Cairo University
Faculty of Engineering
Aerospace & Aeronautical Department
Second Year

MTH 216B

Mathematics III

Project's Preliminary Report

On

Facial Recognition Technology

Firnas Team Members

	Section Number	Bench Number
1. Omar Sabry Abaas	**1**	**25**
2. Karim Ahmed Kamal	**1**	**27**
3. Mohamed Ismail Ibrahim	**2**	**07**
4. Mohamed Saeed Mohamed	**2**	**10**
5. Mohamed Magdy Ahmed	**2**	**14**
6. Nancy Nady Nathan	**2**	**24**

I. Abstract

Nowadays; face recognition is one of our lives essential, and it's a part of computer vision. In this report we made an overview on different methods. Then we used logistic regression with simple neural network, which is a classifier that could be used to classify any type of objects like: human, or cars or even animals such as cats.

II. Table of Contents

III. List of Figures

IV. List of Tables

1. Introduction

From the oldest ages, early humans have used the face recognition method to detect their tribe members or whether someone is angry/happy. By the development reached in our century and the tremendous number of people present now, governments and companies need to recognise the people they're working with.

So, great efforts are made to develop systems that can verify whether he/she is a permissible person or not depending essentially on biometric identifications. As since the use of traditional identifications such as birth certificate or a national ID can be easily forged or hacked, the use of biometric identification ensures that this person is who we want as for example two persons having the same fingerprint doesn't exist.

1.1. Biometric Identification

It is an identification based on the measurements and calculations of the characteristics of each person – the use of biological or physical character to identify a person/identity like his unique fingerprint, face, distinctive voice or DNA.

Types of biometric identification include fingerprint identification, hand geometry, palm vein authentication, iris scan, face recognition, signature, voice analysis, and more.

The use of face recognition method is recommended in most cases as

a. It doesn't require nor physical interaction of a sensor with the test subject neither positioning the test subject in a particular place for a moment so that the sensor take time to identify.
b. For surveillance in public areas, face recognition method is efficient, time saving and it's not an expensive technique.

Despite that, in high security systems the iris scan or fingerprint methods are preferred to be implemented with other methods as they are more accurate and gives no chance for errors. Thus, great efforts are made every-day to develop systems that can detect a face in a photo/video, recognise whose face it is and verify whether he/she is a permissible person or not.

1.2. History of Face Recognition

Woody Bledsoe, Helen Chan Wolf, and Charles Bisson were the pioneers in the field of face recognition. During 1964 and 1965, they worked on the concept of recognizing faces using computers. Despite that, little of the work was published because the funding was provided by an un-named intelligence agency that did not allow much publicity.

They had a large database of images (a book of mug shots) and a photograph, their problem, as expected, was to select a small group of records from the database such that one of the image records matched the photograph

This project was labeled man-machine because both the human and the computer work together, as the human factor extracts the coordinates of a previously wanted features such as the center of pupils, the inside corner of eyes, the outside corner of eyes, point of widows peak, and so on…., from the photographs, which were then used by the computer for recognition.

They used RAND tablet, a graphic tablet that could processes about 40 picture/hour and it takes horizontal and vertical coordinates as inputs on a grid using a stylus that emitted electromagnetic pulses. The system could be used to manually record the coordinate locations of various facial features.

Unfortunately, it is rarely that any two pictures would match in head rotation, lean, tilt, and scale. So, each set of distances is normalized to represent the face in a frontal orientation.

These metrics could then be inserted in a database. Then, when the system was given a new photograph of an individual, it was able to retrieve the image from the database that most closely resembled that individual. At the time, face recognition was unfortunately limited severely by the technology of that era and computer processing power. However, it was an important first step in proving that face recognition was a viable biometric.

1.3. Confusing Definitions

1.3.1. Face Detection

- It is the process of identifying an object as a face and locate it in the input image.
- It is considered the first step in analysing and processing photos.

1.3.2. Face Verification

- It is more like a true or false question, in which the system is given an image and an ID as inputs, and it must verify whether this person is on the database or not, esp. used in banks, offices...., where the employees are known.
- It can be checking that the two input faces are for the same person.
- It is done by something hold by the individual like an ID card or a passport.

1.3.3. Face Recognition

- It is much harder than the face verification.
- This can be done by comparing the similarities/differences in both images, if it is more/less than some threshold parameter, then the other photo belongs to the same person.
- It is the task in which the system decides whether the face is unknown or not.
- The output of the face detection is the input for face recognition.

2. Some methods Used

There are many methods that were developed during the last century, here is some but not all.

1. Elastic Bunch Graph Matching known as EBGM.
2. Linear Discriminant Analysis known as LDA.
3. Local Binary Pattern Histogram knows as LBPH.
4. Principle Component Analysis known as PCA.
5. Artificial Neural Network known as ANN (CNN with logistic regression algorithm)

Now, we'll have a closer look about each method.

