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

# A.I. BASED FACIAL EXPRESSION AND RECOGNITION

A Project Report of Capstone Project – 2

*Submitted by*

**Akash Gupta**
**(1613101078 / 16SCSE101596)**

***in partial fulfillment for the award of the degree***

***of***

**Bachelor of Technology**

**IN**

**Computer Science and Engineering**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**Under the supervision of**
**Dr. Prashant Johri**

**Professor**

**APRIL/MAY-2020**

# DECLARATION

**Project Title**: **A.I. Based Facial Expression and Recognition**

Degree for which the project work is submitted: **Bachelor of Technology in Computer Science and Engineering**

I declare that the presented project represents largely my own ideas and work in my own words. Where others ideas or words have been included, I have adequately cited and listed in the reference materials. The report has been prepared without resorting to plagiarism. I have adhered to all principles of academic honesty and integrity. No falsified or fabricated data have been presented in the report. I understand that any violation of the above will cause for disciplinary action by the Institute, including revoking the conferred degree, if conferred, and can also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been taken.

Signature( )

Akash Gupta

Enrolment No. 1613101078

Date: 08/05/20

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

# SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

## BONAFIDE CERTIFICATE

Certified that this project report "**A.I. Based Facial Expression and recognition**" is the bonafide work of "**Akash Gupta ( 1613101078 )**" who carried out the project work under my supervision.

| SIGNATURE OF HEAD | SIGNATURE OF SUPERVISOR |
|---|---|
| Dr. MUNISH SHABARWAL, | Dr. PRASHANT JOHRI, |
| **Professor & Dean,** | **Professor,** |
| **School of Computing Science & Engineering** | **School of Computing Science & Engineering** |

# ABSTRACT

Authenticity is one of the main aspects needed in today's world, **Age estimation and face recognition** are the most robust techniques to maintain Authenticity. In today's world fraud and scam are on the rise and to curb all of this we have implemented this project. Our field of study is computer vision, **Computer vision** is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. Computer vision uses techniques from machine learning and, in turn, some machine learning techniques are developed especially for computer vision. Face recognition has been implemented by using **neural networks for deep leaning (CNN)**. Though there are other techniques also to implement this like Local binary pattern histograms (LBPH)**, But CNN to this date gives the highest accuracy.**
Age estimation is a relatively new work which is not very successful but we have tried to follow [1] Gil Levi and Tal Hassner(2015) ,Age and Gender Classification Using Convolutional Neural Networks. which can successfully estimate the age of a person from an image or webcam.I used their code for Age estimation using anaconda python and OpenCV.

# ACKNOWLEDGEMENT

The contributions of many different people, in their different ways, have made this possible. We would like to extend our gratitude to our project guide (Dr. Prashant Johri) Who gave us the opportunity to make this project on this topic(**"A.I. Based Facial Expression and Recognition"**), which helped us in doing a lot of research and we came to know about many new things so we are thankful to them.

Secondly, we would like to thank our parents and friends who help me in making this project with a limited frame of time.

# TABLE OF CONTENTS

# List of figures

# Chapter.1 Introduction

## 1.1 Purpose

There are increasing incidents of fraud related to age and identity below are the types related to the same.

All of these crimes are related to wrong identification of age of people. In order to curb this issue our system will be efficient and play an important role in combating this increasing age fraud cases. It is important to validate the authenticity of a person regarding age as date of birth is required in almost every legal official record in government or non-government organization for issue of important documents like citizen card, financial claims etc. Some of the cases which took headlines due to age fraud:

**Credit Card fraud:** The crime of credit card fraud begins when someone either fake credit card details or fraudulently obtains the card number and other account information like age is necessary for the card to be used successfully.

**Employment and Tax Related:** criminal frauds in producing false documents with fake age of a professional to get inside margin line of employment misleading the organisation and taking job over deserved candidates. This also make them eligible for rebate on taxation policy.

**Phone and Utilities:** fraudsters using false identification to use mobile service and utilities offered by service provider as cable accounts are actually the general type of utility fraud we see, followed by the opening of fraudulent household electricity and gas accounts.

**Bank Fraud and grant of Loan:** opening a bank account with fake documents hiding actual identification like name, age, income etc. to fulfil the eligibility is kind of forgery banks are dealing with nowadays. Loan is granted on the basis of credit score which comprises of personal details and by producing forged documents will cost banks a huge stack of money.

**Government benefits:** to enjoy or utilise the benefits offered by government then an individual need to match desired criteria or standards set by governing body but scams and fraud take places due to submitting fake proofs and certificates by citizens.

# Chapter. 2 Literature Survey

This field viz, face recognition and age estimation is a very trending field and much work has been done on it like **[2]** Antitza Dantcheva, Petros Elia, Arun Ross, Proposed using soft biometrics like face features combined with age features to make a robust, identification system, **[3]**Zafeiriou Stefanos, Cha Zhang, and Zhengyou Zhang **,** wrote about a survey on viola Jones vs the deep neural network approach. A lot of work has been on age estimation also like by **[4]** Rasmus Rothe, Radu Timofte , in which they have implemented age estimation using convolutional neural networks using VGG-16 architecture

## Related work:

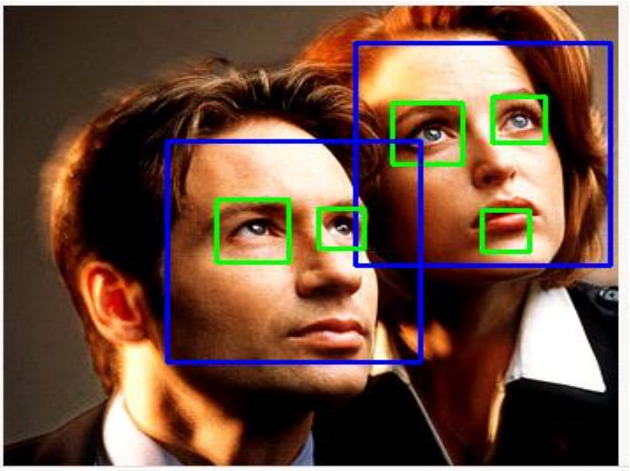
### Face Detection

Over the years there has been a lot of research in the field of Face detection and recognition, and various methodologies that can be used regarding it. We will list a majority of them and will explain about each one in detail. There are other algorithms also but as we have implemented the code in OpenCV and python so we listed only those which are present in OpenCV

- Haar Cascades
- Local Binary Pattern Histograms (LBP Cascades)
- Deep learning

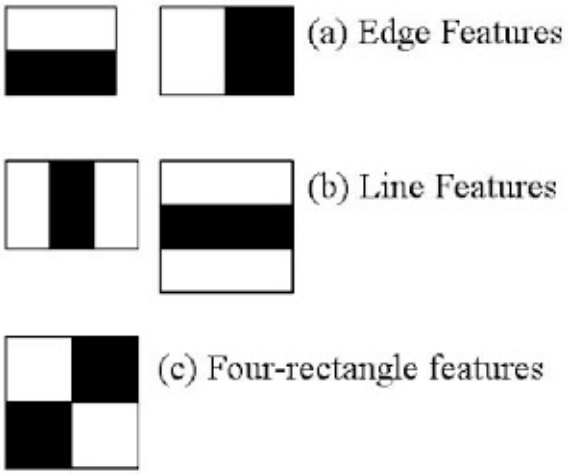
**Haar Cascades:** The first algorithm that was developed for face detection. It was researched by Paul Viola and Micheal Jones [6]. Haar features are digital image features used in object detection, the name is based on Haar wavelets. Viola and Jones used the idea of Haar wavelets and developed the so-called Haar-like features. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, let us say we have an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore, a common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is

defined relative to a detection window that acts like a bounding box to the target object (the face in this case).



(Figure 2.1)

Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in the below image are used. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.




(Figure 2.2)

## Haar Cascade Detection in OpenCv:

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one. First, we need to load the required XML classifiers. Then load our input image (or video) in grayscale mode.

Now we find the faces in the image. If faces are found, it returns the positions of detected faces as Rect(x,y,w,h). Once we get these locations, we can create a ROI for the face and apply eye detection on this ROI , since eyes are always on the face .

## LBP Cascade classifier

It is also an object detection in digital images, but the working of LBP is different that the Haar features. Each training image is divided into some blocks as shown in the picture below.

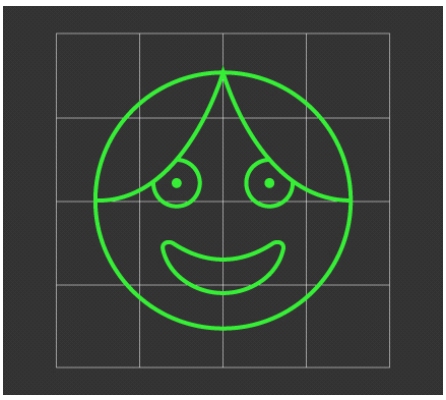


Figure (2.3)

For each block, LBP looks at 9 pixels (3×3 window) at a time, and with a particular interest in the pixel located in the center of the window. Then, it compares the central pixel value with every neighbor's pixel value under the 3×3 window. For each neighbor pixel that is greater than or equal to the center pixel, it sets its value to 1, and for the others, it sets them to 0.

After that, it reads the updated pixel values (which can be either 0 or 1) in a clockwise order and forms a binary number. Next, it converts the binary number into a decimal number, and that decimal number is the new value of the center pixel. We do this for every pixel in a block.

