Indira Gandhi National Open University

MCS-230
Digital Image Processing and Computer Vision

Block

4

COMPUTER VISION-II

PROGRAMME DESIGN COMMITTEE

Prof. (Retd.) S.K. Gupta , IIT, Delhi
Prof. Ela Kumar, IGDTUW, Delhi
Prof. T.V. Vijay Kumar JNU, New Delhi
Prof. Gayatri Dhingra, GVMITM, Sonipat
Mr. Milind Mahajan,. Impressico Business Solutions, New Delhi
Sh. Shashi Bhushan Sharma, Associate Professor, SOCIS, IGNOU
Sh. Akshay Kumar, Associate Professor, SOCIS, IGNOU
Dr. P. Venkata Suresh, Associate Professor, SOCIS, IGNOU
Dr. V.V. Subrahmanyam, Associate Professor, SOCIS, IGNOU
Sh. M.P. Mishra, Assistant Professor, SOCIS, IGNOU
Dr. Sudhansh Sharma, Assistant Professor, SOCIS, IGNOU

COURSE DESIGN COMMITTEE

Prof. T.V. Vijay Kumar, JNU, New Delhi
Prof. S.Balasundaram, JNU, New Delhi
Prof. D.P. Vidyarthi, JNU, New Delhi
Prof. Anjana Gosain, USICT, GGSIPU, New Delhi
Dr. Ayesha Choudhary, JNU, New Delhi
Sh. Shashi Bhushan Sharma, Associate Professor, SOCIS, IGNOU
Sh. Akshay Kumar, Associate Professor, SOCIS, IGNOU
Dr. P. Venkata Suresh, Associate Professor, SOCIS, IGNOU
Dr. V.V. Subrahmanyam, Associate Professor, SOCIS, IGNOU
Sh. M.P. Mishra, Assistant Professor, SOCIS, IGNOU
Dr. Sudhansh Sharma, Assistant Professor, SOCIS, IGNOU

SOCIS FACULTY

Prof. P. Venkata Suresh, Director, SOCIS, IGNOU
Prof. Sandeep Singh Rawat, SOCIS, IGNOU
Dr. Akshay Kumar, Associate Professor, SOCIS, IGNOU
Dr. Sudhansh Sharma, Assistant Professor, SOCIS, IGNOU
Prof. V.V. Subrahmanyam, SOCIS, IGNOU
Prof. Divakar Yadav, SOCIS, IGNOU, New Delhi
Dr. M.P. Mishra, Associate Professor, SOCIS, IGNOU

COURSE PREPARATION TEAM

This Course is adapted from MMTE-003–Digital Image Processing and Pattern Recognition of School of Sciences, IGNOU

Content Editors

Unit 11

Prof. S. Balasundaram
School of Computers and System Sciences,
JNU, New Delhi

Unit 12 and Unit 13

Prof. K.K. Biswas (Retd.)
IIT Delhi

Course Writer

Unit 11

Prof. S. Indu, Delhi Technological University, New Delhi
Prof. Vipula Singh, RNSIT, Banglore

Unit 12 and Unit 13

Prof. Vipula Singh
RNSIT, Banglore

COURSE COORDINATOR

Dr.Sudhansh Sharma,
Assistant Professor,
School of Computers and Information Sciences, IGNOU.

PRINT PRODUCTION

Sh Sanjay Aggarwal
Assistant Registrar, MPDD, IGNOU, New Delhi

June, 2023

Further information on the Indira Gandhi National Open University courses may be obtained from the University's office at Maidan Garhi, New Delhi-110068.
Printed and published on behalf of the Indira Gandhi National Open University, New Delhi by MPDD, IGNOU.
Laser Typesetter: Tessa Media & Computers, C-206, Shaheen Bagh, Jamia Nagar, New Delhi-110025

BLOCK 4 INTRODUCTION

This block deals with Object detection, Object recognition, and Object classification techniques for images.

In Unit 11, deals with the image segmentation which includes the detection of edge, line, boundary and region. Various edge and line detection algorithms are discussed. Various techniques of region based segmentation are discussed along with boundary detection algorithms.

Unit12 discusses the Object recognition using Supervised Learning Approaches, which includes various image classifiers viz. Bayesian and Minimum distance classifiers, the Linear and non-linear discriminant function are also explained.

Unit 13 relates to Object classification using Unsupervised learning approaches, it starts with explanation of clustering along with the need and applications of clustering. The Hierarchial Clustering and Partition based clustering are discussed in detail.

ignou
THE PEOPLE'S
UNIVERSITY

UNIT 11 OBJECT DETECTION

Structure **Page No.**

11.1 INTRODUCTION

The development of computer vision led to various means of understanding images automatically. One of the important applications is object detection in an image. Using image processing algorithms, we can detect semantic objects of a certain class (like individuals, constructions, animals or vehicles) from digital images or videos.

Mainly, object detection algorithm aims to detect objects from a known class like vehicles, human beings, animals, tree etc. In general, there could be very small number of objects may be present in a single image, but similar class can be present in large number of images with various backgrounds.

Such things are to be explored in object detection for example, Figure 1 shows the output of object detection algorithm within a room and a scene on the road. We can see objects lie bottles, glasses, laptop, chair, bag etc inside the room. Where as in the out-door environment, we can see car, two wheelers bus etc

Figure 1 Object detection

Sometimes we are interested in analysis and interpretation of various features which are parts of the image. Object detection involves Image segmentation, which is the process of finding partitions of an image in the form of groups of pixels which are homogeneous with respect to the feature under consideration. We shall begin this unit by discussing the image segmentation and its applications in Sec. 11.2. In Sec. 11.3, we shall discuss various image segmentation techniques. We shall discuss edge, detection with, region detection and boundary detection their applications in the Secs. 11.4, 11.5 and 11.6 respectively.

And now, we will list the objectives of this unit. After going through the unit, please read this list again make sure you have achieved the objectives.

Objectives

After studying this unit, you should be able to

- define Object Detection/image segmentation techniques
- apply edge based segmentation, line based segmentation;
- apply region based segmentation and boundary detection.

We begin the unit by discussing 'what is image segmentation?' in the following section.

11.2 OBJECT DETECTION

Object detection in an image means, identifying objects based on its features. Human beings identify objects instantaneously with out any effort. In computer vision, identifying objects will be based on object models. Object detection models are based on the characteristic features of the image. The block diagram is shown in Figure. 2

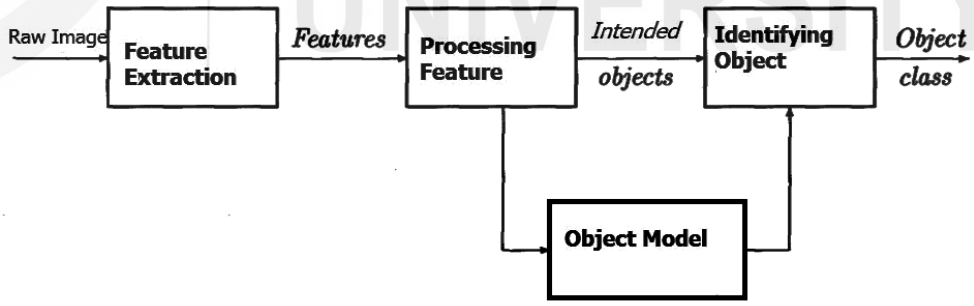


Figure 2 The object detection process

In short, the basic features of object class are to be defined and included in a database of object models. Using feature extraction process, specific features of the object we are looking for are to be identified and matched with the data base for identifying the object class.

We can represent object in multiple ways and accordingly features can be extracted for object identification. The inner region of the object can be represented by some features like gradient, moments, texture etc. Similarly, boundary can be identified based on pattern of pixels, Fourier descriptor etc.

The object detection is majorly classified as (1) Edge Detection, (2) Region Detection and (3) Boundary Detection

Object detection involves Image segmentation which further involves the division of any image into meaningful structures. These structures depict the objects that constitute an image. Segmentation is the first step in image analysis and pattern recognition. Different features in images are identified and extracted by segmentation algorithms. Accuracy of extracted features decide the accuracy of automatic recognition algorithms. Thus, a suitable and rugged segmentation algorithm should be chosen very carefully. Selection of a suitable algorithm is highly application dependent. Image segmentation is one of the most difficult tasks in image processing. Generally, many image processing tasks are aimed at finding a group of pixels in an image that are similar or connected to each other in some way. We are showing the step image analysis, object representation, visualization, understanding and classification in Fig. 3.

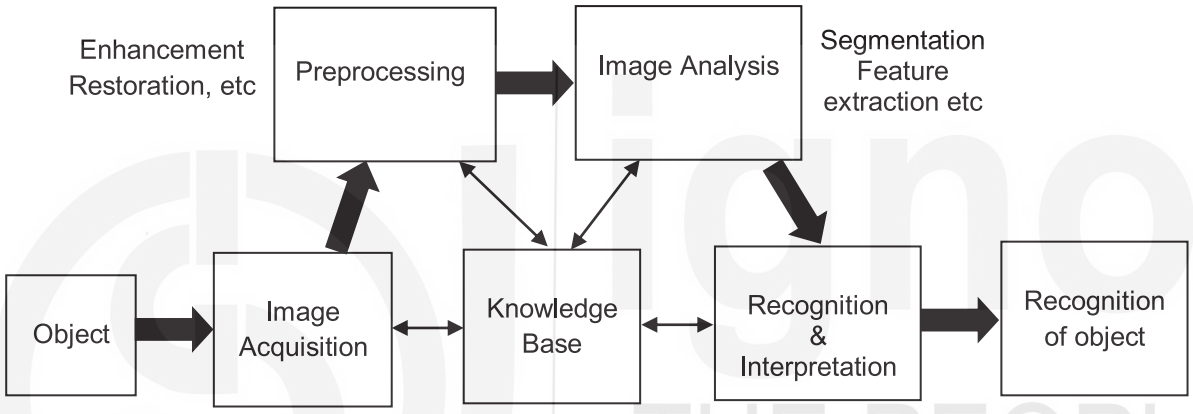


Fig. 3: Image Processing Steps

An image scene is processed for autonomous machine perception and is mapped to knowledge. Here, the system tries to imitate human recognition and ability to make decision according to the information contained in the image, showing in Fig. 4.

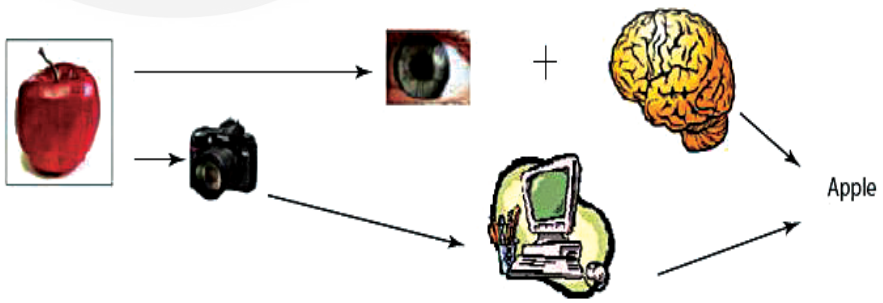


Fig. 4: Image to Knowledge Mapping

Image segmentation is the fundamental step in image analysis, understanding, interpretation and recognition tasks. It is the process of decomposing a scene into different components. Segmentation partitions an image into multiple homogeneous regions with respect to some characteristics. In practice, it groups the pixels having similar attributes into one group. The result of image segmentation is a set of regions that collectively cover the entire image or a set of contours extracted from the image. Each pixel in a particular region is similar to the other pixel with respect to some characteristics such as

colour, edge, texture, etc. Segmentation is an intermediate stage between low level and high level image processing tasks. Low level tasks manipulate pixel values for irregularity correction or for image enhancement, whereas high level tasks manipulate and analyse a group of pixels that convey some information.

Segmentation is the most important step in automated recognition system which has numerous applications and some of them are discussed below:

1. Medical Imaging

Segmentation is used to locate tumors, measure tissue volumes, computer aided surgery, diagnosis, treatment planning, study of anatomical structures etc. Fig. 5 shows the segmented portion of brain.

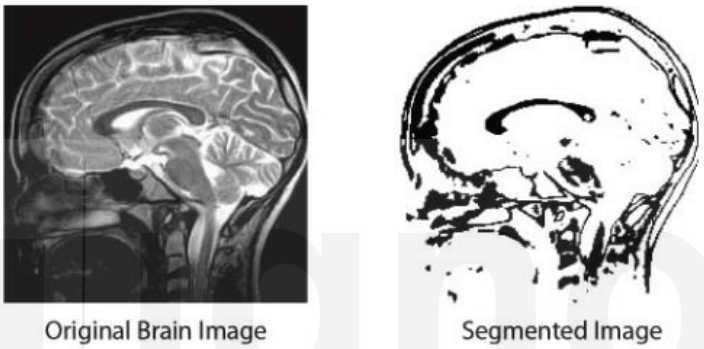


Fig. 5: Example of Image Segmentation in Medical Imaging

2. Satellite Imaging

Segmentation is used to locate objects like roads, forest etc. in satellite images for monitoring of ecological resources like seasonal dynamics of vegetation, deforestation, mapping of underground minerals etc. Segmentation methods based on texture, colour, wavelength etc. are generally used here.

3. Movement Detection

Monitoring crowded urban environment is a goal of modern vision systems. Knowledge of the size of the crowd and tracking its motion can be used to monitor traffic intersection. Intelligent walk signal system can be designed based on the number of people waiting to cross the road. Knowledge of the size of the crowd is helpful in general safety, crowd control and planning urban environment.

4. Security and Surveillance

Security of the national assets such as bridges, dams, tunnels etc is critical in today's world. Automated smart system to detect 'suspicious' movements or activities, to detect left baggage or vehicle is crucial for safely. Automated face detection systems try to match a criminal's face in a crowded place.

5. License Plate Recognition (LPR)

Automated license plate reading is a very useful and practical approach as it helps in monitoring existing and illegally acquired license plates. LPR can be used in private parking management, traffic monitoring,

automatic traffic ticket issuing, automatic toll payment, surveillance and security enforcement. Fig. 6 shows the segmented license plate.

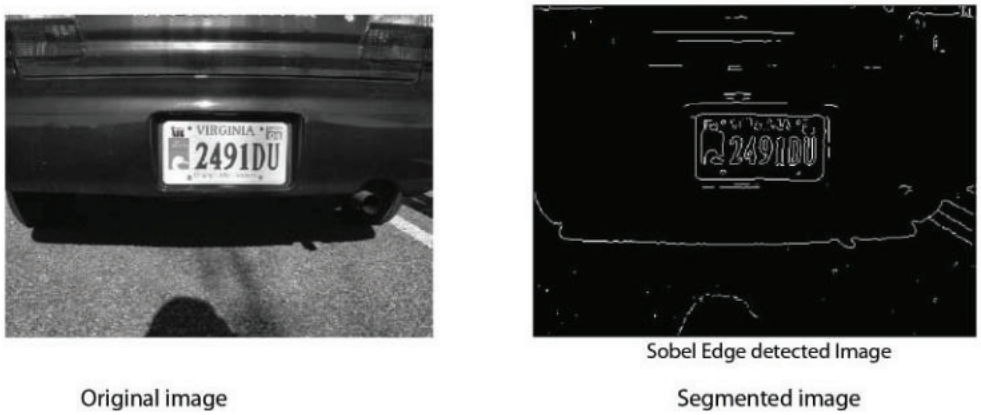


Fig. 6: Example of image segmentation in LPR

6. Industrial Inspection and Automation

Here the objective is to find missing components (Integrated Circuits(IC)), in a Printed Circuit Board (PCB) or a missing pill in a packet of medicine or to check the quality of the baked biscuits etc. These automated systems help in reducing manpower in the plants and in reducing error committed by humans due to repetitive task.

7. Robot Navigation

Robots are used in a variety of industrial, medical and research applications. Pre assigned tasks can be repeatedly and efficiently done by robots in hazardous environment. Image analysis and understanding help robots navigate and execute the given task.

Try the following exercise.

E1) What do you mean by segmentations.

E2) Give two applications of image segmentation techniques.

Now we shall discuss the classification of segmentation in the following section.

11.3 IMAGE SEGMENTATION

Image segmentation helps in simplifying the tasks and goals of computer vision and image processing techniques. Segmenting an image is often considered as the first step for image analysis. Image segmentation is done by splitting an image into multiple parts based on similar characteristics of pixels for identifying objects

There are several techniques through which an image can be segmented, based on division and group-specific pixels which can be further assigned labels, and classified according to these labels. The generated labels, can be used in several supervised, semi-supervised and unsupervised training and testing tasks in machine learning and deep learning applications.

Image segmentation plays a vital importance in computer vision and has

several applications in various industries and research. Some of the commonly used applications are Facial recognition, number plate recognition, image search, analysis of medical image etc

Classification of Image segmentation methods : Researchers are working on image segmentation for over a decade. The commonly used methods involves "Classification based on method of Identification" where the images can be segmented either by grouping similar pixels or by differentiating them by identifying the boundary. The Region based identification method and Boundary based Identification method of segmentation are discussed below:

- **Region based Identification**: In this method, similar pixels are selected based on predefined threshold. Then these selected pixels are grouped together using clustering algorithms like SVM, K-Means, nearest neighbor etc. based on similar attributes or features. These attributes or features can be used for grouping similar pixels for region merging, region growing, region spreading etc.

- **Boundary-based Identification:** In this method separation of regions are done based on the dissimilar features of pixels. The dissimilar pixels are identified using Edge Detection algorithm, Line Detection algorithm etc

Other Image Segmentation methods includes Thresholding Segmentation and the Edge Based Segmentation , the same are discussed below :

1. Thresholding Segmentation

Thresholding based segmentation is considered as the simplest, and easiest method for segmentation. It groups pixels of an image based on a predefined threshold value of pixel intensity. The threshold value of pixel intensity will be determined adaptively on task basis

The threshold value (T) can be a constant value or a dynamic value, based on the application. A constant threshold value helps in differentiating between the backgrounds and segmenting object in an image when the image has less noise. The grey scale image can be converted into a binary image using thresholding technique as shown in the Figure. 7

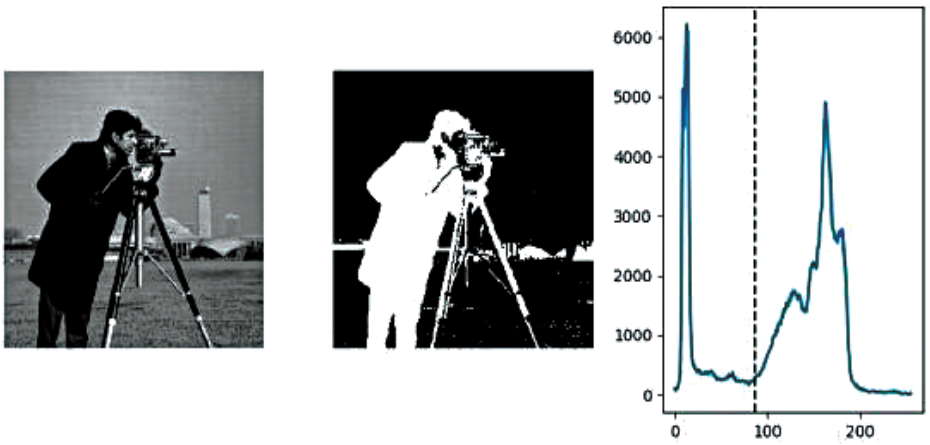


Figure 7

https://scikit-image.org/docs/stable/auto_examples/applications/plot_thresholding.html

(Source: Internet)

The thresholding based segmentation can be further classified as :

1. *Simple Thresholding ,and*
2. *Adaptive Thresholding*

Simple Thresholding : In the Simple Thresholding method (also known as global thresholding), all pixels will be converted into white or black based on the reference pixel intensity value. If the intensity value is less than the reference (threshold value), the pixel will be converted into black and if it is greater, the pixel will be converted into white pixel

Algorithm

1. Initial estimate of T
2. Segmentation using T: G1, pixels brighter than T, G2, pixels darker than (or equal to) T.
3. Computation of the average intensities m1 and m2 of G1 and G2.
4. New threshold value: $T_{new} = (m1 + m2)/2$
5. If $|T - T_{new}| > \Delta T$,
6. back to step 2, otherwise stop.

Global thresholding, using an appropriate threshold T: g(x, y)

= 1, if f (x, y) > T

0, if f (x, y) ≤ T

- **Otsu's Binarization**

In general, a constant threshold value is often chosen through a hit-and-trial method to perform simple thresholding-based segmentation. This constant threshold value can vary according to the application and may not be robust or efficient for two different applications. In Otsu's Binarization method, the threshold value will be decided by the average value of the two peaks obtained from histogram. The main limitation is that it will work only in bimodal image, i.e., images containing only two peaks in a histogram. The main application of this method is scanning documents, Removing unwanted colors, Pattern recognition etc.

Algorithm

1. Otsu's method is aimed in finding the optimal value for the global threshold.
2. It is based on the interclass variance maximization.
3. Well thresholded classes have well discriminated intensity values.
4. M × N image histogram: L intensity levels, [0, . . . , L − 1]
5. n_i - number of pixels of intensity i

$$MN = \sum_{i=0}^{L-1} n_i$$

6. Normalized histogram:

$$p_i = \frac{n_i}{MN}$$

$$\sum_{i=0}^{L-1} p_i = 1, \quad p_i \geq 0$$

7. calculate the between-class variance value
8. the final threshold is the maximum between-class variance value

Example -

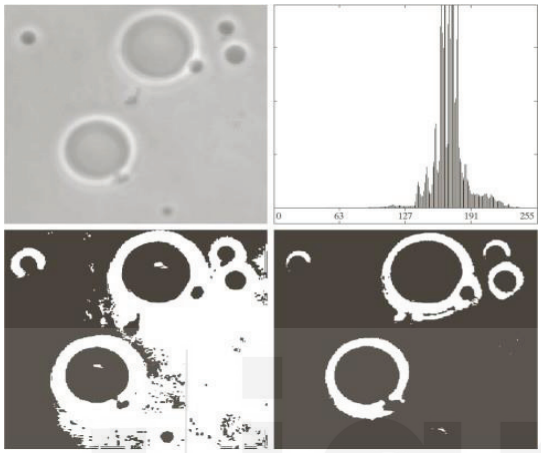


Figure 8

https://www.researchgate.net/publication/263608069_Medical_Image_Segmentation_Methods_Algorithms_and_Applications/figures?lo=1&utm_source=google&utm_medium=organic

a) Original Image
b) Histogram of the image a)
c) Simple threshold, taking T = 0.169 , η = 0.467
d) Otsu's method, taking T = 181, η = 0.944.

Adaptive Thresholding

In this method, we can decide different threshold values for different sections of the image. It is mainly used for images having different backgrounds / properties. Using this method, different lighting conditions can be differentiated and threshold values can be adaptively decided.

Variable Thresholding, if T can change over the image.

- Local or regional thresholding, if T depends on a neighborhood of (x, y).
- Adaptive thresholding, if T is a function of (x, y).
- Multiple thresholding: g(x, y) = a, if f (x, y) > T2

  b, if T1 < f (x, y) ≤ T2

  c, if f (x, y) ≤ T1

After Thresholding Segmentation, the next is Edge Based Segmentation, which is discussed below:

2. **Edge-Based Segmentation**

In this method, the objects are identified based on the edge detection. The edge detection is done based on the pixel properties like texture, contrast, colour, saturation, intensity etc. The results of edge-based image segmentation are shown in Figure 9

Figure 9 Source: Internet

There are two commonly used methods for edge detection

- Search-Based Edge method: In this method, edge detection is done based on edge strength. It is calculated based on the local directional maxima of the gradient magnitude through a computed estimate of the edge's local orientation
- Zero-Crossing Based Edge method: In this method, the edges are identified based on the zero crossings in a derivative expression retrieved from the image. Most popular edge detection methods like Canny, Prewitt, Deriche, and Roberts cross,use the same procedure. The local edges will be grouped to segment the image with subsequent binarization. The detected edge should overlap binary image for object detection.

Marr and Hildreth edge detector (Laplacian of Gaussian)

- The Laplacian is sometimes used on its own for edge detection because its sensitive to noise
- The Laplacian-of-Gaussian (LoG) uses a Gaussian filter to blur the image and a Laplacian to enhance edges.
- Edge localisation is done by finding zero crossings.

Algorithm

- Convolve the image with a two-dimensional Gaussian function
- Compute the Laplacian of the convolved image -> L
- Identify edge pixels as those for which there is a zero-crossing in L. A radially-symmetric 2D Gaussian:

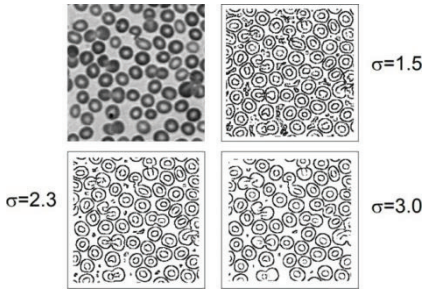
Where $r^2 = x^2 + y^2$

$$G_\sigma(x,y) = e^{\left(\frac{-r^2}{2\sigma^2}\right)}$$

The Laplacian of this is :

$$G(x,y) = \left(\frac{r^2-\sigma^2}{\sigma^4}\right)e^{\left(\frac{-r^2}{2\sigma^2}\right)}$$

- Example



Canny edge detector

The Canny edge detector works on the fact that for edge detection, there is a tradeoff between noise reduction (smoothing) and edge localisation.

Algorithm

- Smooth the image with a Gaussian filter
- Compute the gradient magnitude and orientation
- Apply non-maximal suppression to the gradient magnitude image
- Use hysteresis thresholding to detect and link edges

3. **Region-Based Segmentation**

This algorithm, identify group of pixels with specific characteristics either from a small section or from a bigger portion of input images seed point. Then the algorithm will add more pixels or shrink based on specific characteristics of pixels with all other seed points. Thus, we can get a segmented image. It is further classified into Region Growing and Region Splitting methods

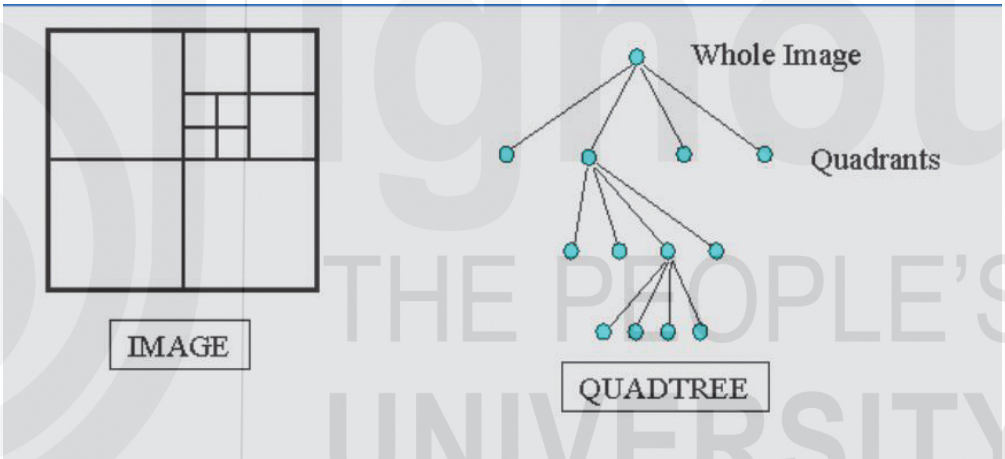


Figure :10

https://www.doc.ic.ac.uk/~dfg/vision/v02d01.gif

(Source:Internet)

- **Region Growing**

In this method, the pixels are merged according to particular similarity conditions wherein at first, small set of pixels are grouped together and merged. This phenomenon is continued iteratively until all the pixels are ground and merged to one another. Basically, the algorithm picks up pixel randomly and finds out matching neighbouring pixels and adds them and continue the same till finding out a dissimilar pixel. After that it will find out another seed point and it will continue.

