GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

FACE RECOGNITION SYSTEM

A Project Report of Capstone Project - 2

Submitted by-

ALANKRIT AGNIHOTRI

(1613101098/16SCSE101259)

in partial fulfillment for the award of the degree

of

B.Tech

in

COMPUTER SCIENCE

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the super vision of

Sreenarayanan N M

Assistant professor

MAY, 2020

GALGOTIAS UNIVERSITY

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report " FACE RECOGNITION SYSTEM " is the bonafide work of " ALANKRIT AGNIHOTRI " who carried out the project work under my supervision.

SIGNATURE OF HEAD

DR.MUNISH SHABARWAL

Phd(Management),Phd(CS)
Professor & Dean,
School of Computing Science and Engineering

SIGNATURE OF SUPERVISOR

Sreenarayanan N M

Assistant professor

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Face detection

ABSTRACT

Accurately identifying the people in photos has until now been a very human process .Even though social media sites have been suggesting photo tags since 2010.There success has been little bit hit and miss specially when drunk selfies are concerned but are starting to step up and things starting to develop at facebook AI labs(artificial intelligence) can now recognize with 97.35% accuracy which is actually no 0.28% less accurate than a human which is surprising because computers are normally more accurate than us. So why are us humans still the best as compared to computers at this. But the human ability to identify people and things by size is actually quite unique form of identification. Most of the animals get to know and recognize each other by smell.

Facial recognition is something that we have evolved to do. Essentially the human brain is really very well trained to recognize recurring patterns and faces are just another patterns and it is an initial step in AI Identification is try to mimic the human method.

The computer will divide the face into visual marks also called as nodal points which include things like depth of the eye sockets or distance between each eye and the width of your nose. The differences between these area were then use to create a unique code a person's own face print but there is a problem as to get a correct match photos are to be almost similar composition and you really get the same view as of before. Our faces are being constant flux. They just not static like finger prints.

There are four main challenges while developing facial recognition. They are known as the A-PIE Problem

- AGEING
- POSE
- ILLUMINATIOn
- EMOTION

CONTENT

Introduction

Face detection is the first step of face recognition as it automatically detects a face from a complex background to which the face recognition algorithm can be applied. But detection itself involves many complexities such as background, poses, illumination etc.

Python is a high-level general-purpose programming language created by Guido van Rossum in 1991. It has a design philosophy that puts emphasis on code readability. It supports multiple programming paradigms including object-oriented, imperative, functional and procedural and has a large standard and comprehensive library. The first release was followed by Python 2.0 in 2000 and Python 3.0 in 2008. At the time of doing this project the latest version is Python 3.7. Python is a good choice for all the researchers in the scientific community because it is

- free and open source
- a scripting language, meaning that it is interpreted
- a modern language (object oriented, exception handling, dynamic typing etc.)
- concise, easy to read and quick to learn
- full of freely available libraries, in particular scientific ones (linear algebra, visualization tools, plotting, image analysis, differential equations solving, symbolic computations, statistics etc.)
- useful in a wider setting: scientific computing, scripting, web sites, text parsing, etc.
- widely used in industrial applications

In comparison with other programming languages such as C/C++, Java, and Fortran, Python is a higher-level language. The computation time is therefore typically a little longer, but it is much easier to program. In the case of C and Fortran, wrappers are also available. PHP and Ruby on the other side are high level languages as well. Ruby can be compared to Python but lacks scientific libraries. PHP on the other hand is a more web-oriented language. Python can

also be compared to Matlab, which has a really extensive scientific library. It is however not open source and free. Scilab and Octave are open source environments similar to Matlab. Their language features are however inferior to the ones available in Python. People in general tend to think that complex problems demand complex processes in order to produce complex solutions. Python was developed with exactly the opposite philosophy. It has an extremely flat learning curve and development process for software engineers. It is used for system administration tasks, by NASA both for development and as a scripting language in several of its systems, Industrial Light & Magic uses Python in its production of special effects for large-budget feature films, Yahoo! uses it (among other things) to manage its discussion groups and Google has used it to implement many components of its web crawler and search engine. As Python is also a language that is easy to learn and both powerful and convenient from the start, we might soon be asking, who is not using it. As already mentioned Python has an extensive set of libraries which can be imported into a project in order to perform specific tasks. The ones that really should be mentioned in any scientific paper dealing with mathematics are NumPy and SciPy. NumPy is a library which provides support for large, multi-dimensional arrays. As images are in fact large two (greyscale) or three (color) dimensional matrices, this library is essential in all image processing tasks. It should also be emphasized that many other libraries (not limited to image processing) use NumPy array representation. SciPy is a library built on the NumPy array object and contains modules for signal and image processing, linear algebra, fast Fourier transform, etc. The last library mentioned in this introductory section is Matplotlib. As the name suggests this library is a plotting library. Although it is used a lot in all areas of science, Image processing relies heavily on it. In this paper the libraries and their capabilities in the area of image processing, image analysis and computer vision in general are discussed. The execution of some of the most common algorithms in the field is also demonstrated together with the resulting images.