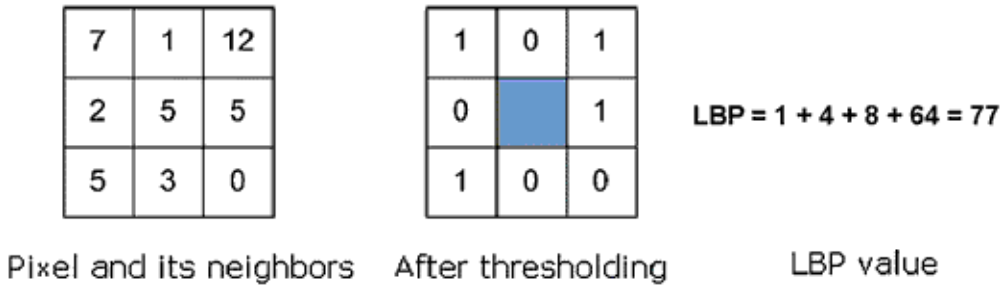
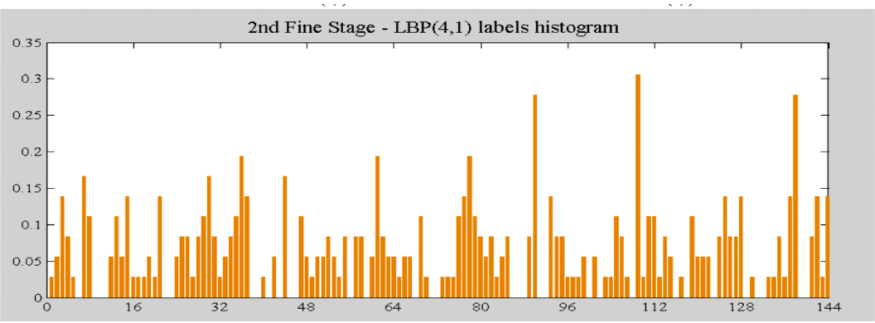



Figure (2.4)

It then converts each block values into a histogram, so now we have gotten one histogram for each block in an image, like this:




(Figure 2.5)

Finally, it then concatenates these block diagrams into one to form a feature vector for one image which contains all the features we are interested in.

## Implementing LBP in OpenCv:

We just need to change a .xml classifier file, in the previous code as

```
#load cascade classifier training file for lbpcascade
lbp_face_cascade = cv2.CascadeClassifier('data/lbpcascade_frontalface.xml')
```

## Deep learning for face detection

Deep learning module was incorporated later in the OpenCV 3.1, Before these the above two methods only could be used for the purpose. This module now supports a number of deep learning frameworks, including Caffe, TensorFlow, and Torch/PyTorch. With OpenCV 3.3, we can utilize pre-trained networks that support various popular deep learning frameworks. The fact that they are pre-trained implies that we don't need to spend much time training the network, rather we can complete a forward pass and utilize the output to make a decision within our application.

Popular architectures that are supported by this module are:

- GoogleLeNet
- AlexNet
- SqueezNet
- VGG
- ResNet

**The Caffee module:** Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by Berkeley AI Research (BAIR) and by community contributors. Yangqing Jia created the project during his PhD at UC Berkeley.

**TensorFlow**: TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

**Pytorch:** PyTorch is an open-source machine learning library for Python, based on Torch, used for applications such as natural language processing. It is primarily developed by Facebook's artificial-intelligence research group, and Uber's "Pyro" software for probabilistic programming is built on it.

The GoogLeNet architecture (now known as “Inception” after the novel micro-architecture) was introduced by Szegedy et al. in their 2014 paper,[7] Going deeper with convolutions. It is important to note at this step that we aren’t training a CNN, rather, we are making use of a pre-trained network. Therefore, we are just passing the blob through the network (i.e., forward propagation) to obtain the result (no back-propagation).

This is a direct implementation of deep learning modules, which have been trained by the processes of feed forward and back propagation as explained earlier in the report. Plus, they are heavily improved by the architectures like **GoogleLeNet.**

## Face Recognition

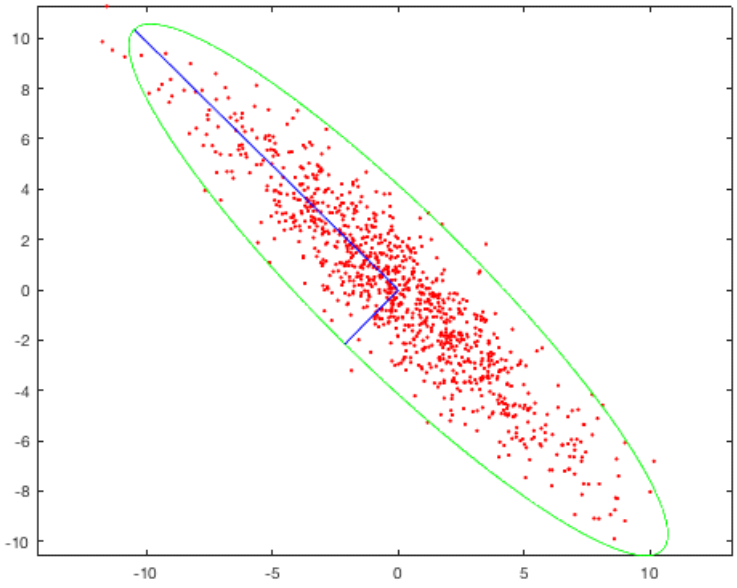
There are many methods for face recognition which have been developed, but since we are using OpenCV face recognition, so we will restrict ourselves to those methods. All the face recognizers in work by first preparing the image or camera feed for face recognition and then the face recogniser is trained to recognize the faces.

There were mainly three types of face recognizers, but we will explain about the fourth one later in the report that we have used. These are namely-

- Eigen Face Recognizer
- Fisher Face Recognizer
- Local Binary patterns Histogram

## Eigen Face recognizer

This method uses a technique called Principal Component analysis (PCA). The idea behind PCA is that we want to select the hyperplane such that when all the points are projected onto it, they are maximally spread out. In other words, we want the axis of maximal variance.




Figure(2.6)

Since the images, can be of high dimension (m x n) and we need to train our network using a lot of images, our method can be painfully slow. To remove this problem, we want **Dimensionality Reduction.** This is what exactly PCA lets us do. In the above image, A potential axis is the x-axis or y-axis, but, in both cases, that's not the best axis. However, if we pick a line that cuts through our data diagonally, that is the axis where the data would be most spread. The longer blue axis is the best axis. Thus, to figure out this axis we use a mathematical term called **Eigen Vectors.**

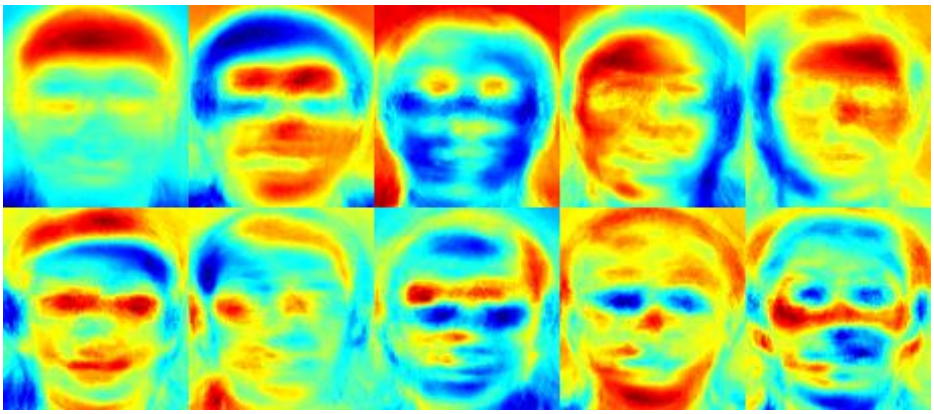
This is how this technique got its name. we compute the covariance matrix of our data and consider that covariance matrix's largest eigenvectors. Those are our principal axes and the axes that we project our data onto to reduce dimensions. Using this approach, we can take high-dimensional data and reduce it down to a lower dimension by selecting the largest eigenvectors of the covariance matrix and projecting onto those eigenvectors. Since we're computing the axes of maximum spread, we're retaining the most important aspects of our data. It's easier for our classifier to separate faces when our data are spread out as opposed to bunched together.

According to OpenCV documentation I quote

"The Eigen face recognizer was trained on the **AT&T Face database"**

The AT&T Face database, sometimes also referred to as ORL Database of Faces, contains ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

“They have used used the jet colormap, so you can see how the grayscale values are distributed within the specific Eigenfaces. You can see, that the Eigenfaces do not only encode facial features, but also the illumination in the images (see the left light in Eigenface #4, right light in Eigenfaces #5):”



(Figure 2.7)

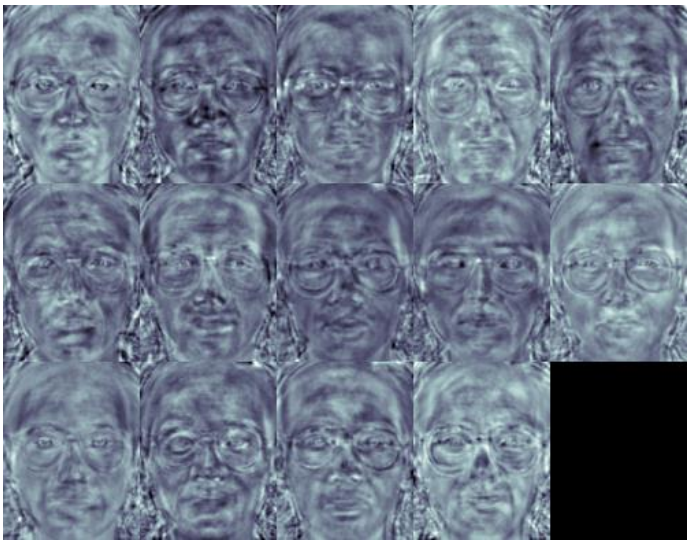
“10 Eigenvectors are obviously not sufficient for a good image reconstruction, 50 Eigenvectors may already be sufficient to encode important facial features. You’ll get a good reconstruction with approximately 300 Eigenvectors for the AT&T Face database. “

(Figure 2.8)

## Fisher Face recognizer

The Eigen face recognizer is a good method, but it loses a lot of discriminative features and throws a lot of components away while representing data. It also cannot discriminate if any external source for example light is creating a variance. Fisher faces looks for linear combinations of pixels that explain the variance between people That means that if something is useful for describing the difference between various instances of the same person, it won't count, whereas if something is usefully to discriminate between different people, it will.