To avoid overfitting, these algorithms will grow multiple region simultaneously. Such algorithms will work even with noisy images

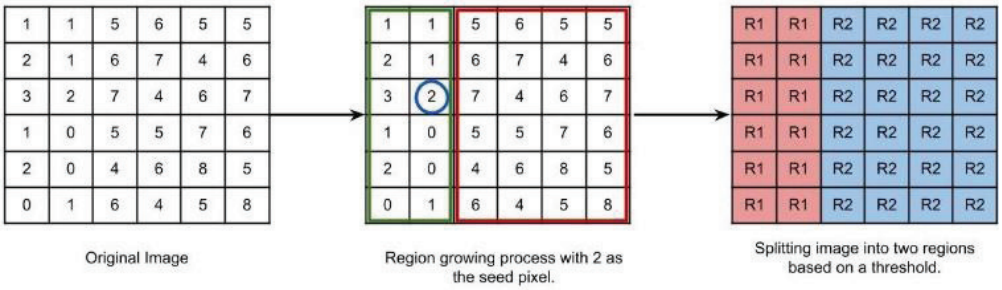


Figure. 11

https://towardsdatascience.com/image-segmentation-part-2-8959b609d268

Source: Internet

- **Region Splitting and Merging**

This algorithm focuses on splitting and merging portions of the image. It splits image based on attributes and then merge regions based on similar attributes. While splitting the whole image will be considered and while region growing it will be concentrating specific points. This algorithm is also called as split-merge algorithms

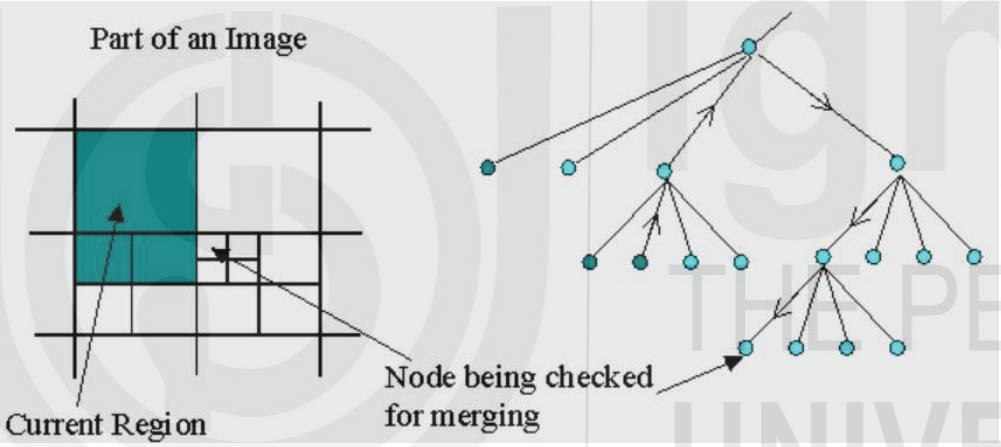
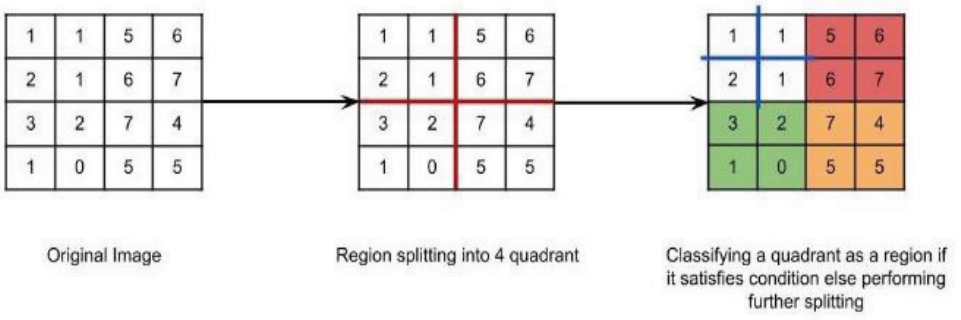


Figure. 12

https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic3.htm

Pictorial representation of how Split and Merge algorithm works



https://towardsdatascience.com/image-segmentation-part-2-8959b609d268

Figure. 13

$|Z_{max} - Z_{min}| <= threshold$

Z_{max} - Maximum pixel intensity value in a region Z_{min} - Minimum pixel intensity value in a region

Some properties that must be followed in region based segmentation -

- Completeness - The segmentation must be complete i.e, Σ Ri = R Every pixel must be in a region
- Connectedness - The points of a region must be connected
- Disjointness - Regions must be disjoint: Ri∩Rj =Ø, for all i = 1,2,...n
- Satisfiability-Pixels of a region must satisfy one common P(Ri) TRUE, for all i property P at least, i.e any region must satisfy a homogeneity predicate P
- Segment ability - Different regions satisfy different P(Ri U Rj) = FALSE properties i.e any two adjacent regions cannot be merged into a single region

Example - Segment the image given by split and merge algorithm

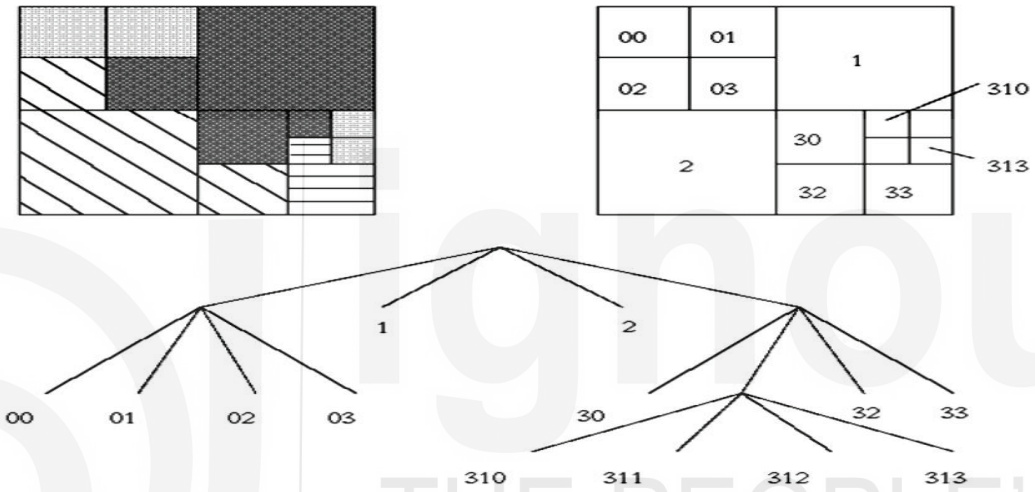


Figure. 14

https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MARBLE/medium/segment/split.htm

3. Neural Networks for Segmentation

Neural networks have become very popular for image segmentation tasks as they not only provide automation, but also provide robust specificity and sensitivity to the developed segmentation masks. Researchers have introduced and analyzed convolutional neural network, generative adversarial networks, deep belief network, and extreme learning machines etc., to perform excellent image segmentation for various applications in healthcare, traffic monitoring, satellite visualization, bakery, plant diseases, etc.

Segmentation through neural networks, especially a convent is done by generating a feature map for an input image data. Then a region based filter is further applied to generate the mask according to ones objectives and applications. Bounding boxes play a very important role in image segmentation. They can be generated through various techniques and consist of coordinates from the segmented part.

A great example of neural network for image segmentation has been released by the experts at Facebook AI Research (FAIR) who created a deep learning architecture called Mask R-CNN which can make a pixel-wise mask for every object present in an image. It is an enhanced version of the Faster R-CNN object detection architecture. The Faster R-CNN uses two pieces of data for every object in an image, the bounding box coordinates and the class of the

object. With Mask R-CNN, you get an additional section in this process. Mask R-CNN outputs the object mask after performing the segmentation.

11.3.1 Image Segmentation Techniques

Image segmentation partitions an image into set of regions. In many applications, regions represent meaningful areas in an image. In other applications, regions might be set of border pixels grouped into structures such as line segments, edges etc. The level of partitioning of an image depends on the problem to be solved. Segmentation should stop when the objects of interests have been isolated. Segmentation has two objectives:

a) To decompose an image into regions for further analysis.

b) To perform a change of representation of an image for faster analysis.

There are a number of segmentation techniques available. Segmentation is application dependent. A single segmentation technique may not be suitable for different applications. Hence, these techniques have to be combined with the domain knowledge in order to effectively solve the problem. Generally, all segmentation algorithms are based on one of the two properties of gray level values of pixels.

a) Discontinuity

In this approach, images are partitioned based on the difference or discontinuity of the gray level values. This means segmentation contour starts from the point where pixels appear to have very different gray values. Edge based segmentation methods fall in this category.

b) Similarity

Images are partitioned based on the similarity of the gray level values of the pixel according to a pre-defined criterion. This means as long as pixels have gray values close to each other, they are considered to be in the same segmentation block. Thresholding, region based clustering and matching based segmentation techniques fall in this category.

We categorize segmentation techniques as follows:

i) Edge Based Segmentation: Edge based method is a commonly used technique to detect boundaries and discontinuities in an image. With this technique, detected edges are assumed to represent object boundaries and used to identify objects. The assumption here is that every part of the object is sufficiently uniform such that they can be separated on the basis of discontinuity alone.

ii) Region Based Segmentation: Edge based techniques find the object boundaries and then locate the objects, whereas region based techniques look for uniformity within a sub-region based on a suitable property like intensity, colour, texture etc. Region based segmentation starts in the middle of an object and then 'grows' outwards till it meets the object boundary.

Try an exercise.

E3) Segmentation algorithms are generally based on which two properties of intensity?

In the following section, we shall discuss the various techniques of image segmentation.

11.4 EDGE DETECTION – EDGE BASED SEGMENTATION

As object can be fully represented by its edges, segmentation of an image into separate objects can be achieved by locating edges of these objects. Edge detection is a fundamental tool in image processing and computer vision. The aim is to identify points in an image at which the brightness change sharply. Changes in image brightness can be due to discontinuities in depth, discontinuities in surface orientation, changes in material property or variation in scene illumination. Result of an edge detection algorithm is a set of connected curves that may indicate object boundaries. This significantly reduces the amount of data to be processed, making the overall object representation algorithm simpler and quicker.

If edge detection step is successful, subsequent task of interpreting the information is also successful. A typical approach to segmentation using edges is

a) Compute an edge image, containing all edges of an original image.

b) Process the edge image so that only closed object boundaries remain.

c) Transform the result to an segmented image by filling in the object boundaries.

The first step of edge detection is discussed in the following sections. Difficulty lies in the second step which often requires removal of edges that are caused by noise or other artifacts, bridging the gaps at locations where no edge was detected and decision to connect the edges that make up a single object.

We discussed already discussed on edges in earlier sections. You may recall that an edge may be loosely defined as a line of pixels showing an 'observable' difference. For example, consider two sub-images shown in Fig. 15. In the sub image Fig. 15(b), there is a clear difference between the gray levels in the second and third columns which can be easily picked by the human eye where as in sub-image of Fig.15 (a) no such difference can be seen.

51	52	53	59
54	52	53	62
50	52	53	68
55	52	53	55

(a)

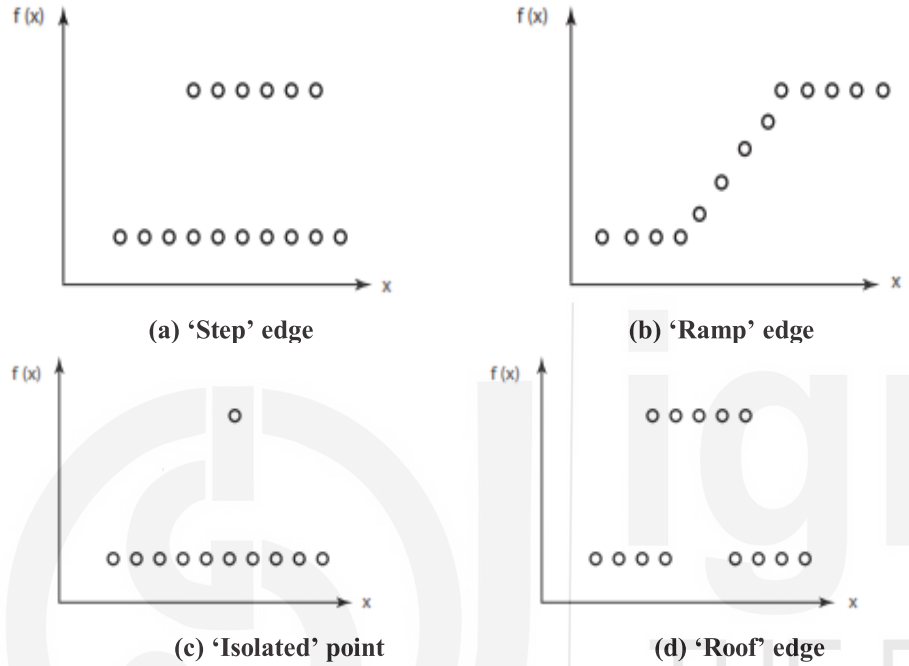
50	53	150	160
51	53	150	180
58	55	154	170
54	56	156	155

(b)

Fig. 15: 2 Sub-Images

Different edge models have been defined based on their intensity profiles. Fig. 16(a) shows a 'step' edge which involves transition between two

intensity values over a distance of one pixel. This is an ideal edge where no additional processing is needed for identification. A 'ramp' edge is shown in Fig. 16(b), where the transition between two intensity levels takes place over several pixels. In practice, all the edges get blurred and noisy because of focusing limitations and inherent noise present in electronic components. A 'point' (Fig. 16(c)) is defined as only one or two isolated pixels having different gray level values as compared to its neighbours. Whereas a roof edge (Fig. 16(d)) is defined as multiple pixels having same or similar gray level values which are different from their neighbours.



(a) 'Step' edge **(b) 'Ramp' edge**

(c) 'Isolated' point **(d) 'Roof' edge**

Fig. 16: Gray Levels Across Edges

As we know that the first order derivative at a point x of a one – dimensional function $f(x)$ is given by

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x) \qquad (1)$$

Second order derivative of $f(x)$ is given by

$$\frac{\partial^2 f}{\partial x^2} = f''(x) = f(x+1) + f(x-1) - 2f(x) \qquad (2)$$

Let the values of the ramp edge in Fig. 16(b) from left to right be

$f(x) =$	20	20	20	20	20	50	100	180	180	180	180	180
first derivative $f'(x) =$ [using Eqn. (1)]	–	0	0	0	0	+30	+50	+80	0	0	0	–
second derivative $f'(x) =$ [using Eqn. (2)]	–	0	0	0	0	30	20	30	−80	0	0	0

Generally, for the implementation of first or second derivative, masks are generated. These masks are convolved with the image to get the result. For 3×3 mask shown in Fig. 17, the output is calculated by

$$g(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x+i, y+j)\, w(i,j) \qquad (3)$$

Where $f(x,y)$ is the image with which mask w is being multiplied.

w(−1,1)	w(0,−1)	w(1,−1)
w(−1,0)	w(0,0)	w(1,0)
w(−1,1)	w(0,1)	w(1,1)

Fig. 17: 3 × 3 mask

Now, before we discuss the edge detection approaches, let us discuss line detection.

A line can be a small number of pixels of a different color or gray level on an otherwise unchanging background. For the sake of simplicity it is assumed that the line is only single pixel thick. Fig. 18 shows line detection masks in various directions. The first mask w_1 in Fig. 18(a) responds strongly to lines (one pixel thick) oriented horizontally and the second mask w_2 in Fig. 18(b) responds to vertical lines. Third w_3 and fourth w_4 in Fig.18(c) and Fig. 18(d) masks respond to lines at 45° and -45° respectively.

−1	−1	−1
2	2	2
−1	−1	−1

(a) Horizontal (w_1)

−1	2	−1
−1	2	−1
−1	2	−1

(b) Vertical (w_2)

−1	−1	−2
−1	2	−1
2	−1	−1

(c) $+45°$ (v (W_3)

2	−1	−1
−1	2	−1
−1	−1	2

(d) $-45°$ (w (W_4)

Fig. 18: Line Detection Mask

Let g_1, g_2, g_3 and g_4 be the response of each of these masks from left to right respectively. If $|g_1| > |g_j|, j = 2,3,4,$ then that point is more likely to be associated with horizontal line. Consider the electronic circuit shown in Fig. 19 (a), the results of applying masks of Fig. 18 (a) to Fig. 18 (d) on this circuit are shown in Fig. 19 (a) to Fig. 19 (g).

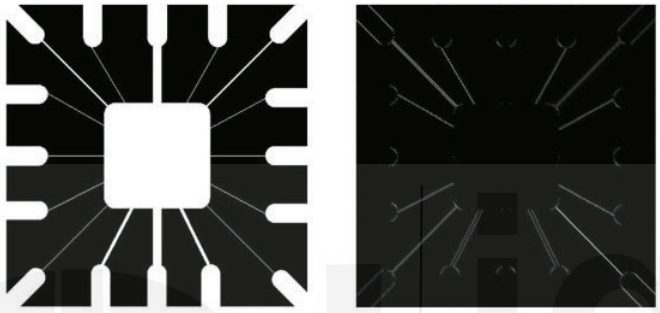
Now let us discuss edge detection approaches.

Edge detection is the most common approach used in segmentation. A typical edge may be the border between a block of red color and a block of yellow. Edge can also be the boundary between two regions with relatively distinct gray-level properties. Computation of a local derivative operator can enhance edges. Fig.20 shows the response of first and second derivative to light strip on dark background (a) and dark strip on light background (b). Edge is the smooth change in grey levels. **First derivative** is positive at the leading edge of transition and negative at the trailing edge of transition. It is zero in areas of constant gray level.

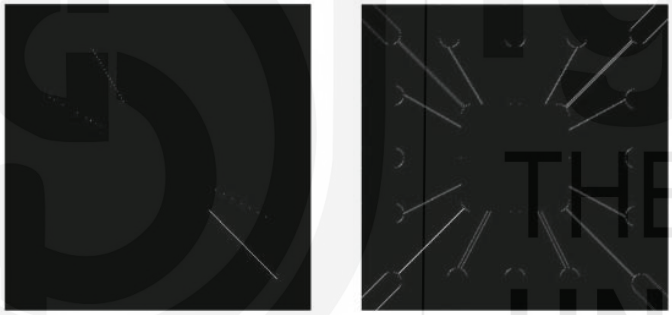
The response of second derivative is different from first derivative. It is positive for dark side, negative for light side and zero in constant area. **'Magnitude' of first derivative** is used to detect the presence of an edge.

'Sign' of second derivative is used to determine whether edge pixel lies on dark side or on light side of edge. '**Zero crossing**' is at the midpoint of a transition in gray level.

Edge detection is a non-trival task. Edge- detection is not as simple as it looks in earlier section. In practice, edges are corrupted by noise and blurring. This can be illustrated by the following examples of edge detection on a one – dimensional array. Looking at Fig. 21(a), we can intuitively say that there is an edge between 4^{th} & 5^{th} pixel. But, if the intensity difference between 4^{th} & 5^{th} pixel is smaller because of noise and if intensity difference between the 5^{th} and 6^{th} pixels is higher as in Fig. 21 (b) it would not be easy to identify the location of the edge precisely. As the edges are corrupted by noise, we observe multiple edges instead of a single edge.

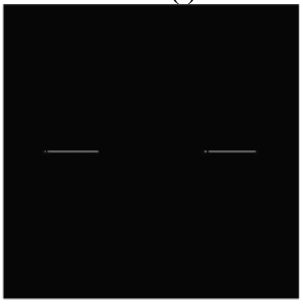


(a) The original image **(b) Results from -45 mask**



(c) -45 line **(d) Result of +45 mask**

(e) +45 line **(f) Result of horizontal mask**



(g) Horizontal line

Fig. 19

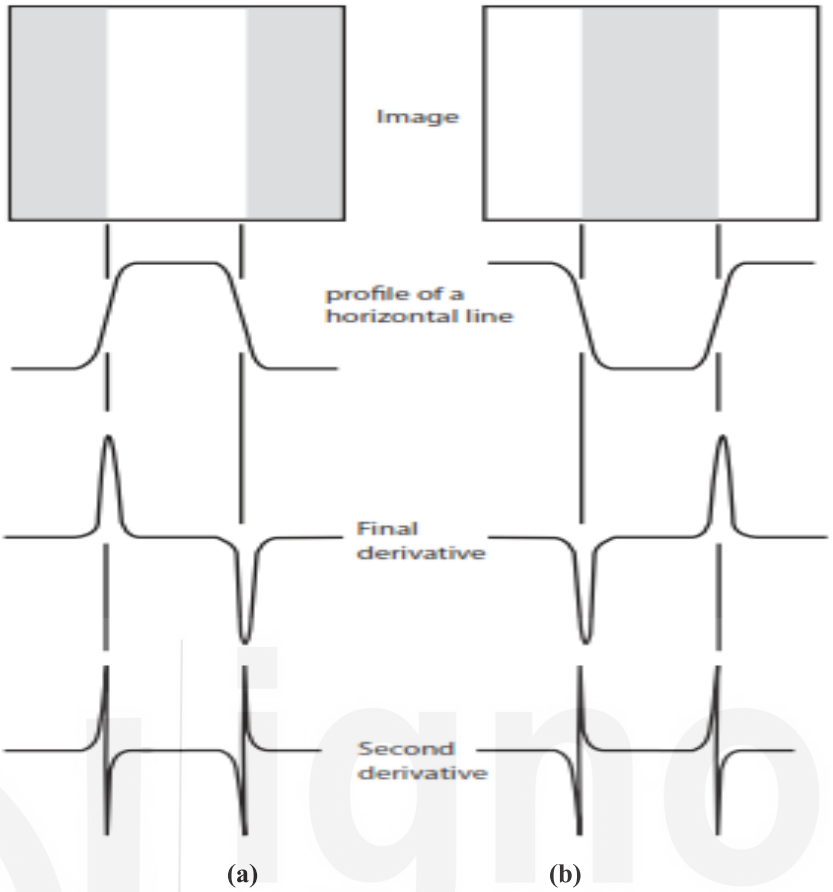


Fig. 20: First and second derivative to a) light stripon dark background, b) dark strip on light background

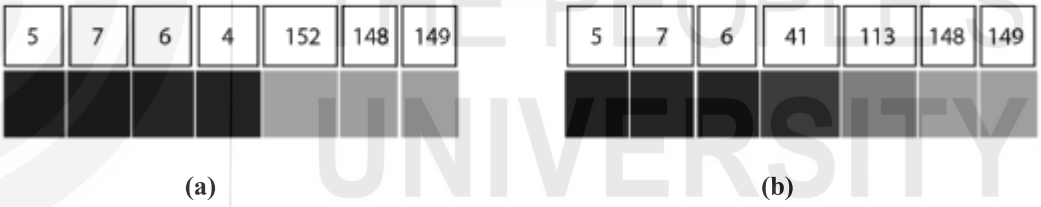


Fig. 21: Example of Edge Detection

11.4.1 Gradient Operator

There are many edge detection methods, broadly, we can classify them into two categories: First order Gradient based search method, and Laplacian based zero crossing method. The first order derivative based search methods detect edges by computing the gradient magnitude and then searching for local directional maxima of the gradient magnitude. The zero crossing based methods search for the zero crossing in a second order derivative computed from the image to find edges.

Gradient is a first order derivative and is defined for an image $f(x, y)$ as

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \qquad (4)$$

It points in the direction of maximum rate of change of f at a point (x, y).

Magnitude of the gradient is $\text{mag}(\nabla f) = \left[G_x^2 + G_y^2\right]^{1/2}$.

This can be approximated as $(\nabla f) = |G_x| + |G_y|$ and direction of the gradient is given by $\alpha(x, y) = \tan^{-1}\left[\frac{G_x}{G_y}\right]$, where α is measured with respect to x-axis.

Now, we present again discussion on various gradicut operators such as, Prewitt operator and Sobel operator.

i) Prewitt Operator

It uses a 3×3 size mask that approximate first derivative. The x direction mask and y direction mask is shown in Fig. 22. The approach used is

$$G_x = (Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)$$
$$G_y = (Z_3 + Z_6 + Z_0) - (Z_1 + Z_4 + Z_7)$$

The convolution results in greatest magnitude indicating gradient direction. These are also called 'Compass operators' because of its ability to determine gradient direction.

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Fig. 22: Prewitt Operator

ii) Sobel Operator

Sobel operator is similar to prewitt's, but it uses a weight of 2 in the centre coefficient to give it little more prominence. Sobel x and y direction mask is given in Fig. 23.

$$G_x = (Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3)$$
$$G_y = (Z_3 + 2Z_6 + Z_0) - (Z_1 + 2Z_4 + Z_7)$$

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & -1 \\ -2 & 0 & 0 \\ -1 & 1 & 1 \end{bmatrix}$$

Fig. 23: Sobel Operator

Sobel operator has the advantage of providing both a derivative and a smoothing effect. This smoothing effect has noise suppression characteristics. Diagonal edges can be detected by prewitt and sobel masks by rotating the earlier masks by 45° counter clockwise. Fig. 24 shows the two masks.

$$\begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

(a) Prewitt **(b) Sobel**

Fig. 24: Prewitt and Sobel Masks for Diagonal Edge Detection.

Now, let us discuss the laplacian operator.

11.4.2 The Laplacian Operator

Laplacian, a second order derivative is defined for a 2D function f(x, y), as

$$\nabla^2 f = \frac{\partial f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Two Laplacian masks are shown in Fig. 25. for Fig. 25 (a), the Laplacian equation is

$$\nabla^2 f = 4Z_5 - (Z_2 + Z_4 + Z_6 + Z_8)$$

And for Fig. 15 (b), the Laplacian equation is

$$\nabla^2 f = 8Z_5 - (Z_1 + Z_2 + Z_3 + Z_4 + Z_6 + Z_7 + Z_8).$$

The centre coefficient of these operators are positive and outer coefficients are negative. Sum of all the coefficients is zero.

0	1	0
1	−4	1
0	1	0

(a)

1	1	1
1	−8	1
1	1	1

(b)

Fig. 25: 3 × 3 Laplacian mask

Laplacian is hardly used in practice for edge detection because it is very sensitive to noise, and produces double edges. Edge direction is not detected by Laplacian. It is used to find the location of edge using zero crossing detection. When the image contains noise, the best approach is to use Laplacian of 2D Gaussian function for edge detection application.

Laplacian of 2D Gaussian function is used for edge detection application.

2D Gaussian filter is given by $H(x, v) = e^{-1x^2 + y^2 / 2\pi^2}$ or $H(r) = e^{-r^2/2\sigma^2}$,

where, $r^2 = x^2 + v^2$ and σ is the standard deviation.

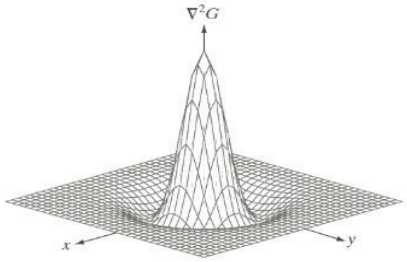
First derivative of Gaussian filter is $H'(r) = \frac{1}{\sigma^2} r e^{-r^2/2\sigma^2}$.

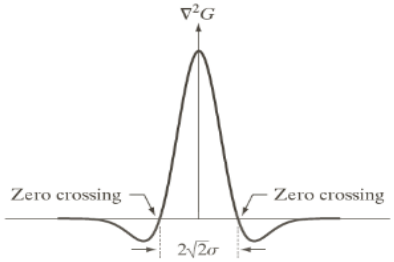
Second derivative of Gaussian filter is $H'(r) = \frac{1}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right) e^{-r^2/2\sigma^2}$.

After returning to original coordinates x, y and introducing a normalizing coefficient C, a convolution mask of LOG (Laplacian of Gaussian) operator is given by

$$H''(x, y) = C\left(\frac{x^2 + y^2 - \sigma^2}{\sigma^4}\right) e^{-(x^2+y^2)/2\sigma^2}.$$

A 5 × 5 LOG mask is given in Fig. 26. Due to its shape, LOG is also known as 'Mexicanhat'. Computing second derivative in this way is robust and efficient. Zero crossings are obtained at r = ±σ.





