

PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology

(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Technology Degree in

Information Technology

By

HARSH GUPTA (37120029)

SYED MUBBASHIR NAJMUL (37120077)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119. TAMIL NADU

MARCH 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **HARSH GUPTA (37120029) & SYED MUBBASHIR NAJMUL (37120077)** who carried out the "RETAIL CRIME PREVENTION USING FACE RECOGNITION" under our supervision from November 2020 to April 2021.

Internal Guide

Dr. SENDURU SRINIVASULU, MTech., Ph.D.,

Head of the Department

Dr. R.SUBHASHINI, M.E.,Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, HARSH GUPTA (37120029) and SYED MUBBASHIR NAJMUL (37120077) hereby declare that the Professional Training Report on “**RETAIL CRIME PREVENTION USING FACE RECOGNITION**” done under the guidance of **Dr. SENDURU SRINIVASULU, M.Tech., Ph.D.**, at Sathyabama Institute Of Science And Technology has submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind Encouragement in doing this project and for completing it successfully. I am grateful to them

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr.SUBHASHINI M.E., Ph.D. ,** Head of the Department of Information and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. SENDURU SRINIVASULU, MTech., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

Every year retail stores incur a huge loss due to theft. Several measures have taken place and failed to prevent the crime efficiently. Certain data provided by surveys show us that the maximum amount of theft and other crimes are committed by repeat offenders in a retail store. Our project focuses on using this particular data to try and prevent crimes with the help of caution. Using the technology of Face Recognition, we can identify a person entering the store, whether they have a previous record of crime inside the premises. If so, our project aims to notify the authorities inside the property to stay active and take necessary precautions in case of a complication. Face being one of the easiest body parts to detect and differentiate by a machine, we use Face Recognition to solve this problem.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	
	LIST OF FIGURES	ix
1.	INTRODUCTION	1
	1.1 FACE DETECTION	1
	1.2 MOTIVATION	1
	1.3 GOALS AND IMPLEMENTATION	2
2.	FACE RECOGNITION PROBLEM	3
	2.1 DEEP LEARNING	5
	2.2 PROBLEMS	6
3.	THEORITICAL BACKGROUND	7
	3.1 ARTIFICIAL NEURAL NETWORK	7
	3.2 WHAT ARE THEY?	7
	3.3 HOW ARE THEY TRAINED	9
	3.4 LAYER TYPES	10
4.	MY PROPOSAL	12
	4.1 PROCESS	14
	4.1.1 FEATURE EXTRACTION	14
	4.2 GLOBAL SYSTEM	16
5.	CONCLUSION AND FUTURE WORK	17
	REFERENCES	18
	APPENDIX	19
	SOURCE CODE	19
	SCREENSHOT	21

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
2.1	EXAMPLE OF TYPICAL GABOR	4
2.2	EXAMPLE OF EIGENFACES	5
2.3	VARIABILITY IN A FACE	6
3.1	THE THREE LAYERS OF ANN	8
4.1	FACENET	12
4.2	FIDUCIAL POINTS	15
5.1	SOURCE CODE SCREENSHOT	21
5.2	OUTPUT SCREENSHOT	21

Chapter 1

INTRODUCTION

Face Recognition (FR) is one of the areas from Computer Vision (CV) that has drawn more interest for long. The practical applications for it are many, ranging from biometrical security, to automatically tagging your friends pictures, and many more. Because of the possibilities, many companies and research centres have been working on it.

1.1FACE DETECTION

That being said, this problem is also a really difficult one, and it has not been until recent years that quality results are being obtained. In fact, this problem is usually split into different sub-problems to make it easier to work with, mainly face detection in an image, followed by the face recognition itself. There are also other tasks that can be performed in-between, such as formalizing faces, or extracting additional features from them. Through the years, many algorithms and techniques have been used, such as eigenfaces or Active Shape models. However, the one that is currently mostly used, and providing the best results, consists in using Deep Learning (DL), especially the Convolutional Neural Networks (CNN). These methods are currently obtaining high quality results, so, after reviewing the current state of art, we decided to focus this project on them.

1.2 MOTIVATION

This project was developed for the company Golden Spear LLC. Their goal is to create an AI enriched system oriented to the fashion world. As such, one of their main departments is devoted to developing such AI technology. This project was born there, as they wanted to recognize faces in uncontrolled environments. The uses for such a project are many, but there are some that are especially most relevant for GoldenSpear. First, to automatically uploaded pictures to make use of this information p.e. for improving recommendations or inferring their dressing style. Other uses would be to be able to unsupervised process media extracted from the internet, in order to transform it into usable information, or, as a longer-term project, to allow shops to link recurrent visits of clients.