When we use a Eigen Faces, it looks for unique features from all the images of a person, so if there were light illumination issues in one or two images it also considered. So Fisher Faces, Changed this and instead looked for useful features from each face of the same person.



Figure(2.9)

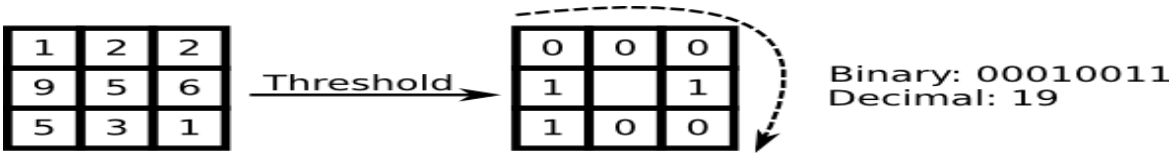
**Eigen faces of the same person**

## Local Binary patterns Histogram

The LBPH approach is well the same we had discussed in Face detection part of this report. It has the same method of creating a matrix of each pixel of the image and then updating pixel values in the near by matrix rows.

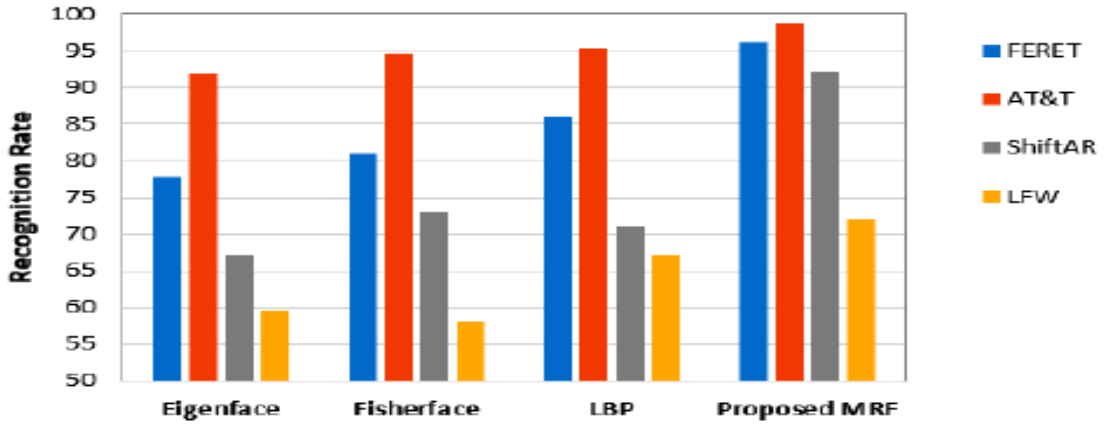
OpenCV documentation quotes it as follows:**"**Eiganand Fisher faces take a somewhat holistic approach to face recognition. **You** treat your data as a vector somewhere in a high-dimensional image space. We all know high-dimensionality is bad, so a lower-dimensional subspace is identified, where (probably) useful information is preserved. The Eigenfaces approach maximizes the total scatter, which can lead to problems if the variance is generated by an external source, because components with a maximum variance over all classes aren't necessarily useful for classification (see http://www.bytefish.de/wiki/pca_lda_with_gnu_octave). So to preserve some discriminative information we applied a Linear Discriminant Analysis and optimized as described in the Fisherfaces method. The Fisherfaces method worked great... at least for the constrained scenario we've assumed in our model.Now real life isn't perfect. You simply can't guarantee perfect light settings in your images or 10 different images of a person. So what if there's only one image for each person? Our covariance estimates for the subspace may be horribly wrong, so will

the recognition. Remember the Eigenfaces method had a 96% recognition rate on the AT&T Facedatabase? How many images do we actually need to get such useful estimates? Here are the Rank-1 recognition rates of the Eigenfaces and Fisherfaces method on the AT&T Facedatabase, which is a fairly easy image database:So some research concentrated on extracting local features from images. The idea is to not look at the whole image as a high-dimensional vector, but describe only local features of an object. The features you extract this way will have a low-dimensionality implicitly. A fine idea! But you'll soon observe the image representation we are given doesn't only suffer from illumination variations. Think of things like scale, translation or rotation in images - your local description has to be at least a bit robust against those things. Just like SIFT, the Local Binary Patterns methodology has its roots in 2D texture analysis. The basic idea of Local Binary Patterns is to summarize the local structure in an image by comparing each pixel with its neighborhood. Take a pixel as center and threshold its neighbors against. If the intensity of the center pixel is greater-equal its neighbor, then denote it with 1 and 0 if not. You'll end up with a binary number for each pixel, just like 11001111. So with 8 surrounding pixels you'll end up with 2^8 possible combinations, called Local Binary Patterns or sometimes referred to as LBP codes. The first LBP operator described in literature actually used a fixed 3 x 3 neighborhood just like this:



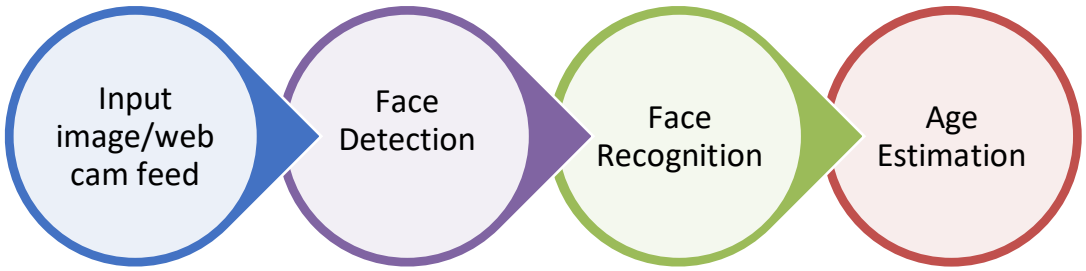

Figure(2.9)

The recognition rates of all the methods described so far were extensively studied by [8] Sultana, Madeena & Gavrilova, Marina & Yanushkevich, Svetlana in their research paper and based on that the following graph was created. It includes different datasets also namely AT&T, LFW, FERET, SHIFTAR.




Figure(2.10)

# Chapter 3 Proposed model




Figure(3.1)

Now, since the parts of the project are presented by the above steps, we would like to explain a variety of things on these parts. But before just moving on to these parts directly we will explain the background behind all of them –

- **Face detection**- The analogy behind it, different methods of implementing it and the technologies used.
- **Face recognition**- The analogy behind it, different methods of implementing it and the technologies used.
- **Age recognition**-This we have only studied in brief so we will just explain how and which method we have used.

## Artificial Intelligence, Machine learning and Neural networks

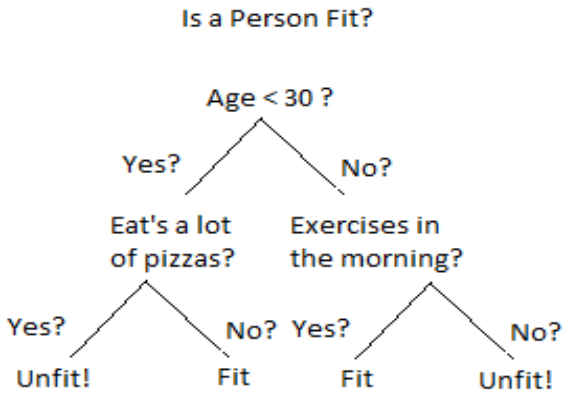
AI is a term that was coined by scientists in Dartmouth conference in 1956.Over the past years the term has become more popular and has been accepted as a key to a bright future for our civilization. The main aim of the researchers in the initial years was to construct complex machines that could think like humans. It should have had all the senses that we humans have and maybe even more, Like the ones we have seen in sci-fi movies “The Terminator”. The term AI is a very broad term in itself, which includes several other terms that we are going to explain further.