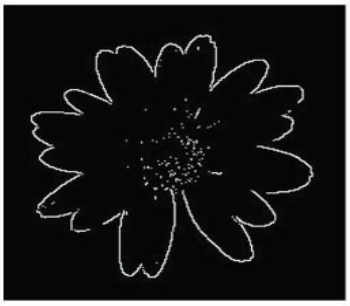
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	−16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Fig. 26: LOG as an image, LOG 3 D plot, 5×5 LOG mask

Fig. 27 (a) is the input image, Fig. 27 (b) is the output of prewitt filter Fig.27 (c) is the output of Robert filter, Fig. 27 (d), Fig. 27 (e), and Fig.27 (f) are outputs of Laplacian, Canny and Sobel filters respectively. As it is clear from the figures, each filter extracts different edges in the image. Laplacian and canny filters extract lot of inner details while sobel and robert filters extract only the boundary. Prewitt filter extracts the entire boundary of the flower without any gaps in the boundary.

(a) Original image

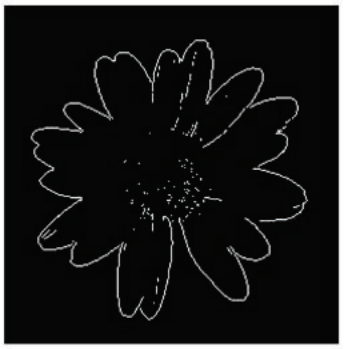
(b) Output of prewitt filter



(c) Output of robert filter

(d) Output of laplacian filter

(e) Output of canny filter



(f) Output of sobel filter

Fig. 27

Try the following exercises.

E4) What is an edge?

E5) List the properties of the second derivative around an edge?

E6) Define Gradient Operator?

E7) Why is a Laplacian generally not used in its original form for edge detection?

E8) Give a 5 × 5 LOG mask.

11.4.3 Line Detection

Line detection algorithm follows the definition of line in mathematics. It will select 'x' edge points and detect all lines lying on theses edge points. Fundamentally Line detection algorithms are based on 'Hough Transform'(HT) and Convolution [2]

- Any line can be represented by the equation (1)

$$Y = a \times X + b \qquad (1)$$

But it will be difficult to represent a vertical line using equation (1). So in HT we use the angle and its distance from origin will be used for representing lines.

If 'r' and 'θ' represents distance of line from origin and its angle,

$$y = x\,\frac{\sin\theta}{\cos\theta} + \frac{r}{\sin\theta} \qquad (2)$$

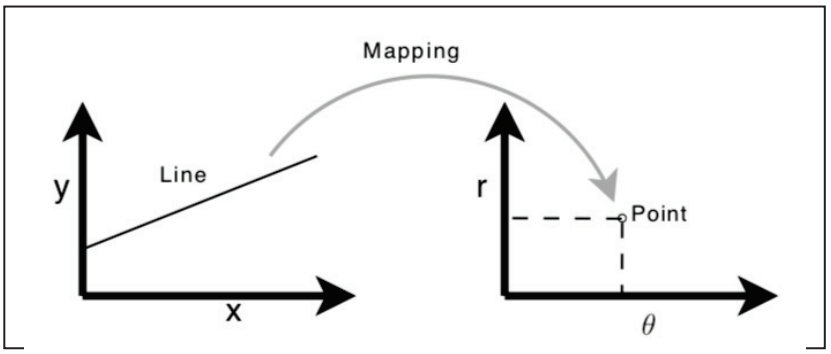


Figure 28(Source: Internet)

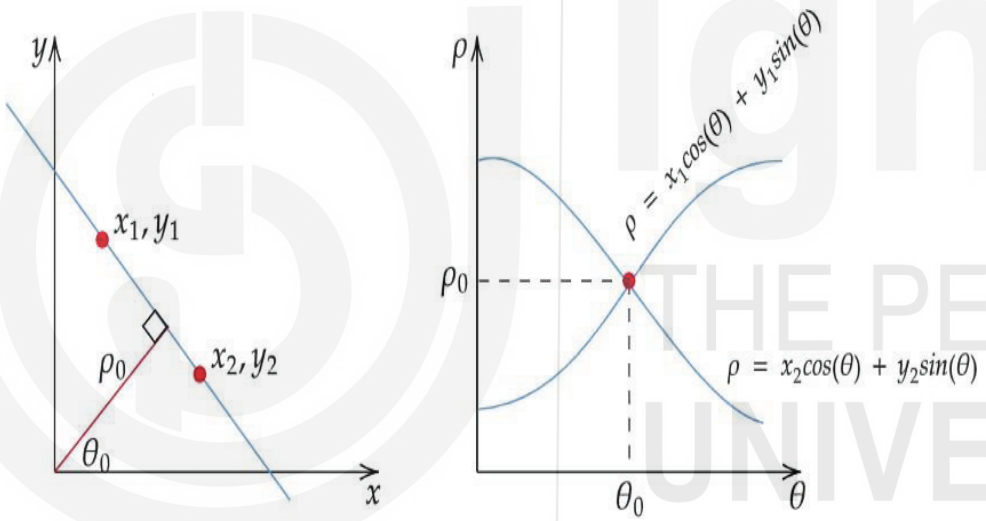
We can represent any line using equation (2) where $\theta \in [0,360[$ and $r \geq 0$.

a) Hough Transform (HT) : The main constraint of any image processing algorithm is the amount of data. We need to reduce data for preserving related information of objects. Edge detection can do this work effectively. The output of edge detector cannot identify lines. HT was initially developed for line detection and later defined for shape detection too

When we represent lines in the form of y = ax + b there is one problem. In this form, the algorithm won't be able to detect vertical lines because the slope a is undefined/infinity for vertical lines This would mean a computer would need an infinite amount of memory to represent all possible values of

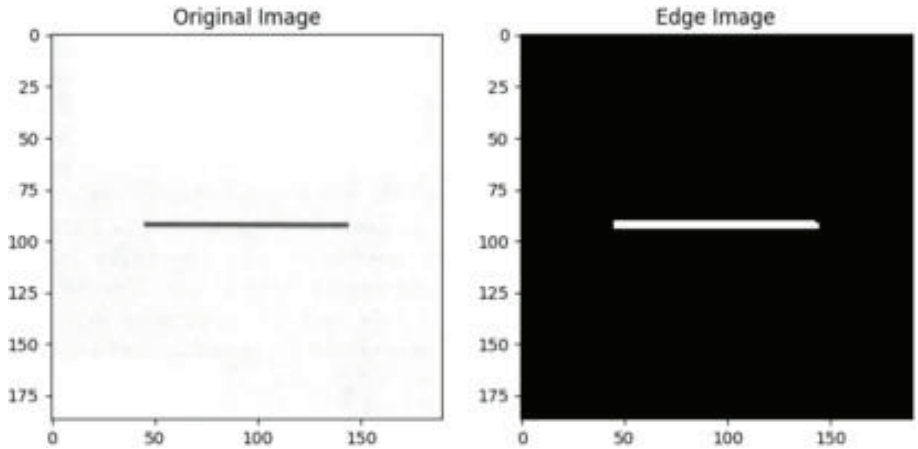
So, in Hough's transform we use the parametric form to describe the lines, i.e $\rho = r\cos(\theta) + c\sin(\theta)$, where ρ is the normal distance of the line from the origin, and θ is the angle that the normal to the line makes with the positive direction of x-axis in the positive direction.

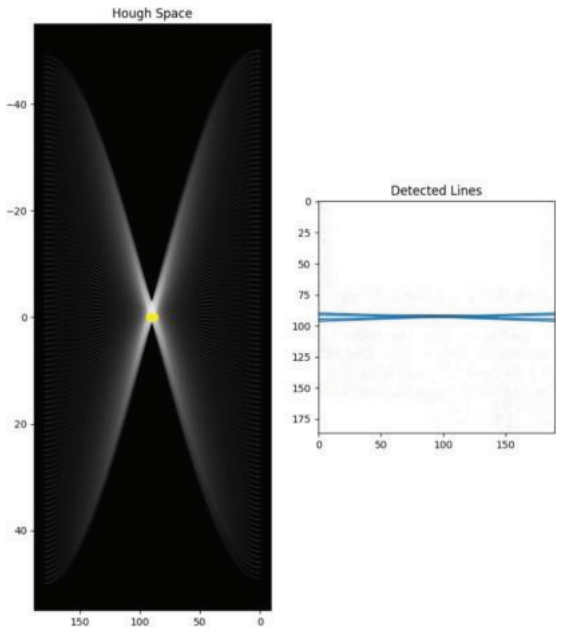
The Hough space thus has 2 parameters - θ and ρ, and a line is represented by a single point, defined by the values of these two coordinates



https://towardsdatascience.com/lines-detection-with-hough-transform-84020b3b1549

Figure 29: Representation of a straight line in the Hough.





https://towardsdatascience.com/lines-detection-with-hough-transform-84020b3b1549

Figure 30 intersection spot in Hough Space (Source: Internet)

The dot in the Hough Space represent that the line exists and is identified by the θ and ρ values.

Edge points produce cosine curves in the Hough Space. Cosine curve can be generated by mapping all edge points from an edge image on to the Hough Space. If two edge points lay on the same line, their corresponding cosine curves will intersect each other on a specific (ρ, θ) pair (intersection points in Figure.30)

Mapping all the edge points from an edge image onto the Hough Space, will generate a number of cosine curves.

1. The first step is to decide the range for ρ and θ. Usually, the range of θ is [0, 180] degrees and ρ is [-d, d] where d is the length of the edge image's diagonal.
2. Then create a 2D array (ρ, θ), then we'll calculate r cos(θ) + c sin(θ) value for each pixel (r,c), for multiple values of ρ and θ, and store the values in the array.
3. Finally, take the highest values in the above array. These will correspond to the strongest lines in the image, and can be converted back to y = ax+b form

Hough transform: The Hough transform is an incredible tool that lets you identify lines. Not just lines, but other shapes as well.

Example: Using Hough transform show that the points (1,1), (2,2), and (3, 3) are collinear find the equation of line.

Solution: The equation of line is y=mx+c, In order to perform Hough transform we need to convert line from (x,y) plane to (m,c) plane Equation of (m,c) plane is

Step 1: y =mx+C

For(1,1) y=mx+c 1=m+c C=-m+1

If c=0 then (0=-m+1) m=1

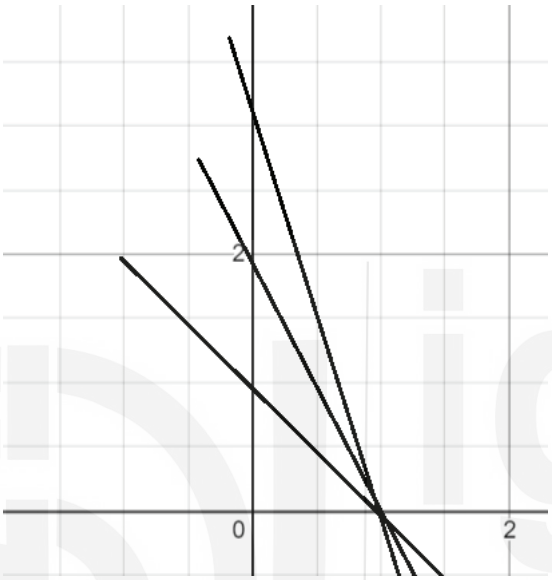
If m=0 then (c=1)c=1 (m,c) = (1, 1)

Similarly for other points

If (x,y)= (2,2) then (m,c) = (1,2)

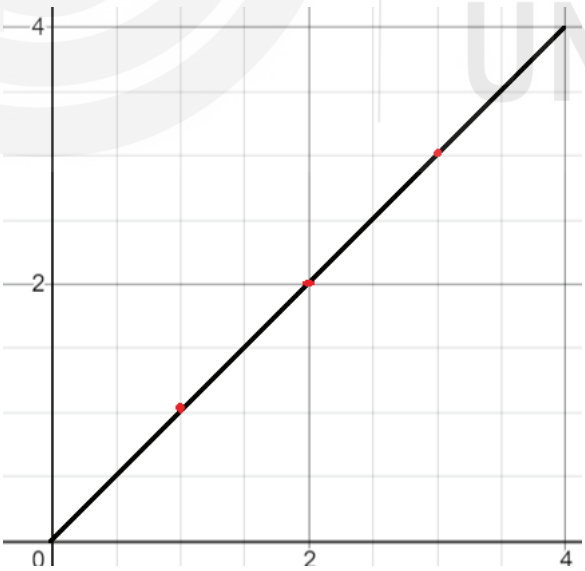
If (x,y)= (3,3) then (m,c) = (1,3)

Step 2: Intersect at the point (0,1) Then (m,c) = (0,1)



Plot a graph for (mc) = (1, 1), (1,2), and (1,3)

Step 3 : The original equation of line is (y=mx+c) put the value of m and c on this eq. Then y=x



points (1,1) , (2,2), and (3, 3) are collinear

b) Convolution Based Technique

Convolution Masks are used to detect lines in this technique. Basically, there are 4 different varieties of convolution masks: Horizontal, vertical, oblique (+45 degrees), and oblique (−45 degrees)

Horizontal (R1)			Vertical (R3)			Oblique (+45 degrees) (R2)			Oblique (-45degrees) (R4)		
-1	-1	-1	-1	2	-1	-1	-1	2	2	-1	-1
2	2	2	-1	2	-1	-1	2	-1	-1	2	-1
-1	-1	-1	-1	2	-1	2	-1	-1	-1	-1	2

Lines are detected using the equation (3) by using the response obtained after convolving these masks with the image.

$$R(x,y) = \max(|R1(x,y)|, |R2(x,y)|, |R3(x,y)|, |R4(x,y)|) \quad (3)$$

$$\text{If } R(x,y) > T, \textit{then discontinuity}$$

Convolution means multiplying the corresponding values and accumulating the same. This method will detect all light lines against dark background

In the following section, we discuss the region based segmentation.

11.5 REGION DETECTION

Region detection is an important aspect in image processing tool for semantically significant spatial information in images. Matching establishes similarity between visual entities, which is crucial for recognition.

In a picture, a region is a collection of connected pixels with comparable features. Because they may correlate to items in a scene, regions are significant for picture interpretation. There are multiple objects in an image which have multiple regions corresponding to various portions of the object. Therefore, it is necessary to partition an image into several regions that correspond to objects or parts of things in order to interpret accurately. In general, pixels in a region will have similar features. Pixels belonging to a specific object can be identified by testing the following conditions

A. The mean of the grey value of pixels of an image and the mean of the grey value of pixels of a specific object in the image will be different

B. The standard deviation of the grey value of pixels belonging to a specific object in the image will lie within a specific range.

C. The texture of the pixels of an object in the image will have a unique property

But the connection between regions and objects is not perfect due to segmentation errors. Therefore, we need to apply object-specific knowledge in later stages for image interpretation.

Region-based segmentation and boundary estimation using edge detection are two methods for splitting a picture into areas.

Further, in boundary detection, semantic boundaries are considered to find different objects or sections of an image. It is different from edge detection because it does not use boundaries between light and dark pixels in an image.

The brief discussion on Region-based segmentation, boundary estimation, and boundary detection is given below:

a) **Region based segmentation:** In region-based segmentation, all pixels of an image belonging to a same area are grouped and labelled together. Here, pixels are assigned to areas based on unique characteristic feature which is different from other part of image. Value similarity and spatial closeness are two important features of this segmentation process. If two pixels are very close to one another and have similar intensity characteristics, they may be allocated to the same region. For example, the similar grey values can represent similar pixels and Eucledian distance can represent the closeness of the pixels.

   The similarity and proximity concepts are based on the idea that points on the same object will project to pixels in the image that have similar grey values. We can also assume to group pixels in the image and then employ domain-dependent information to match areas to object models. In simple cases, thresholding and component labelling can be used to segment data.

b) **Boundary estimation method:** In this method, segmentation is done by finding the pixels (called edges) lying on a region boundary. The difference between neighbouring pixels can be used to determine the region boundary as regions on either side of the boundary may have different grey levels. A large number of edge detectors rely on intensity characteristics to detect edges instead of using derived properties such as texture and motion.

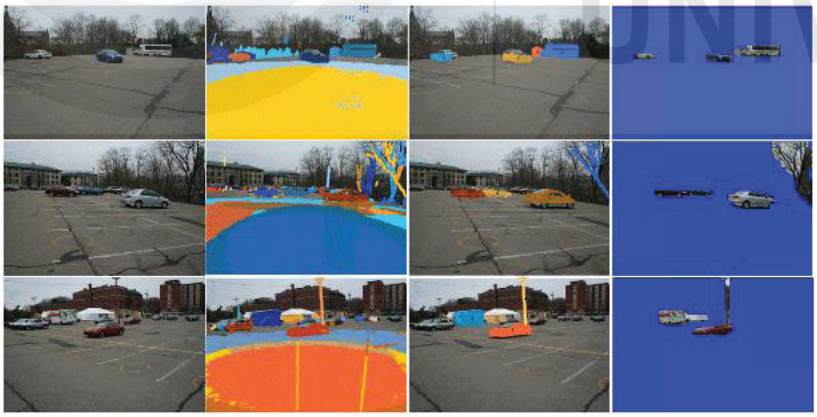


Figure 31 Boundary identified by grouping similar pixels

c) **Boundary Detection:** In boundary detection, semantic boundaries are considered to find different objects or sections of an image. It is different from edge detection because it does not use boundaries between light and dark pixels in an image. A zebra, for example, has several internal boundaries between black and white stripes that humans would not consider part of the zebra's boundary. The focus is more on approximate

boundary detection method using training data because a perfect solution requires high-level semantic knowledge about the scene in the image.

Figure 32

In previous sections, object boundary was detected to locate an object. In this section, object region will be used to locate the object. In theory, locating an object by locating either its boundary or region should result in the same object as boundary and region are just different representations of the same object. But in practice, edge based segmentation approach may give totally different result than a region based approach.

Imperfect images, imperfect methods can be the reason which causes that the result of object detection using boundary based approach to be different from region based approach. Region based methods rely on the assumption that neighbouring pixels within one region have similar values. The common approach is to compare a pixel with its neighbours. A pixel belongs to the same cluster if the pixel and its neighbours satisfy the same similarity criterion. Region based segmentation methods employ two basic steps:

a) Merging
b) Splitting

Image segmentation using **merging** has the following steps

Step 1: Obtain the initial segmentation of the image.

Step 2: Merge two adjacent segments to form a single segment if they are similar in same way

Step 3: Repeat step 2 until no segment to be merged remains.

The initial segmentation can be all individual pixels. The basic idea is to combine two pixels (regions) if they are similar. The similarity criteria can be based on grey level similarity, texture of the segment etc.

Image segmentation using **splitting** has following steps

Step 1: Obtain an initial segmentation of an image.

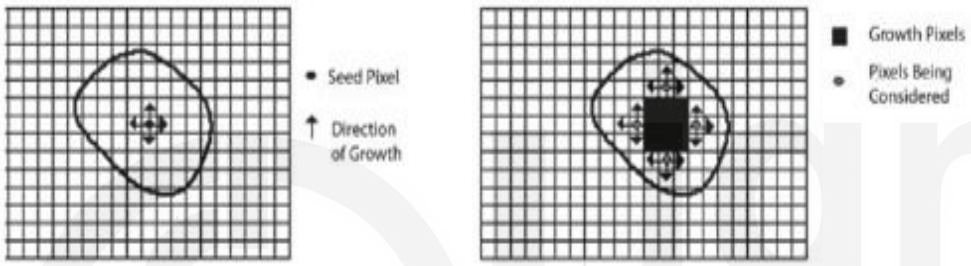
Step 2: Split each segment that is inhomogeneous in some way

Step 3: Repeat step 2 until all segments are homogeneous.

Here, initial segmentation may be the entire image (no segmentation). The criterion for inhomogeneity of a segment may be the variance of gray levels or the difference in its textures etc. Both splitting and merging methods seem to be top-bottom and bottom-top approach of the same method. But there is a basic difference. Merging two segments is straight forward, but in splitting, we need to know the sub-segment boundary.

Let us discuss region growing.

Region growing is a process of merging adjacent pixel segments into one segment. It is one of the simplest and very popular method of segmentation which is used in many applications. It needs a set of starting pixels called '**seed**' points. The process consists of picking a seed from the set and examining all 4 or 8 connected neighbours of this seed and merging similar neighbours to the seed as shown in Fig. 33 (a). The seed point is modified based on all merged neighbours Fig. 33 (b). The algorithm continuous until the seed set is empty.



(a) Start of Growing a Region **(b) Growing Process After a Few Iterations**

Fig. 33: Region Growing

Region growing algorithm for a single seed point and same grey level value as similarity measure to merge the pixels is as follows:

The algorithm uses a data structure entity stack to keep track of seed points. Two operations push and pop are used here. Push: puts the pixel co-ordinates on the top of the stack. Pop: takes a pixel from the top of the stack. Input to the algorithm is an image 'f'. Initial seed has coordinates of (x,y) and grey level value g at $(x,y), f(x,y)=g$. The goal is to grow a region with all pixels having gray level value of g and assigning them grey level $k=1, k\neq 1, k\neq g$. Let (x,y) be the coordinates of initial seed, and let (a,b) be the coordinates of pixel under investigation.

The algorithm

Push (x,y)

Do till stack is not empty

Pop (a, b) /(take input point from top of stack)
If $f(a,b)=g$ / if input = desired value g
Set $f(a,b)=1$ / segmented pixel is assigned as 1/
Push $(a,b+1)$ / Test all four neighbors of (a,b)
Push $(a,b-1)$ / Test all four neighbors of (a,b) by pushing it on the top of the stack/
Push $(a+1,b)$

Push $(a-1, b)$

End

This is a recursive algorithm. The final region is extracted by selecting all pixels having grey level value as 1(k). The algorithm can be modified by changing the similarity measure to incorporate a range of values for merging. The statement if $f(a,b) = g$ can be changed to

$$g_j < f(a,b) < g_2.$$

Thus, if the grey level value of pixel (a,b) is between g_1 and g_2 then, it is segmented. The algorithm can be further modified to incorporate multiple seed points. In the above algorithm, only four neighbours are considered. It can be modified for eight neighbourhood. Instead of using four push instruction, eight push instruction can be used with coordinates of all eight neighbours.

There are several issues about region growing.

1) Suitable selection of seed points is important. Seed point selection is user dependent. Histogram and image properties can be taken into account by the user to select the seed point.

2) Selection criteria for similarity measure is important. It generally depends on the original image and object to be segmented. Band of intensity values, color, texture, shape are some of the similarity measures generally used.

3) Formulation of a stopping rule is also very important in region growing. Growing of a region should stop when no more pixel satisfy the similarity criteria. Algorithm can be made more powerful, if size and shape of the grown region are also considered.

Main advantage of region growing algorithm is that it correctly separates the regions based on the property defined by the user. The algorithm is very simple to implement. Only input needed are the seed points and selection criterion. Multiple criteria can also be applied. The algorithm works well in noisy environment also.

Major disadvantage of region growing is that the seed points are user dependent. Selection of wrong seed points can lead to wrong segmentation results. The algorithm is highly iterative and requires high computational time and power.

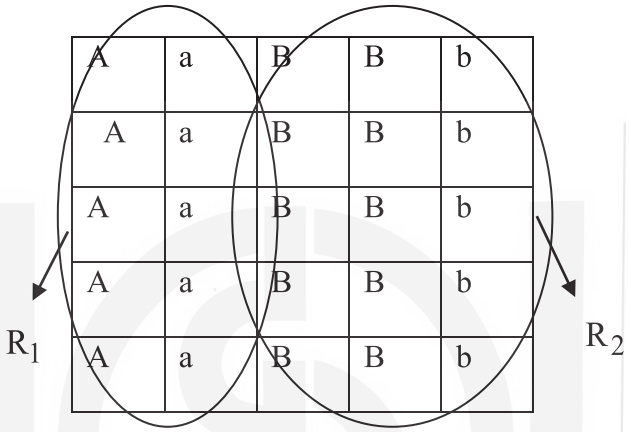
Example 1: In the image segment given in Fig. 34 (a) seed points are given at (3, 2) and (3, 4). Similarity criterion is grey level difference. Find segmented image, if a) $T = 3$ and b) $T = 8$.

Solution: For $T = 3$, region growing starts with pixel (3, 2). All the pixels having grey level difference < 3 are assigned as a and denoted as region R_1. Another region growing starts at (3, 4). All pixels with grey level value < 3 are assigned as b and denoted as region R_2. The output is shown in Fig 34(b). For $T = 8$, all the pixels have grey level difference less than 3 $\Rightarrow$ only one

region is formed, with all pixels being assigned as 'a'. The output is shown in Fig. 34 (c).

	1	2	3	4	5
1	0	0	5	6	7
2	1	1	5	8	7
3	0	1	6	7	7
4	2	0	7	6	6
5	0	1	5	6	5

(a) Input Image Segment for Example 1



(b) Output for T = 3

a	A	a	a	a
a	A	a	a	a
a	A	a	a	a
a	A	a	a	a
a	A	a	a	a

(c) Output for T = 8

Fig. 34

***

Example 2: Use region growing to segment the object in the image given in Fig. 20. The seed is the centre pixel of the image. Region is grown in following directions.

a) In horizontal and vertical directions only (4 neighbourhood)

b) In horizontal and vertical and diagonal directions (8 neighbourhood).

Similarity, criterion is the difference between two pixel values is less than or equal to 5

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	66	60
10	59	10	60	70	63	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Fig. 35: Input image segment

Solution: a) Region growing starts with seed point pixel with grey value 60 in the centre. It moves horizontally up and down, vertically up and down to check how much given pixel value differs from 60. If the difference is less than equal to 5. Then it is assigned as 'a' and merged with the region, else it is assigned as 'b'. Fig 36(a) shows the output.

b) If diagonal elements are also included then the region grows more as shown in Fig. 36(b).

b	b	b	b	b	b	b
b	b	b	b	b	b	b
b	b	a	a	a	b	b
b	a	b	a	b	b	b
b	a	a	a	b	b	b
b	b	b	b	b	b	b
b	b	b	b	b	b	b

(a) Output for 4 Neighborhood

b	b	b	b	b	b	b
b	b	b	b	b	b	b
a	b	a	a	a	b	a
b	a	b	a	b	a	a
b	a	a	a	b	b	a
b	b	b	b	b	b	b
b	b	b	b	b	b	b

(b) Output for 8 Neighborhood

Fig. 36

***

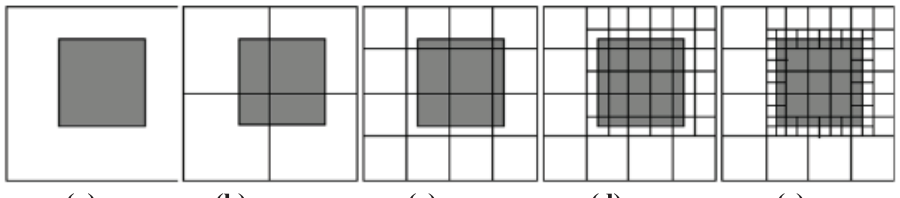
Now, let us discuss Split and Merge Method**.**

As mentioned before, there is a fundamental problem in splitting procedure. It needs suitable sub segments to be established before performing splitting. The problem of how to split has to be solved. We can decide whether a particular segment needs to be split or not by checking if

- Grey level difference (variance) exceeds a threshold or
- The variance of texture measure exceeds threshold or
- Histogram entropy or any other histogram measure exceeds threshold, or
- Edge pixels exist in the segment

We start by splitting the entire image into four quadrants. The object can be in any or some or of all of the four quadrants, as the object can be anywhere in the image.

Then, further subdivision of the quadrants is done. Merging operation is added to the segmentation process, recursive splitting and merging of image segments is done as shown in Fig. 37. Original image is shown in Fig. 37(a) and Fig. 37 (b) shows, entire image split into four segments. Fig. 37(c) shows a further split of four segments. In Fig. 37(d), further segmentation is done if grey level variance in the sub block is non zero. Merging of the sub blocks having same grey levels is shown in Fig. 37(e). Continuing this process, we end up in two segments one being the object and the other being the back ground.



