

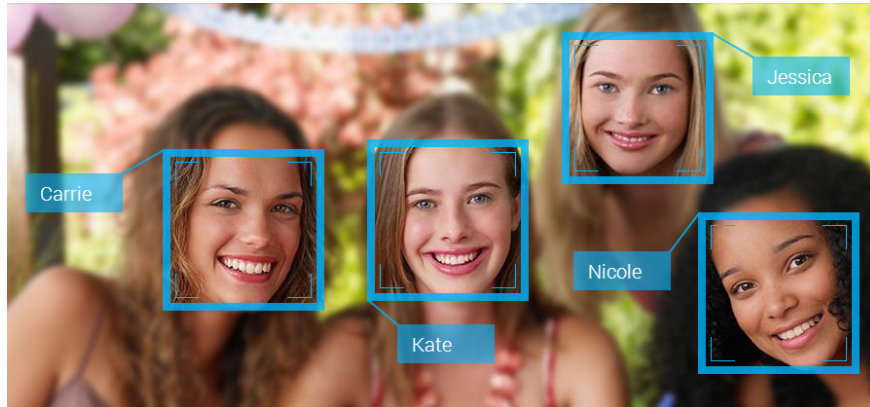
# Face Recognition for Beginners



Divyansh Dwivedi [Follow](#)

Apr 28, 2018 · 9 min read

guided by [Sukant Khurana](#)



Face Recognition of multiple faces in an image

**F**ace Recognition is a recognition technique used to detect faces of individuals whose images are saved in the data set. Despite the point that other methods of identification can be more accurate, face recognition has always remained a significant focus of research because of its non-meddling nature and because it is people's facile method of personal identification.

## Face Recognition Methods:-

There are different methods for face recognition, which are as follows-

### 1.Geometric Based / Template Based:-

Face recognition algorithms are classified as geometry based or template based algorithms. The template-based methods can be constructed using statistical tools like SVM [Support Vector Machines], PCA [Principal Component Analysis], LDA [Linear Discriminant Analysis], Kernel methods or Trace Transforms. The geometric feature based methods analyse local facial features and their geometric relationship. It is also known as a feature-based method.

### 2.Piecemeal / Wholistic:-

The relation between the elements or the connection of a function with the whole face not undergone into the amount, many researchers followed this approach, trying to deduce the most relevant characteristics. Some methods attempted to use the eyes, a combination of features and so on. Some Hidden Markov Model methods also fall into this category, and feature processing is very famous in face recognition.

### **3.Appearance-Based / Model-Based:-**

The appearance-based method shows a face regarding several images. An image considered as a high dimensional vector. This technique is usually used to derive a feature space from the image division. The sample image compared to the training set. On the other hand, the model-based approach tries to model a face. The new sample implemented to the model and the parameters of the model used to recognise the image.

The appearance-based method can classify as linear or nonlinear. Ex- PCA, LDA, IDA used in direct approach whereas Kernel PCA used in nonlinear approach. On the other hand, in the model-based method can be classified as 2D or 3D Ex- Elastic Bunch Graph Matching used.

### **4.Template / Statistical / Neural Networks Based:-**

#### **4.1.Template Matching:-**

In template matching the patterns are represented by samples, models, pixels, textures, etc. The recognition function is usually a correlation or distance measure.

#### **4.2.Statistical Approach:-**

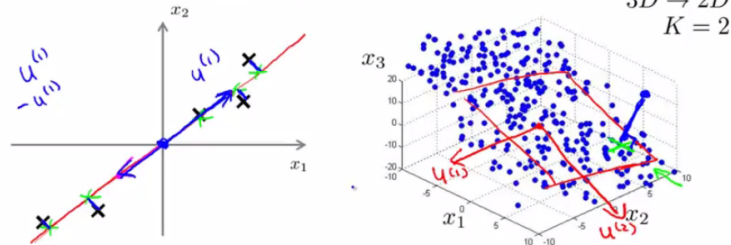
In the Statistical approach, the patterns expressed as features. The recognition function in a discriminant function. Each image represented regarding  $d$  features. Therefore, the goal is to choose and apply the right statistical tool for extraction and analysis.

