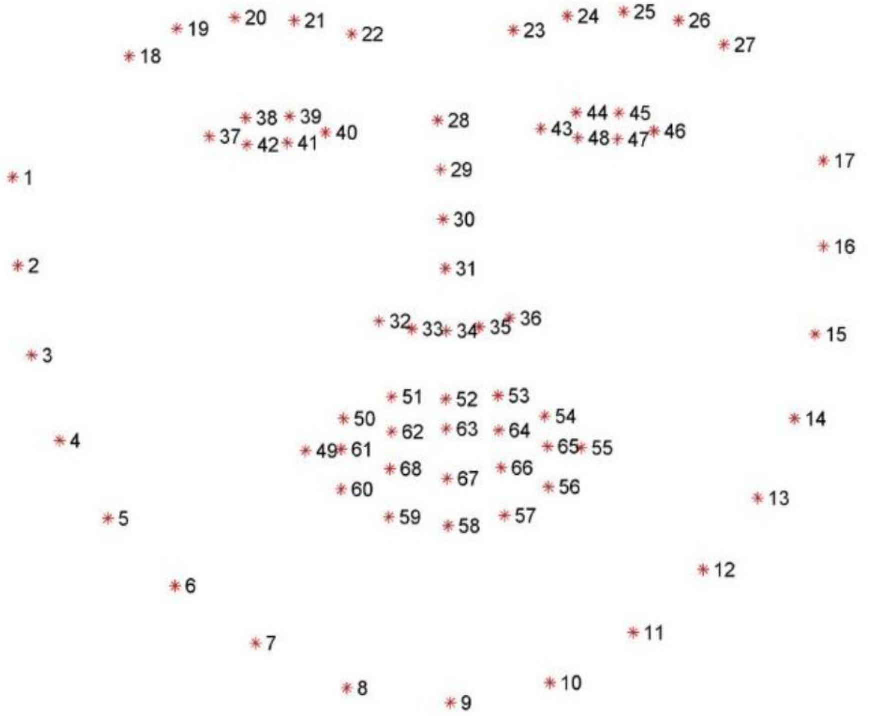
1. dlib
2. face_recognition

Thus in our project, we first load the **RGB** faces and **detect a face** by computing face landmarks, **compute 128-d face encoding** feature vector (from a pretrained network over millions of images) to quantify a face, train a network on the top of the encodings and **recognise faces** in the images given as test images to check the accuracy.

face_recognition Library :
It recognizes and manipulate faces from Python. This provides a simple face_recognition command line tool that lets to do face recognition on a folder of images.

Face recognition algorithms can extract features from a face image namely positions of forehead, eyes, nose, mouth, chin and jaws.