(a) (b) (c) (d) (e)

Fig. 37: Example of Split and Merge Segmentation

Split and merge algorithm uses 'Quad tree' for representing segments. Fig. 38 shows that there is a one to one relation between splitting an image recursively into quadrant sand the corresponding quad tree representation. However, there is a limitation in this representation, as we cannot model merging of two segments at different level of pyramid.

Let R represent the entire image. A homogeneity criterion it is selected. If the region R is not homogeneous (H(R) = false), then split R into four quadrants R_1, R_2, R_3, R_4. Any four regions with the same parent can be merged into a single homogeneous region if they are homogeneous.

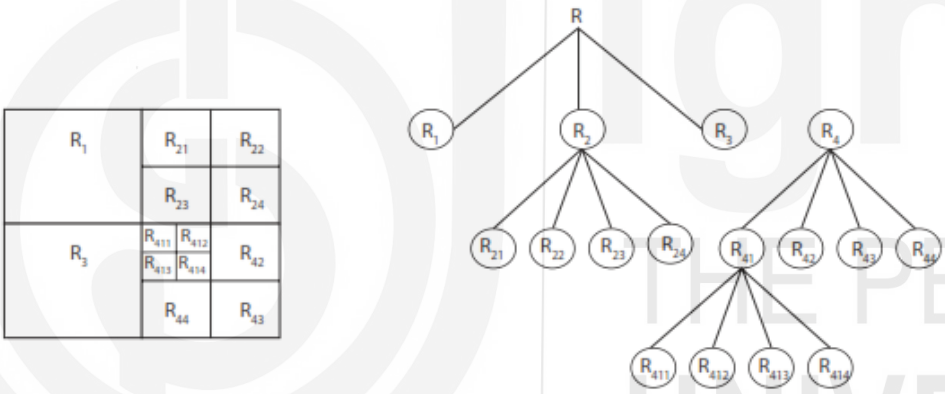
The steps of split and Merge algorithm are as follows:

Step 1: Define an initial segmentation into regions, a homogeneity criterion and a pyramid data structure.

Step 2: If any region R in the pyramid data structure is not homogeneous (H(R) = False), split it into four child regions.

Step 3: When no further splitting is possible, merge two adjacent regions Ri which are homogeneous ($H(R_i \cup R_j)$ = True)

Step 4: Stop when no further merging is possible.



(a) Naming convention of quadrants **(b) Quadtree representation**

Fig. 38: Example of recursive splitting of an image by a quad tree.

Several modification to this basic algorithm are possible. For example, in Step 2, merging of homogeneous regions is allowed which results in a simpler and faster algorithm.

The major advantages of this algorithm is that the image could be split progressively according to our required resolution because the number of splitting levels is decided by the user. Major disadvantage is that it may produce blocky segments as splitting is done in rectangular quadrants. This problem can be solved by splitting at higher level but this will increase the computational time.

Example 3: Segment the image in Fig. 39 (a) by split and merge algorithm. Homogeneity criteria is the grey levels of the pixels.

Solution: The image is divided into four quadrants. Fig. 39 (b) shows the image and its quad tree. Quadrants 2 and 3 are homogeneous. Thus no further splitting is done. Quadrant 1and 4 are non – homogenous hence they are divided further into 4 quadrants. Fig. 39 (c) shows the splitting and

corresponding quad tree. Now only one quadrant, 12 is still non-homogenous. Hence, it is further

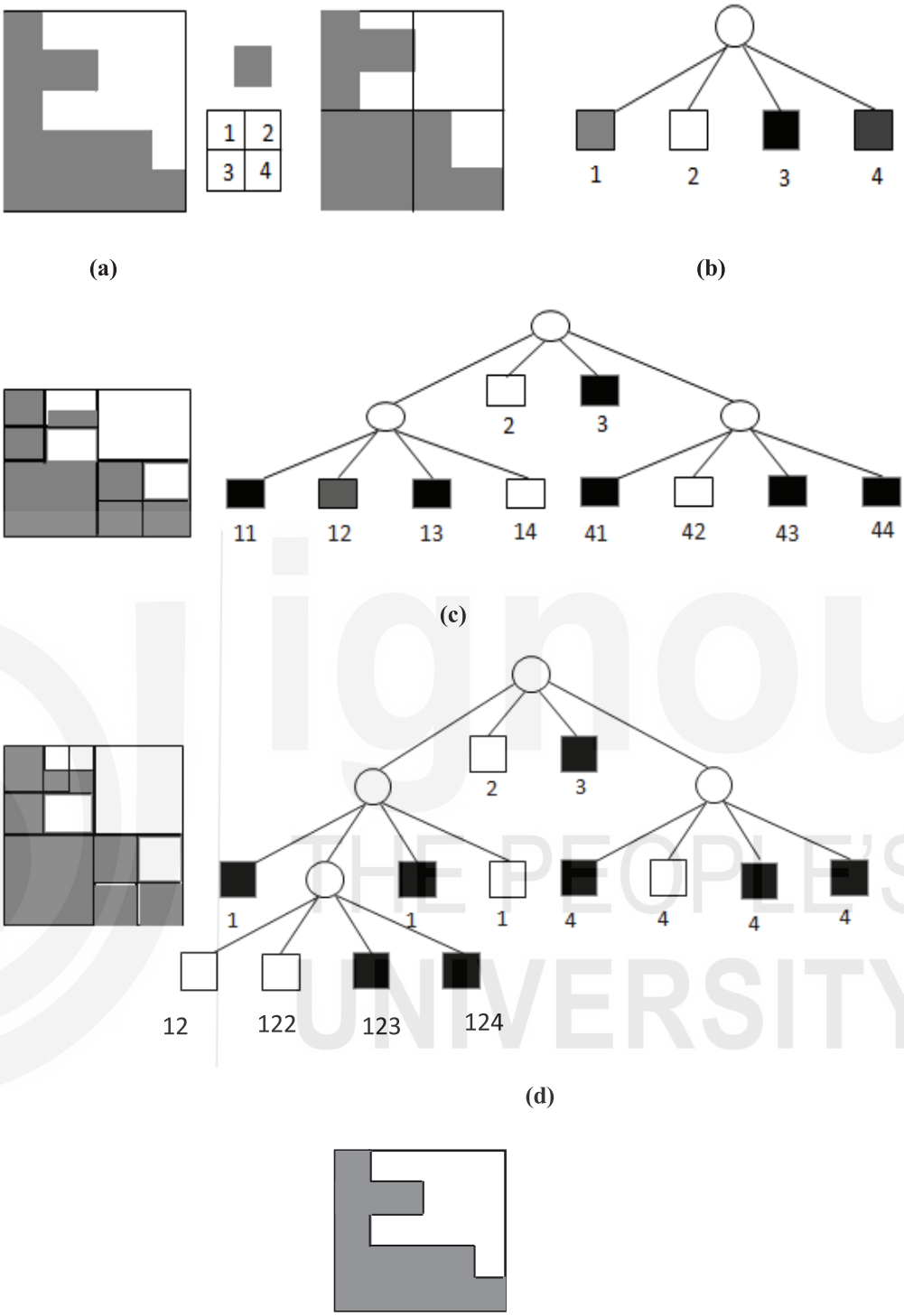


Fig. 39: Split and merge algorithm

subdivided. Fig. 39(d) shows the segmented image and its final quad-tree structure. Now all regions are homogeneous and hence no further splitting is possible.

Now merging operation takes place between adjacent quadrants. Quadrants 43 and 44 are merged into a single region. Quadrants 123 and 124 are merged, quadrants 11 and 14 are merged. Finally merge all quadrants which

are homogenous. Fig. 39(e) is the final segmented image after merging is complete.

***

Now try the following exercises.

E9) Distinguish between image segmentation based on thresholding with image segmentation based on region-growing techniques.

E10) Consider the image segment

128	128	128	64	64	32	32	8
64	64	128	128	128	8	32	32
32	8	64	128	128	64	64	64
8	128	128	64	64	8	64	64
128	64	64	64	128	128	8	8
64	64	64	128	128	128	32	32
8	128	32	64	64	128	128	128
8	8	64	64	128	128	64	64

E11) What are the advantages/disadvantages if we use more than one seed in a region-growing technique?

In the following section, we shall discuss boundary detection.

11.6 BOUNDARY DETECTION

After performing image segmentation, a region may be represented in terms of
– external characteristics (boundaries).
– internal characteristics (texture).

A region may be described by its boundary in terms of features such as its length, the orientation of the straight line joining its extreme points and the number of concavity in the boundary. Many algorithms require the points in the boundary of a region in an ordered clockwise (or anti clockwise) direction. For boundary detection or following or tracking, the following assumptions are made:

1) The image is binary where1=foreground and 0=background
2) The image is padded with a border of 0' s so an object cannot merge with the border.
3) We limit the discussion to single regions. The extension is straight forward.

Moore Boundary Tracking Algorithm:
Given a binary region R or its boundary.

Step 1: Let the starting point b_0, be the uppermost, leftmost point in the image labeled 1.

Step 2: Denote by c_0 the west neighbour of b_0. c_0 is always a background point.

Step 3: Examine the 8-neighbours of b_0, starting at c_0 and proceeding in a clockwise direction.

Step 4: Let b_1 denote the first neighbour encountered whose value is 1.

Step 5: Let c_1 denote the background point immediately preceding b_1 in the sequence.

Step 6: Store the locations of b_0 and b_1 for use in Step 10.

Step 7: Let $b = b_1$ and $c = c_1$.

Step 8: Let the 8-neighbours of b starting at c and proceeding clockwise be denoted as $n_1, n_2 \ldots n_8$. Find the first n_k which is foreground (i.e., a "1") .

Step 9: Let $b = n_k$ and $c = n_{k-1}$

Step 10: Repeat steps 8 and 9 until b=b_0, that is, we have reached the first point and the next boundary point found is b_1.

The sequence of b points found when the algorithm stops is the set of ordered boundary points.

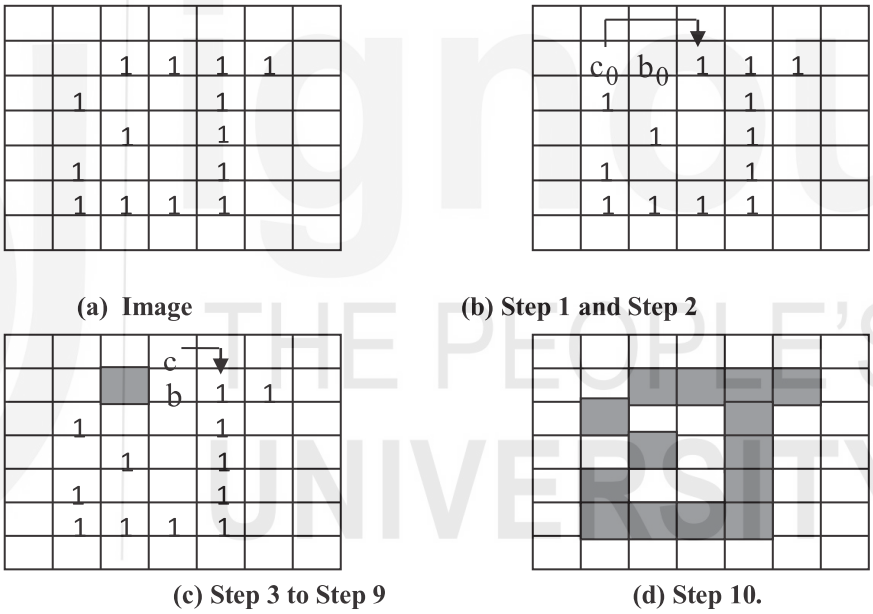


Fig. 40: Illustration in Boundary Following Algorithm.

There is a need for stopping rule as stated in Step 10. We would only include the spur at the right if we stop when we reach the initial point without checking the next point. Starting from topmost left most point in Fig. 41 (a) results in Fig. 41 (b). In Fig. 41 (c) the algorithm has returned to the starting point again. Rest of the boundary could not be traced.

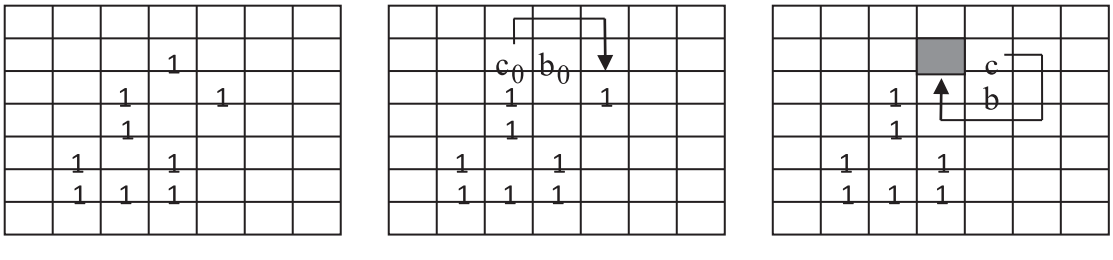
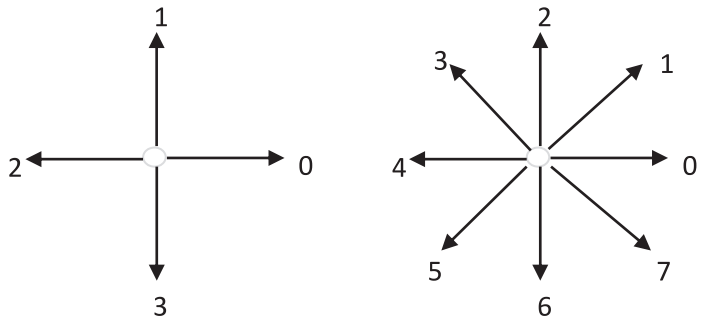


Fig. 41: Example of erroneous result of Boundary Detection Algorithm.

Now, we shall discuss the chain codes.

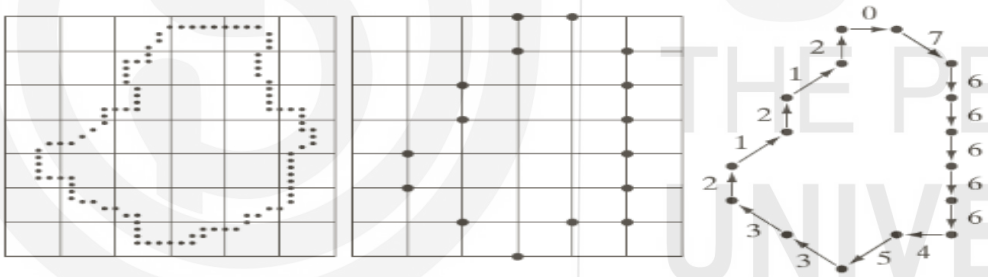
Chain codes are used to represent a boundary by a connected sequence of straight line segments of specified length and direction. Freeman codes [1961] represent a boundary by the sequence of straight line segments of specified length and direction. The direction is coded by a numbering scheme (4 or 8-connectivity) as shown in Fig. 42



(a) 4-direction chain code **(b) 8-direction chain code**

Fig. 42: Direction Numbers

Images are acquired and processed in a grid format with equal spacing in the x and y- directions, so a chain code can be generated by following a boundary in clockwise direction and assigning a direction to the segment connecting every pair of pixels. To avoid noise degradation and long chains a resampling of the image grid is commonly used to describe the boundary at a coarser level as shown in Fig. 43 (a). Fig. 43(b) shows resampled points and Fig. 43(c) shows 8-direction chain code.



(a) Digital Boundary with Resampling Grid **(b) Result of Resampling** **(c) 8-Direction Chain Coded Boundary**

Fig. 43

If we start from topmost leftmost corner, chain code for Fig. 28 is
0 7 6 6 6 6 6 4 5 3 3 2 1 2 1 2

The chain code depends on the starting point. To normalize it, we treat the code as a circular sequence of direction number sand redefine the starting point so that the resulting sequence forms an integer of minimum magnitude. To account for rotation, we use the first differences of the chain code instead of the code itself.

The **first difference** is obtained by counting the number of direction changes in counter clockwise direction that separate two adjacent elements of the code.

For boundary in example in Fig. 28, the chain code is 0 7 6 6 6 6 6 4 5 3 3 2 1 2 1 2.

First difference is **6 7 7 0 0 0 0 6 1 1 0 7 7 1 7 1**

First value is calculated by considering the code as a circular sequence of integers, hence coding the sign changes from 2 to 0 counter clockwise and so on.

Example 4: Find chain code and first difference of the following boundary shapes.

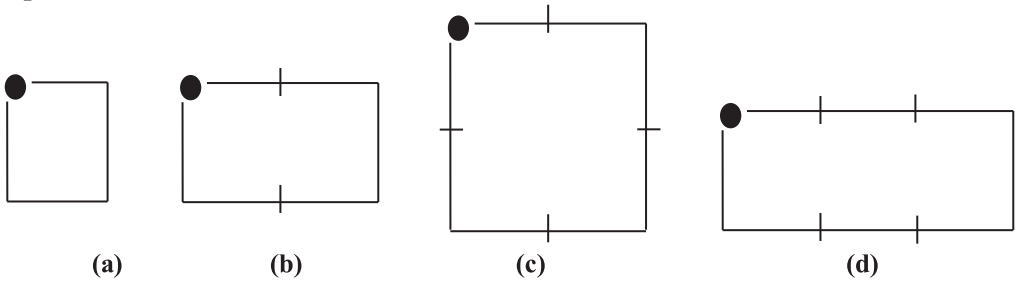
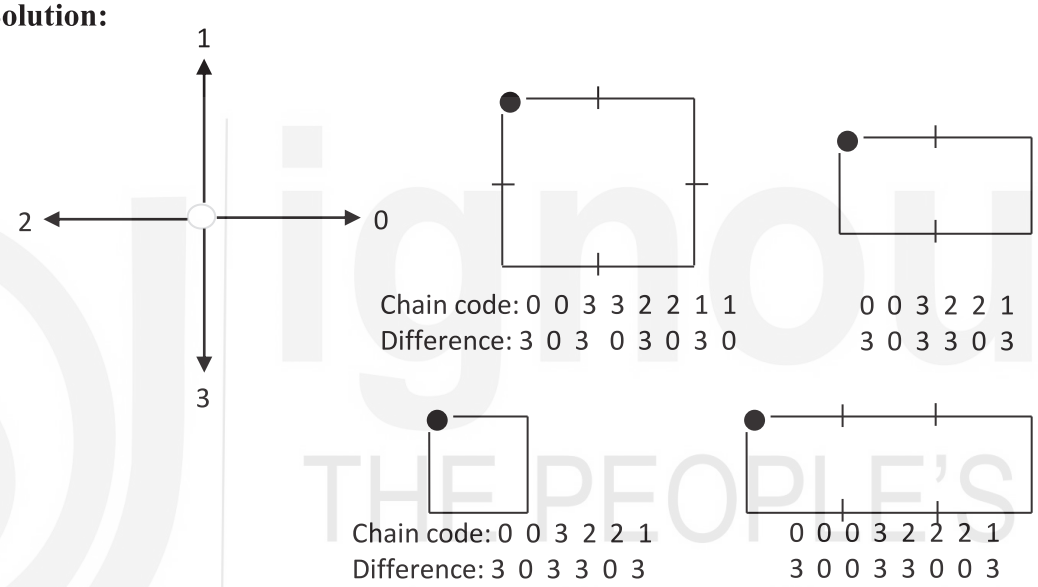


Fig. 44

Solution:

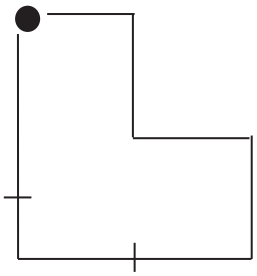


4-directional codes are used in this example. First differences are computed by treating the chain as a circular sequence.

***

Now, try an exercise.

E12) Find chain code and first difference of the following boundary shape.



11.7 FEATURE EXTRACTION

Feature extraction divides a large set of data into smaller groups for quick processing. There are a large number of variables in these huge data sets which require a large amount of processing power. Feature extraction extracts the best feature from by selecting and combining variables into features.

Applications of Feature Extraction

Image Processing, Auto-encoders and Bag of words are some applications of Feature Extraction.

1. Image Processing: In image processing, we experiment with images using different techniques to comprehend them better.

2. Auto-encoders: Auto-encoders do efficient unsupervised data coding. As a result, the feature extraction technique may be used to discover significant features from data to code by learning from original data set and generating new ones.

3. Bag of Words: Feature extraction is an important part of this process. This technique is widely used for natural language processing (NLP). Here, words/features are extracted from a document, website or sentence and are classified according to their frequency of use.

Traditional methods of feature detection

Traditional methods of feature detection include the following:

1. Harris Corner Detection- It detects corners based on differential corner score with reference to direction directly.

2. SIFT (Scale-Invariant Feature Transform)- Generally used for invariance.

3. SURF (Speeded-Up Robust Features)- This technique is a simplified variant of SIFT.

4. FAST (Features from Accelerated Segment Test)- In comparison to SURF, this is a substantially faster corner detecting algorithm.

5. BRIEF (Binary Robust Independent Elementary Features)- This feature descriptor can be used with any other feature detector. By converting floating point integers to binary strings, this approach minimizes memory utilization.

6. Oriented FAST and Rotated BRIEF (ORB) —This OpenCV algorithm uses FAST key-point detector and BRIEF descriptor. It is an alternative to SIFT and SURF.

Deep Learning Techniques for feature extraction

Convolutional neural network (CNN) can replace Traditional feature extractors because of their strong ability and efficiency to extract complex features for expressing more detailed part of an image and can learn task specific features.

1. SuperPoint: It detects points of interest and descriptors using Fully CNN. The extracted features are encoded in VGG style and then using two decoders, it generates inters points and descriptors

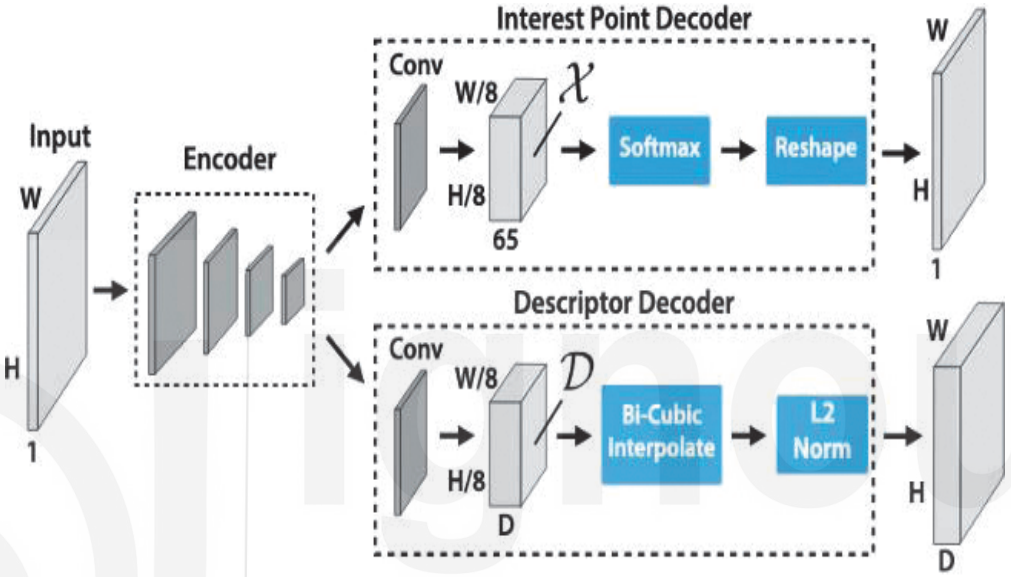


Figure 45 Super Point structure

The SuperPoint structure is an example of a fully connected-convolutional neural network architecture. It is capable of operating on a picture in its entirety and generating interest point detections with fixed-length descriptors in a single forward pass. The approach makes use of a single encoder that is shared by multiple users in order to process the incoming image and minimise the total number of dimensions. Following the encoder, the design then divides into two "heads" that are referred as decoders. One head is responsible for locating potential places of interest, while the other is in charge of describing those potential points of interest. Both of these activities will make use of the majority of the network's parameters. Unlike previous systems, which locate interest points first and then compute descriptors, this one is able to share processing and representation between the two tasks. Traditional systems locate interest points first and then compute descriptors.

As a consequence of this, a system has been developed that is effective for completing tasks such as homography estimation, which require matching geometric shapes.[1]

D2-Net: It is a trainable CNN based local feature detector and dense feature descriptor(feature descriptor has minimum non zero values)

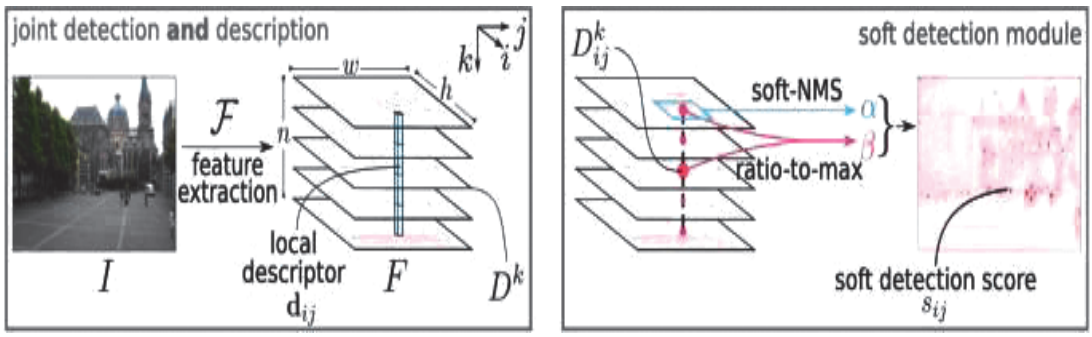


Figure 46: Detect and Describe D2 network

It's a fully convolutional neural network (FCNN) used for extracting feature maps with a double purpose:

i) Obtaining the local descriptors dij at a given spatial position (i,j) is as easy as traversing all the n feature maps Dk;

ii) Keypoint detection scores sij are calculated during training using a soft local-maximum score and a ratio-to-maximum score for each descriptor, and detections are generated by executing a non-local-maximum suppression on a feature map.[2]

2. LF-Net: This approach uses training image pairs with relative pose and depth maps

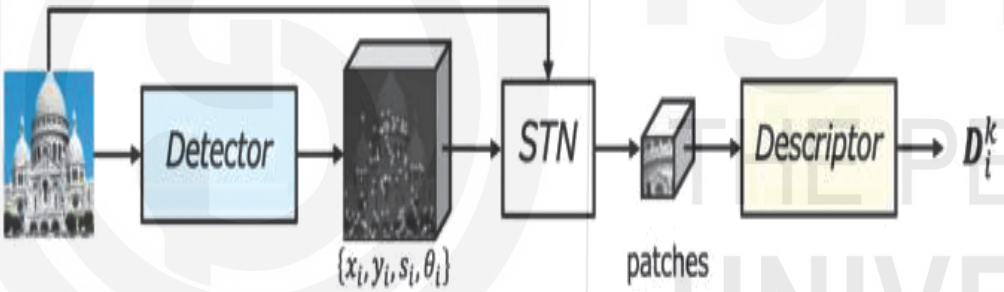


Figure 47: LF-Net Source: Internet

It can be thought of as both a feature detector and a dense feature description at the same time. The key points that are obtained using this method are more stable than their traditional equivalents, which are based on the early detection of low-level structures. This is achieved by delaying the detection until a later stage. We demonstrate that pixel data can be used to train this model [3].

Now let us summaries what we have discussed this unit.

11.8 SUMMARY

In this unit, we have discussed the following:

1. image segmentation techniques;
2. edge based segmentation;
3. line based segmentation;

4. various region based segmentation techniques; and
5. boundary detection algorithm

11.9 SOLUTIONS AND ANSWERS

E1) Image segmentation partitions an image into set of regions. In many applications, regions represent meaningful areas in an image. In other applications, regions might be set of border pixels grouped into structures such as line segments, edges etc. The level of partitioning of an image depends on the problem to be solved. Segmentation should stop when the objects of interests have been isolated. Segmentation has two objectives:

a) To decompose an image into regions for further analysis.

b) To perform a change of representation of an image for faster analysis.