The complex machines that the scientists and researchers wanted to build at that time, that could think like humans had to have some intelligence like humans also. Naturally this intelligence can’t

be put into them magically. We have to create some approaches that could enable the machines to think. **This "intelligence" is what is called "Machine Learning".Machine learning** is the process in which we study the data, learn from it and then make some future predictions. The machine learns the ability to do a task by learning from its experiences again and again until it gets the idea. The most basic example is a new born baby, who doesn't know anything. But the child starts to learn from his/her experiences what is good and what is bad. If the child accidently puts his finger on fire, he feels pain and then immediately removes his finger from fire. Now the next time he knows not to do something like that. Machine learning uses a lot of algorithms that helps it to learn from data and predict the outcomes. We have listed some algorithms below-

- Decision Tree
- Inductive logic programming
- Clustering
- Reinforcement learning
- Bayesian networks

Machine learning can be explained using a simple example which uses Decision Tree to predict the outcome. Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.An example of a decision tree can be explained using below binary tree. Let's say you want to predict whether a person is fit given their information like age, eating habit, and physical activity, etc. The decision nodes here are questions like 'What's the age?', 'Does he exercise?', 'Does he eat a lot of pizzas'? And the leaves, which are outcomes like either 'fit', or 'unfit'. In this case this was a binary classification problem (a yes, no type problem)



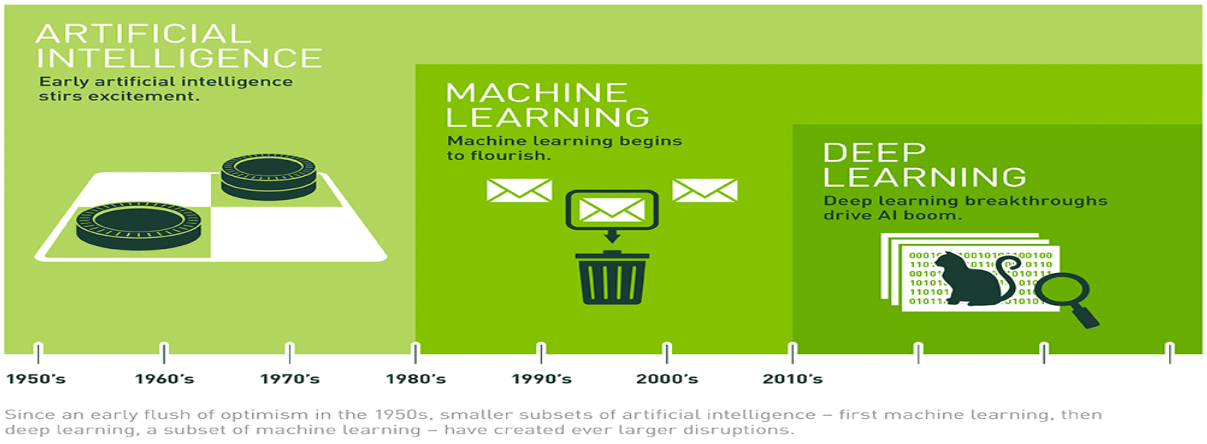

Figure(3.2).

So now using the decision tree we can tell if a person is fit or not. This gives us a basic idea of machine learning and how it can be used.

Well, Machine learning was a good technique but ultimately it did not solve the basic purpose of Artificial Intelligence. Since then a new approach was being developed by the AI researchers, this new technique was based on the biological brain. Our brain contains neurons, and each neuron is connected to some other neuron. The name of this approach **was Neural Networks.**

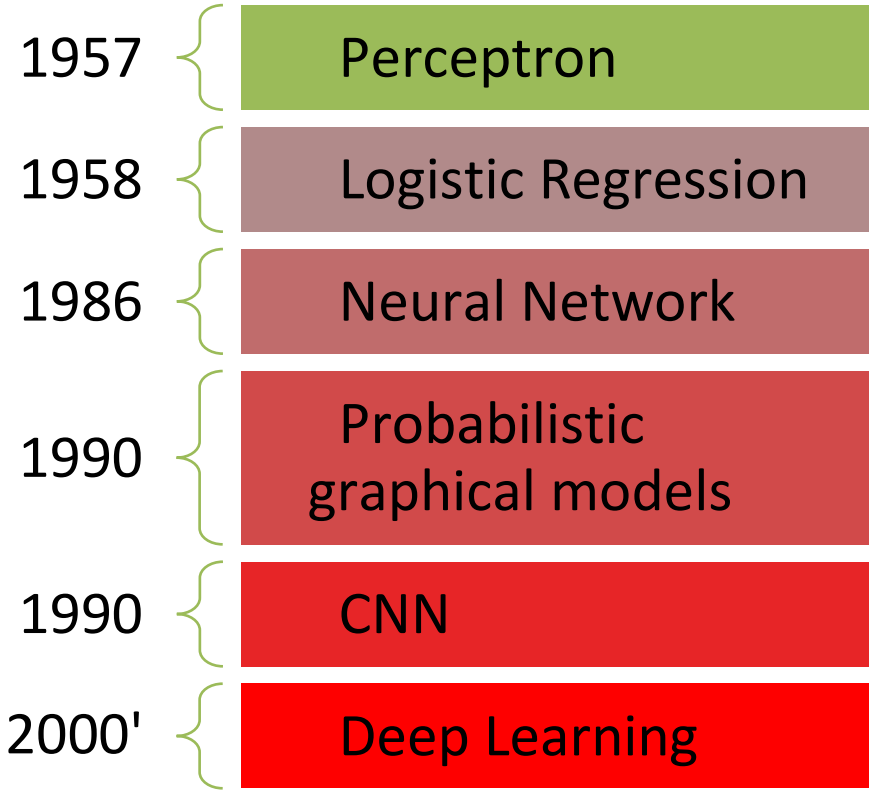
These **Artificial Neural Networks** have discrete layers, connections and directions of data propagation. Each neuron assigns a weight to its input to determine the correctness relative to the output needed.  for example, take an image, chop it up into a bunch of tiles that are inputted into the first layer of the neural network. In the first layer individual neurons, then passes the data to a second layer. The second layer of neurons does its task, and so on, until the final layer and the final output is produced. Neural networks can have many layers but, an efficient way of training a neural net is called **Deep learning.**

**The diagram below will explain the relation b/w all the ideologies.**




Figure(3.3)

**A brief timeline of Machine learning**

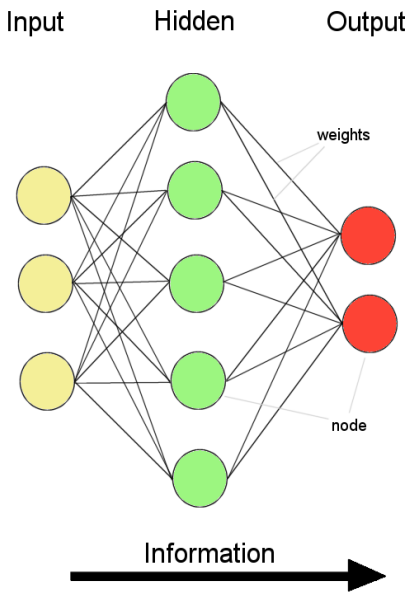



Figure(3.4**)**

Our project is based on **Neural networks, CNN and Deep learning** only so we will restrict our explanation up to that only, but for the sake of understanding we have given the above timeline with all the algorithms.

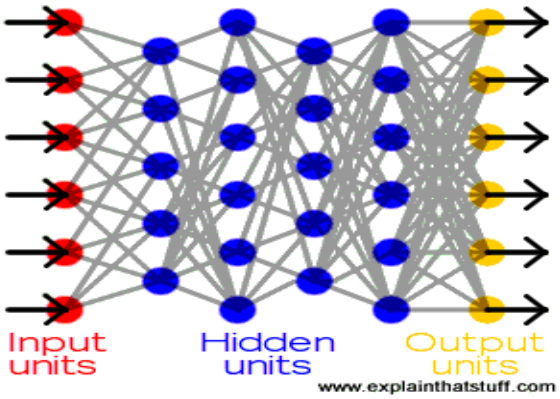
**Neural Networks**The Artificial Neural Nets are based on the biological neurons as mentioned earlier, so there two basic components of a Neural net

- Neurons(nodes)
- Synapses(weights)




(Figure 3.4)

A typical neural network consists of a lot of neurons (units). There are input units as well as output units, the former are designed to receive various types of information from the outside world and then it attempts to learn from that information. The output units respond to the information it has learned. There are one more type of units in between these units, called the hidden units which make up the majority of the neural nets. The connection between these units are called weights. The neural nets are fully connected.




Figure(3.5)

## Learning in Neural Networks

The information in the network flows in two ways:

- **Learning phase**: Patterns of information is fed into the network via the input units, which activates the hidden layers and then the result arrives at the output layer. This design is called The **Feedforward network.** Each unit receives inputs from the units to its left, and the inputs are multiplied by the weights of the connections they travel along. Every unit adds up all the inputs it receives in this way and (in the simplest type of network) if the sum is more than a certain threshold value, the unit "fires" and triggers the units it's connected to (those on its right).
- **Back Propagation:** A feedback process is very important in a network similarly as we humans take feedback about our progress all the time. Hence a neural network also tends to do the same. Once the result is reached at the output layer, it is compared with the result it was supposed to produce. Then the difference between the two is used to adjust the weights of the connections between the units in the network that is going backwards. Hence it is called **Back Propagation.**

Once the network has been trained with enough learning examples, it reaches a point where you can present it with an entirely new set of inputs it's never seen before and see how it responds. For example, suppose you've been teaching a network by showing it lots of pictures of chairs and tables, represented in some appropriate way it can understand, and telling it whether each one is a chair or a table. After showing it, let's say, 25 different chairs and 25 different tables, you feed it a picture of some new design it's not encountered before, let's say a chaise longue, and see what happens. Depending on how you've trained it, it'll attempt to categorize the new example as either a chair or a table, generalizing on the basis of its past experience, just like a human. Hey presto, you've taught a computer how to recognize furniture.

