DESIGN AND IMPLEMENTATION OF AN ONLINE PAYMENT SYSTEM USING FACE RECOGNITION TECHNIQUE

This project report is submitted to
Rashtrasant Tukadoji Maharaj Nagpur University
in the partial fulfilment of the requirement for the award of the degree
of

Bachelor of Engineering in Computer Technology

by

Mr. Bhupender (CT15115)
Mr. Nikhil Nagpure (CT15043)
Mr. Rishi Kumar (CT15113)
Ms. Samiksha Zade (CT15007)

Under the guidance of

Mr. K. Nagaraju
Assistant Professor

2018-2019

DEPARTMENT OF COMPUTER TECHNOLOGY
KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE
RAMTEK – 441 106

DEPARTMENT OF COMPUTER TECHNOLOGY

KAVIKULGURU INSTITUTE OF TECHNOLOGY AND SCIENCE

RAMTEK – 441 106

CERTIFICATE

This is to certify that the project report entitled **‘Design and Implementation of an Online Payment System using Face Recognition Technique’** carried out by Mr. Bhupender (CT15115),Mr. Nikhil Nagpure (CT15043), Mr. Rishi Kumar (CT15113), Ms. Samiksha Zade (CT15007) of B.E. final year of computer technology, during the academic year 2018-2019, in the partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering (Computer Technology)** offered by the **Rashtrasant Tukadoji Maharaj Nagpur University,** Nagpur.

Mr. K. Nagaraju

Guide

Mr. V.P. Mahatme — Dr. B. Ram Rathan Lal

Head of the Department — **Principal**

Date:

Place: Ramtek

DECLARATION

We declare that

a. The work contained in this dissertation has been done by us under the supervision of our guide.

b. The work has not been submitted to any other Institute for any degree or diploma.

c. We have followed the guidelines provided by the Institute in preparing the dissertation report.

d. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever we have used material (data, theoretical analysis, figures and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references.

Project-mates

ACKNOWLEDGEMENT

We are grateful to our respected guide **Mr. K. Nagaraju** for his kind, disciplined and invaluable guidance which inspired us to solve all the difficulties that came across during completion of project.

We are very thankful to our project coordinator **Mr. Amit Ukalkar** for his appreciation and contributions in our project, his supportive nature helps us to practice new thing during project

We express our special thanks to **Mr. V. P. Mahatme**, Head of the Department, for his kind support, valuable suggestion and allowing us to use all facilities that are available in the Department during this project.

Our sincere thanks are due to **Dr. B. Ram Rathan Lal**, Principal, for extending all the possible help and allowing us to use all resources that are available in the Institute.

We are also thankful to our **Parents** and **Friends** for their valuable cooperation and standing with us in all difficult conditions.

Project-mates

ABSTRACT

Face recognition from image or video is a popular topic in biometrics research. Many public places usually have surveillance cameras for video capture and these cameras have their significant value for security purpose. It is widely acknowledged that the face recognition have played an important role in surveillance system as it doesn't need the object's cooperation. The actual advantages of face based identification over other biometrics are uniqueness and acceptance. As human face is a dynamic object having high degree of variability in its appearance, that makes face detection a difficult problem in computer vision. In this field, accuracy and speed of identification is a main issue. Face recognition from the real data, capture images, sensor images and database images is challenging problem due to the wide variation of face appearances, illumination effect and the complexity of the image background. Face recognition is an interesting and successful application of pattern recognition and image analysis. Face processing is based on the fact that the information about a user's identity can be extracted from the images and the computers can act accordingly.

Face recognition and fingerprint scanner are the most effective and relevant applications of image processing and biometric systems. In this project, the system will scan the face of each person coming before the camera and the parameter of comparison for recognition of the faces are extracted and then they are compared with the already stored data in the database. If maximum parameter matches with any particular person's data then authentication for online payment is done. This system unlike fingerprint scanner cannot be deceived in any way and thus contributes to make a better online payment system.

KEYWORDS: face recognition, biometric, database, feature extraction, pattern recognition, viola jones, euclidean distance, open CV, webcam

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Face recognition is an interesting and successful application of pattern recognition and image analysis. Face processing is based on the fact that the information about a user’s identity can be extracted from the images and the computers can act accordingly. The human face is very important in doing interactions with people and conveying people’s identity. Facial recognition offers several advantages. Face recognition technology is gradually evolving to universal biometric solution, since it requires virtually zero efforts from the user end while compared with other biometric options. As compared with other biometrics systems such as retina scanning, fingerprint or palm print scan, face recognition has lots of advantages because in it non-contact process is used. In face recognition technique the image is captured from distance and the identification does not required actual interaction with a person.

Public safety authorities want to locate certain individuals like criminals, suspected terrorists, and missing children. The facial recognition system is useful for the public safety authorities in their mission. Some more applications of face recognition are face recognition is also useful in human computer interaction, virtual reality, database recovery, multimedia, computer entertainment, information security e.g. operating system, medical records, online banking, biometric e.g. Personal Identification - Passports, driver licenses, automated identity verification - border controls, law enforcement e.g. video surveillances, investigation, personal security - driver monitoring system, home video surveillance system. There are many different methods to perform face recognition technique. The simplest method for face recognition is by image processing technique.

This report summarizes the automatic online payment system by face recognition using image processing technique. Face recognition is an interesting and successful application of pattern recognition and image analysis. Face processing is based on the fact that the information about a user’s identity can be extracted from the images and the computers can act accordingly. Once a face is detected, the face region is cropped from the image to be used as “Probe” into the knowledge to check for possible matches. Face detection has many applications, ranging from entertainment, information security, and biometrics.

This project proposes online payment system using face recognition that can make online payment automatically after the face is recognized correctly. First, the face region is extracted from the image by applying pre-processing activities. The method of locating the face region is known as face localization. The local features such as eyes and mouth are extracted from the face region. The distance between the eye balls and the distance between the mouth end points are calculated using the distance calculation algorithm. These values are given as the input to the training algorithm and then these values are used at the time of recognition of the face of registered user i.e. at the time of making online payment the features are extracted from the current image and is compared with the previously generated face database. The online payment of particular person is made in the database as soon as the face has been recognized correctly. The online payment report can further be generated weekly, monthly, date wise and individual report can also be generated of every registered member.

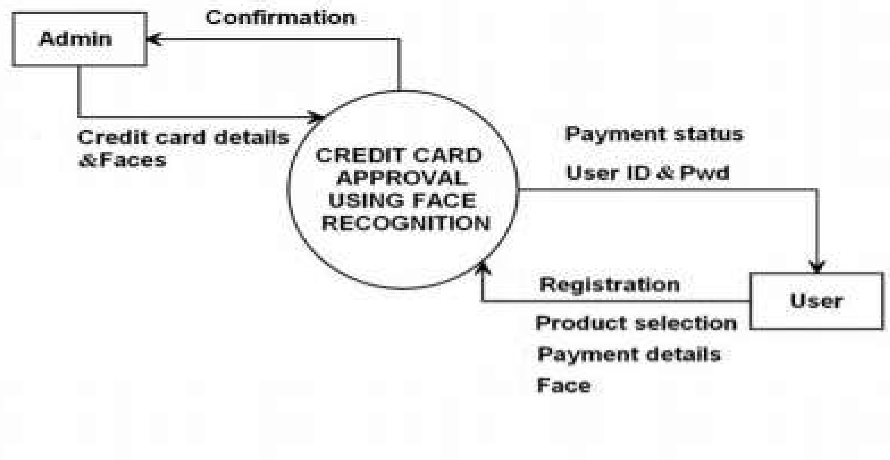


Fig. 1.1 Flow of the system

In the above figure, the input image to the face recognition system is acquired by taking photographs using the digital camera. These images are captured in color mode and are saved in jpg format. However, the proposed method is suitable for working with any file format. Then, feature extraction techniques are applied to train the system and recognize a face accordingly. Once recognized correctly, the online payment is awarded to the registered user.

1.1 Motivation

This project has been developed by getting motivation from the disadvantages of the current online payment system where one time password is used to make the online payment and generate the reports manually as per the requirement. This system consumes more time as well as less flexible. As the world is moving ahead with ―Go Green concept, so the proposed system has also been made to fulfil this concept.

1.2 Organization of Report

Chapter one contains the introduction of project which describe the face recognition system and flow of the system.

Chapter two includes the literature review which refers the literature required to carry out the proposed work.

Chapter three explains the proposed approach and system architecture which includes the detail of how the system has been developed.

Chapter four contains various snapshots of the results generated from the project and mentioned every necessary output needed to be described precisely.

Chapter five discusses the concluded work and how the future work can be extended further.

CHAPTER 2
LITERATURE REVIEW

A business can no longer afford not to offer its customers multiple payment options. Credit and debit cards are fast becoming the most common payment mode of big purchasers. Before the amount moves from the card holder's account into your business account, certain validations, checks and deductions are made. All these tasks are managed by the credit card processor. Cashless transactions benefit your business. Funds are transferred into your merchant account on time with hardly any effort from your side. The fundamental problem faced by the credit card users is to have a secure online transaction using credit cards. Credit card fraud is the biggest risk in credit card transactions. Credit cards are stolen and used to make large purchases, often leading to heavy losses for the credit card processing service and the business. To overcome the traditional method, a face recognition based online payment system came into existence.

2.1 Image

An Image may be defined as a two-dimensional function where and are spatial (plane) coordinates, and the amplitude of at any pair of coordinates is called the intensity or gray level of the image at the point. When x, y, and the intensity values of are all finite, discrete quantities, the image is called as ―digital image. The digital image processing deals with developing a digital system that perform operations on a digital image. Where as to train the computer so as the computer will be able to recognize the person in the image and to provide the online payment the part of artificial intelligence is use called neural network.

- **Purpose of Image Processing**

  The purpose of image processing is divided into 5 groups:

  1. Visualization - Observe the objects that are not visible.
  2. Image sharpening and restoration - To create a better image.
  3. Image retrieval - Seek for the image of interest.

4. Measurement of pattern – Measures various objects in an image.
5. Image Recognition – Distinguish the objects in an image.

2.2 Face Recognition

A facial recognition system is one of the computer applications for automatically identifying person from digital image or a snapshot from video source. In that technique the image and a facial database is compared with selected facial features from the image. It is used in various systems like security systems and can be compared to other biometrics like eye iris recognition or fingerprint system. There are different techniques of facial recognition like traditional, 3D recognition and skin texture analysis.