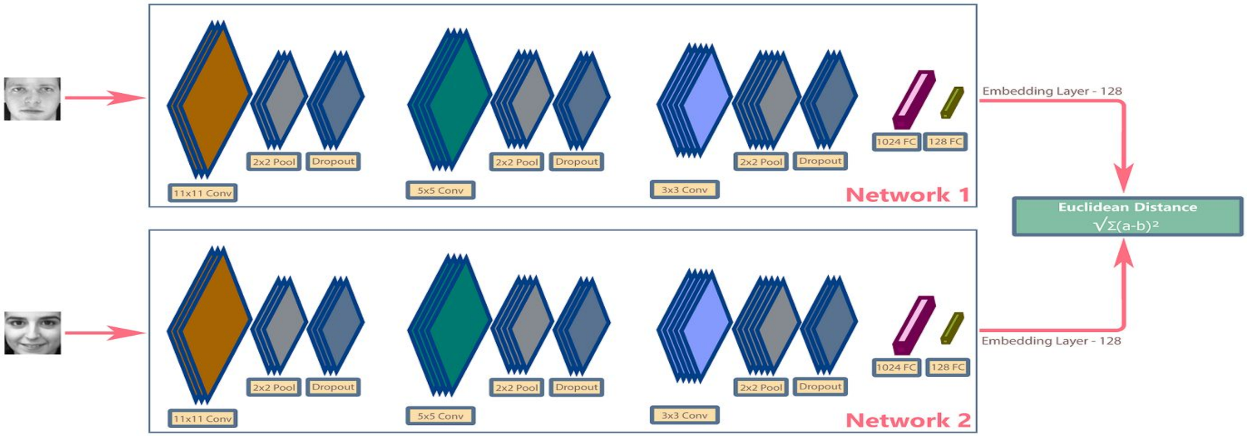
Face Landmarks



There are 68 specific points (called landmarks) that exist on every face. By this, we can compute the face_distance which is the distinguishing factor to compare two faces, based on a tolerance level. The face_recognition library which we are using in our project has face_landmarks and face_distance methods inbuilt.

SIAMESE NEURAL NETWORKS : -->

A Siamese Neural Network is a class of neural network architectures that **contain two or more *identical* sub neural networks**. '*identical*' here means, they have the same configuration with the same parameters and weights, and same operations will be performed on the respective images. Parameter updating is mirrored across both sub-networks. It is used to find the similarity of the inputs, and if found to be non-similar, they are clubbed into different classes. It is good for learning from small samples,i.e., **One-Shot Learning**.
Distance, which is actually **Euclidean Distance metric** here, is lessened if similarity of inputs are found out, through multiple training epochs. Thus this neural network needs more training time than normal networks. It is a problem of Semi-Supervised Learning. Since training of Siamese networks involves pairwise learning, the loss function used here is basically a special loss function, named **Contrastive loss**.



PROJECT IN BRIEF

Firstly we install latest version of Python in our machine, along with OpenCV, dlib, numpy and face_recognition modules, which have siamese neural nets inherent in it.

The dlib library contains implementation of “deep metric learning” which is used to construct face embeddings used for the actual recognition process, and face_recognition library, wraps around dlib’s facial recognition functionality, making it easier to work with.

Then we import libraries into our code, and load our BGR training and testing images (images to compare or detect) into our code using **load_image_file()** method, and also convert them to RGB format, which is readable by OpenCV.

Then, we get the outline location of where exactly the face is in the image using **face_location()** method on the RGB image, and once the face is detected, it is surrounded by a rectangle of desired size using **cv2.rectangle()** method.

