FaceRecognition

# Using face\_recognition python package

## Working: Looks through the no of faces in a directory and encodes them for future detections. Returns the face detected from given image file and Labels them according to prior encoded faces.

## Requirements:

import face\_recognition as fr

import os

import cv2

import face\_recognition

import numpy as np

from time import sleep

# Functions with Description and Source code

## get\_encoded\_faces() looks through the faces folder and encodes all the faces and returns dict of (name, image encoded)

def get\_encoded\_faces():

encoded = {}

for dirpath, dnames, fnames in os.walk("./faces"):

for f in fnames:

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

face = fr.load\_image\_file("faces/" + f)

encoding = fr.face\_encodings(face)[0]

encoded[f.split(".")[0]] = encoding

return encoded

## unknown\_image\_encoded(img) encode a face given the file name

def unknown\_image\_encoded(img):

face = fr.load\_image\_file("faces/" + img)

encoding = fr.face\_encodings(face)[0]

return encoding

## classify\_face(im) will find all of the faces in a given image and label them if it knows what they are.

param im: str of file path

return: list of face names

def classify\_face(im):

faces = get\_encoded\_faces()

faces\_encoded = list(faces.values())

known\_face\_names = list(faces.keys())

img = cv2.imread(im, 1)

face\_locations = face\_recognition.face\_locations(img)

unknown\_face\_encodings = face\_recognition.face\_encodings(img, face\_locations)

face\_names = []

for face\_encoding in unknown\_face\_encodings:

# See if the face is a match for the known face(s)

matches = face\_recognition.compare\_faces(faces\_encoded, face\_encoding)

name = "Unknown"

# use the known face with the smallest distance to the new face

face\_distances = face\_recognition.face\_distance(faces\_encoded, face\_encoding)

best\_match\_index = np.argmin(face\_distances)

if matches[best\_match\_index]:

name = known\_face\_names[best\_match\_index]

face\_names.append(name)

for (top, right, bottom, left), name in zip(face\_locations, face\_names):

# Draw a box around the face

cv2.rectangle(img, (left-20, top-20), (right+20, bottom+20), (255, 0, 0), 2)

# Draw a label with a name below the face

cv2.rectangle(img, (left-20, bottom -15), (right+20, bottom+20), (255, 0, 0), cv2.FILLED)

font = cv2.FONT\_HERSHEY\_DUPLEX

cv2.putText(img, name, (left -20, bottom + 15), font, 1.0, (255, 255, 255), 2)

# Display the resulting image

while True:

cv2.imshow('Video', img)

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

return face\_names

print(classify\_face("test.jpg"))

# output:

