CAPSTONE PROJECT REPORT

FACE DETECTION AND RECOGNITION

NAME: DHIVAKAR.R

COURSE: AI and ML Aug 2020 Batch

Problem Statement:

Build a machine learning model for Face Detection and Recognition

Prerequisites

Prerequisites: Python 3.6: This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/ add-python-to-path-python-is-not-recognized-as-an-internal-or-externalcommand/. Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

Anaconda: Download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6 then run below commands in command prompt/terminal to install these packages pip install -U scikit-learn pip install numpy pip install scipy if you have chosen to install anaconda then run below commands in anaconda prompt to install these packages conda install -c scikitlearn conda install -c anaconda numpy conda install -c anaconda scipy

Dataset Used:

The data source used for this project is captured from live images using opencv module. The screenshot of the process is shared

Implementation

Importing the Required libraries
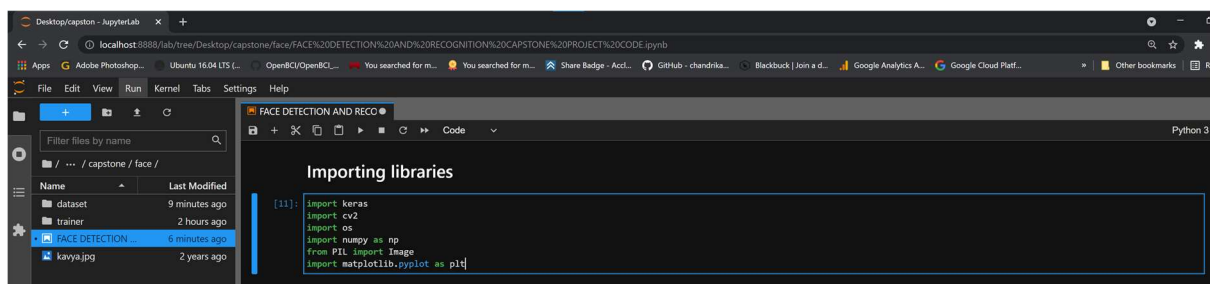
Transforming the dataset :

Transforming the photos

```python
[12]: datagen = keras.preprocessing.image.ImageDataGenerator(
          rotation_range=30,
          width_shift_range=0.0,
          height_shift_range=0.0,
          horizontal_flip=True,
          vertical_flip=True,
          rescale=None,
          preprocessing_function=None
      )
```

Creating a Face Dataset

Creating data for face detection

```python
[4]: def mark_images(f):
         ex = -1
         img = cv2.resize(cv2.imread(f), (640, 480))
         gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
         face_detector = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
         faces = face_detector.detectMultiScale(gray, 1.3, 5)
         for (x,y,w,h) in faces:
             cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
             plt.imshow(gray[y:y+h,x:x+w])
             plt.show()
             face_id = input('\n Assign an ID number and press enter ')
             im = np.expand_dims(np.expand_dims(gray[y:y+h,x:x+w], 0), 3)
             datagen.fit(im)
             for x, val in zip(datagen.flow(im, save_to_dir='C:\\Users\\dhiva\\Desktop\\capstone\\face\\dataset', save_prefix="User_" + str(face_id), save_format='jpg'),range(100))
                 ex = 1
```

Capturing the face data:

capturing face data

```python
[13]: cam = cv2.VideoCapture(0)
      cam.set(3, 640)
      cam.set(4, 480)
      face_detector = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

      face_id = input('\n Assign an ID number and press enter ')
      print("\n  Look the camera and wait ...")

      ex = -1
      while(True):
          ret, img = cam.read()
          gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
          faces = face_detector.detectMultiScale(gray, 1.3, 5)

          for (x,y,w,h) in faces:

              cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)
              cv2.imshow('image', img)
              im = np.expand_dims(np.expand_dims(gray[y:y+h,x:x+w], 0), 3)
              datagen.fit(im)

              for x, val in zip(datagen.flow(im, save_to_dir='C:\\Users\\dhiva\\Desktop\\capstone\\face\\dataset', save_prefix="User_" + str(face_id), save_format='jpg'),range(100)):
                  ex = 1

          k = cv2.waitKey(100) & 0xff
          if k == 27:
              break
          if ex == 1:
              break
      print("\n Exiting Program and cleanup stuff")
      cam.release()
      cv2.destroyAllWindows()

      Assign an ID number and press enter  1

       Look the camera and wait ...

      Exiting Program and cleanup stuff
```
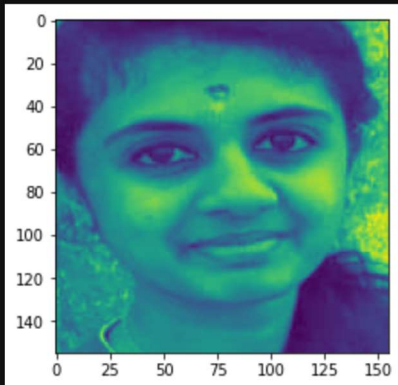
Adding another person image to identify and give difference


```
[14]: mark_images('kavya.jpg')
```



```
Assign an ID number and press enter   2
```