1.3 GOALS AND IMPLEMENTATION

Our goal was to create a complete Face Recognition system, capable of working with any kind of images, and to constantly improve itself. This improvement had to be autonomous, and to allow it to better recognize people in it, and to include new ones. On top of that, the time requirements were also an issue, as this recognition must be done as close to real-time as possible. The task of recognizing faces, especially outside of controlled conditions, is an extremely difficult problem. In fact, there have been many approaches throughout the history that have not succeeded. Apart from the variance between pictures of the same face, such as expression, light conditions or facial hair, it is difficult to determine what makes a face recognizable. As such, our intention at the beginning of this project was not to start from the scratch, but to make use of some of the already existing research. This would allow us to speed up the process, and to make it more feasible to obtain quality results. In order to do so, we researched the history and current state of the field, as more thoroughly explained in Chapter 2. By doing so, we looked for successful ways of addressing the problem in which we could inspire. In the end, we decided to focus on the DeepFace [Taigman et al., 2014] implementation developed in Facebook's AI department. The main reasons are the good results obtained being really close to the state of art, and the quality of the description. It consists in a 3 step process. First, the face in the image is located and frontalized, so that it is looking at the camera. Then, the frontalized face is sent through a CNN, and a set of relevant features are extracted. Finally, these features are used as attributes to compare pairs of images to determine whether they belong or not to the same person. This document is structured as follows. Chapter 2 consists in a review on the history of Face Recognition, including the current state of art methods. In Chapter 3 we provide a theoretical background for Artificial Neural Networks and Deep Learning, the base for our system. The next two chapters [4 and 5] are devoted to describing the developed system, and to present the obtained results.

Chapter 2

FACE RECOGNITION PROBLEM

The recognition of a human face is one of the most basic activities that humans perform with ease on a daily basis. However, when this problem is tried to be solved using algorithms, it proves to be an extremely challenging one. The idea of a machine capable of knowing who is the person in front of them has existed for a long time, the first attempts happening on the 70s [Kelly, 1971]. The researchers have ranged from computer engineers to neural scientists [Chellappa, Wilson, and Sirohey, 1995]. However, during many years no quality solutions were obtained. It has not been until the late 2000s and beginning of the 2010s that functional systems have started to appear. The uses for an automatic face recognition system are many. Typical ones are biometric identification usually combined with other verification methods automatic border control, or crowd surveillance. One of its main advantages is its non intrusivity. Most identification methods require some action from people, either putting the fingerprint in a machine, introducing a password, etc. On the contrary, face recognition can work by simply having a camera recording. Among other uses, some of its most well known uses belong to the social network field. As of 2016, there are already systems being used that rely on face recognition, a brief sample of which are introduced here. This sample is by no means exhaustive, but it tries to show the variety of applications. It comes as no surprise that one of the most uses that draws most attention is to track criminals. As forensic TV series have shown, having a system automatically scanning city cameras to try to catch an escapee would be of great help. Infact, United States is already using this technology. Although far from the quality level depicted in fiction, they are already using it although there is some skepticism regarding whether it works to identify people from afar. Although the large criticism there is involving this kind of methods, there is little doubt that in the future they will become widely used. A not so well known use of face recognition is to authorize payments. As a part of a pilot test, some users are, under some circumstance, asked to take a picture of themselves before the payment is accepted.

In following years there will undoubtedly appear more, many of which we do not currently expect. On a more technical way, there have been, historically, many approaches to the problem. However, there is one key issue in the face recognition problem that most of them have shared, that is, the feature extraction. Most approaches to the problem start by transforming the original images to a more expressive set of features, either manually crafted, or automatically selecting some statistically relevant ones. In fact, working with the raw images is extremely difficult, due to factors such as light, pose, or background, among others. Therefore, by keeping only the information relevant to the face, most of this “noise” is discarded. There have been, traditionally, two main approaches to the problem: the geometric, which uses relevant facial features and the relations between them, and the photometric ones, which extracts statistical information from the image to use in different kinds of comparisons. Regarding the first approach, it is the most intuitive of both of them. Roughly, they rely on finding concrete facial points, and identifying people according to their relative position. There are many possible ways of detecting such features, some of which are introduced here. They are linear filters that are used for edge detection on almost every field of Computer Vision. They are roughly related to the way biological visual systems work [Deco and Lee, 2004]. They are usually applied at different scales to be scale invariant as shown in Figure 2.1, and they are capable of detecting interesting points at any image, such as edges, corners, circles, etc.

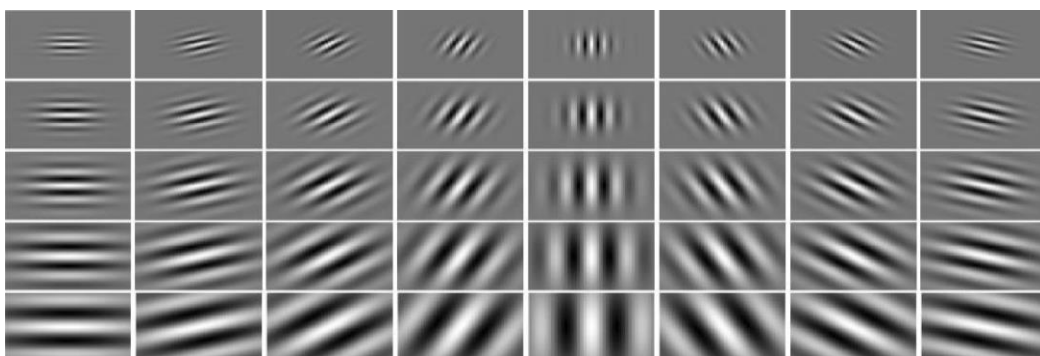


Fig 2.1: Example of typical Gabor

2.1 DEEP LEARNING

In recent years a new method has appeared which has affected the whole Computer Vision community. Since its appearance, Deep Learning, and more interestingly, the way CNNs work is closely related to the way biological visual system works [Itti and Koch, 2001; Kim, Kim, and Lee, 2015].