2.1. Elastic Bunch Graph Matching (EBGM)

All faces have the same topological structure. EBGM uses this structural information and the capability of that structure to translate, scale and rotate. It is an algorithm that uses special points in the human face (like: nose, eyes, etc.…) to put some points called nodes labelled with 2-D distance vectors, each node consists of forty complex **Gabor-wavelet**[1] coefficient sets with different scales and orientations. These nodes are called jets. A labelled graph (image graph) is built by these nodes as the set of nodes are connected to each other by edges as the nodes are labelled with jets and edges which refer to the distance. This graph can be translated, scaled and rotated during matching process as the main structure do. In matching process, a face graph is originated and compared to all face bunch graphs in the data base to find an identification of the person. The matching percentage is determined from the similarity between face graphs of database and the tested image.

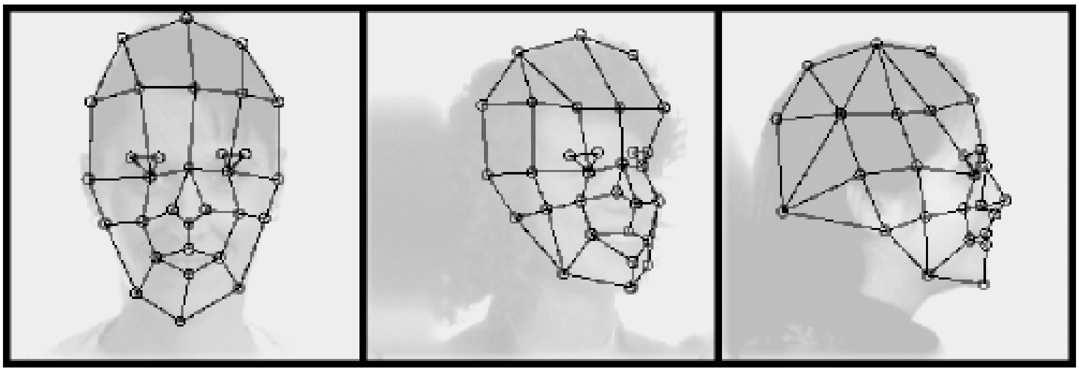

Figure 01

2.1.1. Advantages of EBGM

a. It has a very high accuracy.
b. On increasing the number of landmarks (jets), the efficiency increases.
c. Useful in feature detection.
d. It can be combined with Eigenfaces.
e. It can recognize face by rotational angle up to 22°.
f. Can use only one image as a database of the person.

[1] The Gabor-wavelet transformation is used to produce the local features of the face images. Gabor wavelets Are
biologically motivated convolution kernels in the shape of plan waves restricted by a Gaussian envelop function, the
set of convolution coefficients for kernels of different orientations and frequencies at one image pixel is called a jet.

2.1.2. Disadvantages of EBGM

a. It is a slow method, it even becomes slower with increasing the landmarks.
b. It is sensitive for lightening conditions.
c. A lot of graphs should be placed manually on face.

2.2. Linear Discriminant Analysis

It's a reduction technique used in classification. It gives us small features set that has the most relevant information for the classification. It's an approach for classifying unknown classes of samples into known classes by training samples with known classes. This technique uses maximization between different classes and minimization across the same class. Each class is represented as a block with large differences between each other but small variation in each block(class).

It uses two measures, within-class scatter and between-class scatter, then it calculates the eigenvectors of the projection matrix (W), finally, it compares the test, the image projection matrix to each one of the trained images.

2.2.1. Advantages of LDA

a. It is better than PCA.
b. It supports data classification.

2.2.2. Disadvantages of LDA

It suffers from small-sample size problem.

2.3. Local Binary Pattern Histogram

LBPH is one of the oldest methods used in facial recognition. It is one of the most accurate methods. This technique deals with the pixels and each pixel intensity. It labels the pixels of the image by taking bundles of pixels with each other.