There are many statistical tools, which used for face recognition. These analytical tools used in a two or more groups or classification methods. These tools are as follows-

##### **4.2.1.Principal Component Analysis [PCA]:-**

One of the most used and cited statistical method is the Principal Component Analysis. A mathematical procedure performs a dimensionality reduction by extracting the principal component of multi-dimensional data.

#### Principal Component Analysis (PCA) problem formulation



Reduce from 2-dimension to 1-dimension: Find a direction (a vector  $u^{(1)} \in \mathbb{R}^n$ ) onto which to project the data so as to minimize the projection error.  
 Reduce from n-dimension to k-dimension: Find  $k$  vectors  $u^{(1)}, u^{(2)}, \dots, u^{(k)}$  onto which to project the data, so as to minimize the projection error.

Dimension reduction from 3D to 2D image

#### 4.2.2. Discrete Cosine Transform [DCT]:-

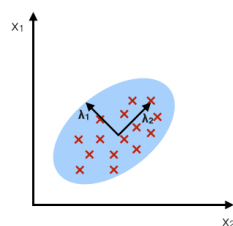
It signifies a series of data points regarding a sum of cosine functions different oscillating frequencies. The Discrete Cosine Transform is based on Fourier discrete transform and therefore, by compacting the variations it can be used to transform images and allowing an efficient dimensionality reduction.

#### 4.2.3. Linear Discriminant Analysis [LDA]:-

LDA is widely used to find the linear combination of features while preserving class separability. Unlike PCA, the LDA tries to model to the difference between levels. For each level the LDA obtains differenced in multiple projection vectors.

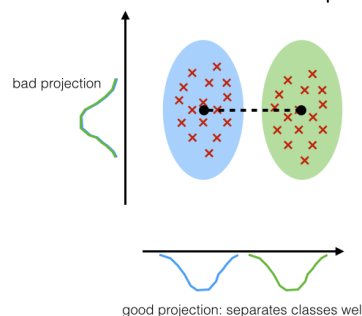
##### PCA:

component axes that maximize the variance



##### LDA:

maximizing the component axes for class-separation



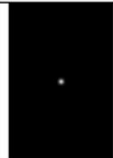



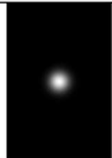















PCA vs LDA

#### 4.2.4. Locality Preserving Projections [LPP]:-

HE and NIYOGI introduced The LPP. It is the best alternative of PCA for preserve locality structure and designing. Pattern recognition algorithms usually search for the nearest pattern or neighbours. Therefore, the locality maintaining the quality of LLP can quicken the recognition.

#### 4.2.5. Gabor Wavelet:-

NI In this algorithm, it signifies that Neuro-physiological data evidence from the visual cortex of mammalian brains suggests that simple cells in the visual cortex can view as a family of self-similar 2D Gabor wavelets. The Gabor functions proposed by Daugman are local spatial bandpass filters that achieve the theoretical limit for conjoint resolution of information in the 2D spatial and 2D Fourier domains.

Source	Magnitude ( $m = 0, n=0$ )	Imaginary	Magnitude ( $m = 2, n=3$ )	Imaginary	Magnitude ( $m = 4, n=7$ )	Imaginary
Original Gabor Wavelet						
						
						

Gabor Wavelet in different magnitude

#### 4.2.6. Independent Component Analysis [ICA]:-

ICA aims to transform the data as linear combinations of the statistically independent data point. Therefore, its goal is to provide an independent instead that uncorrelated image representation. ICA is an alternative to PCA, which give a more powerful data representation. It is a discriminant analysis criterion, which can be used to enhance PCA.