## Speech recognition using neural network learning:

Learning in the neural nets using feedforward and backpropagation can be explained using the example of speech recognition. Suppose there are two persons named "Steve" and "David". They both say the word "Hello", there are two frequency bins for each-
David= 1-0
Steve=0-1

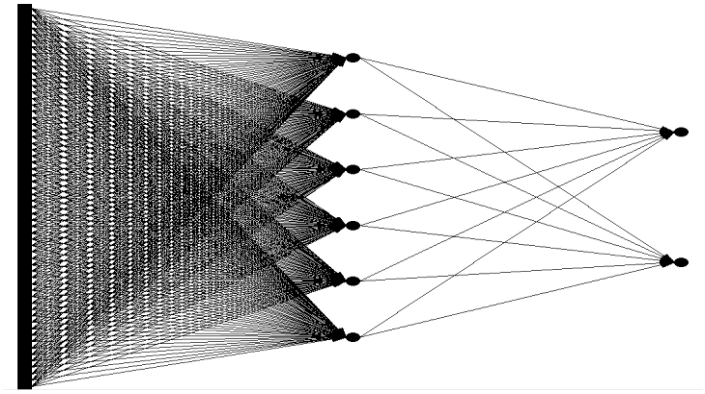
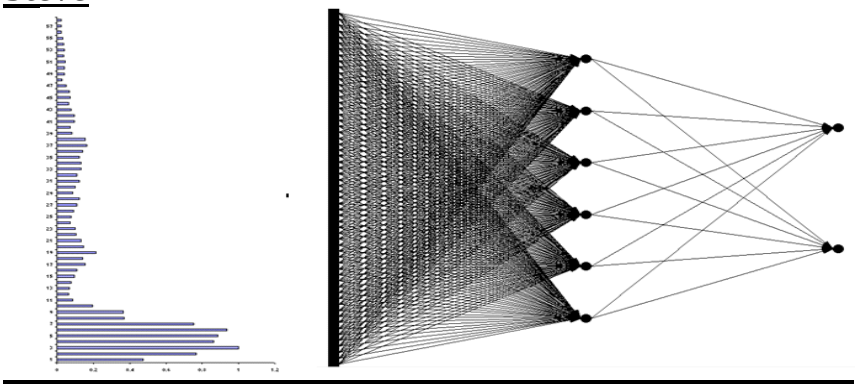


Figure (3.6)

There are 6 hidden layers in the network and 2 output layers. Now in the first phase, the untrained network is given the inputs for which it produces the output as shown. The initial values, output by the network obviously has errors.

**Steve**




Figure(3.7)

**David**

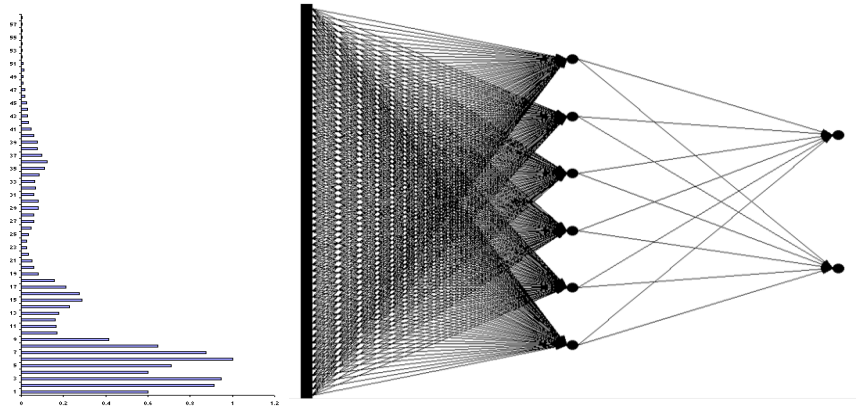
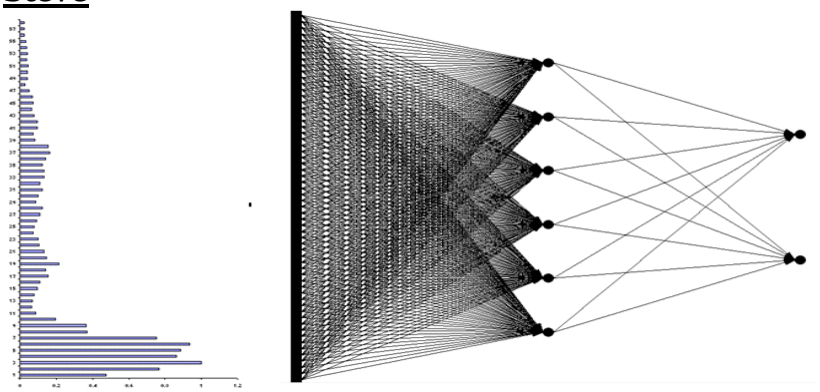



Figure (3.8)

Now in the next phase we will calculate the error in the output and then as we discussed earlier the neural network will make some changes to the weights so that correct output can be produced.

## Calculating error

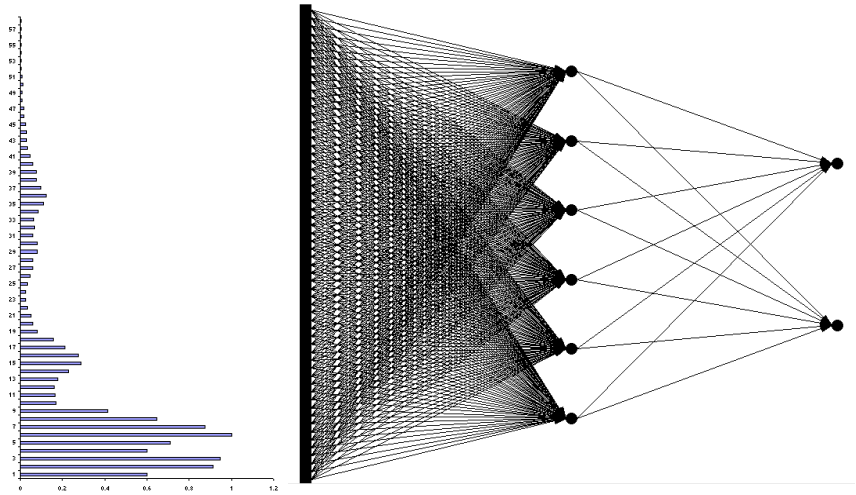
### Steve



0.43-0=0.43

0.26-1=0.74

### David



0.73-1=0.27

0.55-0=0.55

The total error in Steve's network is 0.43+0.74=1.17

The total error in David's network is 0.27+0.55=0.82

After this calculation the network will adjust its weights until its done with correct output.

# Chapter4 Implementation

As Neural network is a class of machine learning algorithms, there are different variations of neural networks. The class of Neural networks contains various architectures like **Convolutional neural networks (CNN), Recurrent neural networks (RNN)** and **Deep belief networks**. The number of (layers of) units, their types, and the way they are connected to each other is called the **network architecture.**

Now, for our project we have used **CNN** Architecture. we will explain this in detail. A CNN consists of a **convolutional layer, a pooling layer** and **fully connected layer.**

- **The convolutional layer**: It is the first layer to extract features from the input image, it creates a relationship between pixels by learning image features suing small squares of input data. It is a mathematical operation.

  Suppose there is a 3 x 3 matrix with image pixel values 0,1 and a filter matrix as shown below

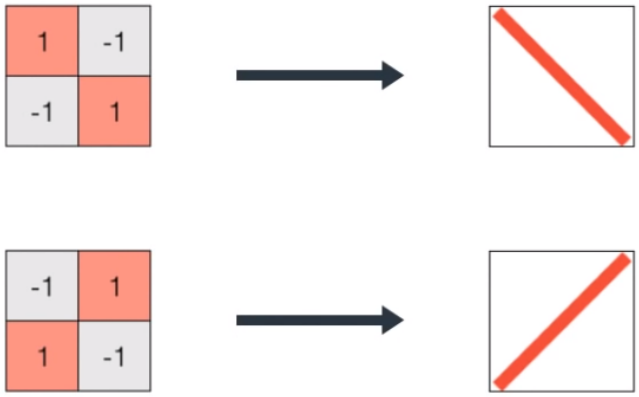
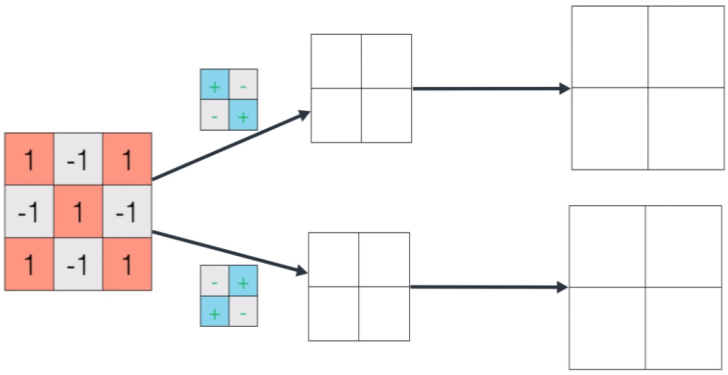


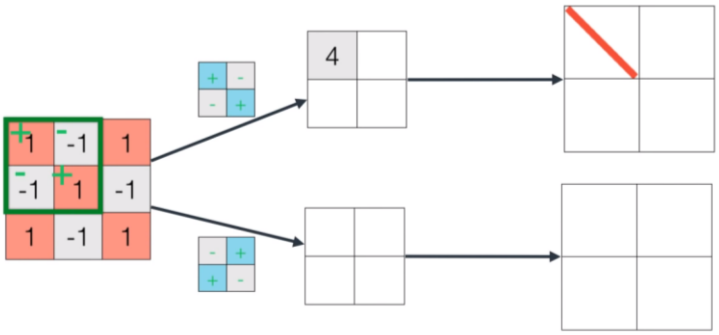

Figure (4.1)

This is the filter matrix that we know creates a forward slash and backward slash, this is the previous knowledge we have for a 2x2 matrix. Now this matrix will be superimposed on the 3x3 matrix from left to right and top to bottom to get smaller outputs from the larger input that is 3x3.