Kaushik S., Dubey R. B., Madan A. (2014) proposed a new application for face recognition task on mobile device. The originality of the work is based on the web service, able to store the normalized distances in the database, to verify the correct building of the classifier's model and to match the test normalized distances with the training normalized distances already stored in the database. This innovative technique is named Bridge Approach (BA). The Bridge Approach allows both to speed up the processing and to use the face recognition system in all locations where an internet network is available. The application uses the OpenCV library for dynamic targeting of face, eyes, nose and mouth and WEKA library for the classification purpose. Experimental results are carried out on 500 face images belonging to 10 individuals (50 images each). Finally, the main goal of the work is to show the performances in terms of computational time and classifier's accuracy.

Another important aspect about the implementation of this mobile face recognition system is the execution of some functions outside the mobile processor (Bridge Approach). Some authors have already considered the use of web service architecture as support to the calculation. So, the operations are faster respect to the processing on the mobile device and the system is able to increase the biometric landmarks in order to improve the recognition accuracy at a lower computational cost. As already discussed in the literature, best performance depends on features, classifiers and data distribution between samples in the training.

It is a type of biometric software application that can identify a specific individual in a digital image by analyzing and comparing patterns. Facial recognition systems are commonly used for security purposes but are increasingly being used in a variety of other applications. Here from an image the face is detected cropped and compared with the faces already stored in the database whereas not all the face is required only few features are required for the comparison and to make database simple and small digital image processing is used to extract the features from the face (image). Face recognition technology have improved dramatically in their performance over the past few years, and this technology is now widely used for various purposes such as for security and for commercial applications.

Kanti J. and Papola A. (2014) described face recognition systems have been grabbing high attention from commercial market point of view as well as pattern recognition field. Face recognition has received substantial attention from researches in biometrics, pattern recognition field and computer vision communities. The face recognition systems can extract the features of face and compare this with the existing database. The faces considered here for comparison are still faces. Machine recognition of faces from still and video images is emerging as an active research area. The present paper is formulated based on still or video images captured either by a digital camera or by a web cam. The face recognition system detects only the faces from the image scene, extracts the descriptive features. It later compares with the database of faces, which is collection of faces in different poses. The present system is trained with the database, where the images are taken in different poses, with glasses, with and without beard. Eigenfaces are a set of eigenvectors used in the computer vision problem of human face recognition. Eigenfaces assume ghastly appearance. The refer to an appearance-based approach to face recognition that seeks to capture the variation in a collection of face images and use this information to encode and compare images of individual faces in a holistic manner. Specifically, the eigenfaces are the principal components of a distribution of faces, or equivalently, the eigenvectors of the covariance matrix of the set of face images, where an image with NxN pixels is considered a point (or vector) in N2-dimensional space. The idea of using principal components to represent human faces was developed by Sirovich and Kirby.(Sirovich and Kirby 1987) and used by Turk and Pentland (Turk and Pentland 1991) for face detection and recognition. The Eigenface approach is considered by many to be the first working facial recognition technology, and it served as the basis for one of the top commercial face recognition technology products. Since its initial development and publication, there have been many extensions to the original method and many new developments in automatic face recognition systems.

2.2.1 Traditional Methods

Some facial recognition algorithm detects facial features by extracting landmarks from an image of the subject. An algorithm may recognize the relative position, size and shape of an eye, nose, jaw and cheek bones. These landmarks are used to search the matching features from other images. Other algorithm compares the face data with the subject and only saving the useful data which is use for facial recognition.

2.2.2 3D Face Recognition

3D face recognition is newly emerging trend to achieve improved accuracy in facial recognition. 3D sensors are used to capture information about the face of an object. This information is then used to identify various features on the object face such as nose, contour of an eye jacket and chin. One of the best advantages of 3D facial recognition is that it is not affected by changes in lightening like other techniques. This type of recognition improves the precision of facial recognition.

Neelesh S., Priyank P., Kavita Sonawane (2012) detected the face is the identity of a person. The methods to exploit this physical feature have seen a great change since the advent of image processing techniques. The accurate recognition of a person is the sole aim of a face recognition system and this identification maybe used for further processing. Traditional face recognition systems employ methods to identify a face from the given input but the results are not usually accurate and precise as desired. The system described in this paper aims to deviate from such traditional systems and introduce a new approach to identify a person using a face recognition system i.e. the generation of a 3D Facial Model. The described working of the face recognition system that will be deployed as an Automated Attendance System in a classroom environment. The techniques and algorithms used along with the constraints and practical difficulties will be highlighted in this paper. The use of Fuzzy Logic and the concepts of Content Based Image Retrieval (CBIR) will be the main aspect of the proposed automated system. The purpose of recognition and identification has gone past the mere need for security. The use of face recognition software has been recently used to ease mundane tasks. The age of automation has given rise to such exploitation and it has been remarkable. The techniques that surround face recognition and identification are the basic concepts of Image Processing. The use of optimized algorithms to refine the results and produce a better and quick result has been a goal that has been achieved in most of the implementations.

Turk M. A. and Pentland A. P. (1991) detected the facial recognition system store video or photographs and try to find recognizable facial characteristics and match them against known facial templates to identify individuals. Most current facial recognition system processes the 2D camera image, although recent products have emerged that try to map the face in 3D using multiple camera angles. Face recognition has the greatest advantage of not requiring any sort of contact, so there are no hygienic concerns. The biometric samples here are typically taken as 2D images of the frontal section of the face, using one or more digital cameras. Typical technical approaches to achieve this recognition include geometrical, eigenfaces, template and graph matching, neural networks and Hidden Markov Models, or a combination of these. Although there are significant advantages to this technology, mainly due to the extrovert characteristics of face images and the user acceptability, the recognition accuracy has shown to be rather inaccurate in practice. The inaccuracy can be explained by a high sensitivity to environmental conditions such as lighting and image background, and also due to changes in the appearance of a face with regards to hairstyle, beard and glasses for example. This paper proposes a system that will try to overcome such problems. Despite the rather poor recognition accuracy, face recognition appears to be among the most interesting biometrics for user authentication in ID document scenarios, simply because of the simple image acquisition and the intuitive concept of comparing face images to images included in the document. Consequently, standards for the layout of facial images, as well as digital storage formats are currently developed.

2.2.3 Skin Texture Analysis

This technology use the visual details of the skin as captured in standard digital image or scan image. This technique of facial recognition turns the unique lines, patterns and spots which appear on person‘s skin in a mathematical space. By addition of skin texture analysis, performance of face recognition can be increase up to 20-25 percent.

There are some notable software with facial recognition ability include Picasa (Google), iPhoto (apple), windows live photo gallery (Microsoft), picture motion browser (Sony).

2.3 Artificial Intelligence (AI)

It is intelligence exhibited by machines. In computer science, an ideal ―intelligent‖ machine is a flexible rational agent that perceives its environment and takes actions that maximize its chance of success at some goal. Neural network is a computer system modeled on the human brain and nervous system whereas there are many face recognition as well as the training techniques such as Eigen face, Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA), Kernel Methods etc. whereas training techniques are Fuzzy logic, Genetic algorithms whereas the developing platform can be used as Matlab and Java.

In mathematics, eigenvalue, eigenvector, and eigenspace are related concepts in the field of linear algebra. The prefix eigen is the German word for innate, distinct, self. Linear algebra studies linear transformations, which are represented by matrices acting on vectors. Eigenvalues, eigenvectors and eigenspaces are properties of a matrix. They are computed by a method described below, give important information about the matrix, and can be used in matrix factorization. They have applications in areas of applied mathematics as diverse as finance and quantum mechanics. In general, a matrix acts on a vector by changing both its magnitude and its direction. Eigen faces is the name given to a set of eigenvectors when they are used in the computer vision problem of human face recognition. Fuzzy Logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based.

2.4 Artificial Neural Network (ANN)

Srivastava S. and Jain D. C. (2016) proposed face recognition technology have improved dramatically in their performance over the past few years, and this technology is now widely used for various purposes such as for security and for commercial applications. Face recognition is an active area of research which is a computer based digital technology. Use of face recognition for the purpose of attendance marking is a smart way of attendance management system. In this paper we present a new way for automated attendance system which makes use of Principal component Analysis (PCA) along with Artificial Neural networks (ANN). As human brain has the learning ability to recognize the persons by their faces even the feature characteristics change with time because the neurons of human brain are trained by reading or learning the face of a person and can identify that face even after several years. Similarly this ability of training and identifying is converted into machine system using Artificial Neural networks (ANN). The basic function for the face recognition system is to compare the face of a person which is to be recognized with the faces already trained in the Artificial Neural Networks and it recognized the best matching face as output even at different lightening conditions, viewing conditions and facial expressions.

These feature vectors are the eigenvectors of covariance matrix and having the face like appearance so that we call them eigenfaces which are used as input to train the Artificial Neural Networks. The learning of the correlated patterns between the input face images is one of the useful properties of Artificial Neural Networks. After training the Artificial Neural Networks, we tested it with known and unknown face images for success and rejection rate analysis.

Warren McCulloch and Walter Pitts (1943) created a computational model for neural networks based on mathematics and algorithms called threshold logic. This model paved the way for neural network research to split into two distinct approaches. One approach focused on biological processes in the brain and the other focused on the application of neural networks to artificial intelligence. Hebbian learning, in the late 1940 psychologist Donald Hebb created a hypothesis of learning based on the mechanism of neural plasticity that is now known as Hebbian learning. Hebbian learning is considered to be a 'typical' unsupervised learning rule and its later variants were early models for long term potentiation. Researchers started applying these ideas to computational models in 1948 with Turing's B-type machines. Farley and Wesley A. Clark (1954) first used computational machines, and then called calculators, to simulate a Hebbian network at MIT. Other neural network computational machines were created by Rochester, Holland, Habit, and Duda (1956). Frank Rosenblatt (1958) created the perceptron, an algorithm for pattern recognition based on a two-layer computer learning network using simple addition and subtraction. With mathematical notation, Rosenblatt also described circuitry not in the basic perceptron, such as the exclusive-or circuit, a circuit which could not be processed by neural networks until after the back propagation algorithm was created by Paul Werbos (1975). Neural network research stagnated after the publication of machine learning research by Marvin Minsky and Seymour Papert (1969), who discovered two key issues with the computational machines that processed neural networks. The first was that basic perceptrons were incapable of processing the exclusive-or circuit. The second significant issue was that computers didn't have enough processing power to effectively handle the long run time required by large neural networks. Neural network research slowed until computers achieved greater processing power.

Back propagation and resurgence, a key advance that came later was the back propagation algorithm which effectively solved the exclusive-or problem, and more generally the problem of quickly training multi-layer neural networks (Werbos 1975). In the mid-1980s, parallel distributed processing became popular under the name connectionism. The textbook by David E. Rumelhart and James McClelland (1986) provided a full exposition of the use of connectionism in computers to simulate neural processes.

Neural networks, as used in artificial intelligence, have traditionally been viewed as simplified models of neural processing in the brain, even though the relation between this model and the biological architecture of the brain is debated; it's not clear to what degree artificial neural networks mirror brain function. Support vector machines and other, much simpler methods such as linear classifiers gradually overtook neural networks in machine learning popularity. But the advent of deep learning in the late 2000s sparked renewed interest in neural networks.

In machine learning and cognitive science, an artificial neural network (ANN) is a network inspired by biological neural networks (the central nervous systems of animals, in particular the brain) which are used to estimate or approximate functions that can depend on a large number of inputs that are generally unknown.

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process.

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an expert in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer what if questions.

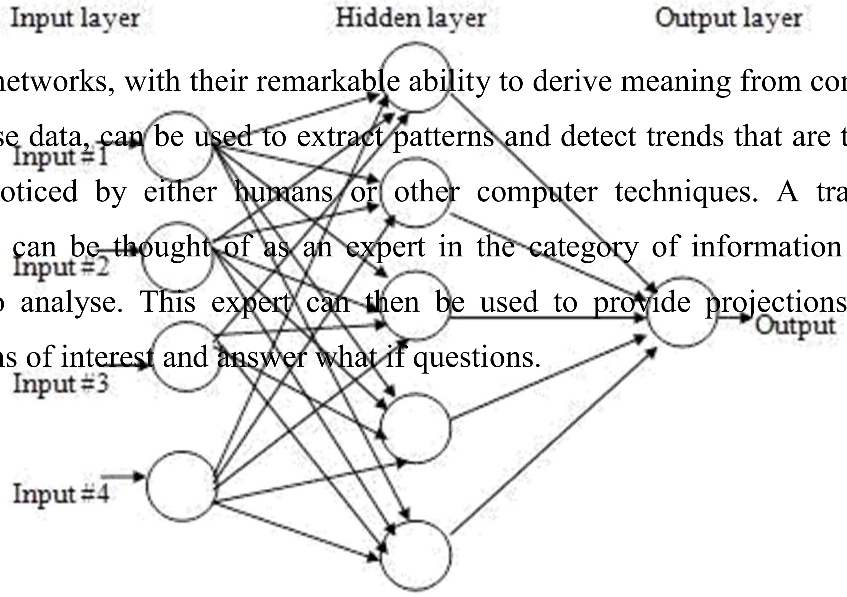


Fig. 2.1 Architecture of neural network

2.4.1 Supervised Learning

In supervised learning, there are given a set of example pairs and the aim is to find a function f in the allowed class of functions that matches the examples. In other words, it is required to infer the mapping implied by the data; the cost function is related to the mismatch between the mapping and the data and it implicitly contains prior knowledge about the problem domain. A commonly used cost is the mean-squared error which tries to minimize the average error between the network's output, and the target value y over all the example pairs. When one tries to minimize this cost using gradient descent for the class of neural networks called Multi-Layer Perceptrons, one obtains the well-known back propagation algorithm for training neural networks. Tasks that fall within the paradigm of supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). The supervised learning paradigm is also applicable to sequential data (e.g., for speech and gesture recognition). This can be thought of as learning with a ―teacher‖ in the form of a function that provides continuous feedback on the quality of solutions obtained thus far.

2.4.2 Unsupervised Learning

In unsupervised learning there are given some data , and the cost function to be minimized can be any function of the data and the network's output . The cost function is dependent on the task (what is trying to be modelled) and it is a priori assumptions (the implicit properties of this model, its parameters and the observed variables). As a trivial example, consider the model , where is a constant and the cost . Minimizing this cost will give us a value of that is equal to the mean of the data. The cost function can be much more complicated. Its form depends on the application: For example in compression it could be related to the mutual information between x and y. In statistical modelling, it could be related to the posterior probability of the model given the data. (Note that in both of those examples those quantities would be maximized rather than minimized) Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering.

2.4.3 Types of Neural Networks

- **Feed Forward Neural Network**

  The feed forward neural networks are the first and arguably simplest type of artificial neural networks devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network.

- **Single-Layer Perceptron**

  The earliest kind of neural network is a single-layer perceptron network, which consists of a single layer of output nodes; the inputs are fed directly to the outputs via a series of weights. In this way it can be considered the simplest kind of feed-forward network. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold (typically 0) the neuron fires and takes the activated value (typically 1); otherwise it takes the deactivated value (typically -1). Neurons with this kind of activation function are also called McCulloch-Pitts neurons or threshold neurons.

- **Multilayer Perceptron**

  The MLP neural network consists of an input layer, one or more hidden layers, and an output layer. Each layer is made up of units. The inputs to the network correspond to the attributes measured for each training tuple. The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer of —neuron-like‖ units, known as a hidden layer. The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary, although in practice, usually only one is used. The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network‘s prediction for given tuple. The network is feed-forward in that none of the weights cycles back to an input unit or to an output unit of a previous layer. It is fully connected in that each unit provides input to each unit in the next forward layer. Each output unit takes, as input, a weighted sum of the outputs from units in the previous layer. It applies a nonlinear (activation) function to the weighted input.

2.4.4 Back Propagation

Back propagation is a common method of training artificial neural networks so as to minimize the objective function. It is a supervised learning method, and is a generalization of the delta rule. As shown in Fig.2.2, it requires a dataset of the desired output for many inputs, making up the training set. It is most useful for feed-forward networks (networks that have no feedback, or simply, that have no connections that loop). The term is an abbreviation for —backward propagation of

errors. Back propagation requires that the activation function used by the artificial neurons (or —nodes‖) be differentiable.

Fig. 2.2 Back Propagation of Multi-layered ANN

2.5 Biometric Identification

It is the technique of automatically identifying or verifying an individual by a physical characteristic or personal trait. The term —automatically means the biometric identification system must identify or verify a human characteristic or trait quickly with little or no intervention from the user.

Biometric is programmed methods of identify a person or verifying the characteristics of a person based on a physiological or behavioural point. Examples of physiological character include hand or finger images, facial character. Behavioural characters are qualities that are learned or acquire. Dynamic signature authentication, speaker verification and keystroke dynamics are examples of behavioural character. Biometric confirmation requires comparing a registered or enrolled biometric sample beside a newly captured biometric sample for example, a fingerprint capture during a login. During enrolment a sample of the biometric attribute is captured, processed by a computer, and stored for later comparison. A neutral face is a relaxed face without contraction of facial muscles and without facial movements. Face recognition systems can achieve high recognition rate for good quality, frontal view, constant lighting and only subtle expression or expressionless face images. The performance of face recognition system significantly decreases when there is a dramatic expression on the face. Therefore, it is important to automatically find the best face of a subject from the images.

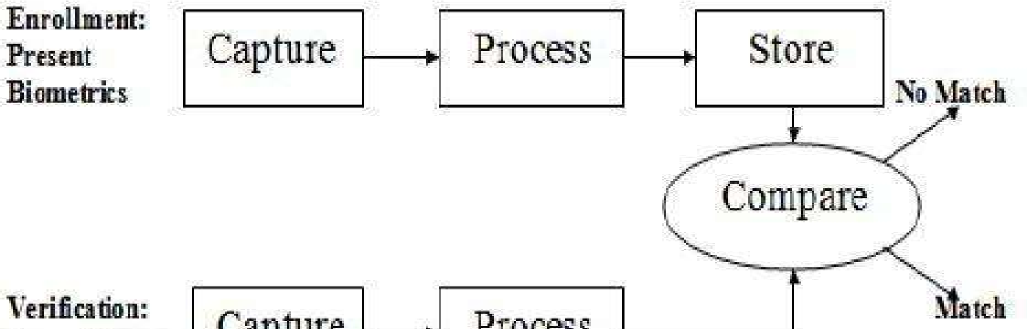


Fig. 2.3 Generic Biometric Process

2.6 MATLAB

MATLAB is a high-performance fourth-generation language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include: Math and computation & algorithm development.

2.6.1 Features of MATLAB

- It also provides an interactive environment for iterative exploration, design and problem solving.
- It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
- It provides built-in graphics for visualizing data and tools for creating custom plots.
- MATLAB‘s programming interface gives development tools for improving code quality, maintainability, and maximizing performance.
- It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

2.6.2 Uses of MATLAB

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

- Signal processing and communication
- Image and video processing
- Control System

- Test and measurement
- Computational finance
- Computational biology

MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands. It is highly encouraged to plot mathematical functions and results of analysis as often as possible. Trying to understand mathematical equations with graphics is an enjoyable and very efficient way of learning mathematics. Being able to plot mathematical functions and data freely is the most important step, and this section is written to assist to do just that.

2.6.3 The Image Acquisition Toolbox

Image Acquisition Toolbox functionality is available in a desktop application. You connect directly to the hardware in the tool and can preview and acquire image data. You can log the data to MATLAB in several formats, and also generate a VideoWriter file, right from the tool. The Image Acquisition Tool provides a desktop environment that integrates a preview/acquisition area with Acquisition Parameters so that you can change settings and see the changes dynamically applied to the image data.

To open the Image Acquisition Tool, do one of the following:

1. Type image tool at the MATLAB command line.
2. Select Image Acquisition on the Apps tab in MATLAB.

Parts of the Desktop:

The Image Acquisition Tool has the following panes:

- Hardware Browser

  Shows the image acquisition devices currently connected to the system. Each device is a separate node in the browser. All of the formats the device supports are listed under the device. Each device's default format is indicated in parentheses. Select the device format or camera file you want to use for the acquisition. When the format is selected, you can then set acquisition parameters and preview the data. See Selecting the Device in Image Acquisition Tool for more information about using the Hardware Browser.

- Preview window

  Use to preview and acquire image data from the selected device format, and to export data that has been acquired in memory to a MAT-file, the MATLAB Workspace, Video Writer, or to tools provided by the Image Processing Toolbox software. See Previewing and Acquiring Data in Image Acquisition Tool for more information about using the Preview window.
- Acquisition Parameters

  Use these tabs to set up general acquisition parameters, such as frames per trigger and color space, device-specific properties, logging options, triggering options, and region of interest. Settings you make on any tab will apply to the currently selected device format in the Hardware Browser. See Setting Acquisition Parameters in Image Acquisition Tool for more information about using the Acquisition Parameters. Also see the Help for each tab while using the tool for more details. When you click any tab, the help for that tab will appear in the Desktop Help pane.
- Information pane

  Displays a summary of information about the selected node in the Hardware Browser.
- Session Log

  Displays a dynamically generated log of the commands that correspond to actions taken in the tool. You can save the log to a MATLAB code file or copy it.
- Desktop Help

  Displays Help for the pane of the desktop that has focus. Click inside a pane for help on that area of the tool. For the Acquisition Parameters pane, click each tab to display information about the settings for that tab.

2.6.4 detectMinEigenFeatures Function

Detect corners using minimum eigenvalue algorithm and return corner Points object collapse all in page. The detectMinEigenFeatures function uses the minimum eigenvalue algorithm developed by Shi and Tomasi to find feature points. Syntax:

points = detectMinEigenFeatures(I)

points = detectMinEigenFeatures(I,Name,Value)

Description:

Example,

points = detectMinEigenFeatures(I) returns a cornerPoints object, points. The object contains information about the feature points detected in a 2-D grayscale input image, I. The detectMinEigenFeatures function uses the minimum eigenvalue algorithm developed by Shi and Tomasi to find feature points.

points = detectMinEigenFeatures(I,Name,Value)) uses additional options specified by one or more Name,Value pair arguments.

2.6.5 Computer Vision Toolbox

It is a field that includes methods for acquiring, processing, analyzing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions. A theme in the development of this field has been to duplicate the abilities of human vision by electronically perceiving and understanding an image. Computer vision is the transformation of data from a still or video camera into either a decision or a new representation. The classical problem in computer vision, image processing, and machine vision is that of determining whether or not the image data contains some specific object.

2.7 Java

Java is the first and foremost an object-oriented programming language. Java is a programming language originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture.

2.7.1 History of Java

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991.Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

Sun Microsystems released the first public implementation as Java 1.0 in 1995. It promised "Write Once, Run anywhere" (WORA), providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run Java applets within web pages, and Java quickly became popular, while mostly outside of browsers, that wasn't the original plan. In January 2016, Oracle announced that Java runtime environments based on JDK 9 will discontinue the browser plugin. The Java 1.0 compiler was re-written in Java by Arthur van Hoff to comply strictly with the Java 1.0 language specification. With the advent of Java 2 (released initially as J2SE 1.2 in December 1998 – 1999), new versions had multiple configurations built for different types of platforms. J2EE included technologies and APIs for enterprise applications typically run in server environments, while J2ME featured APIs optimized for mobile applications. The desktop version was renamed J2SE. In 2006, for marketing purposes, Sun renamed new J2 versions as Java EE, Java ME, and Java SE, respectively.

In 1997, Sun Microsystems approached the ISO/IEC JTC 1 standards body and later the Ecma International to formalize Java, but it soon withdrew from the process. Java remains a de facto standard, controlled through the Java Community Process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine (JVM) as free and open-source software, (FOSS), under the terms of the GNU General Public License (GPL). On May 8, 2007, Sun finished the process, making all of its JVM's core code available under free software/open-source distribution terms, aside from a small portion of code to which Sun did not hold the copyright. Sun's vice-president Rich Green said that Sun's ideal role with regard to Java was as an ―evangelist‖. Following Oracle Corporation's acquisition of Sun Microsystems in 2009–10, Oracle has described itself as the ―steward of Java technology with a relentless commitment to fostering a community of participation and transparency‖. This did not prevent Oracle from filing a lawsuit against Google shortly after that for using Java inside the Android SDK (Software Development Kit). Java software runs on everything from laptops to data centres, game consoles to scientific supercomputers. On April 2, 2010, James Gosling resigned from Oracle.

2.7.2 Principles

There were five primary goals in the creation of the Java language:

- It must be ―simple, object-oriented, and familiar.
- It must be ―robust and secure.
- It must be ―architecture-neutral and portable.
- It must execute with ―high performance.
- It must be ―interpreted, threaded, and dynamic.

2.8 OpenCV

Officially launched in 1999, the OpenCV project was initially an Intel Research initiative to advance CPU-intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel‗s Performance Library Team. In the early days of OpenCV, the goals of the project were described as:

- Advance vision research by providing not only open but also optimized code for basic vision infrastructure. No more reinventing the wheel.
- Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readily readable and transferable.

Advance vision-based commercial applications by making portable, performance-optimized code available for free—with a license that did not require being open or free for them. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation.

The first alpha version of OpenCV was released to the public at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and five betas were released between 2001 and 2005. The first 1.0 version was released in 2006. In mid-2008, OpenCV obtained corporate support from Willow Garage, and is now again under active development. A version 1.1 "pre-release" was released in October 2008.

The second major release of the OpenCV was on October 2009. OpenCV 2 includes major changes to the C++ interface, aiming at easier, more type-safe patterns, new functions, and better implementations for existing ones in terms of performance (especially on multi-core systems). Official releases now occur every six months and development is now done by an independent Russian team supported by commercial corporations.

2.8.1 Features of OpenCV

- Image data manipulation (allocation, release, copying, setting, conversion).
- Image and video I/O (file and camera based input, image/video file output).
- Matrix and vector manipulation and linear algebra routines (products, solvers, eigenvalues, SVD).
- Various dynamic data structures (lists, queues, sets, trees, graphs).
- Basic image processing (filtering, edge detection, corner detection, sampling and interpolation, color conversion, morphological operations, histograms, image pyramids).
- Motion analysis (optical flow, motion segmentation, tracking).
- Object recognition (Eigen-methods, HMM).
- Basic GUI (display image/video, keyboard and mouse handling, scroll-bars).
- Image labeling (line, conic, polygon, text drawing)
- One of OpenCV‘s goals is to provide a simple-to-use computer vision infrastructure that helps people build fairly sophisticated vision applications quickly. The OpenCV library contains over 500 functions that span many areas in vision, including factory product inspection, medical imaging, security, user interface, camera calibration, stereovision, and robotics. Because computer vision and machine learning often go hand-in hand.

2.8.2 OpenCV API’s

- core- It provides the core functionalities of OpenCV and contains basic structures and operations.
- highgui- provides easy interface to create and manipulate windows that can display images, add track bars to the windows, handle simple mouse events as well as keyboard commands, read and write images to/from disk or memory, read video from camera or file and write video to a file.
- imgproc- provides basic image processing operations like contrast, brightness, blur etc.
- video- provides video analysis functionalities.
- calib3d- provides camera calibration and 3D construction. Although it has got most of the images in a 2D format they do come from a 3D world. It allows to find out from 2D image information about 3D.
- features2d-provides feature detection and operations in 2D.
- objdetect- provides object detection functions like face detection, edge detection.
- ml- machine learning library is a set of classes and functions for statistical classification, regression, and clustering of data.

2.9 Haar Cascade Classifier

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here will work with face detection. 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. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

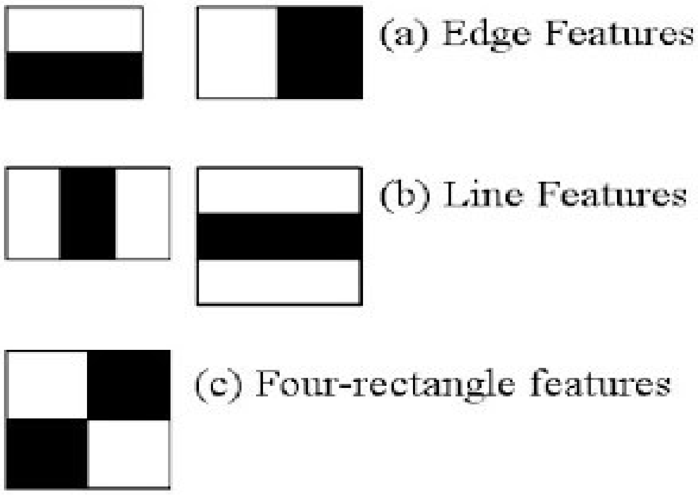


Fig. 2.4 Haar Features

Now, all possible sizes and locations of each kernel are used to calculate lots of features. Even a 24x24 window results over 160000 features). For each feature calculation, we need to find the sum of the pixels under white and black rectangles. To solve this, they introduced the integral image. However large your image, it reduces the calculations for a given pixel to an operation involving just four pixels. Nice, isn't it? It makes things super-fast.

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applied to cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.

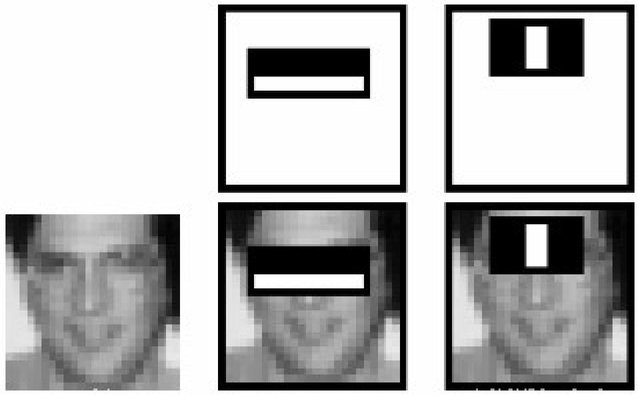


Fig. 2.5 Feature Extraction

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then the same process is done. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found).

The final classifier is a weighted sum of these weak classifiers. It is calledweak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features. (Imagine a reduction from 160000+ features to 6000 features. That is a big gain).

So now you take an image. Take each 24x24 window. Apply 6000 features to it. Check if it is face or not. Wow.. Isn't it a little inefficient and time consuming? Yes, it is. The authors have a good solution for that.

In an image, most of the image is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. This way, we spend more time checking possible face regions.

For this they introduced the concept of Cascade of Classifiers. Instead of applying all 6000 features on a window, the features are grouped into different stages of classifiers and applied one-by-one. (Normally the first few stages will contain very many fewer features). If a window fails the first stage, discard it. We don't consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region. How is that plan!

The authors' detector had 6000+ features with 38 stages with 1, 10, 25, 25 and 50 features in the first five stages. (The two features in the above image are actually obtained as the best two features from Adaboost). According to the authors, on average 10 features out of 6000+ are evaluated per sub-window.

So this is a simple intuitive explanation of how Viola-Jones face detection works. Read the paper for more details or check out the references in the Additional Resources section.

2.10 Face Recognition Techniques

Face recognition can be done in both a still image and video which has its origin in still image face recognition. Different approaches of face recognition for still images can be categorized into tree main groups such as:

i. Holistic approach.
ii. Feature based approach.
iii. Hybrid approach.

2.10.1 Holistic Approach

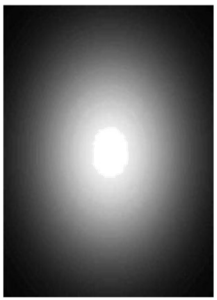
In holistic approach, the whole face region is taken into account as input data into face detection system. Examples of holistic methods are Eigenfaces (most widely used method for face recognition), probabilistic Eigenfaces, fisherfaces, support vector machines, nearest feature lines (NFL) and independent component analysis approaches. They are all based on principal component analysis (PCA) techniques that can be used to simplify a dataset into lower dimension while retaining the characteristics of dataset.

2.10.2 Featurebased Approach

In feature based approaches, local features on face such as nose, and then eyes are segmented and then used as input data for structural classifier. Pure geometry, dynamic link architecture, and hidden Markov model methods belong to this category. Feature based system requires points in the face image with high information content. Typically these are located around the eyes, nose and mouth. User is not interested in the face contour or hair, since most stable and informative features in the human face are located in the center of the face. There are several possible ways to locate points with high information content for this the Gabor filter approach is considered. The Gabor filters are generated from a wavelet expansion of a Gabor kernel, parameterized (determining the wavelength and orientation) by the vector. This filter helps in face recognition.

- **Overview of the Algorithm**

This contains two separate algorithms for training and testing (algorithm 1 and 2). However, the basic procedure is equivalent in both algorithms. When processing a face image (either for training or testing), we filter the image with a set of Gabor filters as described in the next section. Then we multiply the filtered image with a 2-D Gaussian to focus on the centre of the face, and avoid extracting features at the face contour. This Gabor filtered and Gaussian weighted image is then searched for peaks, which we define as interesting feature points for face recognition. At each peak, we extract a feature vector consisting of Gabor coefficients and we also store the location and class label.



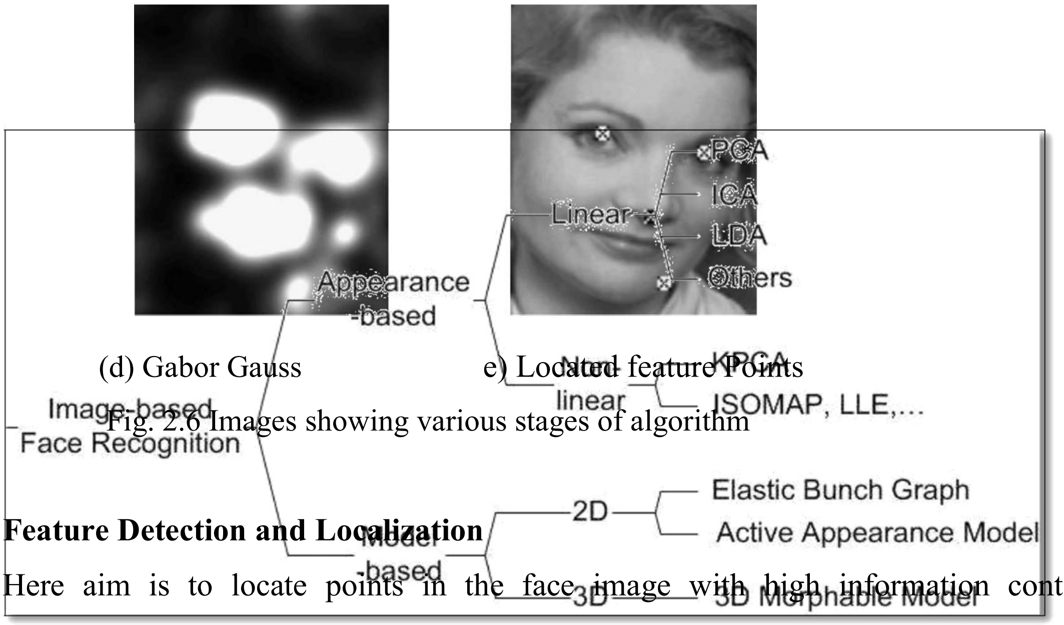
Algorithm 1: The training algorithm

1) Gabor filter face image.
2) Apply Gaussian weighting.
3) Locate peaks in image.
4) Extract feature vector at located peaks.
5) If this is first training image of subject, store feature vector, location and class label for all extracted peaks, else store only those who are misclassified (with respect to the current gallery).

Algorithm 2: The testing algorithm

1) Gabor filter face image.
2) Apply Gaussian weighting.
3) Locate peaks in image.
4) Extract feature vector at located peaks.
5) For each extracted feature vector, compute distance to all feature vectors in gallery
6) Based on class label to the nearest matching feature vectors, assign points to corresponding class

(a) Original image (b) Gabor filtered image (c) 2-D Gaussian



(d) Gabor Gauss e) Located feature Points

Fig. 2.6 Images showing various stages of algorithm

- **Feature Detection and Localization**

Here aim is to locate points in the face image with high information content. Typically these are located around the eyes, nose and mouth. Region of interest isn‘t the face contour or hair, since it is known that the most stable and informative features in the human face are located in the centre of the face. To enforce this Gaussian weighting is applied to focus the attention on the centre of the face.

2.10.3 Hybrid Approach

The idea of this method comes from how human vision system perceives both local feature and whole face. There are modular Eigenfaces, hybrid local feature, shape normalized, component based methods in hybrid approach.

Fig.2.7 Some face recognition techniques

2.11 Principal Component Analysis

Principal Components Analysis (PCA) is a mathematical formulation used in the reduction of data dimensions. Thus, the PCA technique allows the identification of standards in data and their expression in such a way that their similarities and differences are emphasized. Once patterns are found, they can be compressed, i.e., their dimensions can be reduced without much loss of information. In summary, the PCA formulation may be used as a digital image compression algorithm with a low level of loss. In the PCA approach, the information contained in a set of data is stored in a computational structure with reduced dimensions based on the integral projection of the data set onto a subspace generated by a system of orthogonal axes.

The optimal system of axes may be obtained using the Singular Values Decomposition (SVD) method. The reduced dimension computational structure is selected so that relevant data characteristics are identified with little loss of information. Such a reduction is advantageous in several instances: for image compression, data representation, calculation reduction necessary in subsequent processing, etc.

The Principal Component Analysis (PCA) is one of the most important techniques used in image recognition and compression. PCA is a mathematical tool which uses an orthogonal technique to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components.

The purpose of using PCA for face recognition is to express the large 1-D vector of pixels from 2-D facial image into the compact principal components of the feature space. This is named as eigenspace projection.

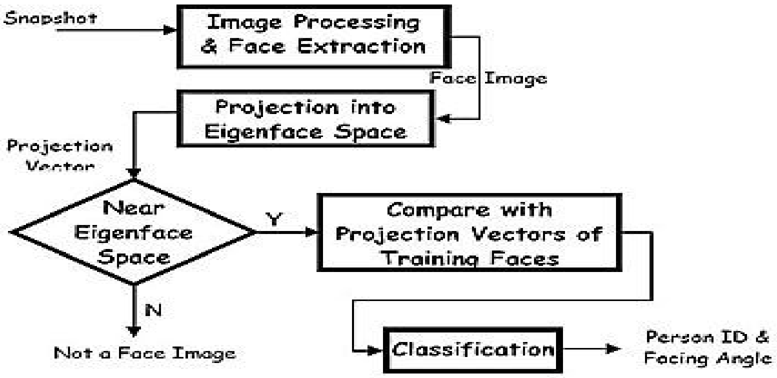

Snapshot
Image Processing & Face Extraction
Face Image
Projection into Eigenface Space
Projection Vector
Near Eigenface Space
Y
Compare with Projection Vectors of Training Faces
N
Not a Face Image
Classification
Person ID & Facing Angle

Fig. 2.8 Flow chart of PCA

- **Mathematical Analysis**
- Get Some Data

  A self-made data set is used which has got only two dimensions to show the PCA analysis at each step.
- Subtract the Mean

  For PCA to work properly, one has to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension.
- Calculate the Covariance Matrix.

  Calculate the Eigen Vectors and Eigen Values of the Covariance Matrix

  Since the covariance matrix is square, it can calculate the eigen vectors and eigen values for this matrix. These are rather important, as they tell us useful information about the data. In the meantime, here are the eigen vectors and eigen values; It is important to notice that these eigen vectors are both unit eigen vectors. Their lengths are both 1. This is very important for PCA, but luckily, most maths packages, when asked for eigenvectors, will give unit eigen vectors.

- **Choosing Components and Forming a Feature Vector**

  In general, once eigen vectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives the components in order of significance.
- **Deriving the New Data Set**

  This is the final step in PCA, and is also the easiest. Once it has been chosen the components (eigen vectors) that wishes to keep in the data and formed a feature vector, it simply take the transpose of the vector and multiply it on the left of the original data set, transposed. It will give the original data solely in terms of the vectors.
- **Eigen Vectors**

  In mathematics, eigenvalue, eigenvector, and eigenspace are related concepts in the field of linear algebra. The prefix eigen is the German word for innate, distinct, self. Linear algebra studies linear transformations, which are represented by matrices acting on vectors. Eigenvalues, eigenvectors and eigenspaces are properties of a matrix. They are computed by a method described below, give important information about the matrix, and can be used in matrix factorization. They have applications in areas of applied mathematics as diverse as finance and quantum mechanics.

In general, a matrix acts on a vector by changing both its magnitude and its direction. However, a matrix may act on certain vectors by changing only their magnitude, and leaving their direction unchanged (or possibly reversing it). These vectors are the eigenvectors of the matrix. A matrix acts on an eigenvector by multiplying its magnitude by a factor, which is positive if its direction is unchanged and negative if its direction is reversed. This factor is the eigenvalue associated with that eigenvector.

An eigenspace is the set of all eigenvectors that have the same eigenvalue, together with the zero vectors. The concepts cannot be formally defined without prerequisites, including an understanding of matrices, vectors, and linear transformations. The technical details are given below.

- **Eigen Faces**

In image processing, processed images of faces can be seen as vectors whose components are the brightness of each pixel. The dimension of this vector space is the number of pixels. The eigenvectors of the covariance matrix associated with a large set of normalized pictures of faces are called eigen faces; this is an example of principal components analysis.

They are very useful for expressing any face image as a linear combination of some of them. In the facial recognition branch of biometrics, eigen faces provide a means of applying data compression to faces for identification purposes. Research related to eigen vision systems determining hand gestures has also been made.

The task of facial recogniton is discriminating input signals (image data) into several classes (persons). The input signals are highly noisy (e.g. the noise is caused by differing lighting conditions, pose etc.), yet the input images are not completely random and in spite of their differences there are patterns which occur in any input signal. Such patterns, which can be observed in all signals could be - in the domain of facial recognition - the presence of some objects (eyes, nose, mouth) in any face as well as relative distances between these objects. These characteristic features are

called eigen faces in the facial recognition domain (or principal components generally).
They can be extracted out of original image data by means of a mathematical tool called Principal Component Analysis (PCA). By means of PCA one can transform each original image of the training set into a corresponding eigen face. An important feature of PCA is that one can reconstruct reconstruct any original image from the training set by combining the eigen faces. Remember that eigen faces are nothing less than characteristic features of the faces. Therefore one could say that the original face image can be reconstructed from Eigen faces if one adds up all the eigen faces (features) in the right proportion. Each eigen face represents only certain features of the face, which may or may not be present in the original image. If the feature is present in the original image to a higher degree, the share of the corresponding eigenface in the ”sum” of the eigen faces should be greater.
If, contrary, the particular feature is not (or almost not) present in the original image, then the corresponding eigenface should contribute a smaller (or not at all) part to the sum of eigenfaces. So, in order to reconstruct the original image from the eigenfaces, one has to build a kind of weighted sum of all eigenfaces. That is, the reconstructed original image is equal to a sum of all eigenfaces, with each eigenface having a certain weight. This weight specifies, to what degree the specific feature (eigenface) is present in the original image.

If one uses all the eigenfaces extracted from original images, one can reconstruct the original images from the eigenfaces *exactly*. But one can also use only a part of the eigenfaces. Then the reconstructed image is an approximation of the original image. However, one can ensure that losses due to omitting some of the eigenfaces can be minimized. This happens by choosing only the most important features (eigenfaces). Omission of eigenfaces is necessary due to scarcity of computational resources.

How does this relate to facial recognition? The clue is that it is possible not only to extract the face from eigenfaces given a set of weights, but also to go the opposite way. This opposite way would be to extract the weights from eigenfaces and the face to be recognized. These weights tell nothing less, as the amount by which the face in question differs from ”typical” faces represented by the eigenfaces. Therefore, using this weights one can determine two important things:

Determine, if the image in question is a face at all. In the case the weights of the image differ too much from the weights of face images (i.e. images, from which we know for sure that they are faces), the image probably is not a face.

Similar faces (images) possess similar features (eigenfaces) to similar degrees (weights). If one extracts weights from all the images available, the images could be grouped to clusters. That is, all images having similar weights are likely to be similar faces.

2.12 Fischerfaces

The Principal Component Analysis (PCA), which is the core of the Eigenfaces method, finds a linear combination of features that maximizes the total variance in data.

While this is clearly a powerful way to represent data, it doesn't consider any classes and so a lot of discriminative information *may* be lost when throwing components away. Imagine a situation where the variance in your data is generated by an external source, let it be the light. The components identified by a PCA do not necessarily contain any discriminative information at all, so the projected samples are smeared together and a classification becomes impossible.

The Linear Discriminant Analysis performs a class-specific dimensionality reduction and was invented by the great statistician Sir R. A. Fisher.. He successfully used it for classifying flowers in his 1936 paper The use of multiple measurements in taxonomic problems . In order to find the combination of features that separates best between classes the Linear Discriminant Analysis maximizes the ratio of between-classes to within-classes scatter, instead of maximizing the overall scatter. The idea is simple: same classes should cluster tightly together, while different classes are as far away as possible from each other in the lower-dimensional representation. This was also recognized by Belhumeur, Hespanhaand Kriegman and they applied a Discriminant Analysis to face recognition in [BHK97]

ALGORITHM DESCRIPTION

Let X be a random vector with samples drawn from c classes:

$$\begin{aligned} X &= \{X_1, X_2, \ldots, X_c\} \\ X_i &= \{x_1, x_2, \ldots, x_n\} \end{aligned}$$

The scatter matrices S_B and *S_{W}* are calculated as:

$$\begin{aligned} S_B &= \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T \\ S_W &= \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_j - \mu_i)^T \end{aligned}$$

, where μ is the total mean:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$$

And μ_i is the mean of class $i \in \{1, \ldots, c\}$:

$$\mu_i = \frac{1}{|X_i|} \sum_{x_j \in X_i} x_j$$

Fisher's classic algorithm now looks for a projection W, that maximizes the class separability criterion:

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

Following [BHK97], a solution for this optimization problem is given by solving the General Eigenvalue Problem:

$$\begin{aligned} S_B v_i &= \lambda_i S_w v_i \\ S_W^{-1} S_B v_i &= \lambda_i v_i \end{aligned}$$

There's one problem left to solve: The rank of S_W is at most $(N - c)$, with N samples and c classes. In pattern recognition problems the number of samples N is almost always smaller than the dimension of the input data (the number of pixels), so the scatter matrix S_W becomes singular. In this was solved by performing a Principal Component Analysis on the data and projecting the samples into the $(N - c)$-dimensional space. A Linear Discriminant Analysis was then performed on the reduced data, because S_W isn't singular anymore.

The optimization problem can then be rewritten as:

$$W_{pca} = \arg\max_W |W^T S_T W|$$

$$W_{fld} = \arg\max_W \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_W W_{pca} W|}$$

The transformation matrix W, that projects a sample into the $(c-1)$-dimensional space is then given by:

$$W = W_{fld}^T W_{pca}^T$$

For this example i am going to use the Yale Facedatabase A, just because the plots are nicer. Each Fisherface has the same length as an original image, thus it can be displayed as an image. The demo shows (or saves) the first, at most 16 Fisherfaces:

The Fisherfaces method learns a class-specific transformation matrix, so the they do not capture illumination as obviously as the Eigenfaces method. The Discriminant Analysis instead finds the facial features to discriminate between the persons.

It’s important to mention, that the performance of the Fisherfaces heavily depends on the input data as well. Practically said: if you learn the Fisherfaces for well-illuminated pictures only and you try to recognize faces in bad-illuminated scenes, then method is likely to find the wrong components (just because those features may not be predominant on bad illuminated images). This is somewhat logical, since the method had no chance to learn the illumination.

The Fisherfaces allow a reconstruction of the projected image, just like the Eigenfaces did. But since we only identified the features to distinguish between subjects, you can’t expect a nice reconstruction of the original image. For the Fisherfaces method we’ll project the sample image onto each of the Fisherfaces instead. So you’ll have a nice visualization, which feature each of the Fisherfaces describes:

The differences may be subtle for the human eyes, but you should be able to see some differences:

2.13 Local Binary Pattern Histogram

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Using the LBP combined with histograms we can represent the face images with a simple data vector.As LBP is a visual descriptor it can also be used for face recognition tasks, as can be seen in the following step-by-step explanation.

Now that we know a little more about face recognition and the LBPH, let's go further and see the steps of the algorithm:

1 .Parameters: the LBPH uses 4 parameters:

- Radius: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- Neighbors: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- Grid X: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- Grid Y: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. Training the Algorithm: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. Applying the LBP operation: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors.

The image below shows this procedure:

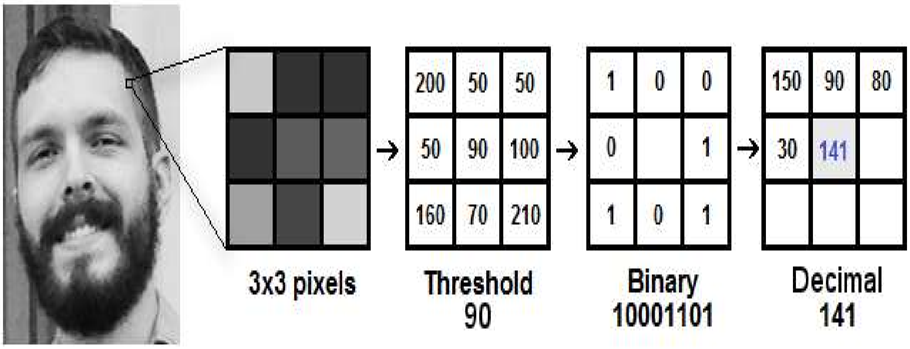


Fig. 2.9 LBPH threshold

Based on the image above, let’s break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
- Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), the final result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image which represents better the characteristics of the original image.

Note: The LBP procedure was expanded to use a different number of radius neighbors, it is called Circular LBP.

