**CODE**

**FACE-TRAIN.py**

import os

import cv2

import numpy as np

from PIL import Image

import pickle

BASE\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

image\_dir = os.path.join(BASE\_DIR, "images")

face\_cascade=cv2.CascadeClassifier('cascades/data/haarcascade\_frontalface\_alt2.xml')

recognizer = cv2.face.LBPHFaceRecognizer\_create()

current\_id = 0

label\_ids = {}

y\_labels = []

x\_train = []

for root, dirs, files in os.walk(image\_dir):

for file in files:

if file.endswith("png") or file.endswith("jpg"):

path = os.path.join(root, file)

label = os.path.basename(root)

print(label)

if not label in label\_ids:

label\_ids[label] = current\_id

current\_id += 1

id\_ = label\_ids[label]

#print(label\_ids)

#y\_labels.append(label) # some number

#x\_train.append(path) # verify this image, turn into NUMPY array, GRAY

pil\_Image = Image.open(path).convert("L") #grayscale

size = (550, 550)

final\_image = pil\_Image.resize(size, Image.ANTIALIAS)

image\_array = np.array(pil\_Image, "uint8")

#print(image\_array)

faces = face\_cascade.detectMultiScale(image\_array, scaleFactor=1.1, minNeighbors=5)

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

roi = image\_array[y:y+h, x:x+w]

x\_train.append(roi)

y\_labels.append(id\_)

#print(y\_labels)

#print(x\_train)

with open("labels.pickle", 'wb') as f:

pickle.dump(label\_ids, f)

recognizer.train(x\_train, np.array(y\_labels))

recognizer.save("trainer.yml")

**PRODUCT.py**

import cv2

import numpy

import time

import pickle

import os

capture=cv2.VideoCapture(0)

capture.set(3,640)

capture.set(4,480)

frame\_set=[]

start\_time=time.time()

BASE\_DIR = os.path.dirname(os.path.abspath(\_\_file\_\_))

image\_dir = os.path.join(BASE\_DIR, "images")

test = ""

for root, dirs, files in os.walk(image\_dir):

for file in files:

if file.endswith("png") or file.endswith("jpg"):

path = os.path.join(root, file)

label = os.path.basename(root)

face\_cascade = cv2.CascadeClassifier('cascades/data/haarcascade\_frontalface\_alt.xml')

eye\_cascade = cv2.CascadeClassifier('cascades/data/haarcascade\_eye.xml')

smile\_cascade = cv2.CascadeClassifier('cascades/data/haarcascade\_smile.xml')

start\_time=time.time()

recognizer = cv2.face.LBPHFaceRecognizer\_create()

recognizer.read("trainer.yml")

labels = {"person\_name": 1}

with open("labels.pickle", 'rb') as f:

og\_labels = pickle.load(f)

labels = {v:k for k,v in og\_labels.items()}

while(True):

ret, frame = capture.read()

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

frame\_set.append(gray)

faces = face\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)

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

#print(x,y,w,h)

roi\_gray = gray[y:y+h, x:x+w] #(ycord\_start, ycord\_end)

roi\_color = frame[y:y+h, x:x+w]

#recognize? deep learned model

nameid, conf = recognizer.predict(roi\_gray)

if conf>=4 and conf<=85:

#print(id\_)

print(labels[nameid])

font = cv2.FONT\_HERSHEY\_SIMPLEX

name = labels[nameid]

test = name

color = (255, 255, 255)

stroke = 2

cv2.putText(frame, name, (x,y), font, 1, color, stroke, cv2.LINE\_AA)

img\_item = "intruder.jpg"

cv2.imwrite(img\_item, roi\_color)

color = (0, 0, 255) #BGR 0-255

stroke = 2

end\_cord\_x = x + w

end\_cord\_y = y + h

cv2.rectangle(frame, (x, y), (end\_cord\_x, end\_cord\_y), color, stroke)

subitems = smile\_cascade.detectMultiScale(roi\_gray)

#cv2.imshow('frame',gray)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

end\_time=time.time()

elapsed = end\_time - start\_time

if elapsed > 20:

break

capture.release()

cv2.destroyAllWindows()

if test == "abdul":

print "Authorized Person"

else:

import Attachment

Attachment

**ATTACHMENT.py**

import smtplib

import base64

import time

filename = "C:/Users/JARJIS/Videos/IAM/intruder.jpg"

fo = open(filename, "rb")

filecontent = fo.read()

encodedcontent = base64.b64encode(filecontent)

sender = 'saferlap@gmail.com'

recipient = 'ccfis.unom2018@gmail.com'

marker = 'AUNIQUEMARKER'

body = """Intruder Alert: The Person Below used your system.....!"""

part1 = """From: Laptop

To: ccfis.unom2018@gmail.com

Subject: Unauthorized Usage - Image

MIME-Version: 1.0

Content-Type: multipart/mixed; boundary=%s

--%s

""" %(marker,marker)

part2 = """Content-Type: text/plain

Content-Transfer-Encoding:8bit

%s

--%s

""" %(body,marker)

part3 = """Content-Type: multipart/mixed; name=\"%s\"

Content-Transfer-Encoding:base64

Content-Disposition: attachment; filename=%s

%s

--%s--

""" %(filename, filename, encodedcontent, marker)

message = part1 + part2 + part3

try:

mail = smtplib.SMTP('smtp.gmail.com', 587)

mail.ehlo()

mail.starttls()

mail.login('saferlap@gmail.com', 'L0n3W01f')

mail.sendmail(sender,recipient,message)

print "Successfully Sent"

time.sleep(60)

import email

import imaplib

username = "saferlap@gmail.com"

password = "L0n3W01f"

mail = imaplib.IMAP4\_SSL("imap.gmail.com")

mail.login(username,password)

mail.select("Inbox")

result, data = mail.uid('search', None, "ALL")

inbox\_item\_list = data[0].split()

most\_recent = inbox\_item\_list[-1]

oldest = inbox\_item\_list[0]

result2, email\_data = mail.uid('fetch', most\_recent, '(RFC822)')

email\_message = email.message\_from\_string(email\_data[0][1])

sub = email\_message['Subject']

if sub == "Lock":

import ctypes

ctypes.windll.user32.LockWorkStation()

else:

print "Allowed by Authorized Person"

except Exception:

print "Error: Unable to send the mail"