(Figure 4.2)

## Filter one



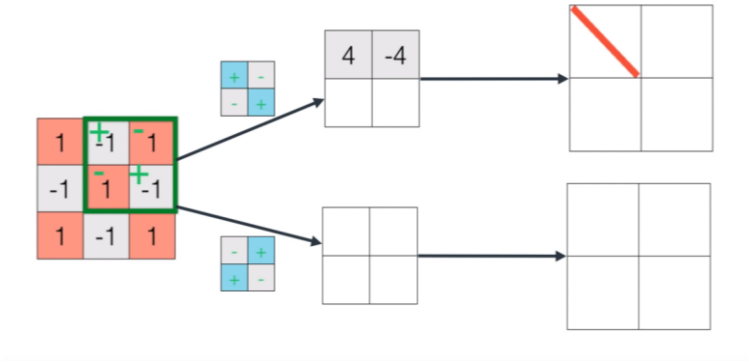
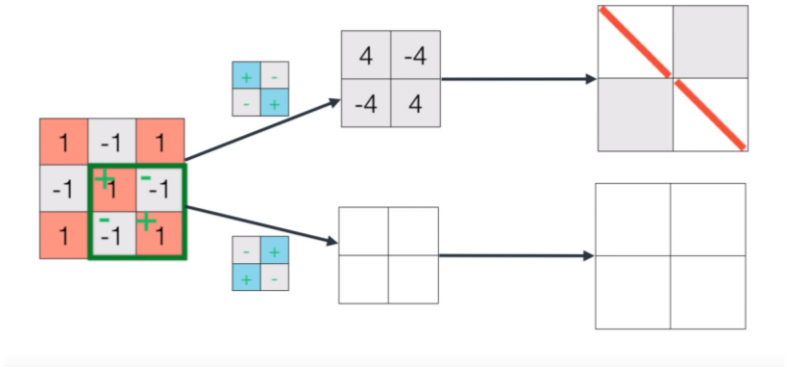
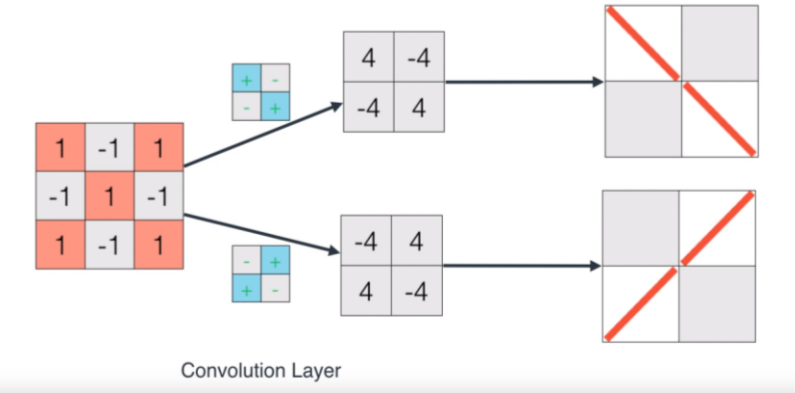




Figure (4.3)




(Figure 4.4)

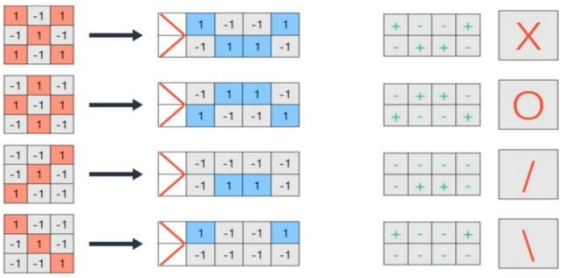
## Filter 2




Figure(4.5)

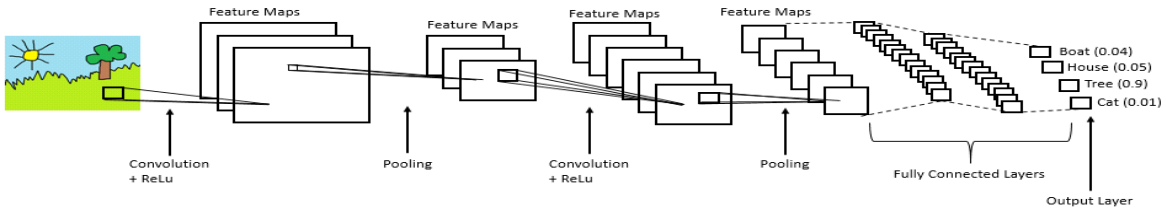
- **Pooling layer:** Pooling layers section would reduce the number of parameters when the images are too large. In the above image the areas that are grey do not satisfy the criterion, so it not useful. This is the work of the pooling layer.

- **Fully Connected Layer:** This layer finds logic and tries to figure out which image that might be.



Figure(4.6)

The process can be visualised by the below image. These are the series of steps that are taken up in a Convolutional Neural Network Architecture for image recognition .




Figure(4.7)

## Deep Learning Face recognition

The most accurate implementation of face recognition is using deep learning. Well with that being said this could only be possible with OpenCV 3 and above. Otherwise implementing this algorithm would have been a lot tougher.

Deep learning combined with face recognition is called **deep metric learning**. This means instead of output as a single label of an image, this algorithm outputs a 128 -d feature vector, i.e a list of 128 real valued numbers that are used to quantify a face.

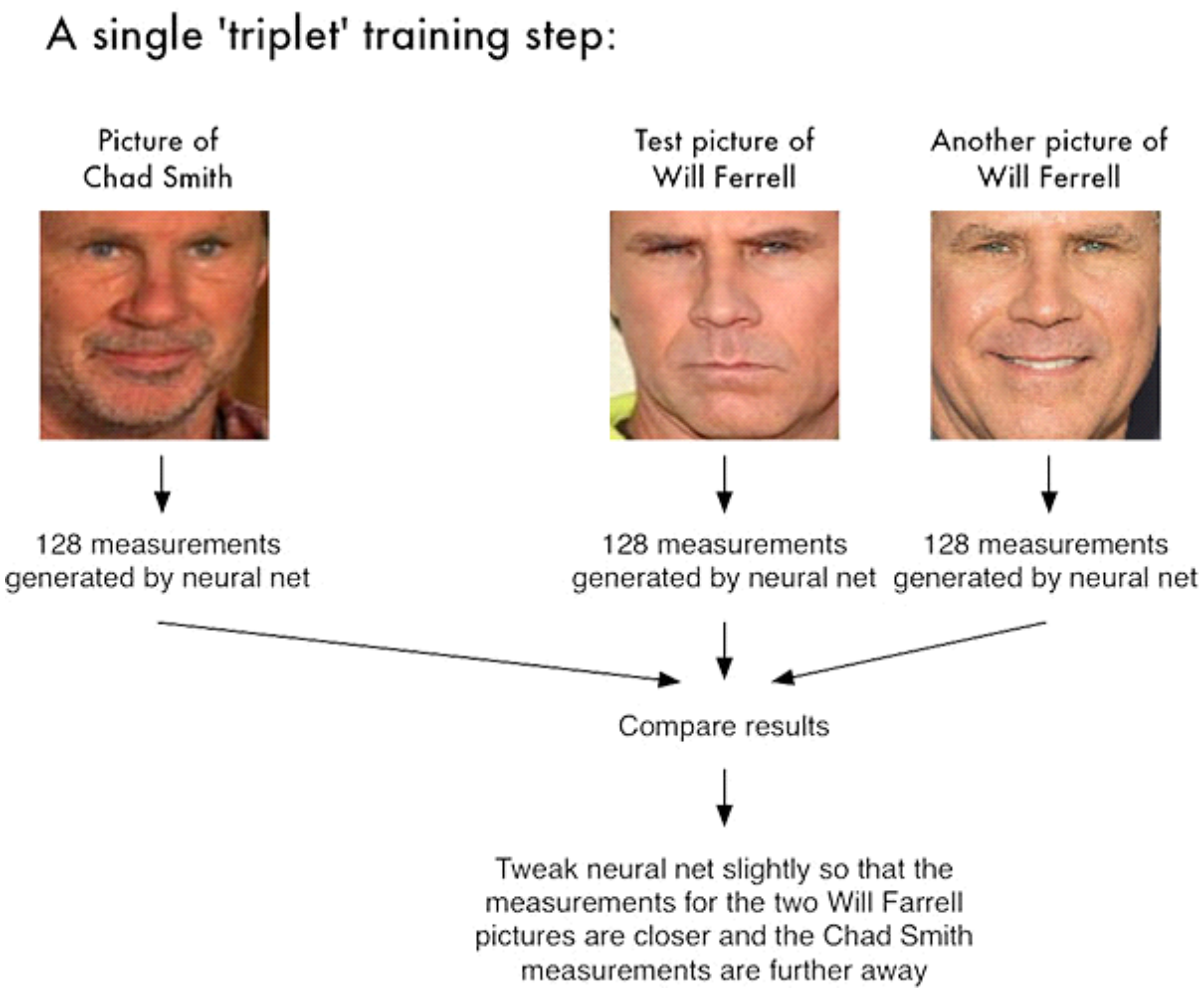



***Image credit:*** **Adam Geitgey's *"Machine Learning is Fun"* blog**

Figure(4.7)

The network quantifies the faces, constructing the 128-d embedding for each. From there, the general idea is that it'll tweak the weights of the neural network so that the 128-d measurements of the two Will Ferrel will be closer to each other and farther from the measurements for Chad Smith. This network architecture for face recognition is based on [5]ResNet-34 from the Deep Residual Learning for Image Recognition paper by Kaiming He., but with fewer layers and the number of filters reduced by half. This network was trained on the dataset LFW (Labelled faces in the wild), by Davis King (creator of Dlib library) and He claims the accuracy to be 99.38%, This accuracy is supported by the dlib documentation.