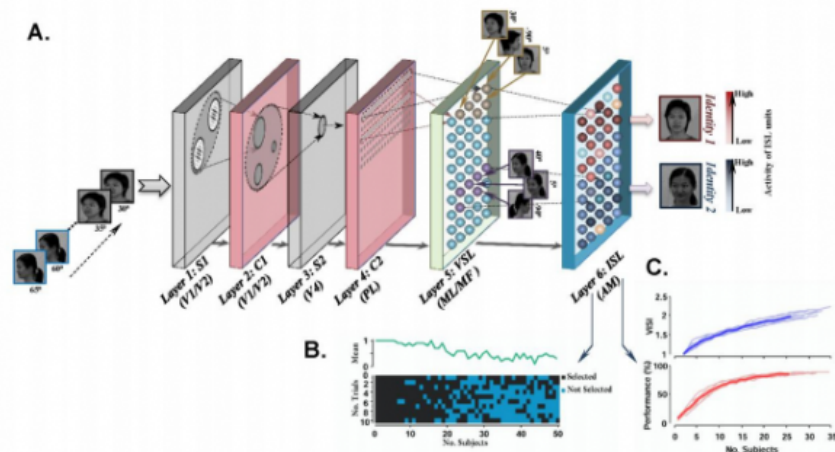
#### 4.2.7. Kernel PCA:-

Scholkopf et al. introduced the use of Kernel functions for performing nonlinear PCA. Its basic methodology is to apply a nonlinear mapping

to the input and then solve a linear PCA in the resulting feature subspace.

#### 4.3. Neural Networks:-

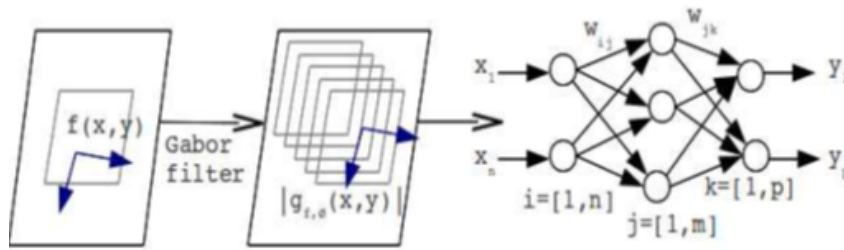
Neural Network has continued to use pattern recognition and classification. Kohonen was the first to show that a neuron network could be used to recognise aligned and normalised faces. There are methods, which perform feature extraction using neural networks. There are many methods, which combined with tools like PCA or LCA and make a hybrid classifier for face recognition. These are like Feed Forward Neural Network with additional bias, Self-Organizing Maps with PCA, and Convolutional Neural Networks with multi-layer perception, etc. These can increase the efficiency of the models.



Deep Neural Network for Face Recognition

##### 4.3.1. Neural Networks with Gabor Filters:-

The algorithm achieves face recognition by implementing a multilayer perceptron with a back-propagation algorithm. Firstly, there is a preprocessing step. Each image normalised in phases of contrast and illumination. Then each image is processed through a Gabor filter. The Gabor filter has five orientation parameters and three spatial frequencies, so there are 15 Gabor wavelengths.