USE OF FACE RECOGNITION

1. Apple first used facial recognition to unlock its iPhone X, and continues with the iPhone XS. Face ID authenticates — it makes sure you're you when you access your phone. Apple says the chance of a random face unlocking your phone is about one in 1 million.Facial recognition software can, in essence, take roll. If you decide to cut class, your professor could know. Don't even think of sending your brainy roommate to take your test
2. Facebook uses an algorithm to spot faces when you upload a photo to its platform. The social media company asks if you want to tag people in your photos. If you say yes, it creates a link to their profiles. Facebook can recognize faces with 98 percent accuracy.
3. Some companies have traded in security badges for facial recognition systems. Beyond security, it could be one way to get some face time with the boss.
4. Retailers can combine surveillance cameras and facial recognition to scan the faces of shoppers. One goal: identifying suspicious characters and potential shoplifters
5. You might be accustomed to having an agent scan your boarding pass at the gate to board your flight. At least one airline scans your face.

FUTURE OF FACE RECOGNITION

The use of spherical canonical images allows us to perform matching in the spherical harmonic transform domain, which does not require preliminary alignment of the images. The errors introduced by embedding into an expressional space with some predefined geometry are avoided. In this facial expression recognition setup, end-to-end processing comprises the face surface acquisition and reconstruction, smoothening, sub sampling to approximately 2500 points. Facial surface cropping measurement of large positions of distances between all the points using a parallelized parametric version is utilized. The general experimental evaluation of the face expressional system guarantees better face recognition rates. Having examined techniques to cope with expression variation, in future it may be investigated in more depth about the face classification problem and optimal fusion of color and depth information. Further study can be laid down in the direction of allele of gene matching to the geometric factors of the facial expressions. The genetic property evolution framework for facial expressional system can be studied to suit the requirement of different security models such as criminal detection, governmental confidential security breaches etc.

WHO CREATED FACE RECOGNITION

A RAND tablet was a device that could be used to input vertical and horizontal coordinates on a grid using a stylus that emitted electromagnetic pulses. This particular system was used to manually record the coordinate locations of several facial features, such as eyes, hairline, mouth and nose. These values could then be inserted in a database. After this, once the user gave the system a new picture of an individual, it was capable to get the image from the database that most resembled that profile. Unfortunately, at that time, the technology of the era and computer processing power severely limited facial recognition. However, it was certainly an important step in proving that facial recognition was a possible viable biometric.

During the 1970s, Harmon, Lesk and Goldstein found a way to add better accuracy to a manual facial recognition system. 21 specific subjective markers were used in order to identify faces automatically. These markers included lip thickness and hair color. As for Bledsoe's system, the actual biometrics had to be computed manually.

At the 2002 Super Bowl, facial recognition was used by law enforcement officials in a major test of the technology. Unfortunately, the test was seen as a failure. False positives and backlash from critics proved that face recognition wasn't quite ready for prime time. One of the most important limitations during that time was that face recognition did not yet work well in large crowds, functionality that is relevant when using face recognition for event security.

HOW FACE RECOGNITION WORK

You might be good at recognizing faces. You probably find it a cinch to identify the face of a family member, friend, or acquaintance. You're familiar with their facial features — their eyes, nose, mouth — and how they come together.

That's how a facial recognition system works, but on a grand, algorithmic scale. Where you see a face, recognition technology sees data. That data can be stored and accessed. For instance, half of all American adults have their images stored in one or more facial-recognition databases that law enforcement agencies can search, according to a Georgetown University study.

So how does facial recognition work? Technologies vary, but here are the basic steps:

Step 1. A picture of your face is captured from a photo or video. Your face might appear alone or in a crowd. Your image may show you looking straight ahead or nearly in profile.