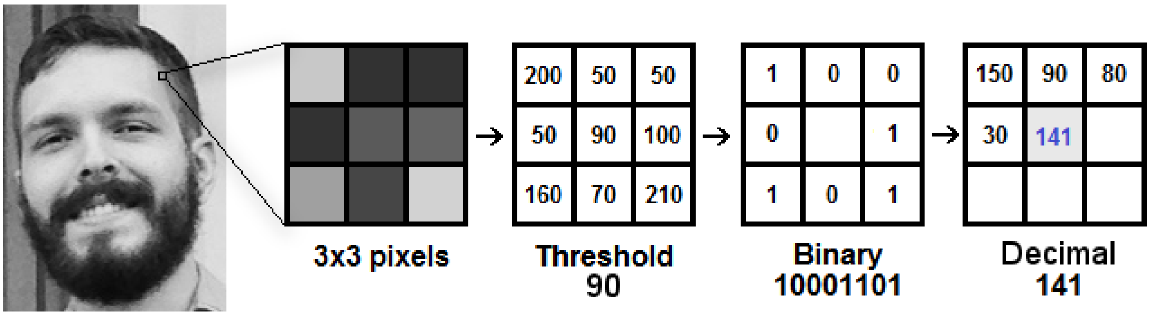

Figure 02

Every group of pixels, as an example: 3x3 pixels, are expressed by a matrix. Every pixel is expressed by its intensity of that pixel 0-255 (8 bits). We take the centre value of the matrix as a reference then we use it to change the matrix to binary form. Each value smaller than the chosen value is replaced by zero and each greater or equal value is replaced by one. The matrix is changed to binary except the central one so we need to avoid the central value. We arrange the binary values in a fixed way (e.g. 10001101) and then convert it to decimal value. We put that value to the centre of the matrix. At the end we could have a new simple image. We can extract histogram of the regions of that image. We then use that method on the tested image and find the closest histogram to it using the suitable approach.

2.3.1. Advantages

a. One of the simplest methods.
b. The change in illumination doesn't affect the accuracy as all pixel values change by the same scale.
c. It can represent local image features.

2.3.2. Disadvantages

It is considered a slow method as it produces long histograms especially on using large database.

2.4. <u>Principle Component Analysis</u>

PCA is one of the most important and successful techniques used in image recognition. It was invented in 1901 by Carl Pearson. PCA is a classic tool reduces the dimensionality of the data space to smaller size of feature space. The main idea of using PCA in face recognition is to extract a small set of characteristic features of face image and construct Eigen faces, which are the principal components of the initial training set of the face image, So PCA transforms 2-D facial image vector of pixels to 1-D vector which is named as Eigen space projection. PCA algorithm are shown in fig ()