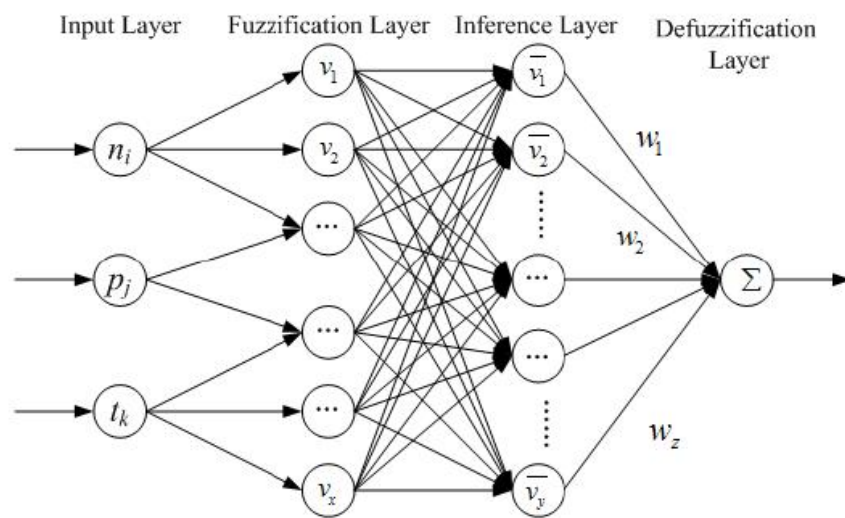
Neural Networks with Gabor filters

#### 4.3.2. Neural Networks and Hidden Markov Models:-

Hidden Markov Models are a statistical tool used in face recognition. They have been used in conjunction with neural networks. It generated in a neural network that trains pseudo 2D HMM. The input of this 2D HMM process is the output of the ANN, and It provides the algorithm with the proper dimensionality reduction.

#### 4.3.3. Fuzzy Neural Networks:-

The fuzzy neural networks for face recognition introduced in 2009. In this a face recognition system using a multilayer perceptron. The concept behind this approach is to capture decision surfaces in nonlinear manifolds a task that a simple MLP can hardly complete. The feature vectors are obtained using Gabor wavelength transforms.



Fuzzy Neural Network

### How the Face Recognition works:-

There are many ways for face recognition. Here we use OpenCV for face recognition. In face recognition, the image first prepared for preprocessing and then trained the face recogniser to recognise the faces. After teaching the recogniser, we test the recogniser to see the

results. The OpenCV face recogniser are of three types, which are as follows-

## 1.EigenFaces Face Recognizer

EigenFaces face recogniser views at all the training images of all the characters as a complex and try to deduce the components. These components are necessary and helpful (the parts that grab the most variance/change) and discard the rest of the images, This way it not only extracts the essential elements from the training data but also saves memory by rejecting the less critical segments.

## 2.FisherFaces Recognizer

Fisherfaces algorithm, instead of obtaining useful features that represent all the faces of all the persons, it removes valuable features that discriminate one person from the others. This features of one person do not dominate over the others, and you have the features that distinguish one person from the others.

## 3.Local Binary Patterns Histograms

We know that Eigenfaces and Fisherfaces are both affected by light and in real life; we cannot guarantee perfect light conditions. LBPH face recogniser is an improvement to overcome this drawback. The idea is not to find the local features of an image. LBPH algorithm tries to find the local structure of an image, and it does that by comparing each pixel with its neighbouring pixels.

## How to Run Face Recognition:-

```
#import OpenCV module
import cv2
import os
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

#function to detect face
def detect_face (img):

#convert the test image to gray image
gray = cv2.cvtColor (img, cv2.COLOR_BGR2GRAY)

#load OpenCV face detector
face_cas = cv2.CascadeClassifier ('-File name.xml-')
```

```

faces = face_cas.detectMultiScale (gray,
scaleFactor=1.3, minNeighbors=4);

#if no faces are detected then return image
if (len (faces) == 0):
    return None, None

#extract the face
faces [0]=(x, y, w, h)

#return only the face part
return gray[y: y+w, x: x+h], faces [0]

#this function will read all persons' training images,
detect face #from each image
#and will return two lists of exactly same size, one
list
def prepare_training_data(data_folder_path):

#-----STEP-1-----
#get the directories (one directory for each subject)
in data folder
dirs = os.listdir(data_folder_path)
faces = []
labels = []
for dir_name in dirs:

#our subject directories start with letter 's' so
#ignore any non-relevant directories if any
if not dir_name.startswith("s"):
    continue;

#-----STEP-2-----
#extract label number of subject from dir_name
#format of dir name = slabel
#, so removing letter 's' from dir_name will give us
label
label = int(dir_name.replace("s", ""))

#build path of directory containin images for current
subject subject
#sample subject_dir_path = "training-data/s1"
subject_dir_path = data_folder_path + "/" + dir_name

#get the images names that are inside the given
subject directory
subject_images_names = os.listdir(subject_dir_path)

#-----STEP-3-----
#go through each image name, read image,
#detect face and add face to list of faces
for image_name in subject_images_names:

#ignore system files like .DS_Store
if image_name.startswith("."):