## Deep learning and Convolutional neural networks

The confusion arises to a lot of people about what is what. So we would like to explain a little about both. Everything that is done in CNN is also done in Deep learning that is described in the below picture-

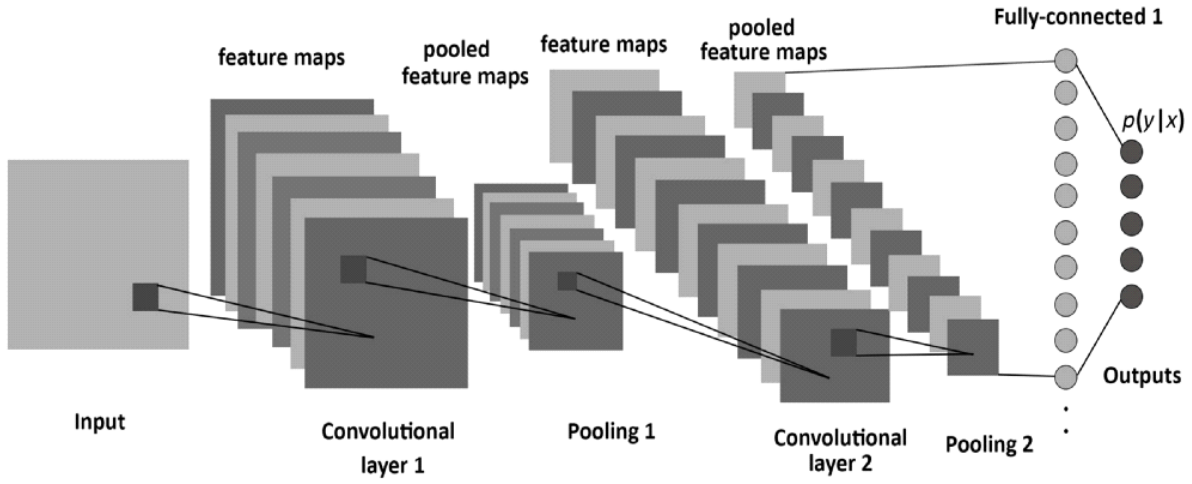
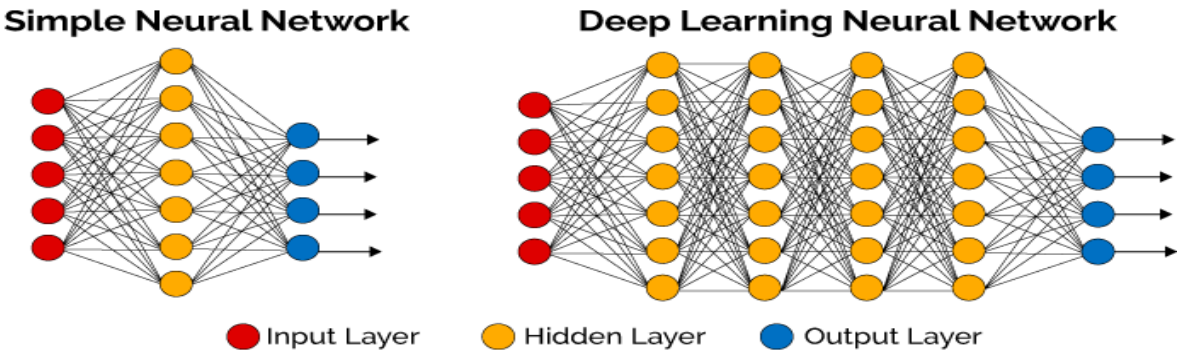



Figure (4.8)

The Deep learning uses CNN , and the clear distinction between both is any CNN that has more than 1 hidden layer is called a Deep learning network. This also has an effect on the training time. Only computers with a GPU can use Deep learning networks to train. The accuracy between both is also different, mainly deep neural networks have a lot of layers and hence more the layers, more is the accuracy.




(Figure 4.9)

## Age Recognition using Deep learning

There have been various attempts on Age recognition, like the early ones include one facial feature (eyes, nose, mouth, chin), are localized and their sizes and distances are measured, ratios between them are calculated and then used for classifying images. Another model similar to this was age progression of subjects under 18 years. All of these methods require an accurate localization of features which is a tough job in itself.

As it is, we know that Deep learning is possible with today's technology and it gives better results than any other methodology, so we used Deep learning for this model taken from [1]. The proposed model was used by the researchers throughout their experiments and is shown below.

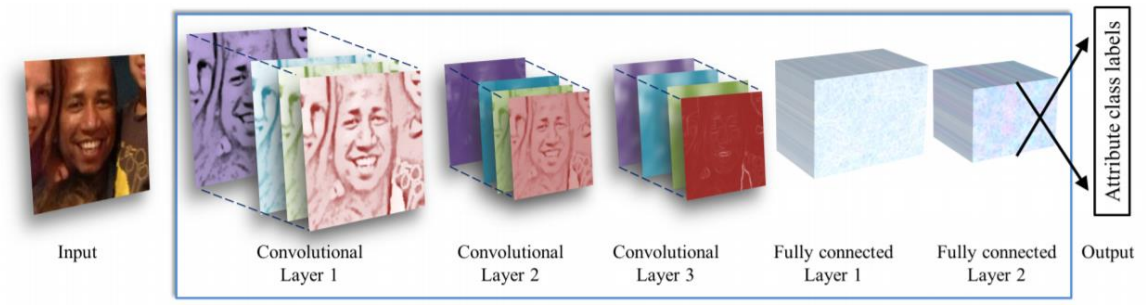



Figure 2. **Illustration of our CNN architecture.** The network contains three convolutional layers, each followed by a rectified linear operation and pooling layer. The first two layers also follow normalization using local response normalization [28]. The first Convolutional Layer contains 96 filters of $7\times7$ pixels, the second Convolutional Layer contains 256 filters of $5\times5$ pixels, The third and final Convolutional Layer contains 384 filters of $3 \times 3$ pixels. Finally, two fully-connected layers are added, each containing 512 neurons. See Figure 3 for a detailed schematic view and the text for more information.

(Figure 4.10)

The network comprises of only three convolutional layers and two fully-connected layers with a small number of neurons, all three color channels are processed directly by the network. Images are first rescaled to 256 × 256 and a crop of 227 × 227 is fed to the network. The three subsequent convolutional layers are then defined as follows.

- 96 filters of size 3 x 7 x 7 pixels are applied to the input in the first convolutional layer, followed by a rectified linear operator (ReLU), a max pooling layer is taking the maximal value of 3 x 3 regions with two-pixel strides and a local response normalization layer.
- The 96 x 28 x 28 output of the previous layer is then processed by the second convolutional layer, containing 256 filters of size 96 x 5 x 5 pixels. Again, this is followed by ReLU, a max pooling layer and a local response normalization layer with same hyper parameters as before.
- Finally, the third and the last convolution layer operates on the 256 x 14 x 14 blob by applying a set of 384 filters of size 256 x 3 x 3 pixels, followed by ReLU and a max pooling layer.
- A first fully connected layer that receives the output of the third convolutional layer and contains 512 neurons, followed by a ReLU and a dropout layer.
- A second fully connected layer that receives the 512- dimensional output of the first fully connected layer and again contains 512 neurons, followed by a ReLU and a dropout layer.
- A third, fully connected layer which maps to the final classes for age.
- Finally, the output of the last fully connected layer is fed to a soft-max layer that assigns a probability for each class. The prediction itself is made by taking the class with the maximal probability for the given test image.