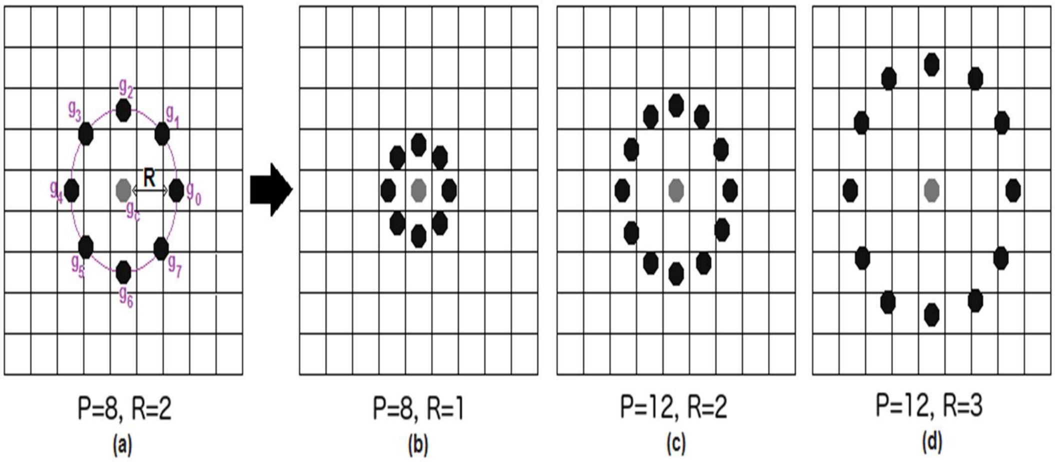


Fig. 2.10 Circular LBP

It can be done by using bilinear interpolation. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data points.

4. Extracting the Histograms: Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids, as can be seen in the following image:

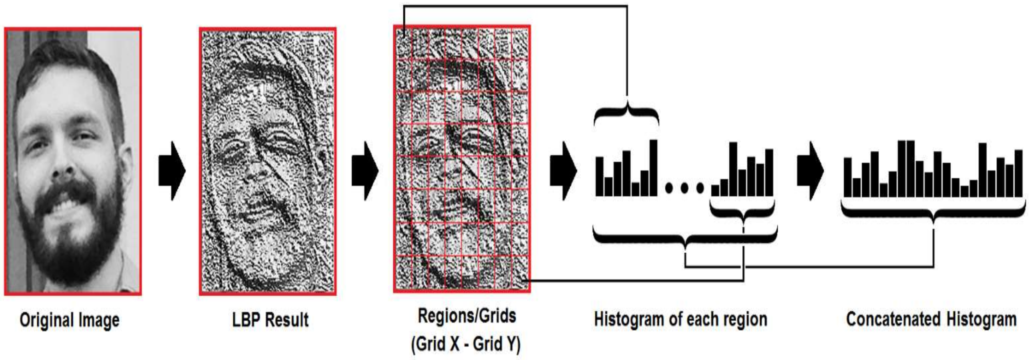


Fig. 2.11 Histogram generation

Based on the image above, we can extract the histogram of each region as follows:As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.
Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final

histogram. The final histogram represents the characteristics of the image original image.

5. Performing the face recognition: In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input image, we perform the steps again for this new image and creates a histogram which represents the image.

- So to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.
- We can use various approaches to compare the histograms (calculate the square, absolute value, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

- So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement.
- We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized if the confidence is lower than the threshold defined
- LBPH is one of the easiest face recognition algorithms.
- It can represent local features in the images.
- It is possible to get great results (mainly in a controlled environment).
- It is robust against monotonic gray scale transformations.
- It is provided by the OpenCV library (Open Source Computer Vision Library).

2.14 Databases

A database is an organized collection of data. It is the collection of schemas, tables, queries, reports, views, and other objects. The data are typically organized to model aspects of reality in a way that supports processes requiring information, such as modeling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

A database management system (DBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyze data. A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases.

Well-known DBMSs include MySQL, PostgreSQL, MongoDB, Microsoft SQL Server, Oracle, Sybase, SAP HANA, and IBM DB2. A database is not generally portable across different DBMSs, but different DBMS can interoperate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS. Database management systems are often classified according to the database model that they support; the most popular database systems since the 1980s have all supported the relational model as represented by the SQL language. Sometimes a DBMS is loosely referred to as a ―database.

2.14.1 Terminology and Overview

Formally, a database refers to a set of related data and the way it is system consisting of an integrated set of computer software that allows users to interact with one or more databases and provides access to all of the data contained in the database (although restrictions may exist that limit access to particular data). The Database management system (DBMS) provides various functions that allow entry, storage and retrieval of large quantities of information and provides ways to manage how that information is organized.

Because of the close relationship between them, the term ―database is often used casually to refer to both a database and the DBMS used to manipulate it. Outside the world of professional information technology, the term database is often used to refer to any collection of related data (such as a spreadsheet or a card index).

This article is concerned only with databases where the size and usage requirements necessitate use of a database management system.

Existing DBMSs provide various functions that allow management of a database and its data which can be classified into four main functional groups:

Data definition – Creation, modification and removal of definitions that define the organization of the data.

Update – Insertion, modification, and deletion of the actual data.

Retrieval – Providing information in a form directly usable or for further processing by other applications. The retrieved data may be made available in a form basically the same as it is stored in the database or in a new form obtained by altering or combining existing data from the database.

Administration – Registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information that has been corrupted by some event such as
an unexpected system failure.

Both a database and its DBMS conform to the principles of a particular database model. ―Database system refers collectively to the database model, database management system, and database.

Physically, database servers are dedicated computers that hold the actual databases and run only the DBMS and related software. Database servers are usually multiprocessor computers, with generous memory and RAID disk arrays used for stable storage. RAID is used for recovery of data if any of the disks fail. Hardware database accelerators, connected to one or more servers via a high-speed channel, are also used in large volume transaction processing environments. DBMSs are found at
the heart of most database applications. DBMSs may be built around a custom multitasking kernel with built-in networking support, but modern DBMSs typically rely on a standard operating system to provide these functions from databases before the inception of Structured Query Language (SQL). The data recovered was disparate, redundant and disorderly, since there was no proper method to fetch it and arrange it in a concrete structure.

2.14.2 DB Languages

Database languages are special-purpose languages, which do one or more of the following:

- Data definition language – defines data types and the relationships among them.
- Data manipulation language – performs tasks such as inserting, updating, or deleting data occurrences.
- Query language – allows searching for information and computing derived information.

Database languages are specific to a particular data model. Notable examples include: SQL combines the roles of data definition, data manipulation, and query in a single language. It was one of the first commercial languages for the relational model, although it departs in some respects from the relational model as described by Codd (for example, the rows and columns of a table can be ordered). SQL became a standard of the American National Standards Institute (ANSI) in 1986 and of the International Organization for Standardization (ISO) in 1987. The standards have been regularly enhanced since and are supported (with varying degrees of conformance) by all mainstream commercial relational DBMSs.

OQL is an object model language standard (from the Object Data Management Group). It has influenced the design of some of the newer query languages like JDOQL and EJB QL. XQuery is a standard XML query languages implemented by XML database systems such as MarkLogic and eXist, by relational databases with XML capability such as Oracle and DB2, and also by in-memory XML processors such as Saxon. SQL/XML combines XQuery with SQL.

A database language may also incorporate features like:

- DBMS-specific Configuration and storage engine management: Computations to modify query results, like counting, summing, averaging, sorting, grouping, and cross-referencing

Constraint enforcement (e.g. in an automotive database, only allowing one engine type per car).

2.14.3 ORACLE

Oracle Corporation is an American multinational computer technology corporation, headquartered in Redwood Shores, California. The company primarily specializes in developing and marketing database software and technology, cloud engineered systems and enterprise software products—particularly its own brands of database management systems. In 2015, Oracle was the second-largest software maker by revenue, after Microsoft.

The company also develops and builds tools for database development and systems of middle-tier software, enterprise resource planning (ERP) software, customer relationship management (CRM) software and supply chain management (SCM) software.

- **Advantages of Choosing Oracle Database**

1) Portability

   Oracle is ported to more platforms than any of its competition, running on more than 100 hardware platforms and 20 networking protocols. This makes writing an Oracle application fairly safe from changes of direction in hardware and operating system, and therefore a safe bet. One caveat, however, is that applications using some constructs (such as field level triggers) may have to be reworked when porting them to a block mode environment. This enables application developers to reduce software development costs and bring products to market quickly and enables user to upgrade hardware while retaining their software applications and minimizing conversation costs.

2) Market Presence

   Oracle is by far the largest RDBMS Vendor, and spends more on R&D than most of its competitors earn in total revenue. Oracle has the largestindependent RDBMS market share in VMS, UNIX and OS/2 Server fields. This market clout means that you are unlikely to be left in the lurch by Oracle and there are always lots of third party interfaces supported and also, proficient staff is relatively easy to get.

3) Version Changes

   Oracle seem very good at informing you in detail as to what is not going to be supported in the next major release and usually have some knob you can twiddle for good backward compatibility, or simply leave it working, but with some warnings in the manual. Backward compatibility is very good meaning you will not be in for an application re-write when you upgrade the DBMS. (Compare this with the Ingres v5->6 OSQL upgrade from hell.) I've worked with Oracle since V4 Beta and have never been in for nasty surprises as far as syntax go. However, see ―Version Changes‖ under disadvantages.

4) Backup and Recovery

   Oracle provides industrial strength support for on-line backup and recovery and good software fault tolerance to disk failure. You can also do point-in-time recovery. Of course, you need the archive mechanisms and storage space to do this, but Oracle supports continuous archiving to tape devices spanning multiple volumes.

5) Performance

Speed of a tuned Oracle database and application is quite good, even with large databases. Oracle refers to >100 GB databases and have personal experience administering 10 GB databases. The performance is not only ―raw‖, but includes consideration of performance with locking and transaction control.

6) Cursor Support

Oracle, like Ingres, but unlike Sybase (until Release 10 I think), supports cursors which ease programming when performance is needed. A cursor basically lets you do row-by-row processing. Oracle supports multiple cursors per Oracle connection in line with ANSI standards.

7) SQL Dialect

The dialect of SQL offered by Oracle is in my opinion superior to the others in the extensions it offers over ANSI-2, which is very much a lowest common denominator. Constructs such as the absolute function and decode keyword are very powerful Oracle additions to the standard SQL.

8) Multiple Database Support

Oracle has a superior ability to manage multiple databases within the same transaction using a two-phase commit protocol. This is best implemented in V7. You can fairly easily move where data is actually stored from node to node in a network and have data mirroring, making it easy to optimize the location of the data from time to time. This is not so easily done with offerings from other vendors or earlier versions of Oracle, where you were not able to update more than one database in the one transaction with any reliability. This meant that you could not move data around without recoding the programs. With V7, the DBA can optimize the location without pre-planning by programmers or re-examination of the code prior to the move.

9) PL/SQL

PL/SQL, the procedural extensions, is a draft ANSI standard for procedural
DBMS languages.