E2) Segmentation is the most important step in automated recognition system which has numerous applications and some of them are discussed below:

1. Medical Imaging:
2. Satellite
3. Movement Detection:
4. Security and Surveillance:
5. License Plate Recognition (LPR)
6. Industrial Inspection and Automation
7. Robot Navigation

E3) Generally, all segmentation algorithms are based on one of the two properties of gray level values of pixels.

i. Discontinuity

In this approach, images are partitioned based on the difference or discontinuity of the gray level values. Edge based segmentation methods fall in this category.

ii. Similarity

Images are partitioned based on the similarity of the gray level values of the pixels according to a pre-defined criterion. Thresholding, region based clustering and matching based segmentation techniques fall in this category.

E4) A typical edge may be the border between a block of red color and a block of yellow. Edge can also be the boundary between two regions with relatively distinct gray-level properties.

E5) 'Sign' of second derivative is used to determine whether edge pixel lies on dark side or on light side of edge. '**Zero crossing**' is at the midpoint of a transition in gray level.

E6) **Gradient Operator**

Gradient is a first order derivative and is defined for an image $f(x,y)$ as

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

It points in the direction of maximum rate of change of f at a point (x,y)

Magnitude of the gradient is $\text{mag}(\nabla) = [G_x^2 + G_y^2]^{1/2}$

This can be approximated as $(\nabla f) = |G_x| + |G_y|$ mag

and direction of the gradient is given by $\alpha(x,y) = \tan^{-1}\begin{bmatrix} G_x \\ G_y \end{bmatrix}$

α is measured with respect to x- axis

E7) Laplacian is hardly used in practice for edge detection because it is very sensitive to noise, and produces double edges. Edge direction is not detected by Laplacian.

It is used to find the location of edge using zero crossing detection.

E8) 5 x 5 LOG mask

0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	−16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

E9) Image segmentation based on thresholding applies a single fixed criterion to all pixels in the image simultaneously. Hence, it is rigid. On the other hand, image segmentation based on the region-based approach is more flexible; hence it is possible to adjust the acceptance criteria in the course of region-growing process so that they can depend on the shape of the growing regions if desired.

E10) Step 1 Computation of the histogram of the input image.
The histogram of the image gives the frequency of occurrence of the gray level.

The histogram threshold is fixed as 32. Now the input image is divided into two regions as follows:

Region 1: Gray level < 32

Region 2: Gray level > 32

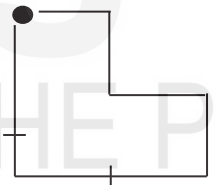
The input image after this decision is given as

$$\begin{vmatrix} 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 2 & 2 & 2 & 2 & 1 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \end{vmatrix}$$

E11) The advantage of using more than one seed is that better segmentation of the image can be expected, since more seeds lead to more homogeneous regions.

The drawback of using more than one seed is that the probability of splitting a homogeneous region in two or more segments increases.

E12)



Chain code: 0 3 0 3 2 2 1 1
Difference: 3 3 1 3 3 0 3 0

REFERENCES

[1] https://openaccess.thecvf.com/content_cvpr_2018_workshops/papers/w9/DeTone_SuperPoint_Self-Supervised_Interest_CVPR_2018_paper.pdf

[2] https://openaccess.thecvf.com/content_CVPR_2019/papers/Dusmanu_D2-Net_A_Trainable_CNN_for_Joint_Description_and_Detection_of_CVPR_2019_paper.pdf

[3] https://www.semanticscholar.org/paper/D2-Net%3A-A-Trainable-CNN-for-Joint-Description-and-Dusmanu-Rocco/162d660eaaa1eb2144d8030102f3e6be1e80ce50

[4] https://web.ipac.caltech.edu/staff/fmasci/home/astro_refs/HoughTrans_lines_09.pdf

[5] https://www2.ph.ed.ac.uk/~wjh/teaching/dia/documents/edge-ohp.pdf

[6] https://en.wikipedia.org/wiki/Line_detection

[7] https://towardsdatascience.com/image-feature-extraction-traditional-and-deep-learning-techniques-ccc059195d04

[8] https://morioh.com/p/14d27a725a0ehttps://www.researchgate.net/figure/An-example-of-edge-based-segmentation-18_fig2_283447261

UNIT 12 OBJECT RECOGNITION USING SUPERVISED LEARNING APPROACHES

Structure **Page No.**

12.1 INTRODUCTION

Obect recognition is the discipline of building machines to perform perceptual tasks which humans are particularly good at. e.g. recognize faces, voice, identify species of flowers identify diseases in plants by observing the leaves, identify animals, etc. There are practical needs to find more efficient ways of doing things. e.g. read hand-written symbols, diagnose diseases, identify incoming missiles from radar or sonar signals. The machines may perform these tasks faster, more accurately and cheaply.

The goal of Object recognition is to clarify complicated mechanisms of decision making processes and automatic these functions using computers.

In a classification task, we are given an object or **pattern** and the task is to classify it into one out of c classes. In the previous units it was discussed that how the decision boundary surface between various classes can be used to assign a class to each point in the test data. In this unit, a different approach is being proposed, where the class assigned will depend on how close the point of the test data (that is, the pattern) is to a particular class. This gives rise to Minimum Distance Classifier.

And now, we will list the objectives of this unit. After going through the unit, please read this list again make sure you have achieved the objectives.

Objectives

After studying this unit, you should be able to

- define pattern recognition
- apply different types of classifiers
- describe discriminant Functions (linear and non-linear)
- use Bayesian classification.

- find minimum distance classifiers;
- apply machine learning algorithm;
- describe supervised learning approach;
- describe unsupervised learning approach.

We begin the unit by discussing some basic concepts in the following section.

12.2 BASIC CONCEPTS

Although humans can perform some of the perceptual tasks with ease, there is not sufficient understanding to duplicate the performance with a computer. Because the complex nature of the problems, many pattern recognition research has been concerned with more moderate problems of pattern classification — the assignment of a physical object or event to one of several pre-specified categories.

For example, a lumber mill producing assorted hardwoods wants to automate the process of sorting finished lumber according to the species of trees. Optical sensing is used to distinguish birch lumber from ash lumber. A camera takes pictures of the lumber and passes to on to a feature extractor.

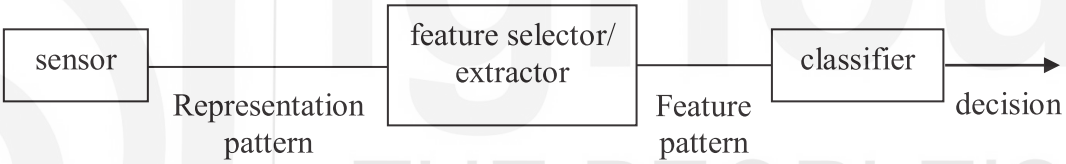


Fig. 1: A pattern classification system

The feature extractor reduces the quantum of data by measuring certain relevant properties that distinguish pictures of birch lumber from pictures of ash lumber. These features are then passed to a classifier that evaluates the evidence presented and makes a final decision about the lumber type.

Suppose that somebody at the lumber mill tells us that birch is often lighter colored than ash. Then brightness becomes an obvious feature. We might attempt to classify the lumber merely by seeing whether or not the average brightness ‘x’ exceeds some critical value.

One characteristic of human pattern recognition is that it involves a teacher. Similarly a machine pattern recognition system needs to be trained. A common mode of learning is to be given a collection of labeled examples, known as training data set. From the training data set, structure information is distilled and used for classifying new inputs.

Try an exercise.

E1) Explain a pattern classification system.

Goal of pattern recognition is to reach an optimal decision rule to categorize the incoming data into their respective categories. A pattern recognition investigation may consist of several stages, enumerated below. Not all stages may be present; some may be merged together so that the distinction between two operations may not be clear, even if both are carried out; also, there may be some application-specific data processing that may not be regarded as one of the stages listed. However, the points below are fairly typical.

1. **Formulation of the problem:** gaining a clear understanding of the aims of the investigation and planning the remaining stages.
2. **Data collection:** making measurements on appropriate variables and recording details of the data collection procedure (ground truth).
3. **Initial examination of the data:** checking the data, calculating summary statistics and producing plots in order to get a feel for the structure.
4. **Feature selection or feature extraction:** selecting variables from the measured set that are appropriate for the task. These new variables may be obtained by a linear or nonlinear transformation of the original set (feature extraction). To some extent, the division of feature extraction and classification is artificial.
5. **Supervised pattern classification:** The system is trained by providing the relevant feature samples with each sample labeled as belonging to a particular class. This help to develop the classifier.
6. **Apply discrimination:** or regression procedures as appropriate. The classifier is designed using a training set of exemplar patterns.
7. **Assessment of results:** This may involve applying the trained classifier to an independent test set of labeled patterns.
8. **Interpretation:** Result interpretation is very important. In case, results are not satisfactory, cycle may be started again from data collection.

We now define the basic terms which are used in the process of pattern recognition.

Feature

Feature is a property (or characteristics) of an object (quantifiable or non quantifiable) which is used to distinguish between (or classify) two different objects. For example, sorting incoming fish on a conveyor according to species using optical sensing. Two category of spices are there: Sea bass, Salmon.

Some properties that could be possibly used to distinguish between the two types of fishes are

- Length
- Lightness (Dark colour or light colour)
- Width
- Number and shape of fins
- Position of the mouth, etc…

This is the set of all suggested features to explore for use in the classifier.

Feature Vector

A Single feature may not be useful always for classification. A set of features used for classification form a feature vector. For example, here the relevant feature vector could be

Fish $x^T = [x1, x2] =$ [Lightness, Width]

Feature Space

The samples of input (when represented by their features) are represented as points in the feature space. If a single feature is used, then the feature space is one- dimensional feature space shown in fig. 2. If number of features is 2, then we get points in 2D space as shown in the Fig. 3. We can also have an n-dimensional feature space.

Fig. 2: Point Representing Samples in 1-D Space

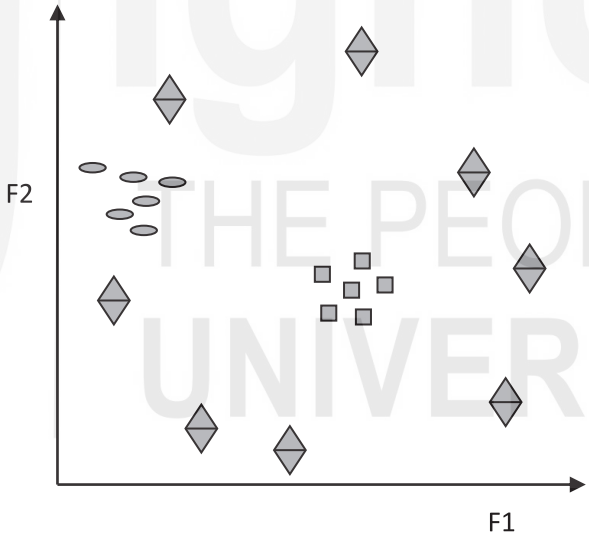
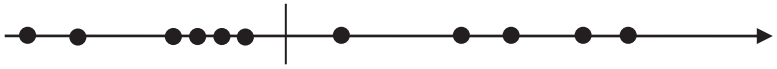


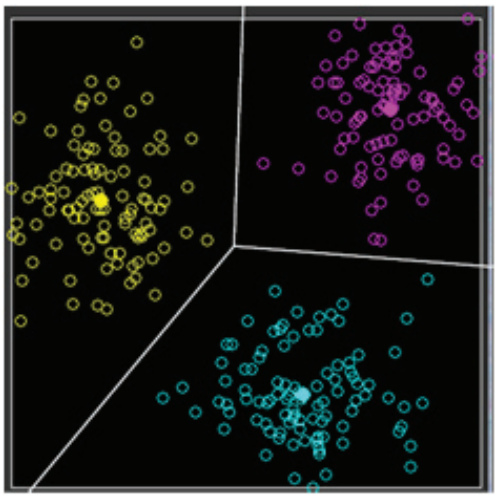
Fig. 3: Sample Points in a 2-Dimensional Feature Space

Decision Region and Decision Boundary

As we know that of pattern recognition is to reach an optimal decision rule to categorize the incoming data into their respective categories (classes). The decision boundary separates points belonging to one class from points of other. The decision boundary partitions the feature space into decision regions. The nature of the decision boundary is decided by the discriminant function which is used for decision. It is a function of the feature vector.



(a) Decision boundary in one-dimensional case with two classes



b) Decision Boundary in 2 (or 3-Dimensional Case with Three Classes

Fig. 4

Hyper Planes and Hyper Surfaces

For two category case, the decision boundary is a line $g(x) = 0$, where $g(x)$ is the discriminant function. When the coordinates of a point are substituted in $g(x)$ and this results in a positive value, we say it belongs to class 1, while a negative value decides the other class. If the number of dimensions is three, then the decision boundary will be a plane or a 3-D surface. The decision regions become semi-infinite volumes. If the number of dimensions increases to more than three, then the decision boundary becomes a hyper-plane or a hyper-surface. The decision regions become semi-infinite hyperspaces.

Learning

The classifier to be designed is built using input samples which is a mixture of all the classes. The classifier learns how to discriminate between samples of different classes. In supervised learning, the classifier is first given a set of training samples and the optimal decision boundary found, and then the classification is done.

Error

The accuracy of classification depends on two things, which are

i) The optimality of decision rule used: The central task is to find an optimal decision rule which categories the training samples correctly, and can generalise to correctly categorising unseen samples as far as possible. This decision theory leads to a minimum error-rate classification.

ii) The accuracy in measurements of feature vectors: This inaccuracy is because of presence of noise. Hence the classifier should deal with noisy and missing features too.

There are various types of classifiers used. We define them as following:

a) **Nonparametric:** Nonparametric techniques do not rely on a set of parameters/weights.

b) **Parametric:** These models are parameterized, with its parameters/ weights to be determined through some parameter optimization

algorithm, which are then determined by fitting the model to the training data set.

c) **Supervised:** The training samples are given as some input/output pairs. The output is the desired response for the input. The parameters/weights are adjusted so as to minimize the errors between the response of the networks and the desired response.

d) **Unsupervised:** Suppose that we are given data samples without being told which classes they belong to. There are schemes that are aimed to discover significant patterns in the input data without a teacher (labeled data samples).

Try the following exercises.

E2) What is a feature space?

E3) You are given set of data $S = \{$dolphin, Pomeranian dog, humming bird, frog, rattlesnake, bat$\}$. Developing a suitable classification strategy of classify the given set S according to their nature of existence.

In the following section, we shall discuss discriminant functions.

12.3 DISCRIMINANT FUNCTIONS

Discriminant functions (DFs) are useful for representing classifiers in a simpler way. If we have a set of K classes then we may define a set of K discriminant functions $y_k(x)$, one for each class. Data point 'x' is assigned to class 'c' if

$$y_c(x) > y_k(x) \qquad \text{for all } k \neq c.$$

Formulating a pattern classification problem in terms of discriminant functions is useful since it is possible to estimate the discriminant functions directly from data. Direct estimation of the decision boundaries is sometimes referred to as discriminative modeling. The choice of discriminant function may depend on prior knowledge about the patterns to be classified or may be a particular functional form whose parameters are adjusted by a training procedure. Many different forms of discriminant function have been considered in the literature, varying in complexity from the linear discriminant function (in which g is a linear combination of the x_i) to multi-parameter nonlinear functions such as the multilayer perceptron.

Here, we will discuss some discriminant functions.

Linear Discriminant Functions (LDF)

If no probability distribution or parameters are known, we can to estimate parameters of the discriminant function with Labeled data. The shape of discriminant functions is known such as shown in Fig. 5. If we have samples from 2 classes $x_1, x_2, \ldots, x_n$, we assume that 2 classes can be separated by a linear boundary $l(\theta)$ with some unknown parameters θ. Fit the "best"

boundary to data by optimizing over parameters θ by minimizing classification error on training data as shown in Fig. 6.

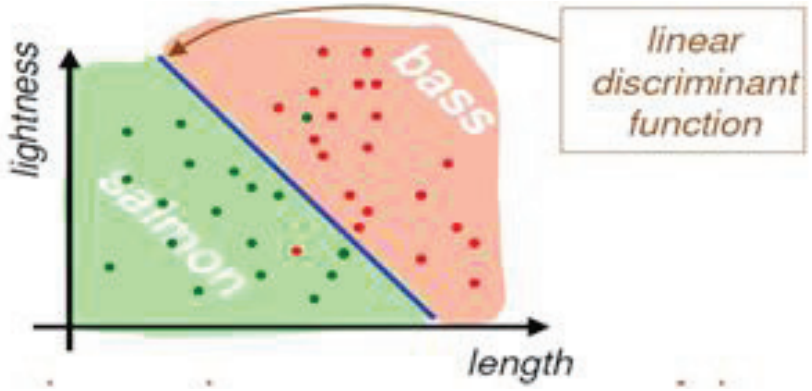


Fig. 5: Linear Discriminant Function

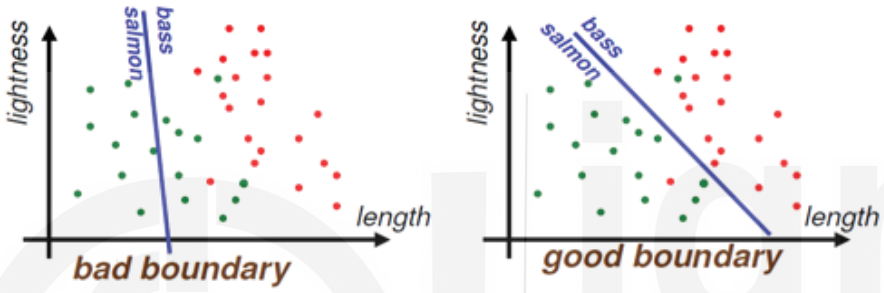


Fig. 6: Linear Discriminant Function for 2 Classes Showing 2 Boundaries

There are many different ways to representa two class pattern classifier. One way is in terms of a discriminant function $g(x)$.

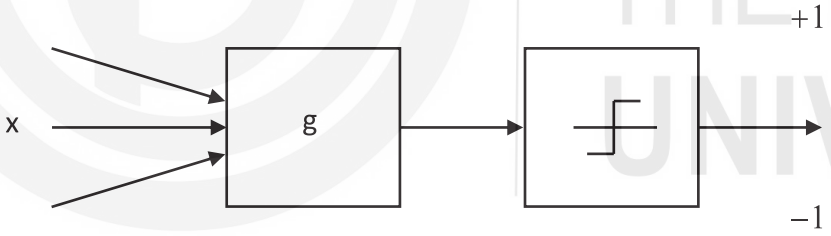


Fig. 7: Block Diagram for Linear Discriminant Function

For a new sample x and a given discriminant function, we can decide on x belongs to Class 1, if $g(x) > 0$, otherwise it belongs to class 2.

A discriminant function that is a linear combination of the components of x can be written as

$$g(x) = w^T x + W_0,$$

where w is called the weight vector and w_0 the threshold weight (also

referred to as bias). These are the parameters that we want to estimate (learn) based on training data. A classifier based entirely on linear discriminant functions is called a linear classifier or a linear machine.

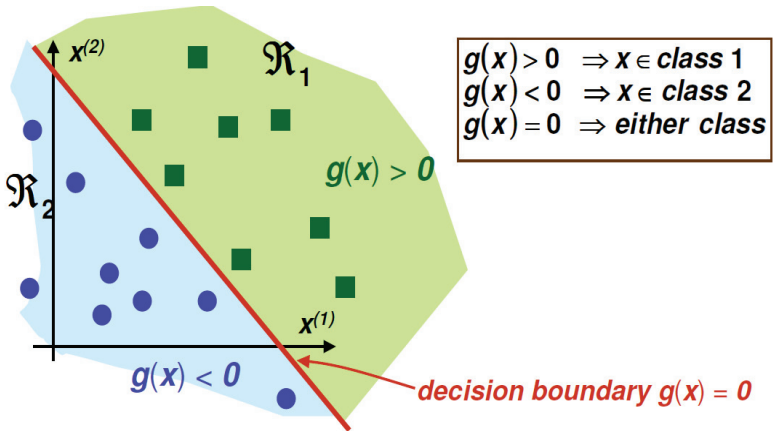
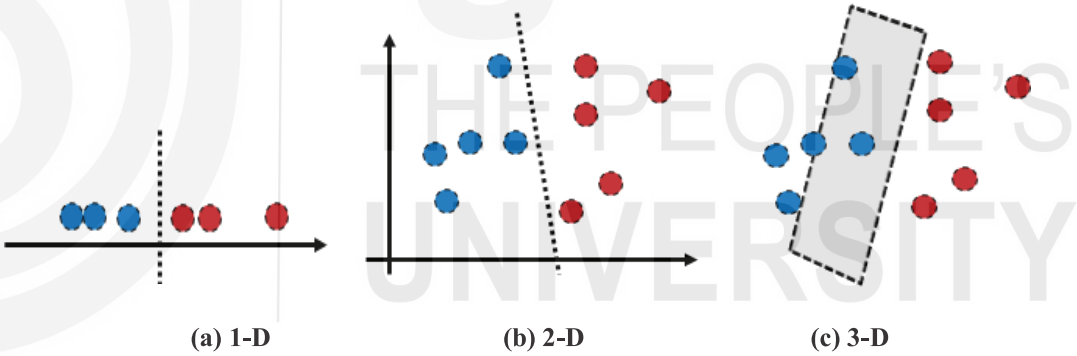


Fig. 8: Linear Discriminant Function for 2 Classes

Decision boundary surface that separates data samples assigned to Class 1 from data samples assigned to Class 2 is given by $g(x) = w^T x + w_0 = 0$.

The equation $g(x) = 0$ defines the decision boundary. This is a hyperplane when $g(x)$ is linear. Set of vectors x are chosen, for some scalars $a_0,\ldots,a_d$ satisfy $a_0 + a_1 x(1) + \cdots + a_d x(d) = 0$. A hyperplane is a point in 1-D, a line in 2-D, a plane in 3-D as shown in Fig. 9.

Let us consider the family of discriminant functions that are linear combinations of the components of $x = [x_1,\ldots:;x_p]^T$,



(a) 1-D (b) 2-D (c) 3-D

Fig. 9: Discriminant Function $g(x)$

You may see clearly that in Fig. 9(a) the discriminant function is simply a cut off, and in Fig. 9(b), the discriminant function is a line and in Fig. 9(c), the discriminant function is a plane.

$$g(x) = w^T x + w_0 = \sum_{i=1}^{p} w_i x_i + w_0$$

This is a linear discriminant function, a complete specification of which is achieved by prescribing the weight vector w and threshold weight w_0. Equation of $g(x)$ is the equation of a hyperplane with unit normal in the direction of w and a perpendicular distance $|w_0|/|w|$ from the origin. The

value of the discriminant function for a pattern x is a measure of the perpendicular distance from the hyperplane shown in Fig. 10.

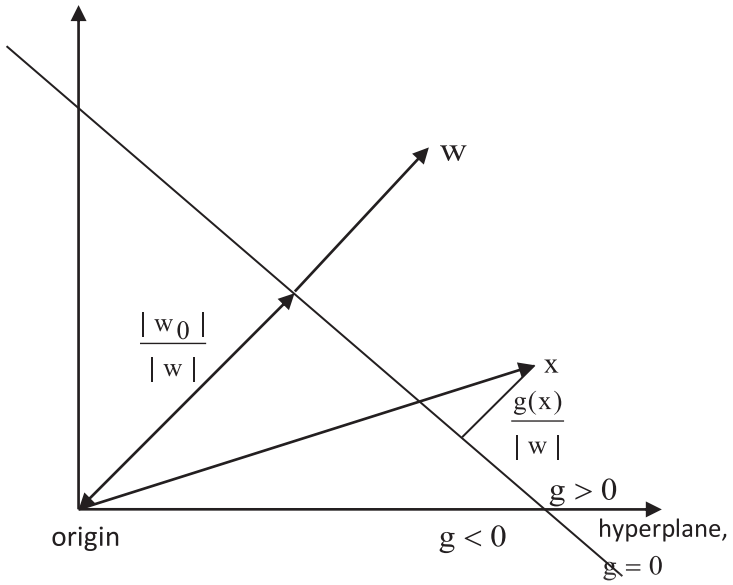


Fig. 10: Geometry of Linear Discriminant Function

Some of the properties of LDA are listed below:

- LDA assumes that the data follows a Gaussian distribution. More specifically, it assumes that all classes share the same covariance matrix.
- LDA finds linear decision boundaries in a $K-1$ dimensional subspace. As such, it is not suited if there are higher-order interactions between the independent variables.
- LDA is well-suited for multi-class problems but should be used with care when the class distribution is imbalanced because the priors are estimated from the observed counts. Thus, observations will rarely be classified to infrequent classes.
- Similarly to Principle Component Analysis (PCA), LDA can be used as a dimensionality reduction technique. Note that the transformation of LDA is inherently different to PCA because LDA is a supervised method that considers the outcomes.

The following are the situation, in which Linear Discriminant Analysis is useful.

- When the classes are well-separated, the parameter estimates for the logistic regression model are surprisingly unstable. Linear discriminant analysis does not suffer from this problem.
- If n is small and the distribution of the predictors X is approximately normal in each of the classes, the linear discriminant model is again more stable than the logistic regression model.
- Linear discriminant analysis is popular when there are more than two response classes.

Let us understand this through the following example.

Example 1: In order to select the best candidates, an over-subscribed secondary school sets an entrance exam on two subjects of English and

Mathematics. The marks of 5 applicants as listed in the Table 1 below and the decision for acceptance is passing an average mark of 75.

(i) Show that the decision rule is equivalent of the method of linear discriminant function.

(ii) Plot the decision hyperplane, indicating the half planes of both Accept and Reject, and location of the 5 applicants.

Table 1

Candidate No.	English	Math	Decision
1	80	85	Accept
2	70	60	Reject
3	50	70	Reject
4	90	70	Accept
5	85	75	Accept

Solution: i) Denote marks of English and Math as x_1 and x_2, respectively. The decision rule is as follows:

If $x_1 + x_2 > 75$, accept, otherwise reject.

This is equivalent to using a linear discriminant function.
$g(x) = x_1 + x_2 - 150$ with the following decision rule:

If $g(x) > 0$, accept, otherwise reject.

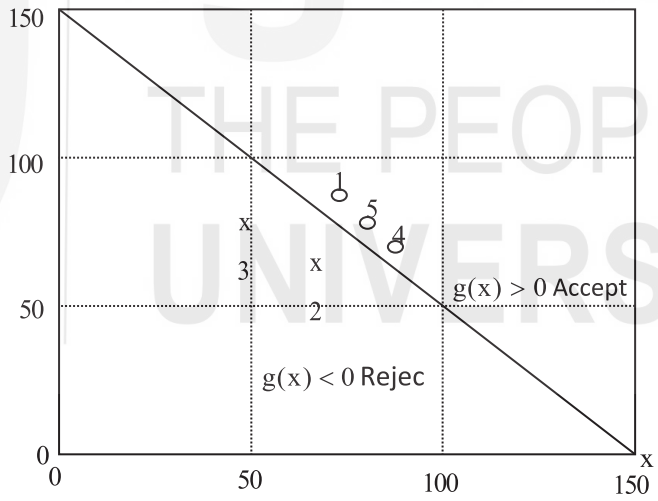


Fig. 11

(ii) To plot $g(x) = 0$, the easiest way is to set $x_1 = 0$, and find the value of x_2 so that $g(x) = 0$.

For this, $0 = 0 + x_2 - 150$, so $x_2 = 150$.

$[0,150]^T$ is on the hyperplane.

Likewise we can also set $x_2 = 0$, find the value of x_1 so that $g(x) = 0$. i.e.