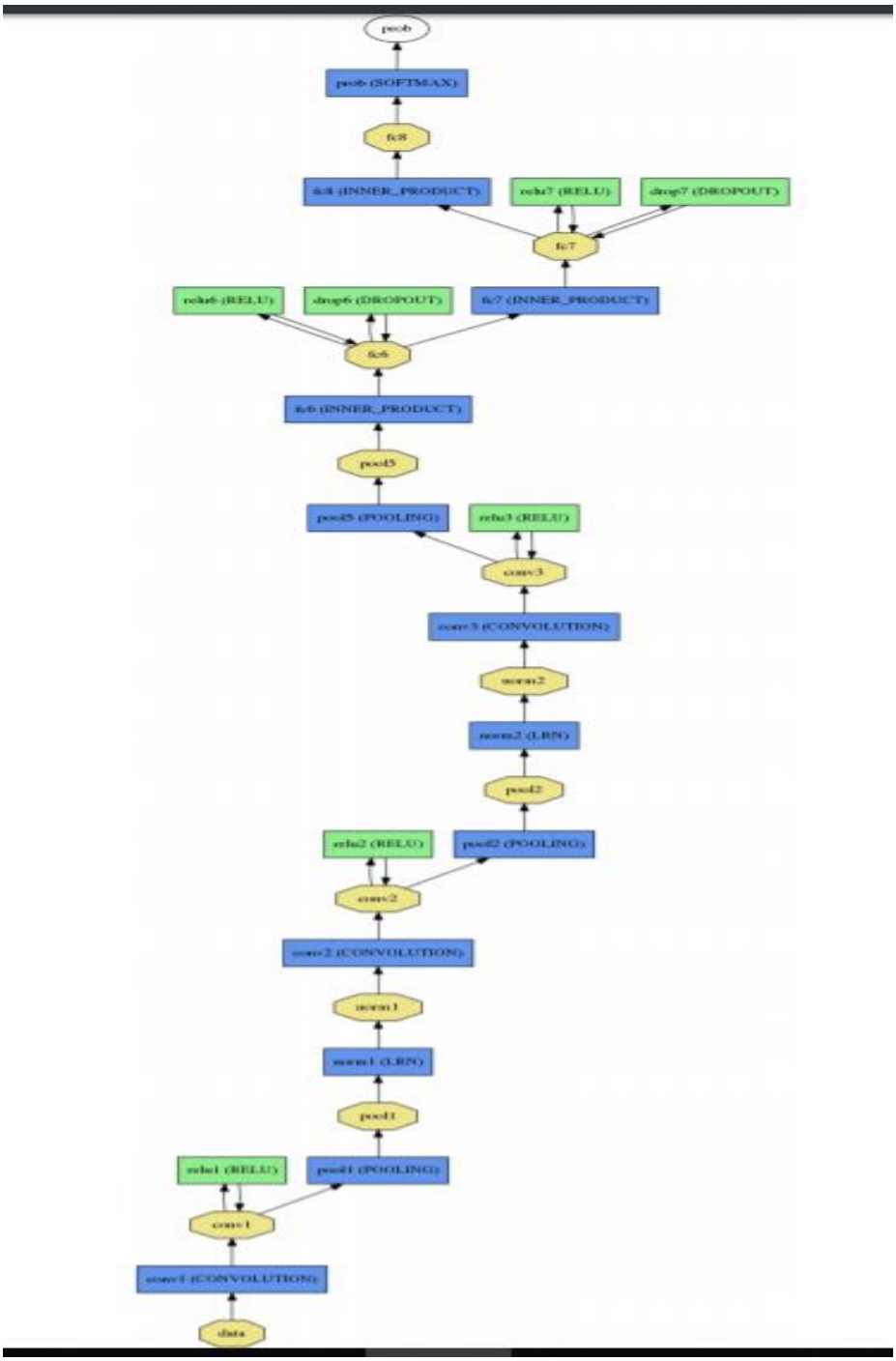



Figure (4.11)

## Training the network

Initialization. The weights in all layers are initialized with random values from a zero mean Gaussian with standard deviation of 0.01. To stress this, they do not use pre-trained models for initializing the network; the network is trained, from scratch, without using any data outside of the images and the labels available by the benchmark. This, again, should be compared with CNN

implementations used for face recognition, where hundreds of thousands of images are used for training. Target values for training are represented as sparse, binary vectors corresponding to the ground truth classes. For each training image, the target, label vector is in the length of the number of classes (two for gender, eight for the eight age classes of the age classification task), containing 1 in the index of the ground truth and 0 elsewhere.

## ACCURACY

They tested the accuracy of their CNN design using the recently released Adience benchmark, designed for age classification. The Adience set consists of images automatically uploaded to Flickr from smart-phone devices. Because these images were uploaded without prior manual filtering, as is typically the case on media webpages (e.g., images from the LFW collection) or social websites (the Group Photos set), viewing conditions in these images are highly unconstrained, reflecting many of the real-world challenges of faces appearing in Internet images. Adience images therefore capture extreme variations in head pose, lightning conditions quality, and more. The entire Adience collection includes roughly 26K images of 2,284 subjects. Table 1 lists the breakdown of the collection into the different age categories. Testing for both age or gender classification is performed using a standard five-fold, subject-exclusive cross-validation protocol, defined in. They used the in-plane aligned version of the faces, originally used in . These images are used rather than newer alignment techniques in order to highlight the performance gain attributed to the network architecture, rather than better pre-processing. They emphasize that the same network architecture is used for all test folds of the benchmark and in fact, for both gender and age classification tasks. This is performed in order to ensure the validity of their results across folds, but also to demonstrate the generality of the network design proposed here; the same architecture performs well across different, related problems. We compare previously reported results to the results computed by our network. Their results include both methods for testing: centre-crop and over-sampling .

# Chapter 5 Results and discussions

## Results.1

| | 0-2 | 4-6 | 8-13 | 15-20 | 25-32 | 38-43 | 48-53 | 60- | Total |
|---|---|---|---|---|---|---|---|---|---|
| Male | 745 | 928 | 934 | 734 | 2308 | 1294 | 392 | 442 | 8192 |
| Female | 682 | 1234 | 1360 | 919 | 2589 | 1056 | 433 | 427 | 9411 |
| Both | 1427 | 2162 | 2294 | 1653 | 4897 | 2350 | 825 | 869 | 19487 |

Table 1. **The AdienceFaces benchmark.** Breakdown of the AdienceFaces benchmark into the different Age and Gender classes.

| Method | Exact | 1-off |
|---|---|---|
| Best from [10] | 45.1 ± 2.6 | 79.5 ±1.4 |
| Proposed using single crop | 49.5 ± 4.4 | 84.6 ± 1.7 |
| Proposed using over-sample | **50.7 ± 5.1** | **84.7 ± 2.2** |

Table 3. **Age estimation results on the Adience benchmark.** Listed are the mean accuracy ± standard error over all age categories. Best results are marked in bold.

## The above are the results of the research paper and below are the results of our dataset

**Test Results for Age Group (25-32)**

(Real Age: 30 , Test Result : 27.2)

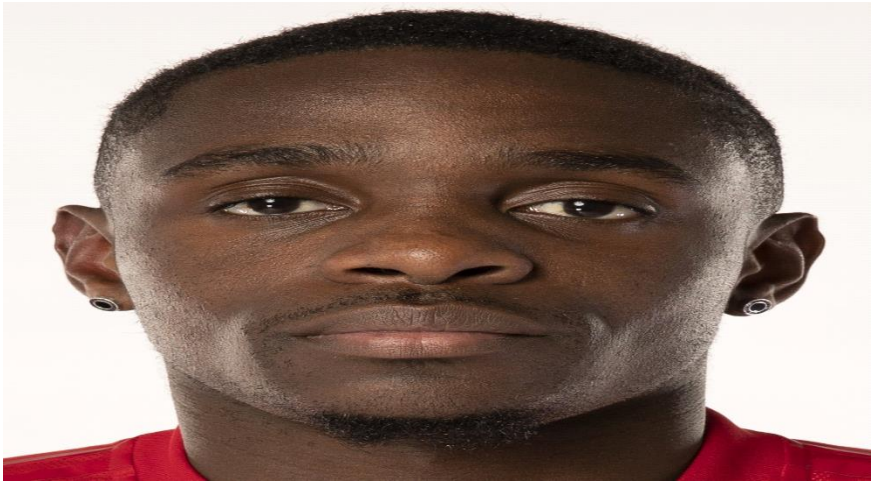
(Real Age :22 , Test Result :24)

(Real Age : 22 , Test Result :21)

(Real Age : 31 .Test Result : 29.6)

(Real Age : 23 . Test Result :20)

(Real Age :21 , Test Result :27)



**Test Results for Age Group (38-43):**

(Real Age : 38 , Test Result :33)


(Real Age :41 , Test Result :41.4)

(Real Age: 39, Test Result :31)

# Chapter 6 Conclusions and Future works

A possible future application for facial recognition systems lies in retailing. A retail store (for example, a grocery store) may have cash registers equipped with cameras; the cameras would be aimed at the faces of customers, so pictures of customers could be obtained. The camera would be the primary means of identifying the customer, and if visual identification failed, the customer could complete the purchase by using a PIN (personal identification number). After the cash register had calculated the total sale, the face recognition system would verify the identity of the customer and the total amount of the sale would be deducted from the customer's bank account. Hence, face-based retailing would provide convenience for retail customers, since they could go shopping simply by showing their faces, and there would be no need to bring debit cards, or other financial media. Wide-reaching applications of face-based retailing are possible, including retail stores, restaurants, movie theaters, car rental companies,hotels, etc.e.g. Swiss European surveillance: facial recognition and vehicle make, model, color and license plate reader.

Some other possible applications that can be developed are :

**1.** In order to prevent the frauds of ATM , it is recommended to prepare the database of all ATM customers with the banks in India & deployment of high resolution camera and face recognition software at all ATMs. So,whenever user will enter in ATM his photograph will be taken to permit the access after it is being matched with stored photo from the database.

**2.** Duplicate voter are being reported in India. To prevent this, a database of all voters, of course, of all constituencies, is recommended to be prepared. Then at the time of voting the resolution camera and face recognition equipped of voting site will accept a subject face 100% and generates the recognition for voting if match is found.

**3.** Passport and visa verification can also be done using face recognition technology as explained above.

**4.** Driving license verification can also be exercised face recognition technology as mentioned earlier.

**5.** To identify and verify terrorists at airports, railway stations and malls the face recognition technology will be the best choice in India as compared with other biometric technologies since other technologies cannot be helpful in crowdy places.

# Chapter 7 References

[1] Gil Levi and Tal Hassner(2015).Age and Gender Classification Using Convolutional Neural [2] Antitza Dantcheva, Petros Elia, Arun Ross. What else does your biometric data reveal? A survey on soft biometrics. IEEE Transactions on Information Forensics and Security, Institute of Electrical and Electronics Engineers, 2015, 11

[3] Zafeiriou, Stefanos, Cha ZNetworks.

hang, and Zhengyou Zhang. "A survey on face detection in the wild: past, present and future." Computer Vision and Image Understanding, vol. 138 pp. 1-24, 2015.

[4] Rothe, Rasmus, Radu Timofte, and Luc Van Gool. "DEX: Deep EXpectation of apparent age from a single image." IEEE International Conference on Computer Vision Workshops, pp. 10-15, 2015

[5] Deep Residual Learning for Image Recognition Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun submitted *Submitted on 10 Dec 2015*

[6] Viola and Jones, "Rapid object detection using a boosted cascade of simple features", Computer Vision and Pattern Recognition, 2001

[7] Szegedy et al, Going deeper with convolutions , 2014

[8] Sultana, Madeena & Gavrilova, Marina & Yanushkevich, Svetlana. (2014). Multi-resolution fusion of DTCWT and DCT for shift invariant face recognition. Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics. 2014. 10.1109/SMC.2014.6973888.