Fig 2.2: Example of eigenfaces:

Whether this is the reason of its success is out of the scope of this document, but it can not be denied that the results they are obtaining make them a choice to consider when faced with CV problems. In fact, a large number of the most successful applications of CV in recent years have used CNNs, and this tendency is expected to continue. Because of this, the work in this thesis makes use of them. Two of the most successful applications of CNNs in the FR problem are DeepFace [Taigman et al., 2014] and FaceNet [Schroff, Kalenichenko, and Philbin, 2015]. These two have provided state-of-art results in recent years, with the best results being obtained by the second ones. Although there are other methods providing close results, such as involving Joint Bayesian methods [Cao et al., 2013; Chen et al., 2013], we decided to focus on CNN. The reasons were not only result driven, but also interest driven, as we were personally interested in working with them.

2.2 PROBLEMS

Unfortunately, even though its potential, automatic face recognition has many problems. One of the most important ones is face variability in a single person. There are many factors that can influence so that two pictures from the same person look totally different, such as light, face expression or occlusion. Actually, when dealing with faces in controlled environments, face recognition systems are already delivering quality results, but they still have problems when faced with faces in the “wild”. Even more, factors such as sunglasses, beards, different hairstyles, or even age, can greatly difficult the task.

Another problem to be taken into account is the environment. Except in controlled scenarios, face pictures have very different backgrounds, which can make the problem of face recognition more difficult. In order to address this issue, many of the most successful systems focus on treating the face alone, discarding all the surroundings.



Fig 2.3: Variability in a face

Chapter 3

THEORETICAL BACKGROUND

This chapter aims to provide an introduction into the concept of Convolutional Neural Networks. In order to do so, it is necessary to understand the concept of Artificial Neural Network, so the first part of the chapter is devoted to do so. After that, Deep Learning and CNN are explained.

3.1 ARTIFICIAL NEURAL NETWORK

Inspired in their biological counterparts, Artificial Neural Networks are sets of interconnected computational nodes, usually with square or cubic shapes. They are a computational approach for problems in which the solution of the problem, or finding a proper representation, is difficult for traditional computer programs. The way they process information could be understood as receiving external inputs that can elicit, or not, a response in some of the nodes of the system neurons. The whole set of responses determines the final output of the network. They have proven their capacity in many problems, such as Computer Vision ones, which are difficult to address by extracting features in a traditional way. This section aims to briefly introduce the main technical concepts of the method, in order to make it easier to understand the Deep Learning explained afterwards.

3.2 WHAT ARE THEY?

The power of ANN comes from a set of computationally simple nodes that combine together, that is, the neurons. These neurons are structured in layers, which are connected between them, similarly to the way biological neurons are connected by axons. These layers are divided into 3 main types: input, hidden and output. The input layer corresponds to the data that the network receives. It could be understood as the input vector from other methods. This layer is connected to a hidden layer, that is, the ones that are not in the extremes. This is where their name comes, as they are not “visible” from the outside. Another interesting interpretation would be that, contrary.

To other methods, once the network is trained, looking at them does not provide any insight on what they do. As such, ANN are sometimes referred as black boxes, as it is high impossible to understand their functioning. There can be multiple hidden layers, each of them connected to the previous one. Every neuron in hidden and output layers are traditionally connected to all neurons from previous layer. Each edge has an associate weight, which indicates how strongly related the two neurons are, either directly or inversely, similarly to the way biological neurons are connected. Finally, the last layer is called output layer, and it delivers the result of the ANN, with one output per class. This is important, as ANN are mostly used for classification problems.