$0 = x_1 + 0 - 150$, so, $x_1 = 150$. $[150,0]^T$ is on the hyperplane.

Plot a straight line linking $[0,150]^T$ and $[150,0]^T$ as shown in Fig. 11.

***

Next, we shall discuss another discriminant function.

Piecewise Linear discriminant Functions

Suppose we have ' m ' classes, define m linear discriminant functions

$$g_i(x) = w_j^t x + w_{i0} \qquad i = 1, \ldots, m$$

Given x, assign class c_i, if

$$g_i(x) \geq g_j(x) \qquad \forall j \neq i$$

Such classifier is called a linear machine that divides the feature space into c decision regions, with $g_i(x)$ being the largest discriminant if x is in the region R_i.

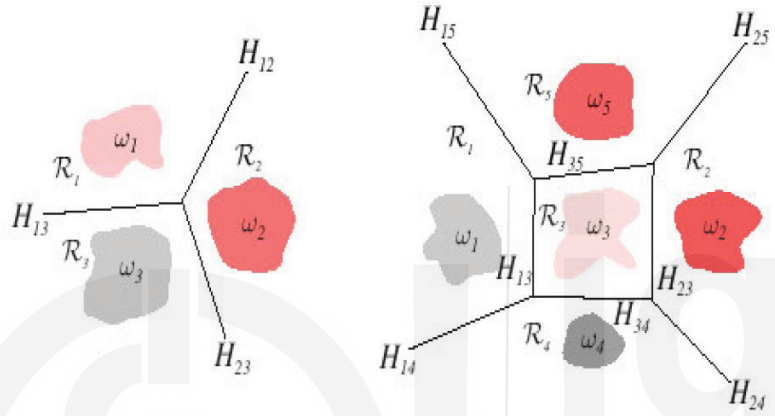


Fig. 12: Linear Discriminant Function for Multiple Classes

For a two contiguous regions R_i and R_j; the boundary that separates them is a portion of hyperplane H_{ij} defined by:

$$g_i(x) = g_j(x) \Leftrightarrow w_i^t x + w_{i0} = w_j^t x + w_{j0}$$

$$\Leftrightarrow (w_i - w_j)^t x + (w_{i0} - w_{j0}) = 0$$

Thus $w_i - w_j$ is normal to H_{ij}.. The distance from x to H_{ij} is given by

$$d(x, H_{ij}) = \frac{g_i(x) - g_j(x)}{\|w_i - w_j\|}.$$

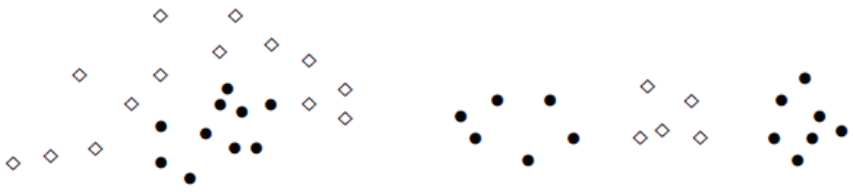


Fig. 13: Groups not Separable by a Linear Discriminant

In a multi-class problem, a pattern x is assigned to the class for which the discriminant function has the maximum value. A linear discriminant function divides the feature space by a hyperplane whose orientation is determined by the weight vector w and distance from the origin by the weight threshold w_0.

Next discriminant function is quadratic discriminant function.

Quadratic Discriminant Function

A quadratic discriminant function is a mapping

$$g: X \to R \text{ with } g(x) = \frac{1}{2} x^T W x + w^T x + w_0.$$

In quadratic discriminant function, the model parameter is $\theta = \{W; w, w_0\}$. Depending on W, the geometry of g could be convex, concave, or neither. Fig. 14 shows a quadratic discriminant function separating an inner and an outer cluster of data points.

A quadratic discriminant function is able to classify data using quadratic surfaces. This example shows an ellipsoid surface for separating an inner and outer cluster of data points. QDF is not really that much different from LDF except that you assume that the covariance matrix can be different for each class and so, we will estimate the covariance matrix separately for each class $k, k, = 1,2,\ldots,K.$

$$\delta_k(x) = -\frac{1}{2}\log\left|\sum_k\right| - \frac{1}{2}(x-\mu_k)^T \sum_k^{-1}(x-\mu_k) + \log \pi_k$$

Classification rule:

$$\hat{G}(x) = \arg\max_k \delta_k(x)$$

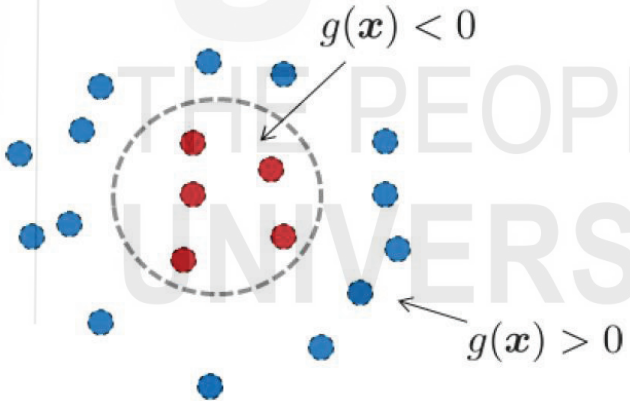


Fig. 14: Quadratic Discriminant Function

The classification rule is similar as well. You just find the class k which maximizes the quadratic discriminant function. The decision boundaries are quadratic equations in x. QDF allows more flexibility for the covariance matrix, tends to fit the data better than LDF, but then it has more parameters to estimate. The number of parameters increases significantly with QDF as a separate covariance matrix is required for every class. If you have many classes and not so many sample points, this can be a problem.

After quadratic discriminant function, let us now discuss the non-linear discriminant function.

Now, try the following exercises.

E4) Explain Linear Discremination Function.

E5) What are the properties of LDA?

In the following section, we shall discuss Bayesian classification.

12.4 BAYESIAN CLASSIFICATION

Goal of most classification procedures is to estimate the probabilities that a pattern to be classified belongs to various possible classes, based on the values of some feature or set of features.

Here, we are discussing Bayesian decision making or Bayes Classifier.

This method refers to choosing the most likely class, given the value of the feature/s. Bayes theorem calculates the probability of class membership. In most cases, we decide which is the most likely class. We need a mathematical decision making algorithm, to obtain classification.

Let us recall Bayes theorem used in probability. Accordingly to it, $P(w_i|\vec{X}) = \frac{P(\vec{X}|w_i)P(w_i)}{P(\vec{X})}$, where $P(w_i)$ is the prior probability for class $w_i, P(X)$ is the probability (Unconditional) for feature vector $X, P(w_i|X)$ is measured-conditioned or posteriori probability and $P(X|w_i)$ is the probability (Class-conditional) of feature vector X in class w_i.

$P(X)$ is the probability distribution for feature X in the entire population. It is also called unconditional density function. $P(w_i)$ is the prior probability that a random sample is a member of the class C_i.
$P(X|w_i)$ is the class conditional probability (or likelihood) of obtaining feature value X given that the sample is from class w_i. It is equal to the number of times (occurrences) of X, if it belongs to class w_i.
The goal is to measure $P(w_i|X)$, posteriori probability, from the above three values. This is the probability of any vector X being assigned to class w_i.

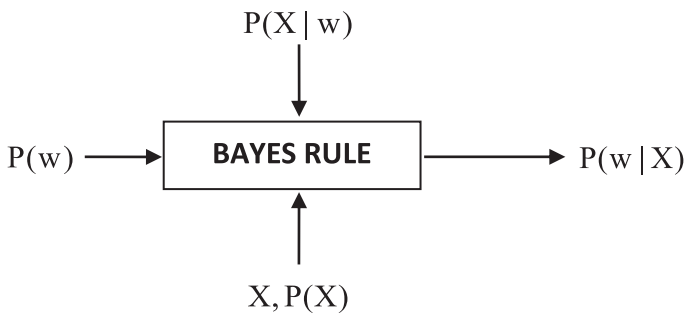


Fig. 15: Bayes Theorem

Here is another model, called the naive Bayes probabilistic model.
The probability model for a classifier is a conditional model

$$P(X|W_1,\ldots,W_n)$$

over a dependent class variable X with a small number of outcomes or classes, conditional on several feature variables W_1 through W_n. The problem is that if the number of features n is large or when a feature can take on a large number of values, then basing such a model on probability tables is infeasible. We therefore reformulate the model to make it more tractable.

Using Bayes' theorem, we write

$$P(X \mid W_1, \ldots, W_n) = \frac{p(X)\, p(W_1, \ldots, W_n \mid X)}{p(W_1, \ldots, W_n)}$$

In simple words, $\text{posterior} = \frac{\text{prior x likelihood}}{\text{evidence}}$

In practice, we are only interested in the numerator of that fraction, since the denominator does not depend on X and the values of the features W_i are given, so that the denominator is effectively constant. The numerator is equivalent to the joint probability model $p(X \mid W_1, \ldots, W_n)$, which can be rewritten as follows, using repeated applications of the definition of conditional probability:

$$\begin{aligned} p(X \mid W_1, \ldots, W_n) &= p(X)\, p(W_1, \ldots, W_n \mid X_n) \\ &= p(X)\, p(W_1 \mid X)\, p(W_2, \ldots, W_n \mid X, W_1) \\ &= p(X)\, p(W_1 \mid X)\, p(W_2 \mid X, W_1)\, p(W_3, \ldots, W_n \mid X, W_1, W_2) \\ &= p(X)\, p(W_1 \mid X)\, p(W_2 \mid X, W_1), \ldots, \\ &\quad p(W_n \mid X, W_1, W_2, \ldots, W_{n-1}) \end{aligned}$$

Now the "naive" conditional independence assumptions come into play: assume that each feature W_i is conditionally independent of every other feature W_i for $i \neq j$. This means that

$$p(W_i \mid X, W_j) = p(W_i \mid X)$$

for $i \neq j$ and the joint expression can be expressed as

$$\begin{aligned} p(X \mid W_i, \ldots, W_n) &= p(X)\, p(W_1 \mid X)\, p(W_2 \mid X) \ldots \\ &= p(C) \prod_{i=1}^{n} p(w_i \mid X). \end{aligned}$$

This means that under the above independence assumptions, the conditional distribution over the class variable can be expressed like this:

$$p(X \mid W_1, \ldots, W_n) = \frac{1}{z} p(C) \prod_{i=1}^{n} p(w_i \mid X),$$

where Z (the evidence) is a scaling factor dependent only on $W_i, \ldots W_n$, i.e., a constant, if the values of the feature variables are known.

Let us now discuss how Parameter estimation is done.

All model parameters (i.e., class priors and feature probability distributions) can be approximated with relative frequencies from the training set. These are maximum likelihood estimates of the probabilities. A class' prior may be

calculated by assuming equiprobable classes (i.e., priors = 1 / (number of classes)), or by calculating an estimate for the class probability from the training set (i.e., (prior for a given class) = (number of samples in the class) / (total number of samples)).

To estimate the parameters for a feature's distribution, one must assume a distribution or generate nonparametric models for the features from the training set. If one is dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution.

For example, suppose the training data contains a continuous attribute, x. We first segment the data by the class and then compute the mean and variance of x in each class. Let μ_c be the mean of the values in x associated with class c, and let σ_c^2 be the variance of the values x in associated with class c. Then, the probability of some value given a class, $p(x = v \mid c)$, can be computed by plugging into the equation for a Normal distribution parameterized by μ_c and σ_c^2. That is,

$$p(x = v \mid c) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(v-\mu_c^2)^2}{2\sigma_c^2}}$$

Another common technique for handling continuous values is to use binning to discretize the values. In general, the distribution method is a better choice if there is a small amount of training data, or if the precise distribution of the data is known. The discretization method tends to do better if there is a large amount of training data because it will learn to fit the distribution of the data. Since naive Bayes is typically used when a large amount of data is available (as more computationally expensive models can generally achieve better accuracy), the discretization method is generally preferred over the distribution method.

If a given class and feature value never occurs together in the training set then the frequency-based probability estimate will be zero. This is problematic since it will wipe out all information in the other probabilities when they are multiplied. It is therefore often desirable to incorporate a small-sample correction in all probability estimates such that no probability is ever set to be exactly zero.

Now, we shall Construct a classifier from the probability model. The discussion so far has derived the independent feature model, that is, the naive Bayes probability model. The naive Bayes classifier combines this model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the *maximum a posteriori* or *MAP* decision rule. The corresponding classifier is the function classify defined as follows:

$$\text{classify}(W_1, \ldots, W_n) = \arg\max p(C = c) \prod_{i=1}^{n} p(W_i = w_i \mid C = c)$$

Despite the fact that the far-reaching independence assumptions are often inaccurate, the naive Bayes classifier has several properties that make it surprisingly useful in practice. In particular, the decoupling of the class

conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution.

This in turn helps to alleviate problems stemming from the curse of dimensionality, such as the need for data sets that scale exponentially with the number of features. Like all probabilistic classifiers under the MAP decision rule, it arrives at the correct classification as long as the correct class is more probable than any other class; hence class probabilities do not have to be estimated very well. In other words, the overall classifier is robust enough to ignore serious deficiencies in its underlying naive probability model.

Properties of Bayes Classifiers

1. **Incrementality:** with each training example, the prior and the likelihood can be updated dynamically. It is flexible and robust to errors.
2. **Combines prior knowledge and observed data:** prior probability of a hypothesis is multiplied with probability of the hypothesis given the training data.
3. **Probabilistic hypotheses:** outputs is not only a classification, but a probability distribution over all classes.
4. **Meta-classification:** the outputs of several classifiers can be combined, e.g., by multiplying the probabilities that all classifiers predict for a given class.

Now, let us see the following examples.

Example 2: (Two class problem): Let us define variables, Cold (C) and not-cold (C'). Feature is fever (f). Prior probability of a person having a cold, $P(C) = 0.01$. Prob. of having a fever, given that a person has a cold is, $P(f \mid C) = 0.4$. Overall prob. of fever $P(f) = 0.02$. Find the Prob. that a person has a cold, given that she (or he) has a fever.

Solution: Using Bayes theorem, the Prob. that a person has a cold, given that she (or he) has a fever is:

$$P(C \mid f) = \frac{p(f \mid c)\,P(c)}{P(f)} = \frac{0.4 * 0.001}{0.02} = 0.2$$

let us take an example with values to verify:

Total Population =1000.

Thus, people hav ing cold = 10.

People having both fever and cold = 4.

Thus, people having only cold = 10 – 4 = 6.

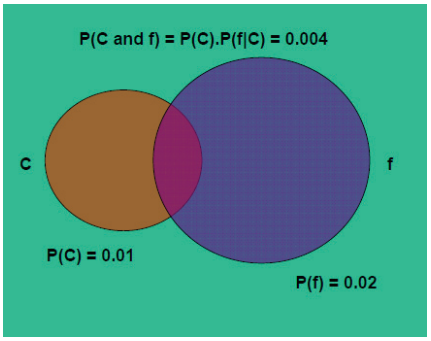


Fig. 15: A Venn Diagram

People having fever (with and without cold) = 0.02 * 1000 = 20.

People having fever without cold = 20 – 4 = 16

So, probability (percentage) of people having cold along with fever, out of all those having fever, is $= 4/20 = 0.2(20\%)$.

Probability of a joint event - a sample comes from class C and has the feature value X:

$$P(C \text{ and } X) = P(C).P(X|C)$$
$$= 0.01 * 0.4$$

Or, $P(C \text{ and } X) = P(X).P(C|X)$
$$= 0.02 * 0.2$$

Also verify, for a K class problem:

$$P(X) = P(w_1)P(X|w_1) + P(w_2)P(X|w_2) + \cdots + P(w_k)P(X|w_k)$$

$$P(w_i|\vec{X}) = \frac{P(\vec{X}|w_i)P(w_i)}{P(w_1)P(X|w_1) + P(w_2)P(X|w_2) + \cdots + P(w_k)P(X|w_k)}$$

We get

$$P(f) = P(C)P(f|C) + P(C')P(f|C')$$
$$= 0.01 * 0.4 + 0.99 * 0.01616 = 0.02$$

***

A Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions. In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

An advantage of the naive Bayes classifier is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

Try following exercises.

E6) Explain Bayes classifier.

E7) Explain properties of Bayes classifier.

12.5 MINIMUM DISTANCE CLASSIFIERS

Minimum distance classifier is a pattern classification scheme defined by distance functions. Here pixels which are close to each other in feature space are likely to belong to the same class. The measure of similarity is the "distance" between pixels in feature space (n-D histogram). All dimensions should be in comparable units and distance may be scaled in pixels, radiance, reflectance etc. the classification is most effective if the clusters are disjoint. It requires the least amount of prior information to operate.

Distance in feature space is the primary measure of similarity in minimum distance classifier algorithms.Pixels that are "close" in feature space willbe grouped in the same class.The relative distances may change when data are calibrated, atmospherically corrected or rescaled in ways that treat different features differently. If two features have different units, theymust be scaled to provide comparable variance. Otherwise the "distance" will be biased.

Distance in feature space is the primary measure of similarity in all clustering algorithms.

1) Euclidean distance:

$$d_i(x) = -D_i(x) = -[(x - z_i)^T (x - z_i)]^{1/2}$$

2) Square of the Euclidean distance:

$$d_i^2(x) = -D_i^2(x) = -[(x - z_i)^T (x - z_i)]$$

3) Square of Euclidean distance after eliminating components that are independent of class:

$$d_i(x) = -[x^T z_i + 1/2(z_i^T z_i)]$$

4) Taxicab distance

$$d_i(x) = -D_i(x) = -\sum_{k=1}^{N_b} | X_k - Z_{ki} |$$

The **decision boundary** for the single prototype, simple distance discriminant function is the set of planar surfaces perpendicular to and bisecting the lines connecting pairs of prototypes as shown in Fig. 1. This is a minimum-distance classifier. If the prototype is the mean value of the training pixels for each class, it is called a minimum-distance-to-mean classifier.

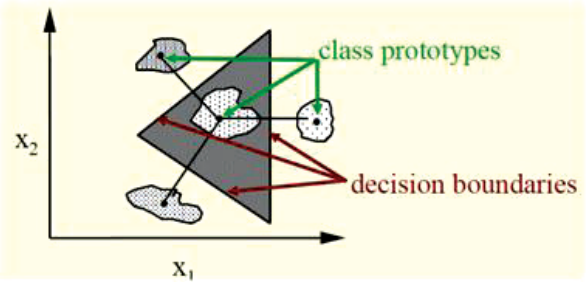


Fig. 1: Decision boundary

The results of clustering will depend strongly on the choice of the prototype. Alternatives for prototype selection are

1. Let the user select prototypes, i.e., one "example" pixel per class. (Reduces the utility of a clustering procedure.)
2. Devise an unbiased procedure for selecting prototypes (random selection, selection at vertices of an arbitrary grid etc)
3. Use the user-selected prototype or unbiased selection procedure as the starting point of an optimization procedure.

We shall discuss Euclidean distance classifier and Mahalanobis distance classifiers here.

The Euclidean Distance Classifier

The optimal Bayesian classifier is significantly simplified under the following assumptions:

- The classes are equiprobable.
- The data in all classes follow Gaussian distributions.
- The covariance matrix is the same for all classes.
- The covariance matrix is diagonal and all elements across the diagonal are equal. That is, $S = \sigma^2 I$, where I is the identity matrix.

Under these assumptions, it turns out that the optimal Bayesian classifier is equivalent to the minimum Euclidean distance classifier.
Each class (pattern) is represented by a single prototype vector, z. Assume that there are ' m ' classes and that these classes are represented by the prototype vectors, $z_1, z_2, z_3, \ldots, z_m$.

The **Euclidean distance**, $D_i(x)$, of a measurement vector, x, from the prototype vector, z_i:

$$D_i(x) = |x - z_i| = [(x - z_i)^T (x - z_i)]^{1/2}$$

The discriminant function is usually defined as the negative of the separation distance:

$$d_i(x) = -D_i(x)$$

The larger (less negative) $d_i(x)$, the closer the measurement vector lies relative to the prototype vector z_i. The maximum value of $d_i(x)$ is zero and occurs when x matches the prototype vector exactly.

Algorithm

Step 1: Select a threshold, T. T is a representative distance in measurement space. The choice of T in this algorithm is entirely arbitrary; it is also the only input required of the user.

Step 2: Select a pixel with measurement vector, x. The selection scheme is arbitrary. Pixels could be selected at random.

Step 3: Let the first pixel be taken as the first cluster center, z_1.

Step 4: Select the next pixel from the image.

Step 5: Compute the distance functions, $D_i(x)$. Compute the distance function for each of the classes established at this point, i.e., compute $D_i(x)$, for $i = 1, \ldots, N$ where $N =$ the number of classes. ($N = 1$ initially.)

Step 6: Compare the $D_i(x)$ with T.

a) if $D_i(x) < T$, then $x \in w_i$.

b) if $D_i(x) < T$, for all i, then let x become a new prototype vector: Assign $x \rightarrow z_{N+1}$. (Do not compute D_{N+1} for pixels already assigned to an existing class.)

Step 7: Return to step #4 until all pixels are assigned to a class.

Step 8: After all pixels have been assigned to a cluster center, re-compute the $D_i(x)$ and reassign pixels according to the minimum $D_i(x)$.

This simple clustering algorithm is extremely sensitive to the value of the threshold, T, and the order in which pixels are selected. For a given T, two different selection patterns can yield very different results. For the same selection pattern, a different value of T T will lead to different results. These flaws are typical of clustering algorithms. All are sensitive to the starting selection of cluster centers and to the particular specification of the clustering criterion. The better algorithms handle the problems cleverly and without the severe problems that would be apparent with the above algorithm.

That is, given an unknown x, assign it to class ω_i if

$$\|x - m_i\| \equiv \sqrt{(x - m_i)^T (x - m_i)} < \|x - m_j\|, \quad \forall_i \neq j$$

It must be stated that the Euclidean classifier is often used, even if we know that the previously stated assumptions are not valid, because of its simplicity. It assigns a pattern to the class whose meanis closest to it with respect to the Euclidean norm.

The Mahalanobis Distance Classifier

In many appliacations, the range of all feature value may differ widely. One could be in hundreds while the other could be in decimal fractions. If this issue is overlooked some feature values will get neglected. If one relaxes the assumptions required by the Euclidean classifier and removes the last one, the one requiring the covariance matrix to be diagonal and with equal elements, the optimal Bayesian classifier takes the form of minimum Mahalanobis distance classifier. That is, given an unknown x, it is assigned to class ω_i if

$$\sqrt{(x-m_i)^T S^{-1}(x=m_i)} < \sqrt{(x-m_j)^T S^{-1}(x-m_j)}, \quad \forall_j \neq i,$$

where S is the common covariance matrix. The presence of the covariance matrix accounts for the shape of the Gaussians distributions of various features.

Try the following exercises.

E8) What distance measure is used by Euclidean Distance Classifier?

E9) What is the discriminant function for Euclidean distance classifier?

In the following section, we shall discuss Machine learning algorithm.

12.6 MACHINE LEARNING ALGORITHMS

In order to understand how Machine learning works, we need some essential ML vocabulary first:

- ***Parameter***: A value that is learnt during the training of a machine learning program. Based on existing parameters, the training algorithm "trains" and tries to continuously improve the parameters. The inference algorithm uses the parameters to calculate results.
- ***Model***: A trained machine learning method. A model is essentially a (sometimes very large) set of parameters and instructions on how to use them – a ready-to-run ML procedure and the result of the training process. The training algorithm generates the model, the inference algorithm uses it to perform a task.
- ***Inference***: Running a model using a second algorithm. This algorithm uses the model to create a prediction based on an input, classify an input or create some interesting value based on an input. Hence the term "inference algorithm" or "prediction algorithm". Inference is a technical term for "execute the ML model and output a result".

As we see, Machine learning use the combination of a training algorithm and a prediction (or inference) algorithm. The training algorithm uses data to gradually determine parameters. The set of all learned parameters is called a model, basically a "set of rules" established by the algorithm, applicable even to unknown data. The inference algorithm then uses the model and applies it to any given data. Finally, it delivers the desired results.

Equipped with the right vocabulary, we can take a closer look at the execution of a machine learning project:

- We select the machine learning method for which we want to train a model. The choice will depend on the problem to be solved, the available data, the experience and also on gut feeling.
- Then we divide the available data into two parts: The training data and the test data. We apply our training data and thus obtain our model. The model is checked on the unknown test data. It is most important that the

test data aren't used during the training phase under any circumstances. The reason is obvious: Computers are great at learning by heart. Complex models like neural networks can actually start to memorize by themselves. The following results might be quite remarkable. There's only one flaw: They're not based on a model formulated by the program, but on "memorized" data. This effect is called "over fitting". However, the test data are supposed to simulate the "unknown" during quality control and to check whether the model has really "learned" something. A good model achieves about the same error rate on the test data as on the training data without ever having seen it before.

- We use the training data to develop the model with the learning algorithm. The more data we have, the "stronger" the model becomes. Using up all available data for the training algorithm is called an "epoch".
- In order to test it, the trained model is used on the test data unknown to it and makes predictions. If we did everything right, the predictions on unknown data should be as good as on the training data – the model can generalize and solve the problem. Now it is ready for practical use.
- At its most basic, machine learning uses programmed algorithms that receive and analyse input data to predict output values within an acceptable range. As new data is fed to these algorithms, they learn and optimise their operations to improve performance, developing 'intelligence' over time.

Machine learning algorithms are organized into taxonomy, based on the desired outcome of the algorithm. Common algorithm type includes:

- **Supervised learning ---** the algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: the learner is required to learn (to approximate the behaviour of) a function which maps a vector into one of several classes by looking at several input-output examples of the function.

The performance and computational analysis of machine learning algorithms is a branch of statistics known as computational learning theory. Machine learning is about designing algorithms that allow a computer to learn. Learning is not necessarily involves consciousness but learning is a matter of finding statistical regularities or other patterns in the data. Thus, many machine learning algorithms will barely resemble how human might approach a learning task. However, learning algorithms can give insight into the relative difficulty of learning in different environments.

Now we shall discuss supervised learning approach in detail in the following section.

12.7 SUPERVISED LEARNING APPROACH

Supervised learning is fairly common in classification problems because the goal is often to get the computer to learn a classification system that we have created. Digit recognition, once again, is a common example of classification learning. More generally, classification learning is appropriate for any problem where deducing a classification is useful and the classification is easy to determine. In some cases, it might not even be necessary to give

predetermined classifications to every instance of a problem if the agent can work out the classifications for itself. This would be an example of unsupervised learning in a classification context.

Supervised learning often leaves the probability for inputs undefined. This model is not needed as long as the inputs are available, but if some of the input values are missing, it is not possible to infer anything about the outputs. Unsupervised learning, all the observations are assumed to be caused by latent variables, that is, the observations is assumed to be at the end of the causal chain. Examples of supervised learning and unsupervised learning are shown in the Fig. 2.

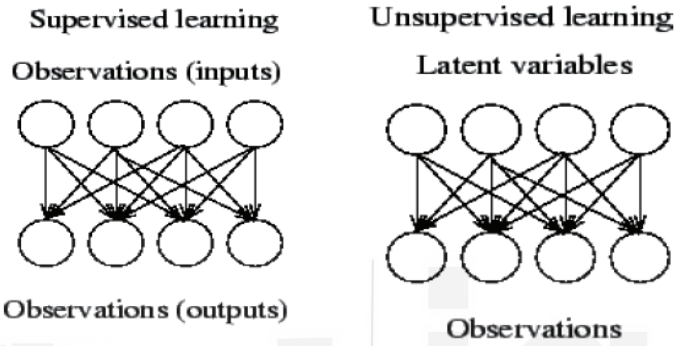


Fig. 2: Examples of Supervised and Unsupervised Learning

Supervised learning is the most common technique for training neural networks and decision trees. Both of these techniques are highly dependent on the information given by the pre-determined classifications. In the case of neural networks, the classification is used to determine the error of the network and then adjust the network to minimize it, and in decision trees, the classifications are used to determine what attributes provide the most information that can be used to solve the classification puzzle. Both of these examples have some "supervision" in the form of pre-determined classifications.