Step 2. Facial recognition software reads the geometry of your face. Key factors include the distance between your eyes and the distance from forehead to chin. The software identifies facial landmarks — one system identifies 68 of them — that are key to distinguishing your face. The result: your facial signature.

Step 3. Your facial signature — a mathematical formula — is compared to a database of known faces. And consider this: at least 117 million Americans have images of their faces in one or more police databases. According to a May 2018 report, the FBI has had access to 412 million facial images for searches.

Step 4. A determination is made. Your faceprint may match that of an image in a facial recognition system database.

Problem Definition

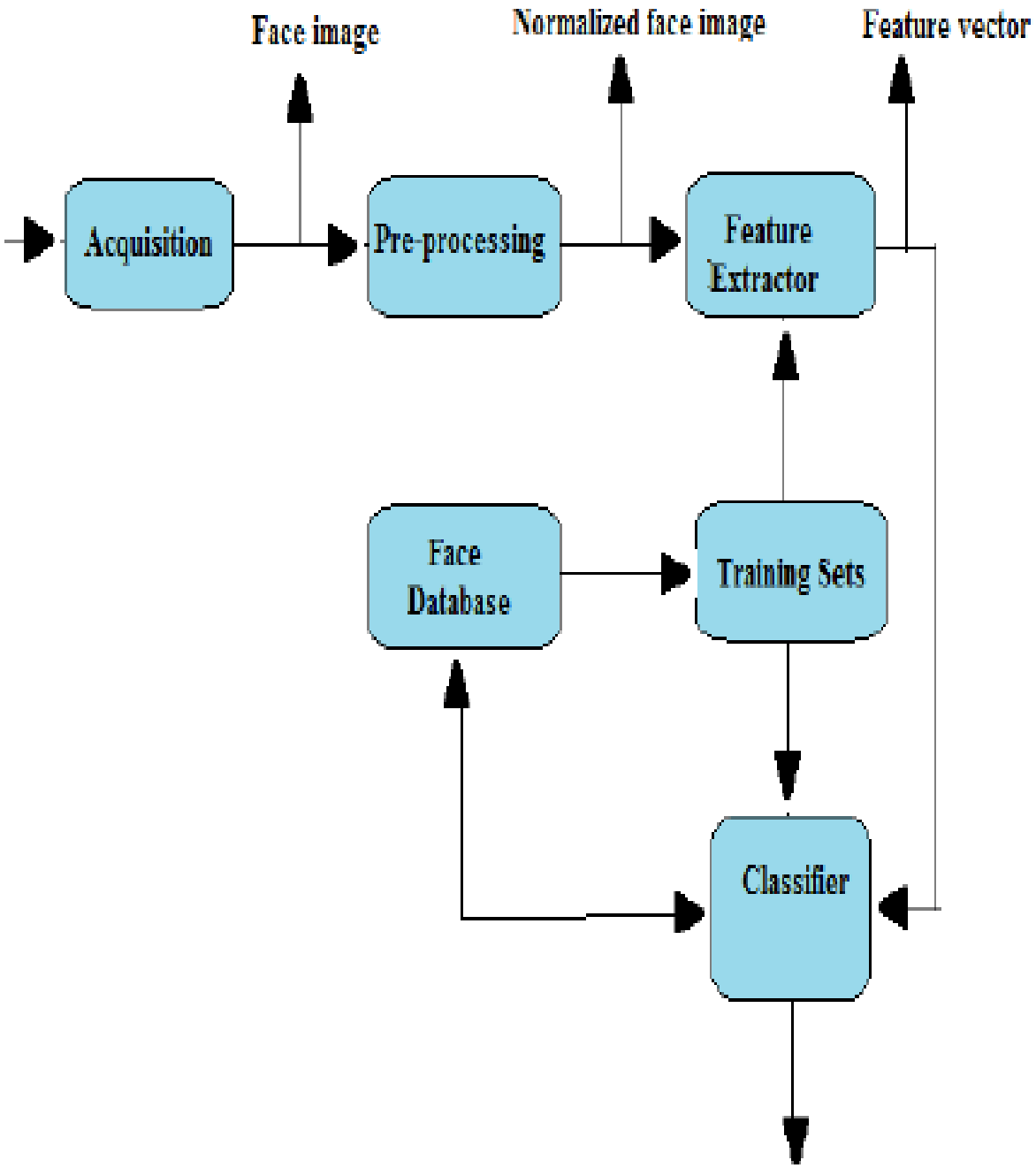
Over the past decade face detection and recognition have transcended from esoteric to popular areas of research in computer vision and one of the better and successful applications of image analysis and algorithm based understanding. Because of the intrinsic nature of the problem, computer vision is not only a computer science area of research, but also the object of neuroscientific and psychological studies also, mainly because of the general opinion that advances in computer image processing and understanding research will provide insights into how our brain work and vice versa. A general statement of the face recognition problem (in computer vision) can be formulated as follows: given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces. Facial recognition generally involves two stages:

- *Face Detection*
  Where a photo is searched to find a face, then the image is processed to crop and extract the person's face for easier recognition.
- *Face Recognition*
  Where that detected and processed face is compared to a database of known faces, to decide who that person is.

Proposed Solution

When image quality is taken into consideration, there is a plethora of factors that influence the system’s accuracy. It is extremely important to apply various image pre-processing techniques to standardize the images that you supply to a face recognition system. Most face recognition algorithms are extremely sensitive to lighting conditions, so that if it was trained to recognize a person when they are in a dark room, it probably won’t recognize them in a bright room, etc. This problem is referred to as "lumination dependent", and there are also many other issues, such as the face should also be in a very consistent position within the images (such as the eyes being in the same pixel coordinates), consistent size, rotation angle, hair and makeup, emotion (smiling, angry, etc), position of lights (to the left or above, etc). This is why it is so important to use a good image preprocessing filters before applying face recognition. You should also do things like removing the pixels around the face that aren't used, such as with an elliptical mask to only show the inner face region, not the hair and image background, since they change more than the face does. For simplicity, the face recognition system presented in this paper is Eigenfaces using grayscale images. The paper demonstrates how easily is to convert color images to grayscale (also called 'grayscale'), and then to apply Histogram Equalization as a very simple method of automatically standardizing the brightness and contrast of your facial images. For better results, you could use color face recognition (ideally with color histogram fitting in HSV or another color space instead of RGB), or apply more processing stages such as edge enhancement, contour detection, motion detection, etc. Also, this code is resizing images to a standard size, but this might change the aspect ratio of the face. In is described a method on how to resize an image while keeping its aspect ratio the same. OpenCV uses a type of face detector called a Haar Cascade classifier. Given an image, which can come from a file or from live video, the face detector examines each image location and classifies it as "Face" or "Not Face." Classification assumes a fixed scale for the face, say 50x50 pixels. Since faces in an image might be smaller or larger than this, the classifier runs over the image several times, to search for faces across a range of scales. This may seem an enormous amount of processing, but thanks to algorithmic tricks, explained in the sidebar, classification is very fast, even when it's applied at several scales. The classifier uses data stored in an XML file to decide how to classify each image location.

ARCHITECTURAL DIAGRAM



PYTHON'S IMAGE PROCESSING LIBRARIES

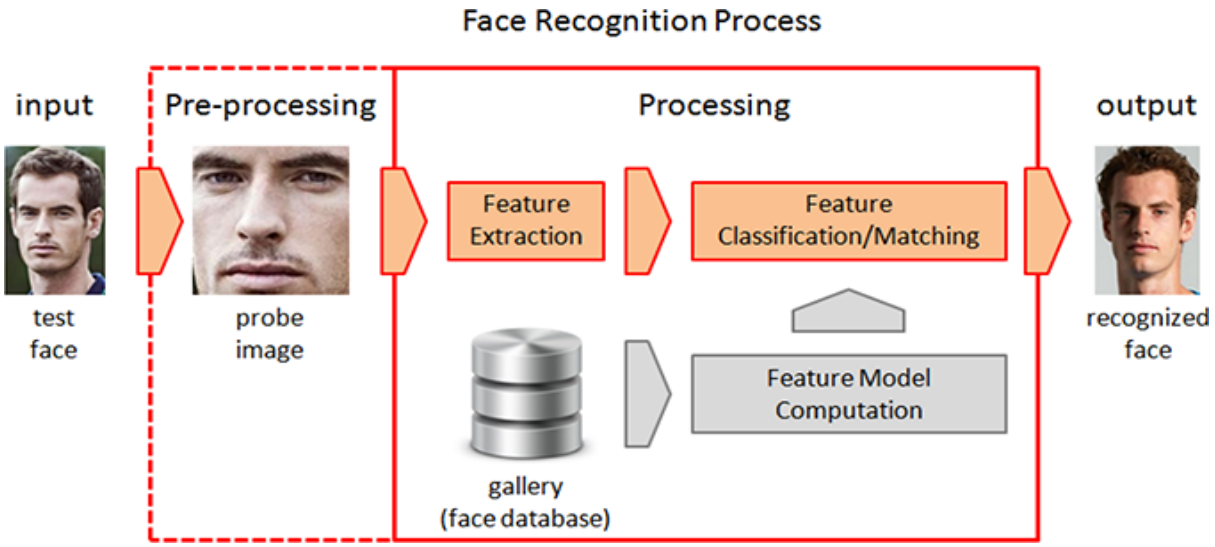
There are several Python libraries related to Image processing and Computer vision. The ones that will be used are:

- PIL/Pillow
  This library is mainly appropriate for simple image manipulations (rotation, resizing, etc.) and very basic image analysis (histogram for example)

- SimpleCV

  Its a library intended (as the name suggests) to be a simplified version of OpenCV. It doesn't offer all the possibilities of OpenCV, but it is easier to learn and use.

- OpenCV
  It is by far the most capable and most commonly used computer vision library. It is written in C/C++, but Python bindings are added during the installation. It also gives emphasis on real time image processing. Among the ones, which will not be presented it might be worth mentioning. It is a simple, user-friendly tool for interactive image classification, segmentation and analysis.



Python Imaging Library (PIL)

It's successor that also supports Python 3 is called Pillow.Consequently not both of them can be installed at the same time. At the time of this project, the last version of Pillow is 5.1.0. The most trivial program (showing an image) can be written as follows:

from PIL import Image im=Image.open('/path/to/the/image/') im.show()

Pillow is able to extract a lot of information from an image and makes it possible to execute several standard procedures for image manipulation, including:

- per-pixel manipulations,
- masking and transparency handling,
- image filtering, such as blurring, contouring, smoothing, or edge finding,
- image enhancing, such as sharpening, adjusting brightness, contrast or color.

Some of them will be demonstrated. Image can for example be easily rotated for specific angle (in our case 45°) and then saved by the following code:

rotated_image=im.rotate(45) rotated_image.save('rotated.jpg')

A color image can also be split into different components (red, green and blue).

r, g, b = im.split() r.show()

Image can also easily be sharpened or blurred. In this case it is however important that we also import the Image filter library. The code needed is shown below.

from PIL import ImageFilter sharp=im.filter(ImageFilter.SHARPEN) blur=im.filter(ImageFilter.BLUR)

Image can for example also be easily cropped with the following command

cropped_im=im.crop((100,100,400,400))

It was demonstrated that PIL/Pillow is very easy if only basic image processing task are needed. For more detailed analysis and computer vision SimpleCV and OpenCV are more appropriate.

SimpleCV

SimpleCV is an interface for Open Source machine vision libraries in Python. As the name suggests it is a simplified version of OpenCV. It is relatively easy to learn and use, as it is concise, has readable interface for cameras, and makes it possible to execute image manipulation, feature extraction, and format conversion. At the time of writing it is however still limited to Python2. As now the majority of Python users have already transferred to Python3, SimpleCV wont be described in detail. Below a simple program can be seen.

from SimpleCV import Image img = Image('lena.jpg') img.show()

As everything included in SimpleCV can be done in OpenCV, which is still developed, emphasis will be put on OpenCV.

OpenCV

OpenCV is an open source computer vision library available written in C and C++ which runs under Linux, Windows, Mac OS X, iOS, and Android. Interfaces are available for Python, Java, Ruby, Matlab, and other languages. A very simple program used just to show an image can be written as follows:

import numpy as np import cv2 img = cv2.imread('lena-color.jpg') cv2.imshow('image',img) cv2.waitKey(0) cv2.destroyAllWindows()

As OpenCV is nowadays the most suitable library for computer vision we will use it in the remaining part of the paper.