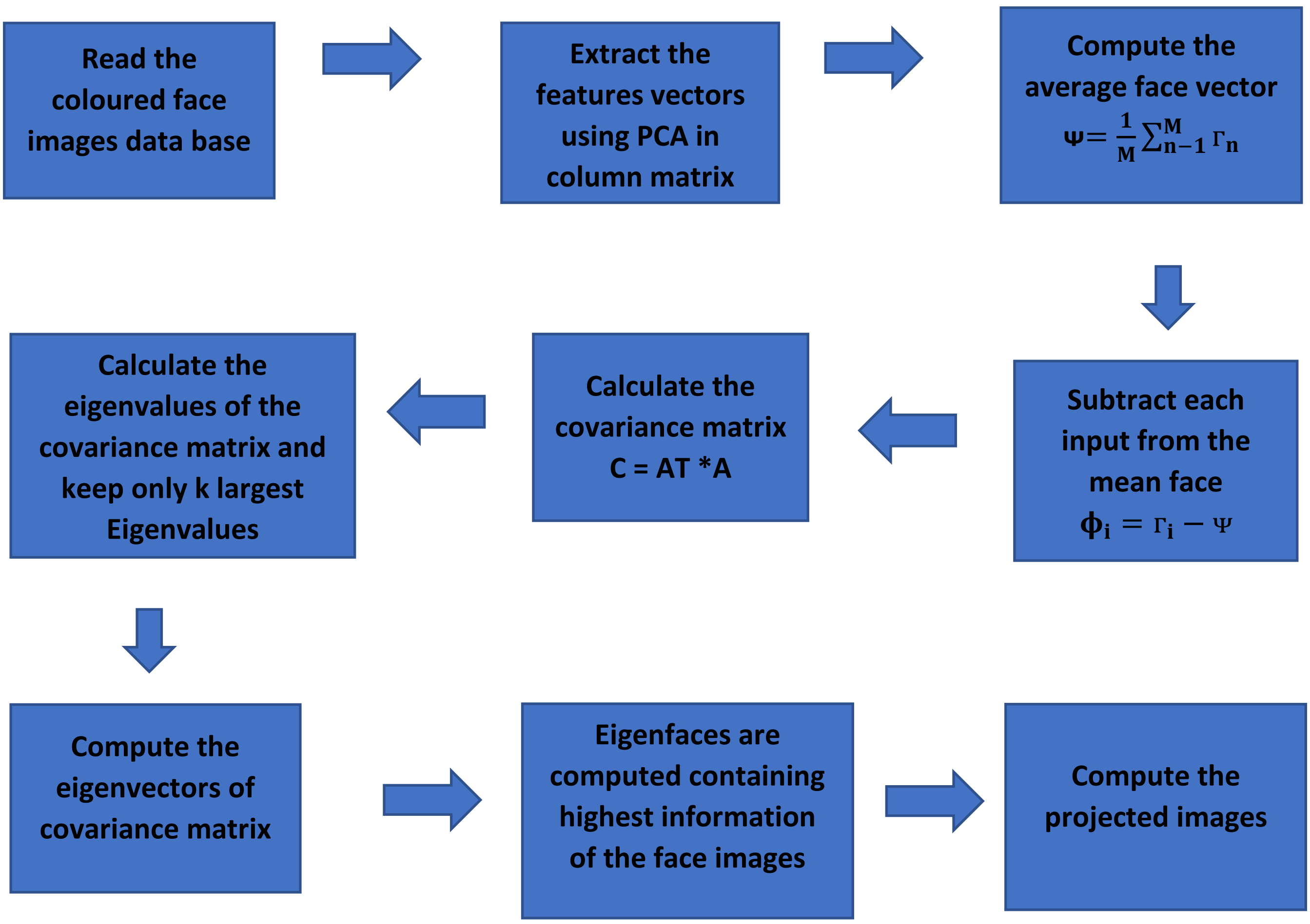

Figure 03

2.4.1. <u>Advantages of PCA</u>

1. The ability to reduce the size of the database

2. Noise reduction from the training database and the small variations are ignored automatically.

2.5. Artificial Neural Network (ANN)

ANN is an attempt to make computers simulate the human brain in the way it solves some problems such as pattern recognition, face recognition, and classification of facial expression.

Human brain has about 100 billion cells called neurons, they are connected to each other through paths that give neurons the ability to send and receive electrical impulses which in turn are responsible for the brain function. Each neuron has a function, it takes some inputs from other cells through the paths referred to and it also considers how strong those connections and then gives an output for a given cell. These neurons don't depend on each other and work in asynchronous way. So, they can solve complex and noisy data problems.

ANN is just a function that takes a few input numbers and gives output few numbers and in between it does some computation with the hidden layers. Each input signal is multiplied by connection strength number which is called weights, then they are added, enter the function, then exit as the output.

Input 1 * Strength of connection1
Input 2 * Strength of connection1 $\xrightarrow{\text{function}}$ output

2.5.1. Advantages of ANN

a. Its ability to detect complex nonlinear relationships between dependent and independent variables.
b. It achieve higher accuracy in detection compared to other techniques because they are trained with large number of samples.
c. The ability to learn how to do tasks (adaptive learning).

2.5.2. Disadvantages of ANN

a. It is a slow technique as it requires a lot of training and case (the training time is about 4 hours while classification time is less than 0.5 second).
b. The modification of ANN is very complex.

The training process is as shown.

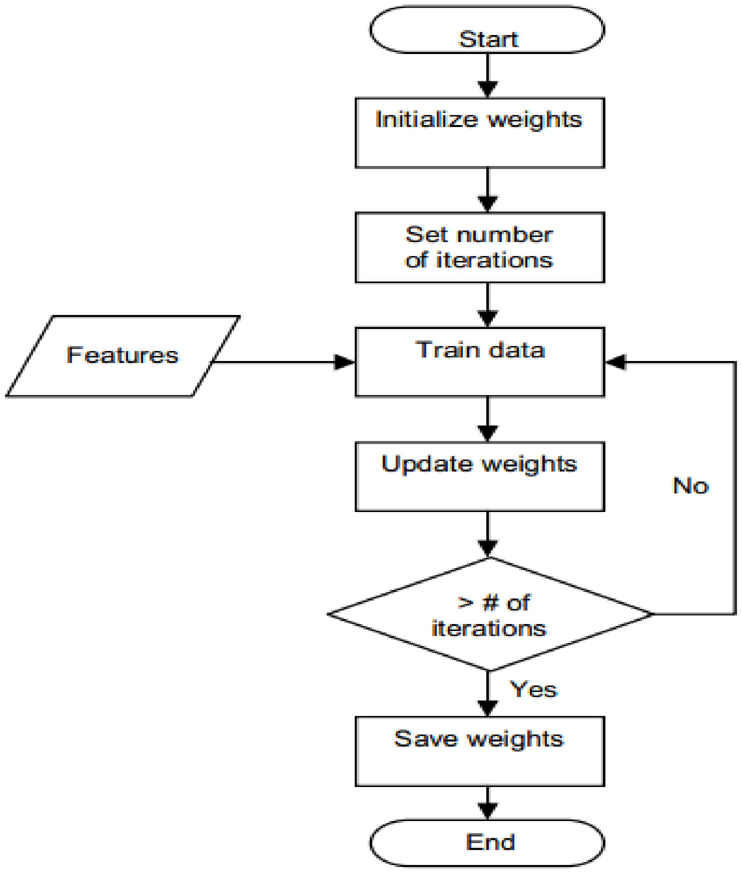
Start
Initialize weights
Set number of iterations
Features
Train data
Update weights
No
> # of iterations
Yes
Save weights
End

Figure 04

3. Logistic Regression with simple neural network (CNN)

Is an algorithm for binary classification, it's a part of ANN algorithms, statistical method for analysing a dataset in which there are one or more independent variables that determine an outcome that has only two possible values, so it is like any regression it is a predictive analysis for a binary outcome (0 or 1).

Logistic regression goal is to find the best fitting model to describe the relation between the dependent variable (the outcome) and the set of independent variables like training sets or testing sets.