PROJECT IN BRIEF

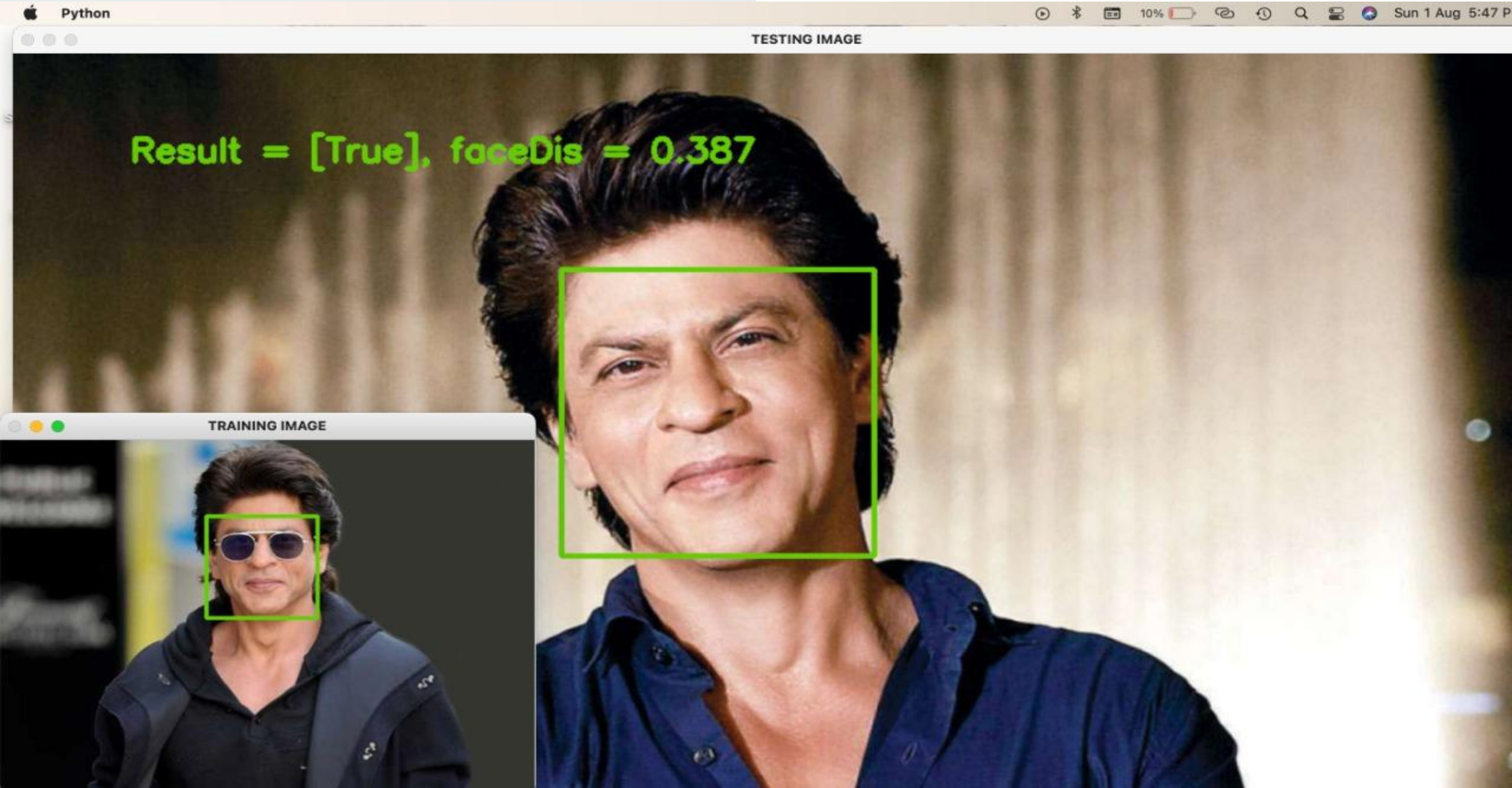
Then face encodings (markings of eyes, nose, mouth, jaws which remain the same for different images of the same person) are taken using **face_encodings()** method which returns a list containing 128 measurements; for both the images. This actually quantifies a face.

Then a comparison between these two returned lists of encoded images is done by the **compare_faces()** method which returns a list of boolean values (True or False).

The **face_distance()** method gets the value of how much the two images differ. There is a **tolerance level of 0.6** beyond which the result is false, that is, the images differ largely.