FACE DETECTION

OpenCV enables us to do even more complex tasks relatively easy. There are for example routines, which detect face. The following sequence of commands does just that.

```
face_cascade=cv2.CascadeClassifier (’C:\\Users\\...\\haarcascade
_frontalface_default.xml’) eye_cascade=cv2.CascadeClassifier
(’C:\\Users\\...\\haarcascade_eye.xml’) img=cv2.imread(’lena.jpg’)
gray=cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)
faces=face_cascade.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) roi_gray = gray[y:y+h, x:x+w]
roi_color = img[y:y+h, x:x+w] eyes = eye_cascade.detectMultiScale
    (roi_gray) for (ex,ey,ew,eh) in eyes: cv2.rectangle(roi_color,(ex,ey),
(ex+ew,ey+eh),(0,255,0),2) cv2.imshow(’img’,img) cv2.waitKey(0)
cv2.destroyAllWindows()
```

Algorithm works perfectly. If however a more complex image is used, the result is not so good. The algorithm itself applies the so called Haar feature-based cascade classifiers. The number of these features can be enormous. But most of them are irrelevant. A good feature is for example the fact that the region of the eyes is usually darker than the region of the nose and cheeks. A second good

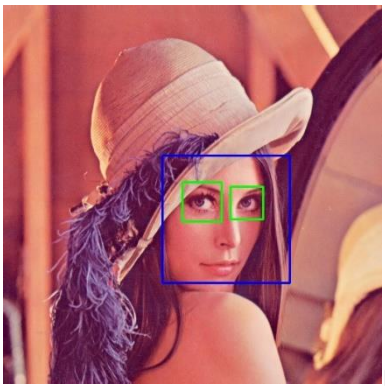


Figure 1. An example of face detection

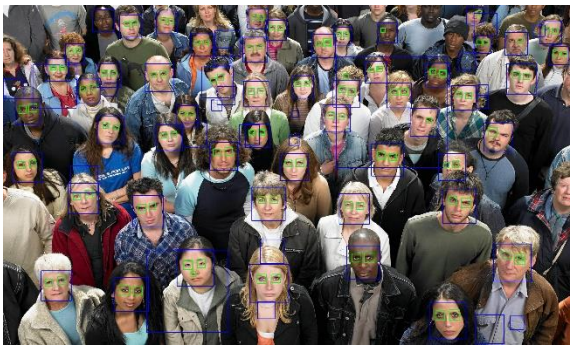


Figure 2. An example of face detection

feature could for example be based upon the fact that the eyes are usually darker than the bridge of the nose. With the increasing number of such features, we can increase the reliability of the algorithm. Misclassifications are of course always a possibility. It should also be noted, that the reliability decreases with the decreasing amount of pixels in the face area.

FACE RECOGNITION

The field of Face recognition could be defined as a research area in which images of faces are grouped into sets belonging to one individuum. Maybe it is easier to understand if we use Facebook as an example. In the past, Facebook was capable of detecting faces ,but then the user had to tag the person by clicking on the image and specifying its name. Now Facebook has the capability to tag everyone in the image automatically. This is achieved by Face recognition algorithms. In Python this task can be achieved using pre-trained convolutional neural networks and OpenCV. The whole program relies heavily on various libraries. So modules paths, face recognition, argparse, picle and os have to be imported into Python project. At first, some images of the person we wish to recognize must be collected. This can be done either manually, or by the application of the Microsoft's Bing API for searching. Ideally, the dataset should contain at least 30 images of each person. No other persons should be present in the images used for training. Two sample images (one for each person) are shown in Fig. 3. The network architecture used for

Figure 3.

face recognition is based on ResNet-34 neural network . The Python's face recognition library however has fewer layers and the number of filters is reduced by half. The network was trained on a dataset of approximately 3 million images.The algorithm is composed of four steps:

1. Finding all the faces In the first step we could use the face detection algorithm described in the previous section, which is the most commonly applied one. The face recognition library however uses a more advanced Histogram of Oriented Gradients (HOG) method .Color images must first be converted to grayscale ones. Then for each pixel in the image we look at the direction in which the image is getting darker. So we get a matrix of gradients . This matrix is to a large

extent independent of the brightness variations in the original image. It his however too big for manipulation. So submatrices

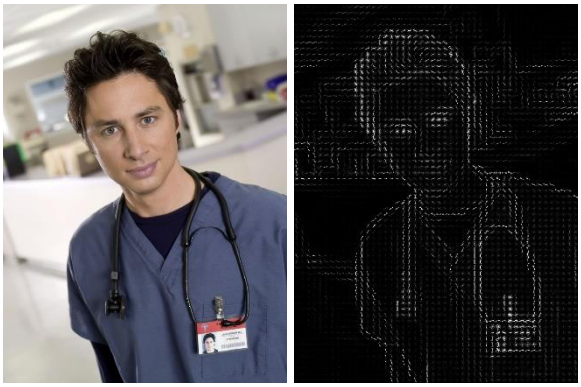


Figure 4. A sample of an original image and its histogram of gradients

of 16x16 size are formed. Then the predominant direction for each submatrix is found.