Now the question is how we will use this algorithm to detect an image or recognize a face? say we have an input of an image as the one we discussed before and we want to output a label y to recognize this image for example as either being a cat in which case the output is 1, or not a cat in which case the output is 0.

First let's see how an image is represented in a computer. For each image the computer store three separate matrices, see the following figure corresponding to the red, green and blue colour channels of this image.

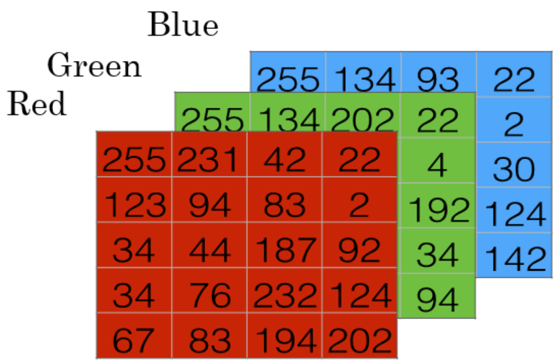

Figure 05

Say that the image is 64 pixels by 64 pixels then you have 3 matrices each is 64 by 64 corresponding to red, green and blue pixel intensity values for this image.

Now we need to turn these pixel intensity values into an input feature vector x. so all we are going to do is define a feature vector x as follows. We just take all the pixel values 255, 231, and so on until we list all the red pixels then eventually 255, 134 and so on till **we** get along feature vector listing all the red, green and blue pixel intensity values of this image by order finally we get a vector x of dimension (64x64x3) that will be 12,888 pixels and we are going to use $n_x = 12.888$ to represent the dimension of the input feature x.

Now let's go further into our logistic model by declaring some notations. A single training example is represented by a pair (x, y) where x is an x-dimensional feature vector and y is the output label vector either 0 or 1. Our training sets will comprise lower-case m training examples so $(x_1 . y_1)$ which is the input and the output of the first training example up to $(x_m . y_m)$ which is our last training example and then that all together is our entire training set. Finally, to output all of the training examples we define a matrix, capital X. by taking the training set inputs x_1 , x_2 and so on and stacking them in columns, then the matrix X will have M columns where M is the number of training examples and its height is n_x. It turns out that to make our implementation of neural network easier we also need to stack the output in columns so we have another matrix, capital Y of dimension 1 by M.

We need an algorithm that can output a prediction which we called $\hat{y}$ (y hat) which is our estimate of Y. more formally we want $\hat{y}$ to be the probability of the chance that Y is equal to 1. In other words, if X is a picture we want $\hat{y}$ to tell us what the chance that this picture is a cat.

So, we have an input X and logistic regression parameters W which is also an x-dimensional vector and b which is just a real number.

Now how we generate $\hat{y}$ we can try a kind of linear function we say that $\hat{y}$ will be $W^T X + b$, this is what we do if we were doing linear regression but this won't work with binary classification as we want $\hat{y}$ to be the chance of y so $\hat{y}$ should really be between 0 and 1, but $W^T X + b$ can be much bigger than one or may be negative which doesn't make sense for probability that you want it to be between 0 and 1.

So, in logistic regression our output is going to be the sigmoid function applied to the quantity $W^T X + b$.

3.1. What is the sigmoid function (α)

A sigmoid function is a mathematical function having a characteristic S-shaped curve or sigmoid curve have domain of all real numbers, with return value monotonically increasing most often from 0 to 1 and defined by the formula: $\alpha(z) = \frac{1}{1+e^{-z}}$

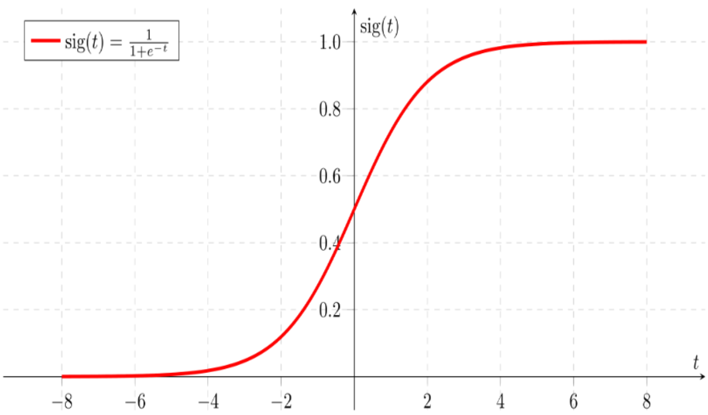

Figure 06

This is what the sigmoid function looks like if we take $Z = W^T X + b$ then we plot Z then the function of Z goes smoothly from 0 up to 1 where Z is a real number. So if Z is a very large number then e^{-z} will be close to zero so the sigmoid of Z will be one over one plus something very close to zero so it will be close to 1. If Z is very small or very large negative number then e^{-z} will be very large number then the sigmoid of Z will be one over one plus something very large so it will be close to 0.

Until now we have seen what the logistic regression model looks like.

Now to train the parameters W and b we need to define a cost function.