Training the Model

Training the model

if there is an error for the .DS_STORE use -->" find . -name ".DS_Store" -delete " for the folder

```python
[15]: path = 'C:\\Users\\dhiva\\Desktop\\capstone\\face\\dataset'
      recognizer = cv2.face.LBPHFaceRecognizer_create()
      detector = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
      def getImagesAndLabels(path):
          imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
          faceSamples=[]
          ids = []
          for imagePath in imagePaths:
              PIL_img = Image.open(imagePath).convert('L')
              img_numpy = np.array(PIL_img,'uint8')
              id = int(os.path.split(imagePath)[-1].split("_")[1])
              faces = detector.detectMultiScale(img_numpy)
              for (x,y,w,h) in faces:
                  faceSamples.append(img_numpy[y:y+h,x:x+w])
                  ids.append(id)

          return faceSamples,ids
      print ("\n Training faces. It will take a few seconds. please Wait ...")
      faces,ids = getImagesAndLabels(path)
      recognizer.train(faces, np.array(ids))
      recognizer.write('C:\\Users\\dhiva\\Desktop\\capstone\\face\\trainer\\trainer.yml')
      print("\n  {0} faces trained. Exiting Program".format(len(np.unique(ids))))
```

```
 Training faces. It will take a few seconds. please Wait ...

  2 faces trained. Exiting Program
```