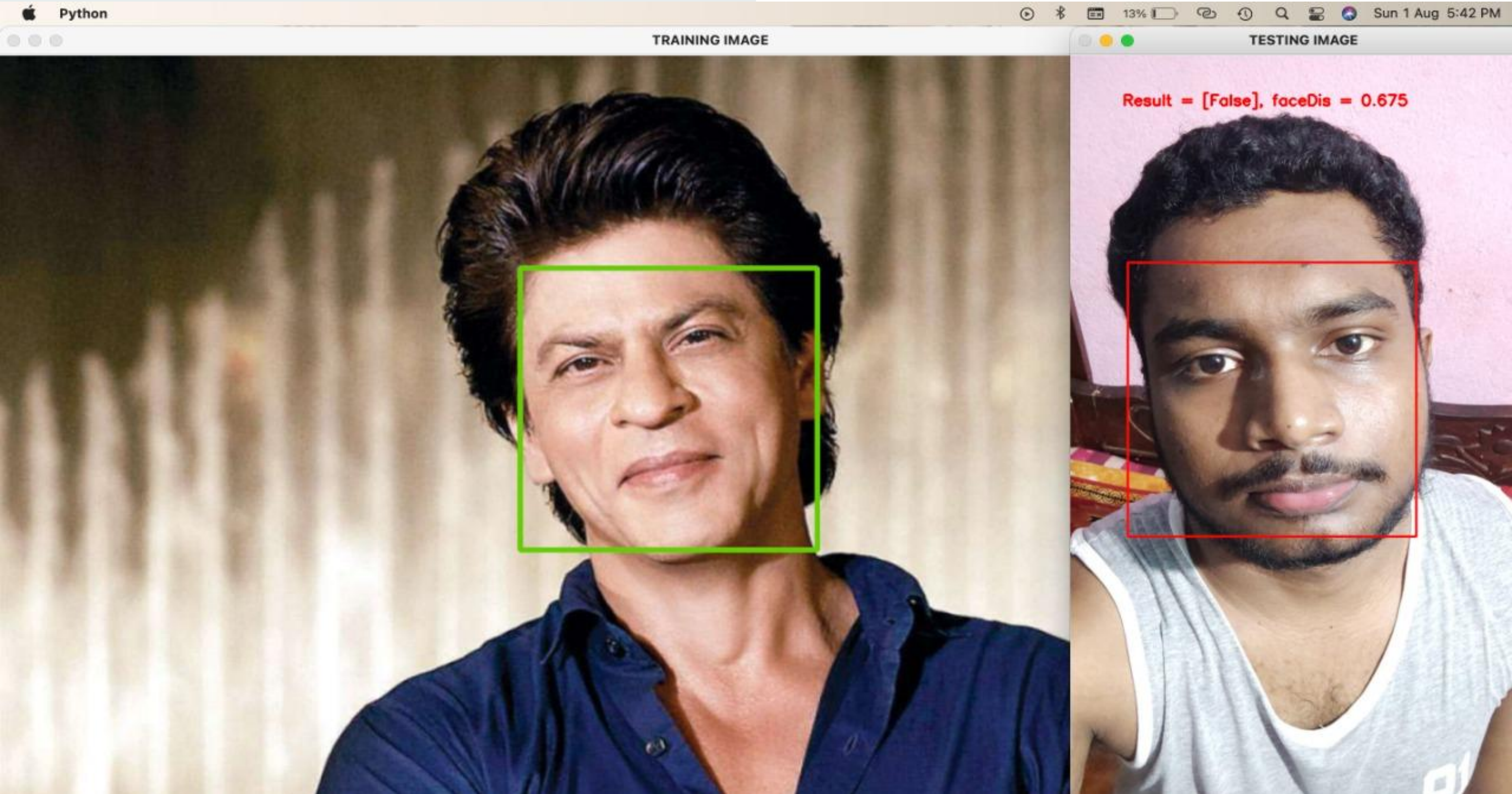
The lower the distance the better the matching and vice versa. Here we have listed 2 instances, one where 2 images match and in other they do not match; in the respective upcoming slides.

Test Image - 01;   Images match



```
Result = [True], faceDis = 0.387
```

Test Image - 02;   Images don’t match



```
Result = [False], faceDis = 0.675
```

PROS and CONS of FACE DETECTION TECHNIQUES

Pros :

- **Improves Security**, both on personal and organizational levels, and helps in tracking down criminals and terrorists.
- Face Detection and Facial Recognition Softwares are quite **easy to integrate** and also **compatible**.
- **Automated Identification** is facilitated by Face Detection, which can be easily replaced by the manual identification system.

Cons:

- There is a **huge data storage burden,** as the algorithms generally manipulate and retrieve data from the stored database.
- The face detection is generally **vulnerable**, as positions and orientations can be manipulated to give erroneous results.
- **Potential breach of Privacy** : Face data can be stored, so as to track the criminals and terrorists, but it is against the Right to Privacy and also is not in compliance with Human Rights if data of private citizens are stored uninformingly.

CONCLUSION AND FUTURE ENDEAVOURS

Face detection and recognition technology has come a long way in the last twenty years. Today, machines are able to automatically verify identity information for secure transactions, for surveillance and security tasks, and for access control to buildings etc, which was manually close to impossible.

In our project, in future, we look to train multiple images through the inherent siamese neural networks. For now, updation of weights is not done, as we are using the pretrained network, but in future we are looking to do so, after figuring out the architecture.

Also we look to build an interface where we can upload the images and thus, by integrating our model with it, we can clearly predict results. That will be a real time solution as we would be able to capture the images in real time and thus train our networks accordingly.

THANK YOU !!