First; we find a loss (error) function for each training example that tell us how well our algorithm is doing.

The loss function (L):

$$L(\hat{y}.y) = -(y \, log \, \hat{y} + (1-y)\log(1-\hat{y}) \,)$$

We need $L(\hat{y}.y)$ to be as small as possible so to understand if this makes sense, let's look at the two cases.

First, let's say Y is equal to 1 then the loss function $L(\hat{y}.y)$ = - $y \, log \, \hat{y}$. So, we need $log \, \hat{y}$ to

as big as possible and that means we want $\hat{y}$ to be large (close to 1).

the other case if Y is equal to zero, the loss function $L(\hat{y}.y)$ = -log$(1-\hat{y})$. So, we need $\log(1-\hat{y})$

to be large to be as big as possible this mean we want $\hat{y}$ to be small (close to 0).

This saying that the loss function will push the parameters to make $\hat{y}$ close to 0 or 1.

3.2. <u>The Cost Function (J)</u>

Now we define the cost function (J) which measures how well our entire training set doing, this function is applied to the parameters W and b will be the average sum of the loss function (L) for every training example.

$$J(W.b) = \frac{1}{M}\sum_{i=1}^{M} L(\hat{y}^{(i)}.y^{(i)}) = -\frac{1}{M}\sum_{i=1}^{M}[y^{(i)}\log \hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})]$$

Where:

y	: the real value of the picture
$\hat{y}$	: the predicted value of the picture
M	: is the number of training example

know we should know how to choose parameters W and b to minimize the cost function we will use gradient descent technique, let's know more about it:

3.3. Gradient Descent Method

It's a technique which depend on back propagation

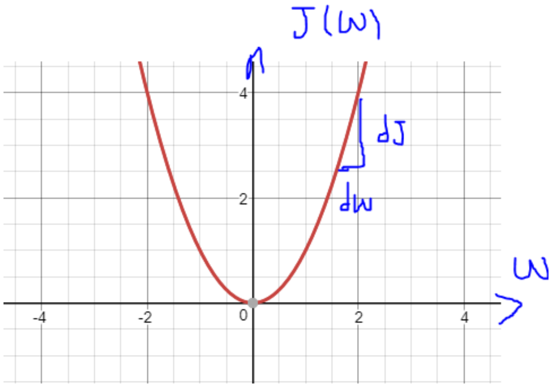

Figure 07

We need to reach value of W by which the cost function j gets minimum

So that we will assume any value of w then we would update its value by back propagation by next equation

W = W – $\alpha \frac{\partial J}{\partial W}$

where α = constant called learning rate and it will be discussed later

Similar for b = b – $\alpha \frac{\partial J}{\partial b}$

following graph will construct that concept

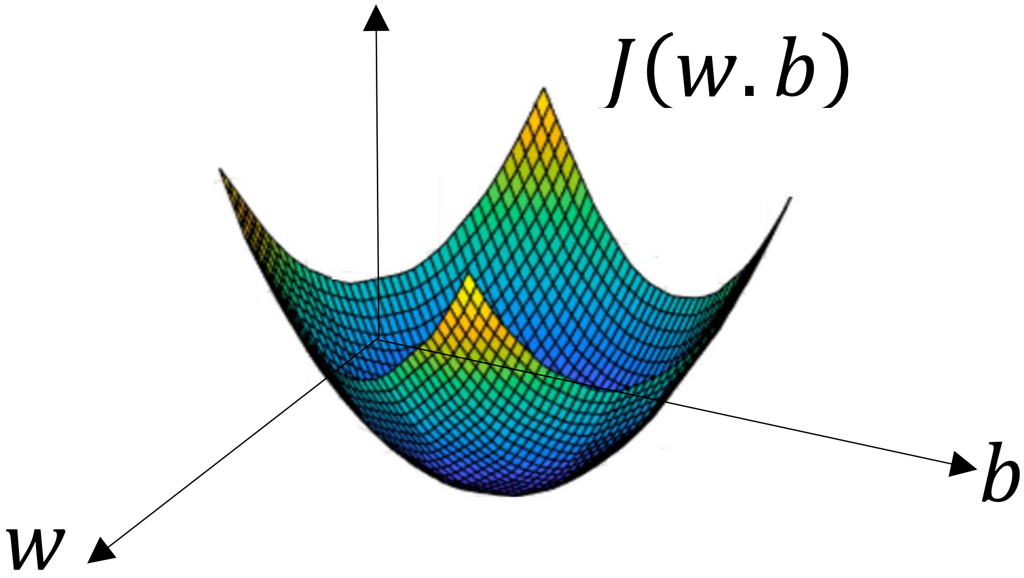

Figure 08

Now let's go on.