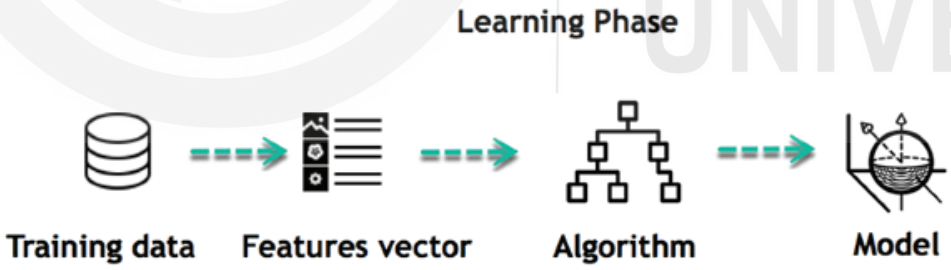


Fig. 3: Learning phase of a supervised learning algorithm

Inductive machine learning is the process of learning a set of rules from instances (examples in a training set), or more generally speaking, creating a classifier that can be used to generalize from new instances. The process of applying supervised ML to a real world problem is described below

Step 1 **(Collect the dataset):** If a requisite expert is available, then s/he could suggest which fields (attributes, features) are the most informative. If not, then the simplest method is that of measuring everything available in the hope that the right (informative, relevant) features can be isolated.

Step 2 **(Data preparation and data pre-processing):** Depending on the circumstances, there are a number of methods to choose from to handle missing data, outlier (noise) detection. There is a variety of procedures for sampling instances from a large dataset. Feature subset selection is the process of identifying and removing as many irrelevant and redundant features as possible. This reduces the dimensionality of the data and enables data mining algorithms to operate faster and more effectively.

Step 3 **(Define a training set):** The goal of the learning algorithm is to minimize the error with respect to the given inputs. These inputs, often called the "training set", are the examples from which the agent tries to learn. But learning the training set well is not necessarily the best thing to do. For instance, if I tried to teach you exclusive-or, but only showed you combinations consisting of one true and one false, but never both false or both true, you might learn the rule that the answer is always true. Similarly, with machine learning algorithms, a common problem is over-fitting the data and essentially memorizing the training set rather than learning a more general classification technique. As you might imagine, not all training sets have the inputs classified correctly. This can lead to problems if the algorithm used is powerful enough to memorize even the apparently "special cases" that don't fit the more general principles. This, too, can lead to over fitting, and it is a challenge to find algorithms that are both powerful enough to learn complex functions and robust enough to produce generalizable results.

Step 4: Then we need to select a suitable algorithm and perform training and evaluate the results.

Step 5: If we are satisfied with the results, classifier is trained else parameter tuning is done which includes updating of data preprocessing and training set.

Supervised Machine Learning Categorization

It is important to remember that all supervised learning algorithms are essentially complex algorithms, categorized as either classification or regression models as shown in Fig. 4.

1) **Classification Models —** Classification models are used for problems where the output variable can be categorized, such as “Yes” or “No”, or “Pass” or “Fail.” Classification Models are used to predict the category of the data. Real-life examples include spam detection, sentiment analysis, scorecard prediction of exams, etc.

2) **Regression Models —** Regression models are used for problems where the output variable is a real value such as a unique number, dollars, salary, weight or pressure, for example. It is most often used to predict numerical values based on previous data observations. Some of the more familiar regression algorithms include linear regression, logistic regression, polynomial regression, and ridge regression.

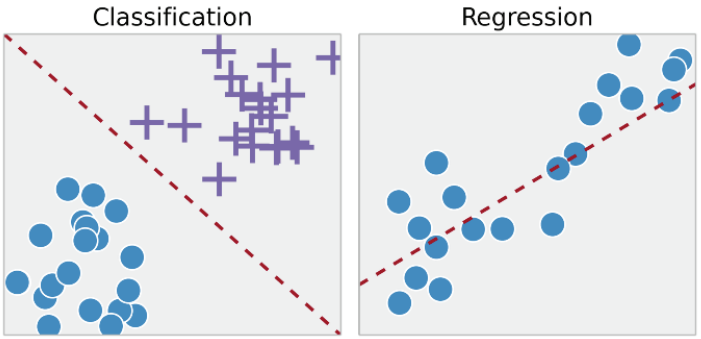


Fig. 4: Categories of Supervised Machine Learning algorithms

There are some very practical applications of supervised learning algorithms in real life, including:

- Text categorization
- Face Detection
- Signature recognition
- Customer discovery
- Spam detection
- Weather forecasting
- Predicting housing prices based on the prevailing market price
- Stock price predictions, among others

Try the following exercises.

E10) What are the different categories of supervised machine learning algorithms?

E11) What are different applications of supervised machine learning algorithms?

Let us list the Differences between Supervised and Unsupervised Learning Approaches.

Parameters	**Supervised machine learning technique**	**Unsupervised machine learning technique**
Process	In a supervised learning model, input and output variables will be given.	In unsupervised learning model, only input data will be given
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data which is not labeled
Algorithms Used	Support vector machine, Neural network, Linear and logistics regression, random forest, and Classification trees.	Unsupervised algorithms can be divided into different categories: like Cluster algorithms, K-means, Hierarchical

		clustering, etc.
Computational Complexity	Supervised learning is a simpler method.	Unsupervised learning is computationally complex
Use of Data	Supervised learning model uses training data to learn a link between the input and the outputs.	Unsupervised learning does not use output data.
Accuracy of Results	Highly accurate and trustworthy method.	Less accurate and trustworthy method.
Real Time Learning	Learning method takes place offline.	Learning method takes place in real time.
Number of Classes	Number of classes is known.	Number of classes is not known.
Main Drawback	Classifying big data can be a real challenge in Supervised Learning.	You cannot get precise information regarding data sorting, and the output as data used in unsupervised learning is labeled and not known.

The following are the various situations, where supervised or unsupervised learning approaches are used.

- In Supervised learning, the machine is trained using data which is well "labeled."
- Unsupervised learning is a machine learning technique, where you do not need to supervise the model.
- Supervised learning allows you to collect data or produce a data output from the previous experience.
- Unsupervised machine learning helps you to finds all kind of unknown patterns in data.
- For example, you will able to determine the time taken to reach back come base on weather condition, Times of the day and holiday.
- For example, Baby can identify other dogs based on past supervised learning.
- Regression and Classification are two types of supervised machine learning techniques.
- Clustering and Association are two types of Unsupervised learning.
- In a supervised learning model, input and output variables will be given while with unsupervised learning model, only input data will be given

Now the question arises, When to Choose Supervised Learning / Unsupervised Learning?

In manufacturing, a large number of factors affect which machine learning approach is best for any given task. And, since every machine learning problem is different, deciding on which technique to use is a complex process.

In general, a good strategy for honing in on the right machine learning approach is to:

- **Evaluate the data.** Is it labeled/unlabelled? Is there available expert knowledge to support additional labeling? This will help to determine whether a supervised, unsupervised, semi-supervised or reinforced learning approach should be used
- **Define the goal.** Is the problem recurring, defined one? Or, will the algorithm be expected to predict new problems?
- **Review available algorithms** that may suit the problem with regards to dimensionality (number of features, attributes or characteristics). Candidate algorithms should be suited to the overall volume of data and its structure
- **Study successful applications** of the algorithm type on similar problems

Let us now summarize, what we have discussed in this unit.

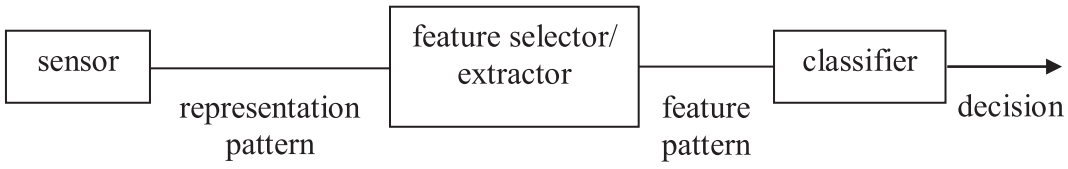
12.8 SUMMARY

We have discussed the following points.

1) Concept of Classification.
2) Various terminologies in pattern recognition problem.
3) Linear discriminant function, piecewise discriminant function, quadratic discrimint function and non-linear discriminant function.
4) Bayes classifier combines prior knowledge with observed data: assigns a posterior probability to a class based on its prior probability and its likelihood given the training data. It computes the maximum a posterior (MAP) hypothesis or the maximum likelihood (ML) hypothesis.
5) Naive Bayes classifier assumes conditional independence between attributes and assigns the MAP class to new instances.
6) Likelihoods can be estimated based on frequencies. Sparse data poses a huge problem. A possible solution is to use m -estimate.
7) Concept of minimum distance classifier.
8) Supervised Learning.
9) Unsupervised Learning.
10) Differences between Supervised Learning/ Unsupervised Learning.

12.9 SOLUTION/ANSWERS

E1)

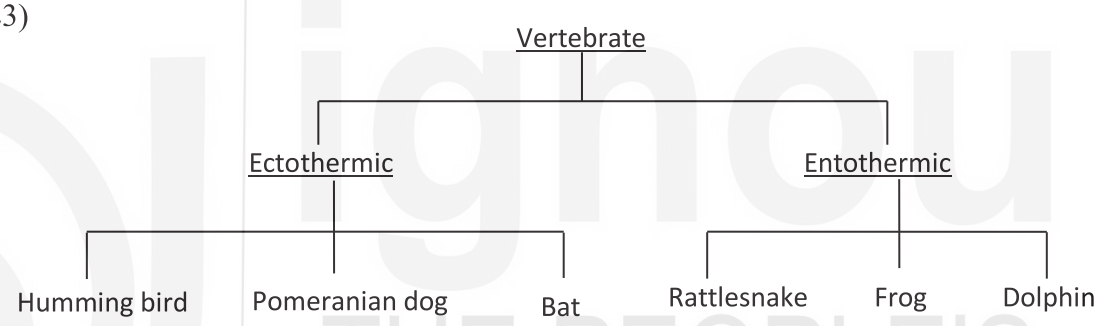


Optical sensing is used to distinguish two patterns. A camera takes pictures of the object and passes to on to a feature extractor. The feature extractor reduces the data by measuring certain "properties" that distinguish pictures of one object to the other. These features are then passed to a classifier that evaluates the evidence presented and makes a final decision about the object type.

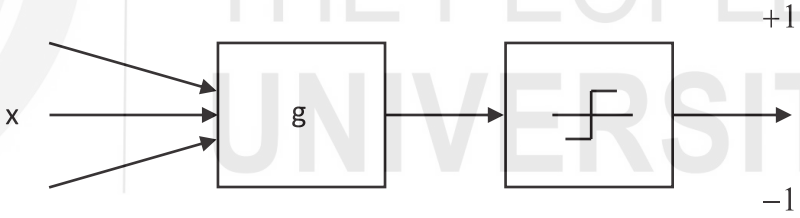
One characteristic of human pattern recognition is that it involves a teacher. Similarly a machine pattern recognition system needs to be trained. A common mode of learning is to be given a collection of labeled examples, known as training data set. From the training data set, structure information is distilled and used for classifying new inputs.

E2) The samples of input (when represented by their features) are represented as points in the feature space. If a single feature is used, then work on a one- dimensional feature space. If number of features is 2, then we get points in 2D space. We can also have an n-dimensional feature space.

E3)



E4)



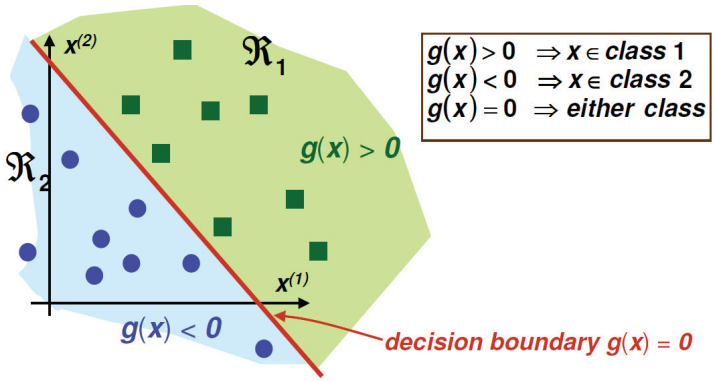
For a new sample x and a given discriminant function, we can decide on x belongs to

Class1 if $g(x) > 0$, otherwise it's Class 2.

A discriminant function that is a linear combination of the components of x can be written as

$$g(x) = w^T x + w_0$$

where w is called the weight vector and w_0 the threshold weight. These are the parameters that we want to estimate based on training data. A classifier based entirely on linear discriminant functions is called a linear classifier or a linear machine.



E5) LDF assumes that the data are Gaussian. More specifically, it assumes that all classes share the same covariance matrix.

- LDF finds linear decision boundaries in a $K-1$ dimensional subspace. As such, it is not suited if there are higher-order interactions between the independent variables.
- LDF is well-suited for multi-class problems but should be used with care when the class distribution is imbalanced because the priors are estimated from the observed counts. Thus, observations will rarely be classified to infrequent classes.
- Similarly to PCA, LDA can be used as a dimensionality reduction technique. Note that the transformation of LDA is inherently different to PCA because LDA is a supervised method that considers the outcomes.

E6) The naive Bayes classifier combines bayes model with a decision rule. One common rule is to pick the hypothesis that is most probable; this is known as the *maximum a posteriori* or *MAP* decision rule. The corresponding classifier is the function classify defined as follows:

$$\text{classify}(W_1,\ldots,W_n) = \arg\max\, p(C=c)\prod_{i=1}^{n} p(W_i = w_i \mid C = c)$$

Despite the fact that the far-reaching independence assumptions are often inaccurate, the naive Bayes classifier has several properties that make it surprisingly useful in practice. In particular, the decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution.

E7)

- **Incrementality:** with each training example, the prior and the likelihood can be updated dynamically. It is flexible and robust to errors.
- **Combines prior knowledge and observed data:** prior probability of a hypothesis is multiplied with probability of the hypothesis given the training data.
- **Probabilistic hypotheses:** outputs is not only a classification, but a probability distribution over all classes.
- **Meta-classification:** the outputs of several classifiers can be combined, e.g., by multiplying the probabilities that all classifiers predict for a given class.

E8) The **Euclidean distance**, $D_i(x)$, of a measurement vector, x, from the prototype vector, Z_i:

$$D_i(x) = -|(x - z_i| = [(x - z_i)^T (x - z_i)]^{1/2}$$

E9) The discriminant function is usually defined as the negative of the separation distance:
$d_i(x) = -D_i(x)$

E10) Supervised learning algorithms are essentially complex algorithms, categorized as either classification or regression models

1) **Classification Models** — Classification models are used for problems where the output variable can be categorized, such as "Yes" or "No", or "Pass" or "Fail." Classification Models are used to predict the category of the data. Real-life examples include spam detection, sentiment analysis, scorecard prediction of exams, etc.

2) **Regression Models** — Regression models are used for problems where the output variable is a real value such as a unique number, dollars, salary, weight or pressure, for example. It is most often used to predict numerical values based on previous data observations. Some of the more familiar regression algorithms include linear regression, logistic regression, polynomial regression, and ridge regression.

E11) Applications of supervised learning algorithms in real life, including:

- Text categorization
- Face Detection
- Signature recognition
- Customer discovery
- Spam detection
- Weather forecasting
- Predicting housing prices based on the prevailing market price
- Stock price predictions, among others.

UNIT 13 OBJECT CLASSIFICATION USING UNSUPERVISED LEARNING

Structure **Page No.**

13.1 INTRODUCTION

Clustering is a technique for identifying similarity groups in data, called clusters. i.e., it groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters. Clustering is often called an unsupervised learning task as no class values denoting an a priori grouping of the data instances are specified beforehand, as is done in supervised learning. Due to historical reasons, clustering is often considered synonymous with unsupervised learning.

And now, we will list the objectives of this unit. After going through the unit, please read this list again make sure you have achieved the objectives.

Objectives

After studying this unit, you should be able to

- define clustering
- define and use different clustering techniques
- apply Hierarchial Clustering
- use partition based clustering,
- apply K - NN clustering

13.2 INTRODUCTION TO CLUSTERING

The problem of pattern clustering may be regarded as one of discriminating the input data, not between individual patterns, but between populations. These populations are searched for a match with the new object with the help of its features. The objective is to categorize patterns into classes so that patterns belonging to different classes are well separated. The process may start with or without any knowledge of the feature space. When we have prior knowledge about the class of a subset of data, it is called **supervised classification or classification**. When nothing is known a prior, the scheme

is called **unsupervised classification or clustering**. In a case of supervised classification, the labelled subset of data is called **training data**.

Being unsupervised in nature, clustering is a very difficult task, as the same data may reveal many different inherent structures depending on the shape and size of its distribution.

A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way". A cluster is therefore a collection of objects which are "similar to each other and are "dissimilar" to the objects belonging to other clusters. Cluster analysis is also used to form descriptive statistics to ascertain whether or not the data consists of a set distinct subgroups, each group representing objects with substantially different properties. The latter goal requires an assessment of the degree of difference between the objects assigned to the respective clusters.

The notion of cluster is not well defined. To better understand the difficulty of deciding what constitutes a cluster, consider Fig. 1, which shows twenty points and three different ways of dividing them into clusters. The shapes of the markers indicate cluster membership. Fig. 1(b) and Fig.1 (d) divide the data into two and six parts, respectively. The apparent division of each of the two larger clusters into 3 sub-clusters may simply be an artifact of human visual system. In Fig. 1(c), there are four clusters. The fig illustrates that the definition of a cluster is imprecise and that the best definition depends on the nature of data and the desired result.

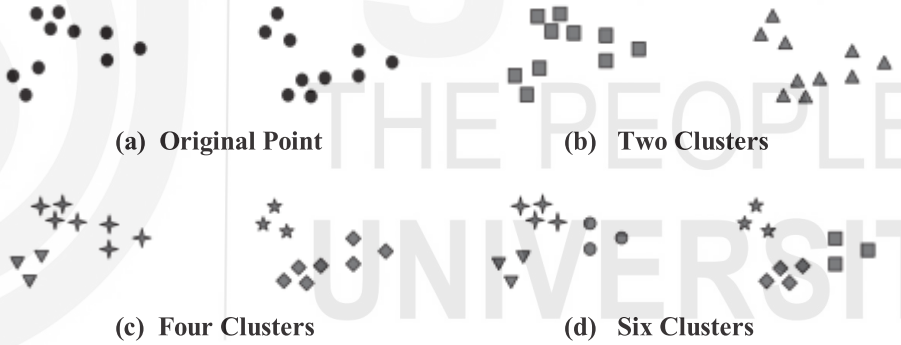


Fig. 1: Different ways of Clustering Same Data Points

The various examples of clustering applications are as follows:

1) **Marketing**: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs.
2) **Land use**: Identification of areas of similar land use in an earth observation database.
3) **Insurance**: Identifying groups of motor insurance policy holders with a high average claim cost.
4) **City-planning**: Identifying groups of houses according to their house type, value, and geographical location.
5) **Earth-quake studies**: Observed earth quake epicenters should be clustered along continent faults.
6) **Image processing**: Clustering parts of image having similar RGB

values, so that image is clustered into regions such as sky, greenery, road, house, etc.

In fact, clustering is one of the most utilized data mining techniques. It has a long history, and used in almost every field, e.g., medicine, psychology, botany, sociology, biology, archeology, marketing, insurance, libraries, etc. In recent years, due to the rapid increase of online documents, text clustering becomes important.

Let us see some real-life examples of clustering.

Example 1: Groups people of similar size to gather to make "small", "medium" and "large" T-Shirts.

– Tailor-made for each person: too expensive

– One-size-fits-all: does not fit all.

***

Example 2: In marketing, segment customers according to their similarities

– To do targeted marketing.

***

Example 3: Given a collection of text documents, we want to organize them according to their content similarities,

– To produce a topic hierarchy

***

See, how quality of clustering is decided?

A **good clustering** method will produce high quality clusters with

a) high intra-class similarity (in the same class)
b) low inter-class similarity (between two classes)

The quality of a clustering result depends on both the similarity measure used by the method and its implementation. The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.

We can measure the quality of clustering by dissimilarity/similarity metric. Similarity is expressed in terms of a distance function, which is typically metric: $d(i, j)$. There is a separate "quality" function that measures the "goodness" of a cluster. The definitions of distance functions are usually very different for interval-scaled, boolean, categorical, and ordinal variables. Weights should be associated with different variables based on applications and data semantics. It is hard to define "similar enough" or "good enough". The answer is typically highly subjective.

Let us define a cluster in the following definition.

Clusters can be defined as collection of similar object groups together. A cluster is a set of entities which are alike and at the same time entities from different cluster are not alike.

In general Clusters may be defined as collection of points in a test space such

that the distance between any two points in the cluster is less than the distance between any point in the cluster and any point outside the cluster.

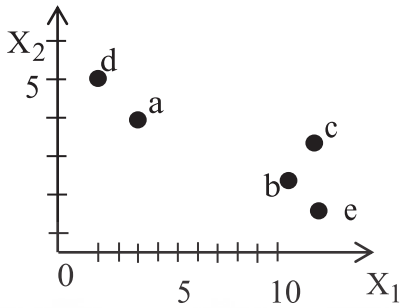
In general, similarity and dissimilarity between data points is measured as a function of the distance between them. The objects may also be grouped into clusters based on different shapes and sizes.

Cluster analysis embraces a variety of techniques, the main objective of which is to group observations or variables into homogeneous and distinct clusters. A simple numerical example will help explain these objectives.

The daily expenditures on food (X1) and clothing (X2) of five persons are shown in Fig. 2.

Person	X_1	X_2
a	2	4
b	8	2
c	9	3
d	1	5
e	8·5	1

(a) Illustrative Data



(b) Grouping of Observations

Fig. 2

Fig. 2 suggests that the five observations form two clusters. The first consists of persons a and d, and the second of b, c and e. It can be noted that the observations in each cluster are similar to one another with respect to expenditures on food and clothing, and that the two clusters are quite distinct from each other.

These conclusions concerning the number of clusters and their membership were reached through a visual inspection of Fig. 2. This inspection was possible because only two variables were involved in grouping the observations. The question is: Can a procedure be devised for similarly grouping observations when there are more than two variables or attributes?

It may appear that a straightforward procedure is to examine all possible clusters of the available observations, and to summarize each clustering according to the degree of proximity among the cluster elements and of the separation among the clusters. Unfortunately, this is not feasible because in most cases in practice the number of all possible clusters is very large and out of reach of current computers. Cluster analysis offers a number of methods that operate much as a person would in attempting to reach systematically a reasonable grouping of observations or variables.

Since clustering is the grouping of similar instances/objects, some sort of measure that can determine whether two objects are similar or dissimilar is required. There are two main type of measures used to estimate this relation: distance measures and similarity measures.
Many clustering methods use distance measures to determine the similarity or dissimilarity between any pair of objects.

Depending on which formula is used to compute the distance between two data points can lead to different classification results. Domain knowledge must be used to guide the formulation of a suitable distance measure for each particular application. For high dimensional data, a popular measure is the

Minkowski Metric: $d(x_i, x_j) = \left(\sum_{k=1}^{d} | x_{i,k} - x_{j,k} |^p \right)^{\frac{1}{p}}$,

where d is the dimensionality of the data.

Special Cases:

If $p = 2$, then the distance is Euclidean distance, and if $p = 1$, then the distance is Manhattan distance.

The commonly used Euclidean distance between two objects is achieved when $p = 2$.

$$d_{ij} = ((x_{i1} - x_{j1})^2 + (x_{i2} - x_{j_2})^2 + \cdots + (x_{id} - x_{jd})^2)^{\frac{1}{2}}.$$

Another well-known measure is the Manhattan distance which is defined when $p = 1$.

$$d_{ij} = |((x_{i1} - x_{j1}) + (x_{i2} - x_{j2}) + \cdots + (x_{id} - x_{jd}))|.$$

The Mahalanobis distance is another very important distance measure used in statistics that measures the statistical distance between two populations of Gaussian mixtures having mean μ_i and μ_j and a common covariance matrix $\sum_{ij}$. This measure is given by

$$d_{ij} = (\mu_i - \mu_j)^T \sum_{ij} (\mu_i - \mu_j).$$

The distance measure described in the last section may be easily computed for continuous-valued attributes. In the case of instances described by categorical, binary, ordinal or mixed type attributes, the distance measure should be revised.

In the case of binary attributes, the distance between objects may be calculated based on a contingency table. A binary attribute is symmetric if both of its states are equally valuable. In that case, using the simple matching coefficient can assess dissimilarity between two objects:

$$d(x_i, x_j) = \frac{r + s}{q + r + s + t}$$

where q is the number of attributes that equal 1 for both objects; t is the number of attributes that equal 0 for both objects; and s and r are the number of attributes that are unequal for both objects.

A binary attribute is asymmetric, if its states are not equally important (usually the positive outcome is considered more important). In this case, the denominator ignores the unimportant negative matches (t). This is called the Jaccard coefficient:

$$d(x_i, x_j) = \frac{r + s}{q + r + s}.$$

When the attributes are nominal, two main approaches may be used:

i) Simple Matching: $d(x_i, x_j) = \frac{p - m}{p}$, where, p is the total number of

attributes and m is the number of matches.

ii) Creating a binary attribute for each state of each nominal attribute and computing their dissimilarity as described above.

Try the following exercise.

E1) What are different distance measures used for clustering?

Now, we shall discuss major clustering applications in the following section.

13.3 MAJOR CLUSTERING APPROACHES

We shall begin this section by describing the major clustering approaches.

1) **Partitioning algorithms:** Construct various partitions and then evaluate them by some criterion.
2) **Hierarchy algorithms:** Create a hierarchical decomposition of the set of data (or objects) using some criterion .
3) **Density-based:** Density based clustering approach is based on connectivity and density functions.
4) **Grid-based:** It is based on a multiple-level granularity structure.
5) **Model-based:** A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other.
6) **Sample based**
7) **High-dimensional algorithm**
8) **Constraint based**

Clustering algorithms may be classified as listed below:

- ***Exclusive Clustering***

  In exclusive clustering data are grouped in an exclusive way, so that a certain datum belongs to only one definite cluster. K -means clustering is one example of the exclusive clustering algorithms.

- ***Overlapping Clustering***

  The overlapping clustering uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership.

- ***Hierarchical Clustering***

  Hierarchical clustering algorithm has two versions: agglomerative clustering and divisive clustering.

- ***Agglomerative clustering*** It is based on the union between the two nearest clusters. The beginning condition is realized by setting every datum as a cluster. After a few iterations it reaches the final clusters wanted. Basically, this is a bottom-up version

- ***Divisive clustering*** It starts from one cluster containing all data items. At each step, clusters are successively split into smaller clusters according to some dissimilarity. Basically this is a top-down version.

- ***Probabilistic Clustering***

  Probabilistic clustering, e.g. mixture of Gaussian, uses a completely probabilistic approach.

The following requirements should be satisfied by clustering algorithm.

1) Scalability
2) Dealing with different types of attributes
3) Discovering clusters of arbitrary shape
4) Ability to deal with noise and outliners
5) High dimensionality
6) Insensitivity to the order of attributes
7) Interpretability and usability

Major problems encountered with clustering algorithms are:

- Dealing with large number of dimensions and a large number of objects can be prohibitive due to time complexity.
- The effectiveness of an algorithm depends on the definition of similarity measure.
- The outcome of an algorithm can be interpreted in different ways.

Now, try an exercise.

E2) Classify the clustering algorithms, along with examples.

In the following section, we shall discuss clustering methods.

13.4 CLUSTERING METHODS

In this section, we shall cover the major clustering techniques.

1. **Hierarchic versus Non-hierarchic Methods:** This is a major distinction involving both the methods and the classification structures designed with them. The hierarchic methods generate clusters as nested structures, in a hierarchical fashion; the clusters of higher levels are aggregations of the clusters of lower levels. Non- hierarchic methods result in a set of un-nested clusters. Sometimes, the user, even when he utilizes a hierarchical clustering algorithm, is interested rather in partitioning the set of the entities considered.

