**Face Recoginition using Python**

**Alok Awasthi (17BEC0789)**

[alok29awasthi@gmail.com](mailto:alok29awasthi@gmail.com)

**Objective – To apply the application of opencv for Face Recognition.**

**Application – Attendance System**

**Abstract – In this day by day improving and busy world we are least concerned about the fact of giving attendance. So, keeping this in mind this project is made to decrease the time and any efforts of the people for such activities. You must have seen these kind of softwares already working in the cell phones. But now the goal is to take it to every corner for decreasing the errors made by humans. You must be thinking that for such concerns we have got a biometric systems or card swipe systems installed in offices but if you people have noticed these biometric sometimes take so many tries and as I said earlier in this busy world every second is important and so we need to evolve with the evolving technology. This systems uses different libraries of python for its functionality.**

**Key words – Python, OpenCv, Numpy, Pillow, Pickle, OS**

1. **INTRODUCTION**

As explained earlier, this project is to simplify the bothersome and fussy system of attendance and also to move on as the technology moves on. To do the following work I have used Python. The most used library that is used in this project is OpenCv. The following libraries were also used in the completion of the project: Numpy, Pillow, Pickle and OS

1. **METHODOLOGY**

Here we basically took steps forward little by little going through how to work with the opencv and finding the optimum solution to attendance problem. We have used python to work out with it and find the code. First we work out with Opencv that requires Numpy too. We first try to work with pictures and how to make them grayscale and all. Then we tried to draw on the pictures and put text on them. Then we started to work with the videos and tried the same function on them as used on the pictures as videos are just frames that are changed at a very high rate.

1. **WORKING**

In this project, we are going to use the haar cascades first to detect the face in the picture. For your information these haar cascades are already available with the python package as .xml file extension.

Then we made .yml extension file and trained our program to recognize the faces of the people and then name them accordingly.

1. **HAAR CASCADES**

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

First we need to load the required XML classifiers. Then load our input image (or video) in grayscale mode.

1. **OS LIBRARY**

The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The \*os\* and \*os.path\* modules include many functions to interact with the file system.

**In os.path.dirname(os.path.abspath(\_\_file\_\_))**

BASE\_DIR is pointing to the *parent* directory

\_\_file\_\_ is the pathname of the file from which the module was loaded, if it was loaded from a file.

**In os.walk()**

The method walk() generates the file names in a directory tree by walking the tree either top-down or bottom-up.

**In os.path.basename(path)**

Returns the base name of pathname path.

[**In os.path.abspath(path)**](http://www.tutorialspoint.com/python/os_path_abspath.htm)

Returns a normalized absolutized version of the pathname path.

1. **NUMPY LIBRARY**

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

* a powerful N-dimensional array object
* sophisticated (broadcasting) functions
* tools for integrating C/C++ and Fortran code
* useful linear algebra, Fourier transform, and random number capabilities

1. **IMAGE MODULE OF PIL**

The [Image](http://pillow.readthedocs.io/en/3.1.x/reference/Image.html#module-PIL.Image) module provides a class with the same name which is used to represent a PIL image. The module also provides a number of factory functions, including functions to load images from files, and to create new images.

**In Image.open(path).convert("L")**

Convert image to grayscale in that path.

1. **PICKLE LIBRARY**

Pickle has two main methods. The first one is dump, which dumps an object to a file object and the second one is load, which loads an object from a file object.

* pickle.dump() – It dumps the data into a file.
* pickle.load() – It loads data from a file.

**Steps to do the following:**

1. Start with the working of web cam and working with videos.
2. Including the Haar Cascades for image (as in this case face) detection.
3. Drawing a rectangle over the face and if you want eyes and mouth too.
4. Using OS to find images from the specific directories.
5. Finding labels from directories.
6. Converting the image to Numpy array.
7. Save the label IDs into a file f using Pickle library.
8. Training the recognizer.
9. Load label names from Pickle and put text over the rectangle i.e., on the person’s face.
10. **ALGORITHMS**

* **Recognizer Trainer**

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

for file in files:

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

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

label = os.path.basename(root).replace(" ","-").lower()

if label in label\_ids:

pass

else:

label\_ids[label] = current\_id

current\_id += 1

id\_ = label\_ids[label]

size=(550,550)

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

faces=face\_cascade.detectMultiScale(image\_array,1.3,5)

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

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

x\_train.append(roi)

y\_labels.append(id\_)

* **Detection**

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

roi\_gray=gray[y:y+h, x:x+w]

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

id\_, conf=recognizer.predict(roi\_gray)

if conf>=45 and conf<=85:

print(id\_, conf)

print(labels[id\_])

font=cv2.FONT\_HERSHEY\_SIMPLEX

name=labels[id\_]

color=(255,255,255)

stroke=2

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

cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)

* **Attendance**

names=names + [name]

temp = {}

for name in names:

name = name.lower()

if name in temp:

temp[name] += 1

else:

temp[name] = 1

countnames = {}

for key, value in temp.items():

if value in countnames:

countnames[value].append(key)

else:

countnames[value] = [key]

if list(countnames.keys())[0] >=10:

n=countnames[list(countnames.keys())[0]]

names.remove(n[0])

faces[n[0]]='Present'

break;

1. **REFRENCE**

* <https://www.youtube.com/user/sentdex>
* <https://stackoverflow.com/>
* <https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_table_of_contents_feature2d/py_table_of_contents_feature2d.html#py-table-of-content-feature2d>
* <https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_objdetect/py_table_of_contents_objdetect/py_table_of_contents_objdetect.html#py-table-of-content-objdetection>