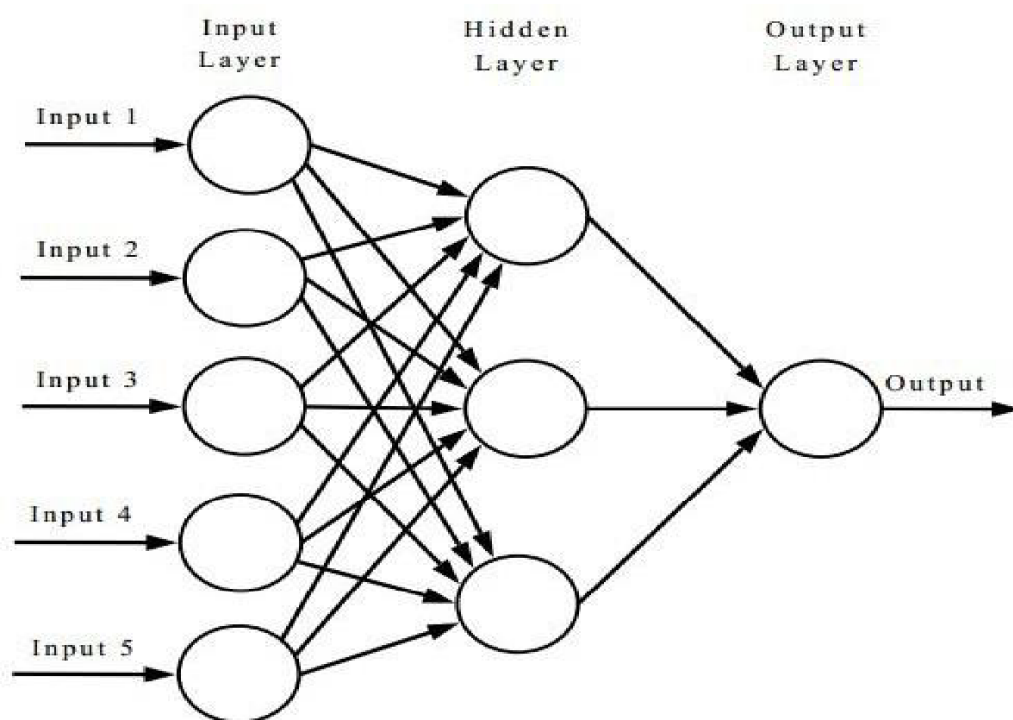


Fig 3.1: The three layers of an ANN:

This is, roughly speaking, the basic structure of an ANN. There are many variations over it, such as Recurrent Neural Networks, in which connections form a directed circle, but they are all based in this. They can be understood as a function f that maps an input X into an output Y . The training task, then, consists in learning the weight associated to each edge.

3.3 HOW ARE THEY TRAINED

One of the main requirements for training this kind of algorithms is data. All learning algorithms use data in their training processes, but ANN require more than most. As it will be explained in following chapters, this became a real issue during the project.

Given the data, there are various learning algorithms, from which gradient descent combined with backpropagation can be considered, given its widely spread use, the most successful of all of them. In fact, to a certain degree it could be considered that using it is enough for training most ANNs.

This algorithm starts by initializing all weights in the network, which can be done following various strategies. Some of the most common ones include drawing them from a probability distribution, or randomly setting them, although low values are advisable.

The process followed afterwards consists of 3 phases that are repeated many times over. In the first one, an input instance is propagated through all the network, and the output values are calculated. Then, this output is evaluated, using a loss function, with the correct output, and this is used to calculate how far off the network is. The final phase consists in updating each weight in order to minimize the obtained error. This is done by obtaining the gradient of each neuron, that could be understood as a “step” towards to actual value. When these three phases are repeated for all input instances we consider this an epoch. The algorithm can run for as many epochs as specified, or as required to find the solution.

There are two main ways of applying backpropagation that need to be mentioned, stochastic and batch. The stochastic approach is the one presented, in which weights are updated after each instance. This introduces a certain amount of randomness, preventing the algorithm from getting stuck in local optima. The other approach, instead, applies the weight update only after having processed a set of instances, using the average error. This usually leads makes the algorithm converge faster to a local minima, which may actually be a good result. A compromise between them both can be achieved using the mini-batch strategy. This one uses small batches with randomly selected samples, which combines both strategies benefits.

Therefore, it can be concluded that the learning task for neural networks consists in finding the right weights. The algorithm explained here is the one most

commonly used, although many other architectures use some variations over this basic algorithm.

There are also other ways of learning the weights, such as genetic algorithms [Montana and Davis, 1989] or simulated annealing [Aarts and Korst, 1989; Yamazaki, Souto, and Ludermir, 2002]. However, with some interesting exceptions, these methods are rarely used, being backpropagation the most commonly used algorithm by far. The choice of the algorithm will affect the performance of the whole method, so it is an important issue to take into account.

We have provided a description of the, arguably, most commonly used ANN, that is, the Multi Layer Perceptron. Variations over it lead to other well known types such as Recurrent Neural Networks (RNN) or Radial Basis Function network (RBF).

3.4 LAYER TYPES

In this section we introduce the 4 most used layer types, together with a fifth not so common, but relevant for our project.

Convolutional Layer: The most iconic layer has already been introduced. It is inspired in traditional MLPs, but having some major differences. The main ones are that each layer has a single set of weights for all neurons shared weights, and that each neuron only processes a small part of the input space. It uses all the parameters introduced in previous section.

Pooling Layer: It is common to include these layers between successive stacks of convolutional ones, in order to progressively reduce the size of the image representation. It works by taking each channel of its receptive field, and resizes it by keeping only the maximum of its values. It is usually used with 2×2 kernels, and a stride of 2, which halves each side size. This reduces the overall size in 75% by picking the larger of 2 patches. This kind of layers does not have weights that need training, and it only uses the stride and kernel size parameters. Its utility consists in reducing the amount of weights to learn, which reduces computational time as well as probability of overfitting.

Fully Connected Layer: These layers are basically neural layers connected to all neurons from previous layer, as the ones from regular ANNs. In this case, they do not use any of the introduced parameters, using instead the number of neurons. The

output they produce could be understood as a compact feature vector representing the input image. They are also used as output layers, with one neuron per output, as usual.