```



```
continue;
```

```
#build image path  
#sample image path = training-data/s1/1.pgm  
image_path = subject_dir_path + "/" + image_name
```

```
#read image  
image = cv2.imread(image_path)
```

```
#display an image window to show the image  
cv2.imshow("Training on image...", image)  
cv2.waitKey(100)
```

```
#detect face  
face, rect = detect_face(image)
```

```
#-----STEP-4-----  
#we will ignore faces that are not detected  
if face is not None:
```

```
#add face to list of faces  
faces.append(face)
```

```
#add label for this face  
labels.append(label)  
cv2.destroyAllWindows()  
cv2.waitKey(1)  
cv2.destroyAllWindows()  
return faces, labels
```

```
#let's first prepare our training data  
#data will be in two lists of same size  
#one list will contain all the faces  
#and other list will contain respective labels for  
each face  
print("Preparing data...")  
faces, labels = prepare_training_data("training-data")  
print("Data prepared")
```

```
#print total faces and labels  
print("Total faces: ", len(faces))  
print("Total labels: ", len(labels))
```

```
#create our LBPH face recognizer  
face_recognizer = cv2.face.createLBPHFaceRecognizer()
```

```
#train our face recognizer of our training faces  
face_recognizer.train(faces, np.array(labels))
```

```
#function to draw rectangle on image  
#according to given (x, y) coordinates and  
#given width and height  
def draw_rectangle(img, rect):  
    (x, y, w, h) = rect  
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
```

```
#function to draw text on give image starting from  
#passed (x, y) coordinates.  
def draw_text(img, text, x, y):  
cv2.putText(img, text, (x, y), cv2.FONT_HERSHEY_PLAIN,  
1.5, (0, 255, 0), 2)
```

```
#this function recognizes the person in image passed  
#and draws a rectangle around detected face with name  
of the subject  
def predict(test_img):
```

```
#make a copy of the image as we don't want to chang  
original image  
img = test_img.copy()
```

```
#detect face from the image  
face, rect = detect_face(img)
```

```
#predict the image using our face recognizer  
label= face_recognizer.predict(face)
```

```
#get name of respective label returned by face  
recognizer  
label_text = subjects[label]
```

```
#draw a rectangle around face detected  
draw_rectangle(img, rect)  
  
#draw name of predicted person  
draw_text(img, label_text, rect[0], rect[1]-5)  
return img
```

```
#load test images  
test_img1 = cv2.imread("test-data/test1.jpg")  
test_img2 = cv2.imread("test-data/test2.jpg")
```

```
#perform a prediction  
predicted_img1 = predict(test_img1)  
predicted_img2 = predict(test_img2)  
print("Prediction complete")
```

```
#create a figure of 2 plots (one for each test image)  
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 5))
```

```
#display test image1 result  
ax1.imshow(cv2.cvtColor(predicted_img1,  
cv2.COLOR_BGR2RGB))
```

```
#display test image2 result  
ax2.imshow(cv2.cvtColor(predicted_img2,  
cv2.COLOR_BGR2RGB))
```

```
#display both images  
cv2.imshow("Tom cruise test", predicted_img1)  
cv2.imshow("Shahrukh Khan test", predicted_img2)  
cv2.waitKey(0)  
cv2.destroyAllWindows()  
cv2.waitKey(1)  
cv2.destroyAllWindows()
```

*This blog is for beginners who want to start their carrier in the field of Computer Vision or AI by learning about what is face recognition, its types, and how it works.*

