## Face Recognition

## DR: Mahmoud Mounir

Mahmoud Emad Abdellatif 15

Nader Waled Mohamed 18

Mmahmoud Saeed Mohamed Hassan 15

**CV2 , Face Recognition**

CV2 is the most popular library for computer vision. Originally written in C/C++, it now provides bindings for Python.

Open CV2,Face Recognition uses machine learning algorithms to search for faces within a picture. Because faces are so complicated, there isn’t one simple test that will tell you if it found a face or not. Instead, there are thousands of small patterns and features that must be matched. The algorithms break the task of identifying the face into thousands of smaller, bite-sized tasks, each of which is easy to solve. These tasks are also called classifiers.

For something like a face, you might have 6,000 or more classifiers, all of which must match for a face to be detected (within error limits, of course). But therein lies the problem: for face detection, the algorithm starts at the top left of a picture and moves down across small blocks of data, looking at each block, constantly asking, “Is this a face? … Is this a face? … Is this a face?” Since there are 6,000 or more tests per block, you might have millions of calculations to do, which will grind your computer to a halt.

To get around this, CV2,Face Recognition uses cascades. What’s a cascade? The best answer can be found in the dictionary: “a waterfall or series of waterfalls.”

Like a series of waterfalls, the CV2,Face Recognition cascade breaks the problem of detecting faces into multiple stages. For each block, it does a very rough and quick test. If that passes, it does a slightly more detailed test, and so on. The algorithm may have 30 to 50 of these stages or cascades, and it will only detect a face if all stages pass.

The advantage is that the majority of the picture will return a negative during the first few stages, which means the algorithm won’t waste time testing all 6,000 features on it. Instead of taking hours, face detection can now be done in real time.

to make GUI to Display Information about the person.



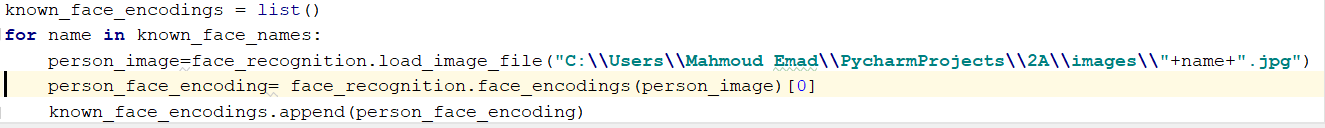
to open the names.txt file to read names of persons in the database and stored in list .



This is a demo of running face recognition on live video from your webcam. It's a little more complicated than the  
other example, but it includes some basic performance tweaks to make things run a lot faster:  
1. Process each video frame at 1/4 resolution (though still display it at full resolution)  
2. Only detect faces in every other frame of video.  
  
PLEASE NOTE: This example requires CV (the `cv2` library) to be installed only to read from your webcam.  
CV is \*not\* required to use the face recognition library. It's only required if you want to run this  
specific demo. If you have trouble installing it, try any of the other demos that don't require it instead.  
  
Get a reference to webcam #0 (the default one)



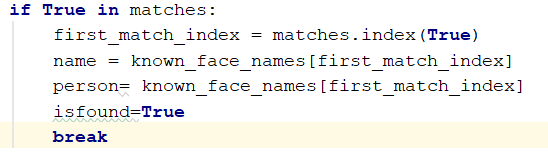
Load a sample pictures from image file and learn how to recognize it. And stored in list from face\_recognation data type.



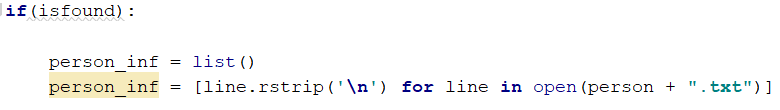
The project then runs the camera and identifies the faces that it sees through many tests that it can identify faces and then compares each of the faces that it has through several tests and calculating the probability that this face is the face in the picture if it This person does not exist in the database. The program creates a red box around the face of the person and passes an )unknown(.



If the person is known, the program passes the name of that person,



opens the file that contains this person's data,



closes the camera, On the destination of user drawings.