- First, we will get the image RGB and put in x
- Then get parameters W and b by training
- Then z = $w^T x + b$
- $\hat{y} = \sigma(z)$
- Get loss and cost function which help in determine most fitting parameters

3.4. Code Documentation

First, we involve a python file with our code and here we would explain it:

- Libraries used:

  1 – Numpy : its useful for matrices processing

  2 – PIL : useful for getting RGB of images

  3 – glob : useful for import array of images

- Overview:
  It divides into two steps: training and testing
  On testing:

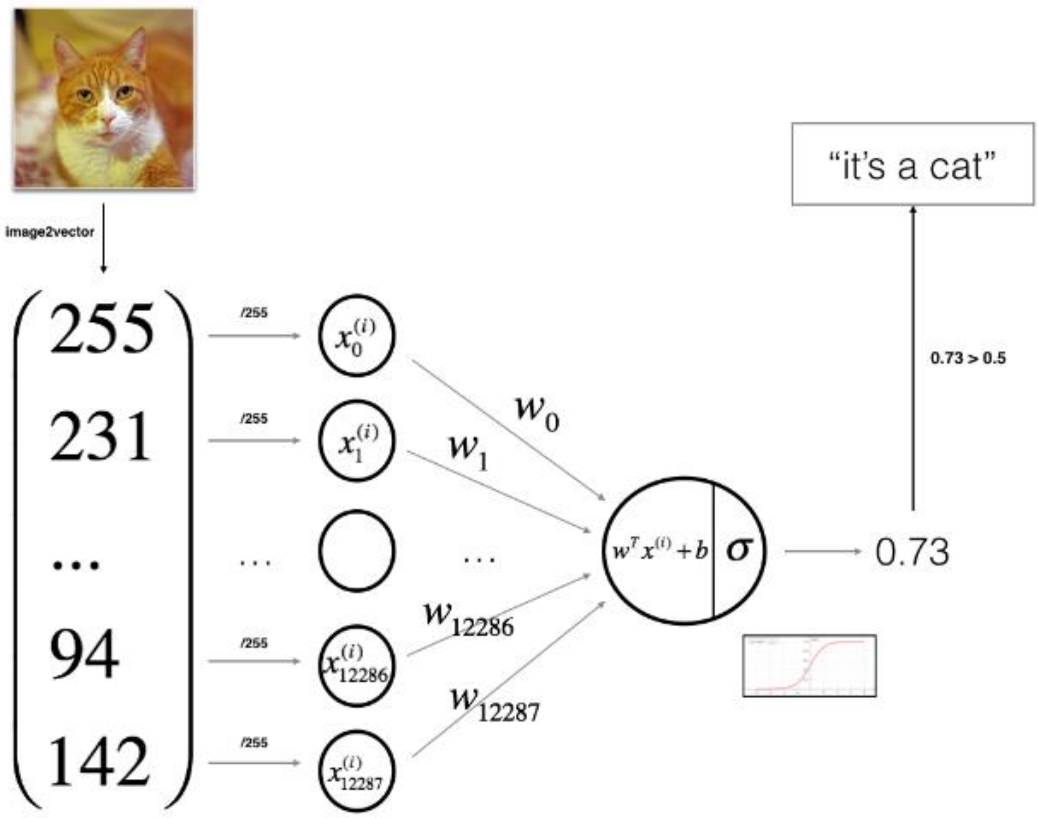

Figure 09

This figure introduces the process after we get most fitting parameters W and b, so we firstly invert the RGB matrices to one vector as shown, and we deal with probability so we need percent of red in pixel or green or blue not its value, so we dived by 255 as we use a 8-bit coloured image in it the maximum value it 255.

Then we calculate z where z = $w^T x + b$, after that we need to get it in range of 0 to 1, so that we calculate sigmoid function $\sigma(z)$ where $\sigma(z) = \frac{1}{1+e^{-z}}$

So, If the value of the sigmoid is greater than 0.5, the algorithm predicts a cat picture

If not, it predicts non-cat picture.

On training:

We import two types of arrays of images: training-set & test-set

In training set an array we import every image and get its RGB matrices and flatten it to a vector then we add all vectors to a 2-D matrix every column in it contain an image or an example for training, Then we create a matrix called train-set-y in it we insert 0 or 1, 0 for non-cat and 1 for cat picture

And its size is (1*number of training examples).

Then we get this matrix inside the previous algorithm as shown in the test section and let it to predict if the training set images are cat or non-cat picture , then we take these values and subtract it from the original values in train-set-y and get the loss of this training and when we get the average of all losses we get the cost function j ,from which we can get most fitting parameters(W, b) using gradient descent algorithm which has been explained

$$\mathrm{L} = \frac{1}{2}(\hat{y} \; - \mathrm{y})^2$$

$$J(W.b) = \frac{1}{M}\sum_{i=1}^{M} L(\hat{y}^{(i)}.y^{(i)})$$

- Accuracy and learning rate
  Learning rate is a constant used in gradient descent algorithm, it is an estimated constant which is vary from model to another , and it affect the accuracy deeply , so we should be careful when we choose it , next we will introduce some data based on our tests .