10) Declarative Integrity

Oracle V7 onwards supports declarative database integrity (the current ANSI standard) and V6 permits you to enter the declarations.

With V6, you can get the toolset to read the declarations and automatically generate the required code. With V7 onwards, not even this is required, as the database engine automatically enforces the integrity. This means that you can open up the database to end-users through simple third party interfaces as they simply cannot break the business rules even if they try. It makes it easy to administer changes in business and data rules as there is only one spot where the change needs to be made. This lowers the cost of required modification to the system because it does not have to edit all applications code that works with the table.

2.14.4 Microsoft Excel

Microsoft Excel is a spreadsheet developed by Microsoft for Windows, macOS, Android and iOS. It features calculation, graphing tools, pivot tables, and a macro programming language called Visual Basic for Applications. It has been a very widely applied spreadsheet for these platforms, especially since version 5 in 1993, and it has replaced Lotus 1-2-3 as the industry standard for spreadsheets. Excel forms part of Microsoft Office.

2.14.5 Microsoft Access

Microsoft Access is a database management system (DBMS) from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software-development tools. It is a member of the Microsoft Office suite of applications, included in the Professional and higher editions or sold separately.

Microsoft Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other applications and databases. Software developers, data architects and power users can use Microsoft Access to develop application software. Like other Microsoft Office applications, Access is supported by Visual Basic for Applications (VBA), an object-based programming language that can reference a variety of objects including DAO (Data Access Objects), ActiveX Data Objects, and many other ActiveX components. Visual objects used in forms and reports expose their methods and properties in the VBA programming environment, and VBA code modules may declare and call Windows operating system operations.

2.15 Python

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library.

Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

2.15.1 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and UNIX shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

2.15.2 Features of Python

- Python is Interpretable: Python is processed at runtime by the interpreter. You do not need to compile the program before executing it. This is similar to PERL and PHP.
- A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases: Python provides interfaces to all major commercial databases.GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of UNIX.
- Scalable: Python provides a better structure and support for large programs than shell scripting.
- Apart from the above-mentioned features, Python has a big list of good features, few are listed below:
  1. It supports functional and structured programming methods as well as OOP.
  2. It can be used as a scripting language or can be compiled to byte-code for building large applications.
  3. It provides very high-level dynamic data types and supports dynamic type checking.
  4. IT supports automatic garbage collection.
  5. It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

2.16 Linear Discriminant Analysis

The basic idea of LDA is grouping of similar classes of data whereas PCA works directly on data. It seeks to find directions along which the classes are best separated. In facts, PCA is used as a basic step for dimensionality reduction and removal of the null spaces of the two scatter matrices. Then LDA is performed in the lower dimensional PCA subspace as it was done for example in Fisher faces. It is concluded that the discarded null spaces can contain significant discriminatory information. To avoid this direct LDA (D-LDA) methods have been introduced. The basic premise behind the D-LDA approaches is that the information residing in the null space of the within-class scatter matrix is more significant for discriminate tasks than the information out of the null space.

Linear discriminant analysis (LDA) is a generalization of Fisher's linear discriminant, a method used in statistics, pattern recognition and machine learning to find a linear combination of features that characterizes or separates two or more classes of objects or events. The resulting combination may be used as a linear classifier or, more commonly, for dimensionality reduction before later classification.

LDA is closely related to analysis of variance (ANOVA) and regression analysis, which also attempt to express one dependent variable as a linear combination of other features or measurements. However, ANOVA uses categorical independent variables and a continuous dependent variable, whereas discriminant analysis has continuous independent variables and a categorical dependent variable (i.e. the class label). Logistic regression and probity regression are more similar to LDA than ANOVA is, as they also explain a categorical variable by the values of continuous independent variables. These other methods are preferable in applications where it is not reasonable to assume that the independent variables are normally distributed, which is a fundamental assumption of the LDA method.

LDA is also closely related to principal component analysis (PCA) and factor analysis in that they both look for linear combinations of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data. PCA on the other hand does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities. Discriminant analysis is also different from factor analysis in that it is not an interdependence technique: a distinction between independent variables and dependent variables (also called criterion variables) must be made.

- **Applications of LDA**

1) Bankruptcy Prediction

In bankruptcy prediction based on accounting ratios and other financial variables, linear discriminant analysis was the first statistical method applied to systematically explain which firms entered bankruptcy vs. survived. Despite limitations including known nonconformance of accounting ratios to the normal distribution assumptions of LDA, Edward Altman's 1968 model is still a leading model in practical applications.

2) Face Recognition

In computerized face recognition, each face is represented by a large number of pixel values. Linear discriminant analysis is primarily used here to reduce the number of features to a more manageable number before classification. Each of the new dimensions is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher faces, while those obtained using the related principal component analysis are called eigenfaces.

3) Marketing

In marketing, discriminant analysis was once often used to determine the factors which distinguish different types of customers and/or products on the basis of surveys or other forms of collected data. Logistic regression or other methods are now more commonly used.

4) Biomedical Studies

The main application of discriminant analysis in medicine is the assessment of severity state of a patient and prognosis of disease outcome. For example, during retrospective analysis, patients are divided into groups according to severity of disease – mild, moderate and severe form. Then results of clinical and laboratory analyses are studied in order to reveal variables which are statistically different in studied groups. Using these variables, discriminant functions are built which help to objectively classify disease in a future patient into mild, moderate or severe form.

5) Earth Science

This method can be used to separate the alteration zones. For example, when different data from various zones are available, discriminate analysis can find the pattern within the data and classify it effectively.

2.17 Face Databases

While there are many databases in use currently, the choice of an appropriate database to be used should be made based on the task given (aging, expressions, lighting etc.). Another way is to choose the data set specific to the property to be tested (e.g. how algorithm behaves when given images with lighting changes or images with different facial expressions). If, on the other hand, an algorithm needs to be trained with more images per class (like LDA), Yale face database is probably more appropriate than FERET.

2.17.1 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).

2.17.2 Yale Face Database

It is also known as Yalefaces. The AT&T Face database is good for initial tests, but it‘s a fairly easy database. The Eigenfaces method already has a 97% recognition rate on it, so you won‘t see any great improvements with other algorithms. The Yale Face database (also known as Yalefaces) is a more appropriate dataset for initial experiments, because the recognition problem is harder. The database consists of 15 people (14 male, 1 female) each with 11 grayscale images sized 320 \times 243 pixel. There are changes in the light conditions (center light, left light, right light), facial expressions (happy, normal, sad, sleepy, surprised, wink) and glasses (glasses, no-glasses).

2.17.3 Extended Yale Face Database

The Extended Yale Face database B contains 2414 images of 38 different people in its cropped version. The focus of this database is set on extracting features that are robust to illumination; the images have almost no variation in emotion/occlusion.

2.18 Aims and Objectives

Aim

To design and implement Online Payment System using Face Recognition Technique.

Objective

- To detect faces using Haar Cascade Classifier.
- To understand Viola and Jones Features.
- To understand the concept of Face Recognition using machine learning.
- To identify faces using Local Binary Pattern Histogram.
- To implement Online Payment System using Face Recognition.

References

1. ACI Committee 229. (1994). “Controlled low-strength materials (CLSM), ACI 229R-94.”*Concrete International*, 16(7), 55-64.
2. BASF (1990). “Guidelines for use of EPS board products in building construction”, *Technical Bulletin E-I Parsippany, N J, USA,65(02), 96-104.*
3. Chandra S. and Choudhary R. (2013). “Performance characteristics of bituminous concrete with industrial wastes as filler*”, Journal of Materials in Civil Engineering,* ASCE, 25(11), 1666-1673.
4. Kaushik S., Dubey R. B. and Madan A. (2014). “Study of Face Recognition Techniques”, *International Journal of Advanced Computer Research*, 4(4), 123-128.
5. Neelesh S Salian., Priyank Patel., Shrenik S. and Kavita Sonawane (2012). “Automated Online payment System using Fuzzy Logic and Content based Image Retrieval”, *International Journal of Computer Applications*, 37(12), 0975 – 8887.
6. Kanti J. and Papola A. (2014). “Smart Online payment using Face Recognition with Percentage Analyzer”, *International Journal of Advanced Research in Computer and Communication Engineering*, 19(08), 224-236.
7. Srivastava S. and Jain D. C. (2016). “A Review on Face Recognition Online payment System”, *International Journal of Computer Applications*, 143(8), 0975-8884.
8. Turk M. A. and Pentland A. P. (1991). “Face Recognition Using Eigenfaces” , *IEEE Conference on Computer Vision and Pattern Recognition*, 35(01), 581-591.
9. Li X. and Areibi S. (2004). “A Hardware/Software Co-design Approach for Face Recognition”, *16th International Conference on Microelectronics*, 12(03), 12-19.
10. Tunisia. Roshan., Tharanga J. G., Samarakoon S. M. S. C., Karunarathne T. A. P., Liyanage K. L. P. M., Gamage M. P. A. W. and Perera D. (2013). “Smart Online payment Using Real Time Face Recognition”, *SAITM Research Symposium on Engineering Advancements*, 4(1), 41-43.
11. John C. Russ. (1995). “Image Processing Handbook”, *CRC Press*, 10(3), 44-49.

12. Pandya J. M., Rathod D. and Jadhav J. (2013). "A Survey of Face Recognisation approach", *International Journal of Engineering Research and Applications* (IJERA), 3(1), 632-635.

13. Yang M., Kriegman D. J. and Ahuja N. (2009). "Detecting Faces in Images," *The IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1), 1-4.

14. Javed M. and Gupta B. (2003). "Performance Comparison of the Various Face Detection Techniques", *International Journal of Scientific Research Engineering and the Technology* (IJSRET), 2(1), 19-27

15. Wiskott L., Fellous T. and Kriiger N. (2014). "Face Recognisation by Elastic Bunch Graph Matching", *IEEE Transactions on pattern analysis and machine intelligence*, 7 (01), 775-779.

16. Kirby M. and Sirovich L. (1990). "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Trans. Pattern. Anal. Intell.* 12(03), 49-78.

17. Jaiswal S., Bhadauria S. and Jadon R. (2016). "Comparison between Face Recognisation Algorithm - EigenFaces, FisherFaces and Elastic Bunch Graph Matching," *The Journal of Global Research in Computer Science*, 2(7), 20-25.