Face Detection

Face detection ¶

```python
[7]: recognizer = cv2.face.LBPHFaceRecognizer_create()
     recognizer.read('C:\\Users\\dhiva\\Desktop\\capstone\\face\\trainer\\trainer.yml')
     faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
     font = cv2.FONT_HERSHEY_SIMPLEX
     id = 0
     names = ['none','dhivakar','kavya']
```

```python
[8]: cam = cv2.VideoCapture(0)
     cam.set(3, 640)
     cam.set(4, 480)

     minW = 0.1*cam.get(3)
     minH = 0.1*cam.get(4)
```

```python
[9]: while True:
    ret, img =cam.read()
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    ret, img =cam.read()
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor = 1.2,
        minNeighbors = 5,
        minSize = (int(minW), int(minH)),
        )
    for(x,y,w,h) in faces:
        cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)
        id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

        if (confidence < 100):
            id = names[id]
            confidence = "  {0}%".format(round(confidence))
        else:
            id = "unknown"
            confidence = "  {0}%".format(round(confidence))

        cv2.putText(
                    img,
                    str(id),
                    (x+5,y-5),
                    font,
                    1,
                    (255,255,255),
                    2
                    )
        cv2.putText(
                    img,
                    str(confidence),
                    (x+5,y+h-5),
                    font,
                    1,
                    (255,255,0),
                    1
                    )

    cv2.imshow('camera',img)
    k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break
cam.release()
cv2.destroyAllWindows()
```
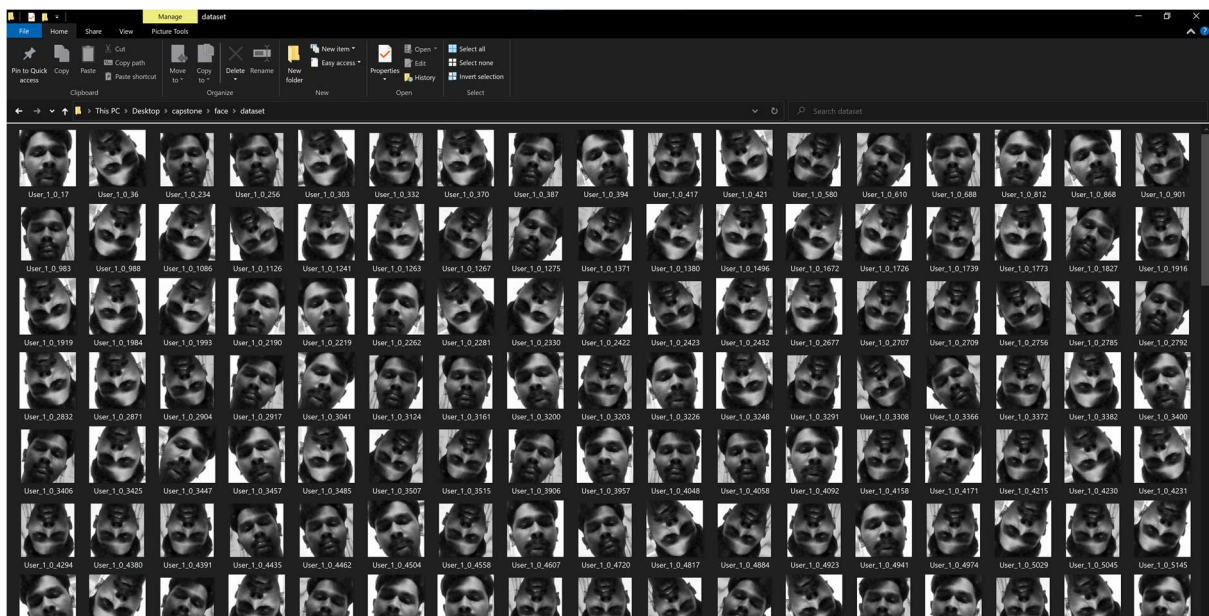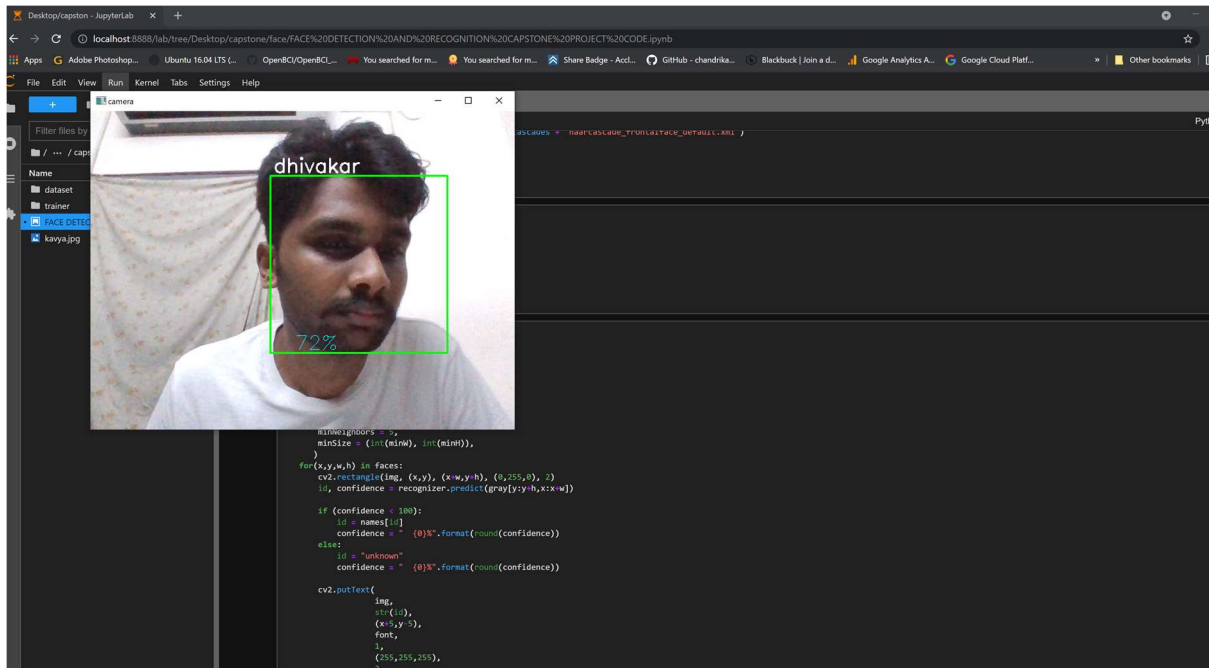
Data set image folder

Output

Person 1 identified

Person 2 identified