Learning rate	Test accuracy
$5 * 10^{-8}$	36.36 %
$5 * 10^{-5}$	54.54 %
$5 * 10^{-3}$	100 %

Table 01

It is shown in the following figure.
So that we need for every model to test our leaning rate very well, not to lose any accuracy

```
def start():
    train_set_x = data_set('D:\\train')
    test_set_x = data_set('D:\\test')
   # train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
    #test_set_y =  np.array([0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0])
    train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
    test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
    d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=1000, learning_rate=0.000000005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
m3a
Cost after iteration 700: 0.690762
Cost after iteration 800: 0.690424
Cost after iteration 900: 0.690086
test[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]]
train accuracy: 83.33333333333334 %
test accuracy: 36.36363636363637 %
```

```
    train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
    test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
    d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=1000, learning_rate=0.000005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
m3a
Cost after iteration 700: 0.251023
Cost after iteration 800: 0.229302
Cost after iteration 900: 0.210568
test[[1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0.]]
train accuracy: 100.0 %
test accuracy: 54.54545454545455 %
```

```
    #test_set_y =  np.array([0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0])
    train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
    test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
    d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=1000, learning_rate=0.005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
m3a
Cost after iteration 700: 0.000282
Cost after iteration 800: 0.000247
Cost after iteration 900: 0.000220
test[[0. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0.]]
train accuracy: 100.0 %
test accuracy: 100.0 %
```

Figure 10

Know there is also a main parameter that affect the accuracy it is number of iterations that make to choose the parameter (W, b) in gradient descent method

Number of iterations	Accuracy
10	81.81 %
100	90.1 %
100	100 %

Table 02

The following picture show data of our tests

```
#test_set_y =  np.array([0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0])
train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=1000, learning_rate=0.005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
om3a
Cost after iteration 700: 0.000282
Cost after iteration 800: 0.000247
Cost after iteration 900: 0.000220
test[[0. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0.]]
train accuracy: 100.0 %
test accuracy: 100.0 %
```

tests

```
#test_set_y =  np.array([0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0])
train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=100, learning_rate=0.005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
om3a
['D:\\train\\lfso.1.jpg', 'D:\\train\\lfso.10.jpg', 'D:\\train\\lfso.11.jpg', 'D:\\train\\lfso.12.jpg', 'D:\\train\\lfso.13.jpg
['D:\\test\\ekavaz.16.jpg', 'D:\\test\\hartb.7.jpg', 'D:\\test\\hensm.14.jpg', 'D:\\test\\lfso.11.jpg', 'D:\\test\\lfso.15.jpg'
Cost after iteration 0: 0.693147
test[[1. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0.]]
train accuracy: 100.0 %
test accuracy: 90.9090909090909 %
```

```
    train_set_y = np.array([1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0])
    test_set_y =  np.array([0,0,0,1,1,1,1,0,0,0,0])
    d = model(train_set_x, train_set_y, test_set_x, test_set_y, num_iterations=10, learning_rate=0.005, print_cost=True)


if __name__ == '__main__':
    start()
```

```
start()
m3a
C:\Users\DELL\PycharmProjects\untitled\venv\Scripts\python.exe "C:/Users/DELL/.PyCharmCE2017.3/config/scratches/neural netw
['D:\\train\\lfso.1.jpg', 'D:\\train\\lfso.10.jpg', 'D:\\train\\lfso.11.jpg', 'D:\\train\\lfso.12.jpg', 'D:\\train\\lfso.13
['D:\\test\\ekavaz.16.jpg', 'D:\\test\\hartb.7.jpg', 'D:\\test\\hensm.14.jpg', 'D:\\test\\lfso.11.jpg', 'D:\\test\\lfso.15.
Cost after iteration 0: 0.693147
test[[1. 0. 1. 1. 1. 1. 1. 0. 0. 0. 0.]]
train[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0.]]
train accuracy: 100.0 %
test accuracy: 81.81818181818181 %
```

Figure 11

But now what if we increase the training data or number of iterations too much, it will cause an overfitting, and we will discuss this in the final report.

Note we attached with the draft report a python file contains the code and two files of image one for the training set images and another for test set images.

4. Conclusion

There are lots of methods that could do the mission but each one has its advantages and disadvantages. For different applications, different methods fit it. We used Logistic regression that mostly fit application of self-driving car. It's a good start for that field as it is perfect in classify different objects like: trees, people and obstacles.

V. References

- https://www.coursera.org/learn/convolutional-neural-networks/lecture/lUBYU/what-is-face-recognition
- http://www.technovelgy.com/ct/Technology-Article.asp?ArtNum=12
- https://pdfs.semanticscholar.org/2542/64840a613a946ea42f7b3ad31007b6c6e62b.pdf
- https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b
- https://www.youtube.com/watch?v=wpAwdsubl1w
- http://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?article=1642&context=etd
- https://pdfs.semanticscholar.org/2542/64840a613a946ea42f7b3ad31007b6c6e62b.pdf