Locally Connected Layer: The last presented layer is really similar to the Convolutional Layer, but it does not use the shared weight strategy. This strategy is justified in normal Convolutional Layers because the relevant features are usually independent of their position in the image. However, there are cases in which this may not hold true. If you know, for example, that all your images will have a face centred to the same position, it makes sense to look for different features at the eye zone than at the mouth zone. This is achieved by giving each neuron its own set of weights, similarly to regular ANN, but they still only process their receptive field. These layers are commonly used after some Convolutional and Pooling ones due to 2 reasons. The first one is that, in order for the features from, p.e, eyes and mouth, to be different, they need to be a relatively abstract ones. Basic structures, such as edges or corners, are relevant in both cases. As already explained, this abstraction level is achieved by applying successive neuron layers, which builds “complex” features by means of simpler ones. Thus, the utility of using some Convolutional Layers before. The other reason is that using Locally Connected Layers introduces a large amount of weights into the network, making it more prone to overfitting. Then, it is better to use them when the image has already been reduced by previous layers. All in all, this type of layer is not very commonly used due to it requiring a fixed spatial distribution. However, if this condition is fulfilled, and you have enough data to prevent overfitting, they are an excellent choice.

There are other considerations to take into account when dealing with CNNs, such as using Dropout a regularization method that randomly disables neurons to prevent overfitting, but we deem the provided information enough for the purpose of this document.

Chapter 4

MY PROPOSAL

When I had to face the problem, my first step was to research the history and current state of the field, as explained in Chapter 2. This gave me a good basis on what I could expect, which methods I could use, and which ones I could discard. First of all I briefly considered using a geometric approach, by using some of the existing software in order to detect relevant facial features, and later use them as features in a classifier. This appealed to me due to its intuitiveness: the process could be easily understood, regardless the specific mathematical details. However, the existing research lead me, quite predictably, to the field of CNNs. As already explained, they are providing new result benchmarks in many Computer Vision applications, including Face Recognition. Additionally, due to the interest in the field, there is a large amount of research in it. Even more, even though some of this research belongs to companies and it is therefore private, there are many papers that are publicly available. These two factors quality of the results and availability of information lead us to choose to use CNNs in our Face Recognition system.

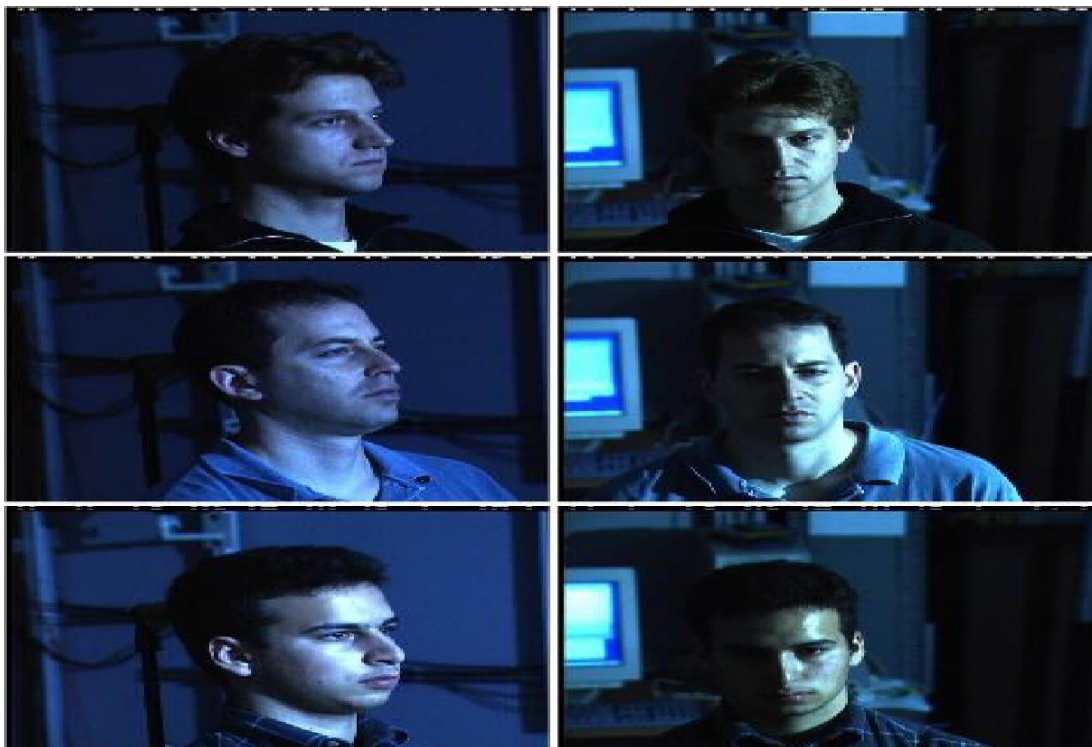


Fig 4.1 : Facenet

This would mean that we would not have time to properly test many CNN configurations, and, therefore, we would be unlikely to succeed. This is why we decided to base our system in some of the already existing methods. Even though this would mean that we could claim less of the resulting work, we would have a much

higher chance to succeed, and we could make more thorough tests. In order to determine which paper to choose, we gathered some of the state of art researches that were publicly described.

From all the papers and methods we considered, we ended up selecting two of the most interesting ones. The first one was FaceNet [Schroff, Kalenichenko, and Philbin, 2015], developed as part of Google's AI research. In the last decades, Google has made huge investments on expanding its AI capabilities. One of its most well known examples is the DeepMind company [Deep- Mind, 2016]. Due to the company's vast resources, it has obtained some of the current state-of-art AI systems. In the FaceNet case, its results are among the best ones worldwide. In fact, even though it does perform extremely well almost perfectly in benchmark datasets (Section 5.4.1), this is not its main feature. Instead, it is developed taking into account much more difficult situations than the ones in such datasets.