2. Posing and Projecting Faces. This step deals with the problem that faces in an image might be looking to some other direction and not straight into the camera. There are several solutions to this problem. Python's library uses the approach with 68 landmarks that are present on any face. Then a ma-

Figure 5. A face with the landmarks

chine learning algorithm is trained to locate these 68 landmarks on any face. After that the face is transformed using affine transformations so that eyes and mouth are centered as best as possible.

3. Encoding Faces
   The Deep Convolutional Neural Network is trained to generate 128 measurements for each face. The training process uses three images at a time (the image of a known person, another image of the same person and an image of some other person). This step requires a large dataset and a lot of computer power. But it has to be executed only once. Some pretrained neural networks are also available on internet.

4. Finding the persons name from the encoding
   The last step is actually a very basic one. The face being analyzed is compared to the faces that we have in our database. Python's library uses Support vector machine (SVM) to do that. In principle any other classification algorithm could be used.

The performance of the algorithm is presented on the image.

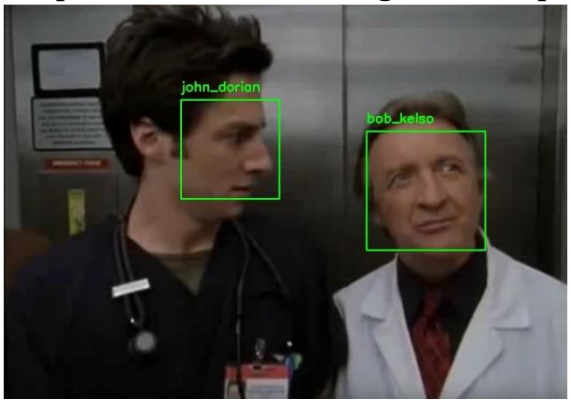


Figure 6. An example of the Face detection algorithm

ADVANTAGES OF FACE RECOGNITION

• **Security Through Biometric Authentication:** One of the benefits of facial recognition system centers on its application in biometrics. It can be used as a part of identification and access control systems in organizations, as well as personal devices, such as in the case of smartphones.

• **Automated Image Recognition:** The system can also be used to enable automated image recognition capabilities. Consider Facebook as an example. Social networking site can recognize photos of its users and allow automated linking or tagging to individual user profiles.

• **Deployment in Security Measures:** Similar to biometric application and automated image recognition, another advantage of facial recognition system involves its application in law enforcement and security systems. Automated biometric identity allows less intrusive monitoring and mass identification.

• **Human-Computer Interaction:** The system also support applications. Filters in Snapchat and Instagram use both AR and facial recognition. System facilitates further human-computer interaction.

• **Equips Devices with Added Functionalities:** It is also worth noting that equipping devices with facial recognition capabilities means expanding their capabilities. For example, iPhone devices from Apple use Face ID for biometric identification and supporting its AR capabilities

CONCLUSION

The paper is divided into two parts. In the first one a short overview of the most common Python's libraries related to image processing and computer vision is given. The second part uses the OpenCV library. In this part the libraries for face detection and face recognition are described and analyzed. Face detection and face recognition are namely two areas of intensive research, because they enable a better interaction between computer system or robots on one side and humans on the other. Python's face recognition library turns out to be fast and very reliable tool for face detection and face recognition. As Python is a high level programming language the library is well suited to be used just as a face detection(recognition) procedure in a wider project without the need for a detailed knowledge of the theoretical background of the employed algorithms. So, in my opinion it has a bright future.

In future it would be very interesting to investigate the possibilities of Python and its libraries for emotion detection. This field is a very hot topic in human machine interface research. Using the results of this research it is possible to vastly improve the social aspects of robots or software packages that can adapt to the user. It namely gives a very reliable feedback in human machine interaction.