This system, which uses Deep Learning, has been trained using a massive 200 million instances private dataset. They must be credited with being able of making use of that much data in such a successful way. Given the quality of the results, the robustness of the method and the fact that they provide a description on how they implemented it, we considered that it would be interesting to inspire our work in it.

For the other option, we also considered one of the other technological leader companies, in this case, Facebook. Maybe more well known by the average user, their face recognition capabilities can be found on a daily basis when using their website. As such, it can be easily checked that it performs surprisingly well. The face recognition method they developed, called Deep- Face [Taigman et al., 2014], is also publicly available. Even though the results in some of the most well known benchmark datasets are not as good as FaceNet's, they were close to them nevertheless. The fact that they have it working in the most used social network is further proof of its capabilities. On top of that, the description of the method was 18 extremely straightforward, which made it easy to comprehend and implement. Finally, the training dataset consisted of roughly 4.4 million images, from an also private dataset.

4.1 PROCESS

Making use of DeepFace guidelines, we have developed a system capable of performing face recognition among a set of stored faces. New people can be

introduced into the system both automatically and manually, which makes it suitable for different scenarios. On top of that, it can work with videos, by identifying the people who appear there, and surround them with the face bounding box. The system is composed of two main parts: the extraction of features by means of a CNN, and using these features to identify a person. In this section we provide an in-depth explanation of how each of them works, as well as the explanation of how the video analysis works, and an online web tool we developed.

4.1.1 Feature extraction

The first part of the system consists in automatically extracting relevant features from the image. These features will later be used to determine whether two images belong or not to the same person. As such, it is extremely important for it to work properly. In this sense, we have used the implementation proposed in [Taigman et al., 2014]. They used a two step process in order to do so: first, they frontalized the face so that they all looked at front, followed by a CNN configuration that made use of that.

The intuition behind is that face recognition is difficult to address due to the large variability between face poses. If the CNN received the images as they are, it would have to deal with people looking in all directions, background noise, different positions of the face in the image, etc. Therefore, it makes sense to first try to reduce this variability by centering faces in the image.

Because of that, the first step would be to have all faces in the middle of the image, and transform them so that they look at the front. Doing so makes it much easier to successfully analyze them. In our system we have made use of the face frontalization library from [Hassner et al., 2015], as they provided a reliable 19 – implementation that worked fast enough for our requirements. It worked by detecting a set of fiducial points, as introduced in Chapter 2, corresponding to relevant face positions forming a basic face shape.



Fig 4.2: Fiducial points

Set of all fiducial points used in the frontalization phase. Notice how all of them are properly placed regardless of partial occlusions. However, in pictures with more than one person it failed sometimes not many at “choosing” the right one. We addressed this problem by using a Faster R-CNN [Ren et al., 2015] to locate faces, together with their bound boxes. This network would be later further used in other problems. In order to select the “main” face in the picture, we looked for the one with a centered bounding box, together with a high confidence of the network. These two factors were linearly combined to produce an overall value for each located face. This way we got rid from people in the background, who were usually in the sides of the image or more blurry. By doing so we effectively get rid of this problem in practically all cases. In those cases in which the network failed at detecting a face we skipped this step and processed the 20 image directly. Even though we risked to fail at choosing the proper face, we acceptable. Anyway, we expert to address this issue in the future.

After using the library on face images, we produced 140 140 RGB images, which an oval in the middle with the frontalized face, and the remaining space black. There are examples of both successful frontalizations as well as unsuccessful ones, but we had to find a way of automatically deciding which group each image belonged to.

4.2 GLOBAL SYSTEM

The Global System (GS) is composed of two main parts: feature extraction and person identification. The first part has already been explained in Section 4.1.1, so in this section we will focus in the second part, that is, how exactly do we use these features in order to identify a person, and also how we trained and tested the whole GS. We have addressed the building of the system as an online learner, so that even during the test phase the system can still learn.

In order to train the GS, a set of labeled people is needed, so that each person has its name available. For each of these images, the system processes them using 21 one of the CNN explained in a previous section including the frontalization step giving 4096 features.

Then, it tries to decide whether to keep it as a representative feature vector or to discard it. In order to do so, we have a pair of parameters: the minimum number of feature vectors required (Min) and the maximum number allowed (Max). The first one determines how many images at least we would like to have from each person. Therefore, each time a new feature vector f is received not only during training. we check how many feature vectors are already stored from that person.

Chapter 5

CONCLUSION AND FUTURE WORK

The computational methods, which were implemented in this project, were used after extensive research. Also, the successful results show that the done research is reliable. Face detection is not only a base program for further machine learning programs but also used separately. The above given face detection program can be used in several gadgets used for capturing pictures by detecting their faces. Face detection allows your camera to focus on your face, which further can be used to create certain editing modes. These modes are quite frequently used in today's day and age and has proved to be an instant hit.

For future work we might add certain features in the application to make it more easy to use for the client.

- With the help of an API link we provide to the client, they can access the front end of this project.
- By entering the given path of the database, the program is successfully running, giving us the output whether the sample is matched or not.

REFERENCES

1. Sathyabama Institute of Science and Technology
2. Udemy.com
3. Youtube
4. Blog.datumbox

APPENDIX

```
import cv2

import os

#import face_recognition

from flask import request

def image_compare(path_img1, path_img2):

    image1 = cv2.imread(path_img1)

    #rgb1 = cv2.cvtColor(image1, cv2.COLOR_BGR2RGB)


    #boxes = face_recognition.face_locations(rgb1)

    #encodings1 = face_recognition.face_encodings(rgb1, boxes)

    image2 = cv2.imread(path_img2)

    #rgb2 = cv2.cvtColor(image2, cv2.COLOR_BGR2RGB)

    #boxes = face_recognition.face_locations(rgb2)

    #encodings2 = face_recognition.face_encodings(rgb2, boxes)

    #matches = face_recognition.compare_faces([encodings1[0]], encodings2[0])

    hist1 = cv2.calcHist([image1],[0],None,[256],[0,256]) #distance formula

    hist2 = cv2.calcHist([image2],[0],None,[256],[0,256]) #matrix

    a=cv2.compareHist(hist1,hist2,cv2.HISTCMP_BHATTACHARYYA)

    #if(matches[0] == True):

        #return("Face Matched")

    #else:

        #return("Face Not matched")
```

```
if(a<0.6):

    return("Face Matched")

else:

    return("Face Not Matched")

from flask import Flask

app = Flask(__name__) #we are creating a server

@app.route('/face_match', methods=['post'])

def face_match1():

    input = request.get_json()

    print("Hello1")

    path1 = input["wanted"]

    print("Hello2")

    path2 = input["suspected"]

    print("Hello3")

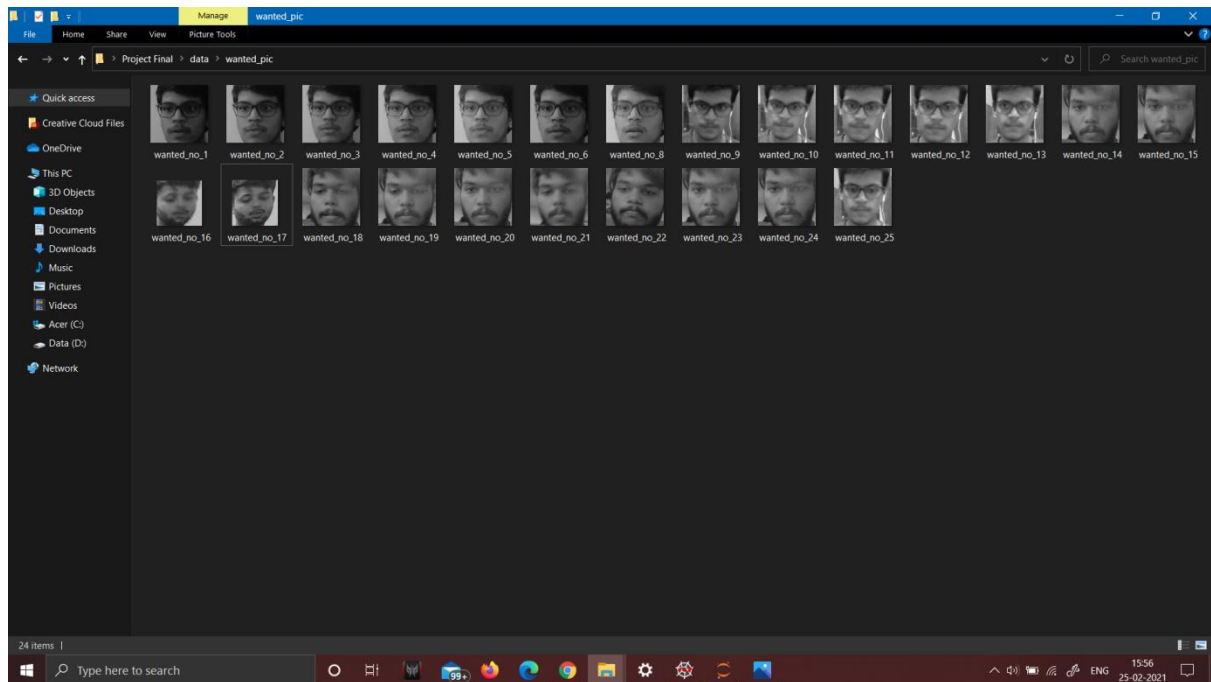
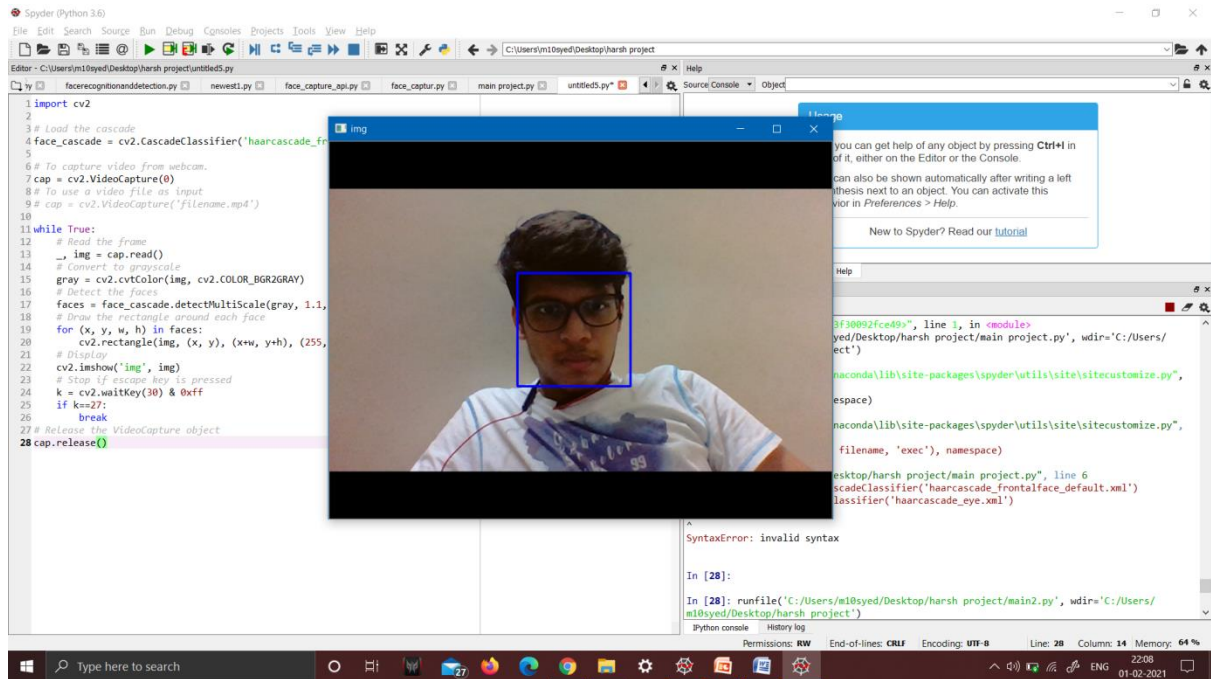
    response = image_compare(path1, path2)

    return response, 200

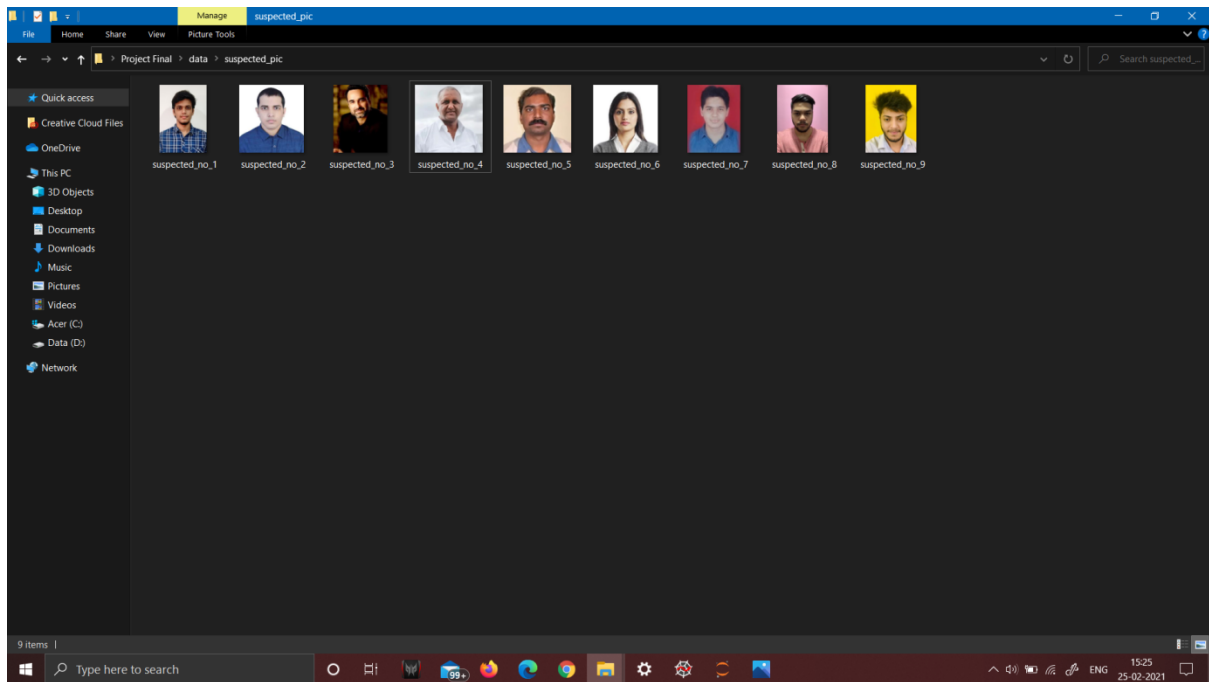

if __name__ == '__main__':

    app.run(port = 8080)
```

SCREENSHOTS



Sample database



Suspected database

```
In [13]: import json
import urllib3
import requests

url = 'http://127.0.0.1:5001/face_match'
path1 = r'C:\Users\m10syed\Desktop\Project Final\data\wanted_pic\wanted_no_1.jpg'
path2 = r'C:\Users\m10syed\Desktop\Project Final\data\suspected_pic\suspected_no_1.jpg'
payload = {"wanted": path1, "suspected": path2}
headers = {'content-type': 'application/json'}

response1 = requests.post(url, data=json.dumps(payload), headers=headers)

In [14]: response1.text
Out[14]: 'Face Matched'
```

Final Output