2. **Agglomerative versus Divisive Methods:** Agglomerative method is a bottom up approaching and involves merging smaller clusters into larger ones while the Divisive method is a top-down approach where large clusters are split into smaller ones. Agglomerative methods have been developed for processing mostly similarity/dissimilarity data while the divisive methods mostly work with attribute-based information, producing attribute-driven subdivisions (conceptual clustering).

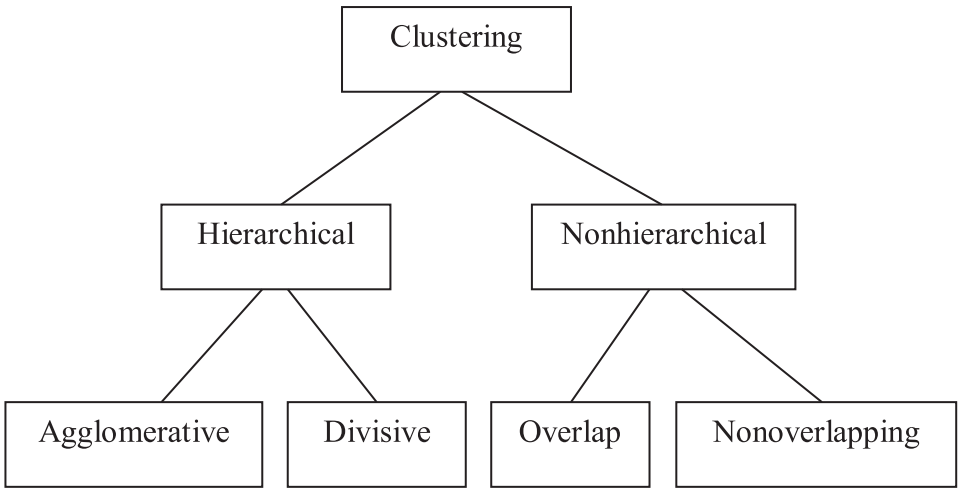


Fig 3: Classical Taxonomy of Clustering Methods

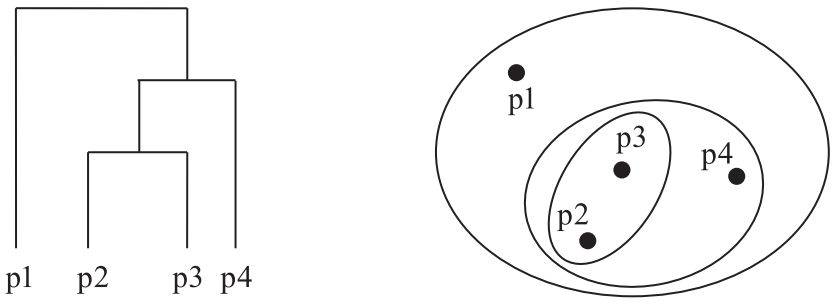
Try an exercise.

E3) How are different clustering methods classified?

Now, let us discuss hierarchical clustering is detail in the following section.

13.5 HIERARCHICAL CLUSTERING

These methods construct the clusters by recursively partitioning the instances in either a top-down or bottom-up fashion. A hierarchy can be represented by a tree structure such as the simple one shown in Fig. 4. For example, patients in an animal hospital are composed of two main groups, dogs and cats, each of which can be sub-divided to further subgroups. Each of the individual animals, 1 through 5, is represented at the lowest level of the tree. Hierarchical clustering refers to a clustering process that organizes the data into large groups, which contain smaller groups and so on. A hierarchical clustering can be drawn as a tree or dendrogram. The finest grouping is at the bottom of the dendrogram, each sample by itself forms a cluster. The coarsest grouping is at the top of the dendrogram, where all samples are grouped into a cluster.



(a) Dendrogram **(b) Nested Clusters**

Fig. 4: A Hierarchical Clustering of Four Points

Logically, several approaches are possible to find a hierarchy associated with the data. The popular approach is to construct the hierarchy level-by-level, from bottom to top (agglomerative clustering) or from top to bottom (divisive clustering). Let us discuss hicrachical clustering methods one by one in detail.

Agglomerative Hierarchical Clustering

Agglomerative hierarchical techniques are the more commonly used methods for clustering. Each object initially represents a cluster of its own. Then clusters are successively merged until the desired cluster structure is obtained. Divisive hierarchical clustering. All objects initially belong to one cluster. Then the cluster is divided into sub-clusters, which are successively divided into their own sub-clusters. This process continues until the desired cluster structure is obtained. The result of the hierarchical methods is a dendrogram, representing the nested grouping of objects and similarity levels at which groupings change. A clustering of the data objects is obtained by cutting the dendrogram at the desired similarity level. The merging or division of clusters is performed according to some similarity measure, chosen so as to optimize some criterion (such as a sum of squares).

The steps of general agglomerative clustering algorithm are as follows:

Step 1: Begin with N clusters. Each cluster consists of one sample.

Step 2: Repeat Step 2 a total of $N-1$ times.

Step 3: Find the most similar clusters C_i and C_j and marge C_i and C_j into one cluster. If there is a tie, merge the first pair found.

Types of Agglomerative Clustering

Nearest neighbor Method (also called Single-link clustering connectedness, the minimum method or the nearest neighbor method) methods that consider the distance between two clusters to be equal to the shortest distance between any member of one cluster to any member of the other cluster.

If the data consist of similarities, the similarity between a pair of clusters is considered to be equal to the greatest similarity from any member of one cluster to any member of the other cluster. This method has a tendency to cluster together at an early stage objects that are distant from each other in the same cluster because of a chain of intermediate objects in the same cluster. Such clusters have elongated sausage-like shapes when visualized as objects in space.

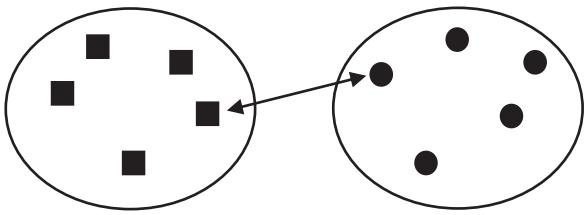


Fig. 5: Cluster Distance in Nearest Neighbour Method

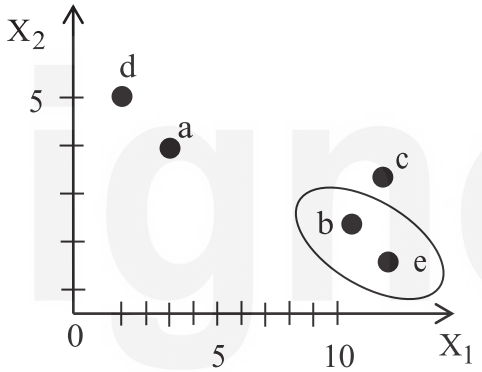
Example 4: Let us suppose that Euclidean distance is the appropriate measure of proximity. Consider the five observations given as a, b, c, d and are shown in Fig. 6(b), and are forming its own cluster. The distance between each pair of observations is shown in Fig. 6(a).
For example, the distance between a and b is

$$\sqrt{(2-8)^2+(4-2)^2} = \sqrt{36+4} = 6.325.$$

Observations b and e are nearest (most similar) and, as shown in Fig. 6(b), are grouped in the same cluster. Assuming the nearest neighbor method is used, the distance between the cluster (be) and another observation is the smaller of the distances between that observation, on the one hand, and b and e, on the other.

Cluster	a	b	c	d	e
a	0	6.325	7.071	1.414	7.159
b		0	1.414	7.616	1.118
c			0	8.246	2.062
d				0	8.500
e					0

(a)



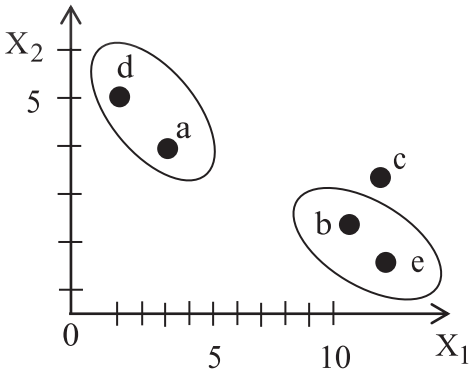
(b)

Fig. 6: Nearest Neighbour Method, (Step 1).

For example, $D(be, a) = \min\{ D(b, a), D(e, a)\} = \min\{ 6.325, 7.159\} = 6.325.$

The four clusters remaining at the end of this step and the distances between these clusters are shown in Fig. 7(a).

Cluster	(be)	a	c	d
(be)	0	6.325	1.414	7.614
a		0	7.071	1.414
c			0	8.246
d				0



(a) **(b)**

Fig. 7: Nearest Neighbour Method, (Step 2).

Two pairs of clusters are closest to one another at distance 1.414; these are (ad) and (bce). We arbitrarily select (ad) as the new cluster, as shown in Fig. 7(b).

The distance between (be) and (ad) is
$D(be, ad) = \min\{ D(be, a), D(be, d)\} = \min\{ 6.325, 7.616\} = 6.325$, while that between c and (ad) is

$$D(c, ad) = \min\{ D(c, a), D(c, d)\} = \min\{ 7.071, 8.246\} = 7.071.$$

The three clusters remaining at this step and the distances between these clusters are shown in Fig. 8 (a). We merge (be) with c to form the cluster (bce) shown in Fig. 8 (b).

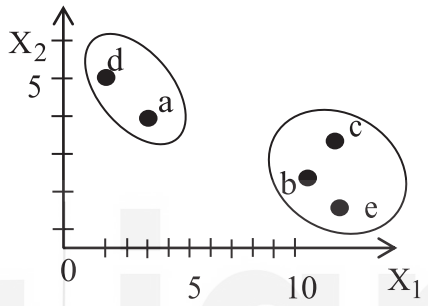
The distance between the two remaining clusters is

$$D(ad, bce) = \min\{ D(ad, be), D(ad, c)\} = \min\{ 6.325, 7.071\} = 6.325.$$

The grouping of these two clusters, it will be noted, occurs at a distance of 6.325, a much greater distance than that at which the earlier groupings took place. Fig. 9 shows the final grouping.

Cluster	(be)	(ad)	c
(be)	0	6.325	1.414
(ad)		0	7.071
c			0

(a)

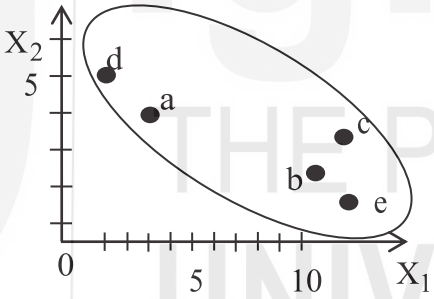


(b)

Fig. 8: Nearest Neighbour Method, (Step 3).

Cluster	(bce)	(ad)
(bce)	0	6.325
(ad)		0

(a)



(b)

Fig. 9: Nearest Neighbour Method, (Step 4).

The groupings and the distance between the clusters are also shown in the tree diagram (dendrogram) of Fig.10. One usually searches the dendrogram for large jumps in the grouping distance as guidance in arriving at the number of groups. In this example, it is clear that the elements in each of the clusters (ad) and (bce) are close(they were merged at a small distance), but the clusters are distant (the distance at which they merge is large).

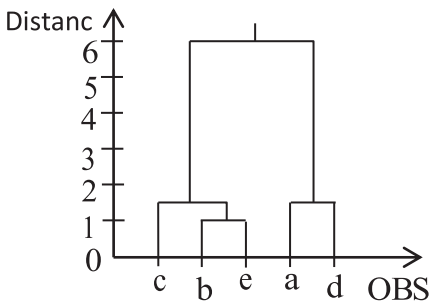


Fig. 10: Nearest neighbour method, (Dendrogram)

***

Complete-link clustering (also called the diameter method, the maximum method or the furthest neighbour method) - methods that consider the distance between two clusters to be equal to the longest distance from any member of one cluster to any member of the other cluster. The nearest neighbour is not the only method for measuring the distance between clusters. Under the furthest neighbor (or complete linkage) method, the distance between two clusters is the distance between their two most distant members. This method tends to produce clusters at the early stages that have objects that are within a narrow range of distances from each other. If we visualize them as objects in space the objects in such clusters would have a more spherical shape as shown in Fig. 11.

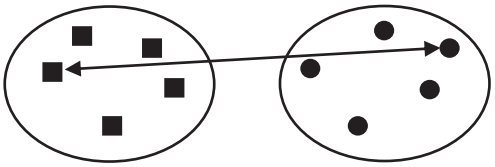


Fig. 11: Cluster Distance (Furthest Neighbour Method)

maximum distance $\cong$ minimum similarity

$$d_{complete}(A,B) := \max_{a\in A, b\in B} d(a,b) \cong \min_{a\in A, b\in B} s(a,b).$$

Now let us understand this through following example:

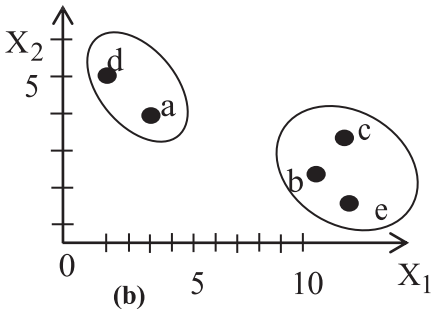
Example 5: Consider the example data presented in Fig. 6. Therefore, the furthest neighbor method also calls for grouping band e at Step 1. However, the distances between (be), on the one hand, and the clusters (a), (c), and (d), on the other, are different:

$$D(be,a) = \max\{D(b,a), D(e,a)\} = \max\{6.325, 7.159\} = 7.159$$
$$D(be,c) = \max\{D(b,a), D(e,c)\} = \max\{1.414, 2.062\} = 2.062$$
$$D(be,d) = \max\{D(b,d), D(e,d)\} = \max\{7.616, 8.500\} = 8.500$$

The four clusters remaining at Step 2 and the distances between these clusters are shown in Fig. 12(a).

Cluster	(be)	a	c	d
(be)	0	7.159	2.062	8.500
a		0	7.071	1.414
c			0	8.246
d				0

(a)



(b)

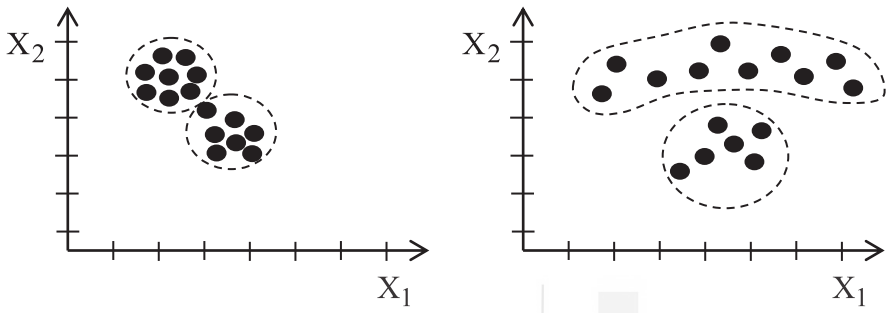
Fig.12: Furthest Neighbour Method (Step 2).

The nearest clusters are (a) and (d), which are now grouped into the cluster (ad). The remaining steps are similarly executed.

***

You may confirm from the Example 4 and Example 5 that the nearest and furthest neighbour methods produce the same results. In other cases, however, the two methods may not agree. Consider Fig. 13(a) as an example.

The nearest neighbour method will probably not form the two groups perceived by the naked eye. This is so because at some intermediate step the method will probably merge the two "nose" points joined in Fig. 13(a) into the same cluster, and proceed to string along the remaining points in chain-link fashion. The furthest neighbour method, will probably identify the two clusters because it tends to resist merging clusters the elements of which vary substantially in distance from those of the other cluster. On the other hand, the nearest neighbour method will probably succeed in forming the two groups marked in Fig. 13(b), but the furthest neighbor method will probably not.



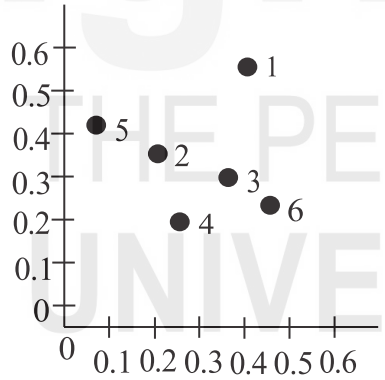
(a) (b)

Fig. 13: Two Cluster Patterns

Now, try the following exercises:

E4) Consider the data given in Fig. 17(a) to Fig. 17(c).

Point	x - Coordinate	y - Coordinate
p1	0.40	0.53
p2	0.22	0.38
p3	0.35	0.32
p4	0.26	0.19
p5	0.08	0.41
p6	045	



(a) x – y coordinates for 6 points **(b) Graph for 6 two-dimensional points**

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

(c) Euclidean Distance Matrix for 6 Points

Fig. 17

Perform clustering using

(i) single link clustering

(ii) complete link clustering

(iii) average link clustering

(iv) Ward's method

In the following section, we shall discuss partitioned clustering.

13.6 PARTITIONAL CLUSTERING

Partitioning clustering begins with a starting cluster partition which is iteratively improved until a locally optimal partition is reached. The starting clusters can be either random or the cluster output from some clustering pre-process (e.g. hierarchical clustering). In the resulting clusters, the objects in the groups together add up to the full object set. Partitioning procedures differ with respect to the methods used to determine the initial partition of the data, how assignments are made during each pass or iteration, and the clustering criterion used. The most frequently used method assigns objects to the clusters having the nearest centroid. This procedure creates initial partitions based on the results from preliminary hierarchical cluster procedures such as the average linkage method or Ward's method, a procedure that resulted in partitioning methods being referred to as two-stage cluster analysis. Some partitioning methods use multiple passes during which cluster centroids are recalculated and objects are re-evaluated, whereas other methods use a single-pass procedure. Partitioning methods also differ with respect to how they evaluate an object's distance from cluster centroids. Some procedures use simple distance and others use more complex multivariate matrix criteria. Finally, most partitioning methods require that the user specify a priori how many clusters will be formed.

Let us discuss an important algorithm known as Frogy's algorithm, which is used for partitional clustering.

Frogy's algorithm: One of the simplest partitional clustering algorithm is Frogy's algorithm. Input to the algorithm consists of data, k, number of clusters to be constructed and k samples called **seed points**. Seed points could be chosen randomly or some knowledge of the desired cluster structure could be the starting point.

The following steps are performed:

Step 1: Initialize the cluster centroid to the seed points.

Step 2: For each sample, find the cluster centroid nearest to it. Put the samples in the cluster identified with the nearest cluster centroid.

Step 3: If no samples changed clusters in Step 2, stop.

Step 4: Compute the centroids of the resulting clusters and go to step 2.

Let us apply these steps in the following example.

Example 6: Perform partitional clustering using Frogy's method for the data given in Fig. 18 (a) with k-2 (two clusters). Use first two sample points (4,4) and (8,4) as seed points.

	x	y
1	4	4
2	8	4
3	15	8
4	24	4
5	24	12

(a) x-y Coordinates for 5 Points

Sample	Nearest cluster centroid
(4,4)	(4,4)
(8,4)	(8,4)
(15,8)	(8,4)
(24,4)	(8,4)
(24,12)	(8,4)

(b) First Iteration

Sample	Nearest cluster centroid
(4,4)	(4,4)
(8,4)	(4,4)
(15,8)	(17.75,7)
(24,4)	(17.75,7)
(24,12)	(17.75,7)

(c) Second Iteration

Sample	Nearest cluster centroid
(4,4)	(6,4)
(8,4)	(6,4)
(15,8)	(21,8)
(24,4)	(21,8)
(24,12)	(21,8)

(d) Third Iteration

Fig. 18

For Step 2, find the nearest cluster centroid for each sample. Fig. 18(b) shows the results. The clusters {(4,4)} and {(8,4), (15,8), (24,4), (24,12)} are produced.

For Step 4, we compute the centroid of the clusters. The centroid of first cluster is (4,4). The centroid of second cluster is (17.75,7) as

(8+15+24+24)/4=17.75 and (4+8+4+12)/4 =7.

As samples change clusters, go to Step 2.

Find cluster centroid nearest each sample. Fig. 18(c) shows the results. The clusters {(4,4),(8,4), } and { (15,8), (24,4), (24,12)} are produced.
For Step 4, we compute the centroid (6,4) and (21,8) of the clusters. As sample (8,4) changed cluster, return to Step 2.

Find cluster centroid nearest each sample. Fig. 17(d) shows the results. The clusters {(4,4),(8,4), } and { (15,8), (24,4), (24,12)} are produced.

For Step 4, we compute the centroid (6,4) and (21,8) of the clusters. As no sample changed clusters, the algorithm terminates.

***

Try an exercise.

E5) Consider the data

Sample	x	y
1	0	0
2	1	0
3	0	2
4	2	2
5	3	2
6	6	3
7	7	3

Perform a partitional clustering using

(i) $k = 2$ and use the first two samples in the list as seed points.
(ii) $k = 3$ and use the first three samples in the list as seed points.

In the following section, we discuss k -means clustering.

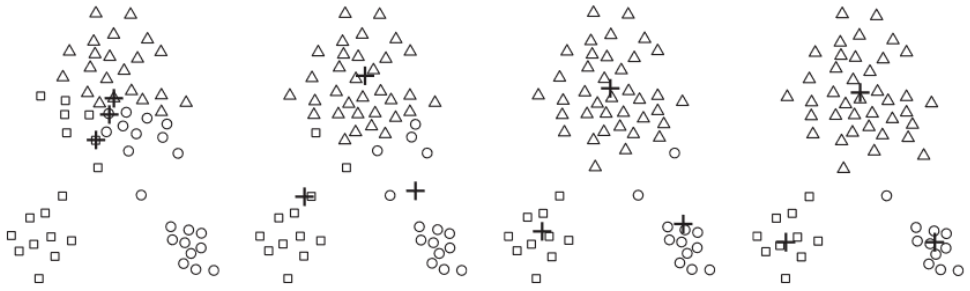
13.7 K-MEANS CLUSTERING

The K -means clustering technique is simple, and we first choose k initial centroids, where k is a user-specified parameter, namely, the number of clusters desired. Each point is then assigned to the closest centroid, and each collection of points assigned to a centroid is a cluster. The centroid of each cluster is then updated based on the points assigned to the cluster. We repeat the assignment and update steps until no point changes clusters, or equivalently, until the centroids remain the same. In its simplest form, the k -means method follows the following steps.

Step 1: Specify the number of clusters and, arbitrarily or deliberately, the members of each cluster.
Step 2: Calculate each cluster's \centroid" (explained below), and the distances between each observation and centroid. If an observation is nearer the centroid of a cluster other than the one to which it currently belongs, re-assign it to the nearer cluster.
Step 3: Repeat Step 2 until all observations are nearest the centroid of the cluster to which they belong.
Step 4: If the number of clusters cannot be specified with confidence in advance, repeat Steps 1 to 3 with a different number of clusters and evaluate the results.

The operation of K -means are shown in Fig. 19, which shows how, starting from three centroids, the final clusters are found in four assignment-update steps. In these and other figures displaying K -means clustering, each subfigure shows (1) the centroid sat the start of the iteration and (2) the assignment of the points to those centroids. The centroids are indicated by the "+" symbol. All points belonging to the same cluster have the same marker shape.

In the first step, shown in Fig. 19(a), points are assigned to the initial centroids, which are all in the larger group of points. For this example, we use the mean as the centroid. After points are updated again. In steps 2, 3, and 4, which are shown in Fig. 19(b), (c), and (d), respectively, two of the centroids move to the two small groups of points



(a) First Iteration (b) Second iteration (c) Third Iteration (d) Fourth Iteration

Fig.19: Using the K -Means Algorithm

at the bottom of the figures. When the K-means algorithm terminates in Fig. 19(d), because no more changes occur, the centroids have identified the natural groupings of points. Centroid at the beginning of the step and the assignment of points to those centroids. In the second step, points are assigned to the updated centroids, and the centroids.

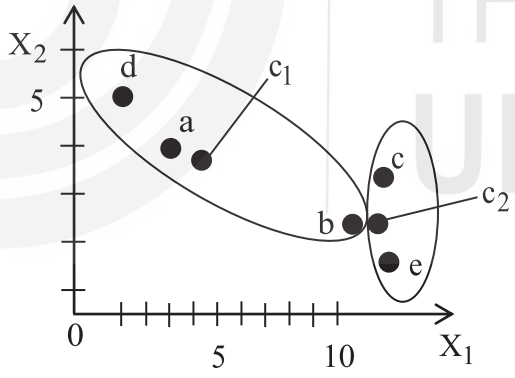
Let us understand this in the following example.

Example 7: Suppose two clusters are to be formed for the observations listed in Fig. 20(a). We begin by arbitrarily assigning a, b and d to Cluster 1, and c and e to Cluster 2. The cluster centroids are calculated as shown in Fig. 20(a).

The cluster centroid is the point with coordinates equal to the average values of the variables for the observations in that cluster. Thus, the centroid of Cluster 1 is the point $(X_1 = 3.67, X_2 = 3.67)$, and that of Cluster 2 the point (8.75, 2). The two centroids are marked by C_1 and C_2 in Fig. 20(a). The cluster's centroid, therefore, can be considered the center of the observations in the cluster, as shown in Fig. 20(b). We now calculate the distance between a and the two centroids.

Cluster 1			Cluster 2		
Observation	X_1	X_2	Observation	X_1	X_2
a	2	4	c	9	3
b	8	2	e	8.5	1
d	1	5			
Average	3.67	3.67	Average	8.75	2

(a)



(b)

Fig. 20: Means Method (Step 1)

$$D(a, abd) = \sqrt{(2-3.67)^2 + (4-3.67)^2} = 1.702.$$

$$D(a, ce) = \sqrt{(2-8.75)^2 + (4-2)^2} = 7.040.$$

Observe that a is closer to the centroid of Cluster 1, to which it is currently assigned. Therefore, a is not reassigned. Next, we calculate the distance between b and the two cluster centroids:

$$D(b, abd) = \sqrt{(8-3.67)^2 + (2-3.67)^2} = 4.641.$$

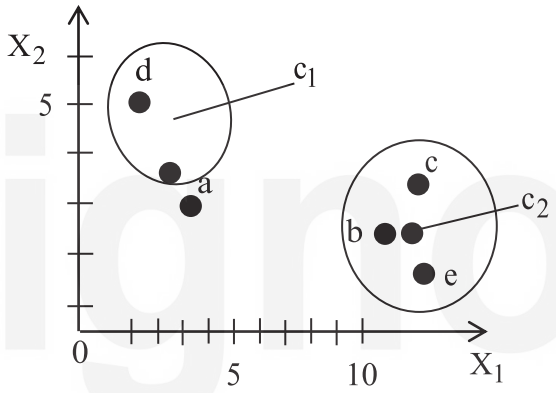
$$D(b, cc) = \sqrt{(8-8.75)^2 + (2-2)^2} = 0.750.$$

Since b is closer to Cluster 2's centroid than to that of Cluster 1, it is reassigned to Cluster 2. The new cluster centroids are calculated as shown in Fig. 21(a).The new centroids are plotted in Fig. 21(b). The distances of the observations from the new cluster centroids are shown in Fig. 21(c).

(an asterisk indicates the nearest centroid):

Cluster 1			Cluster 2		
Observation	X_1	X_2	Observation	X_1	X_2
a	2	4	c	9	3
d	1	5	e	8.5	1
			b	8	2
Average	1.5	4.5	Average	8.5	2

(a)



(b)

	Distance from	
Observation	Cluster 1	Cluster 2
a	0.707*	6.801
b	6.964	0.500*
c	7.649	1.118*
d	0.707*	8.078
e	7.826	1.000*

(c)

Fig. 21: Means Method (Step 2)

Every observation belongs to the cluster to the centroid of which it is nearest, and the k-means method stops. The elements of the two clusters are shown in Fig. 21(c).

***

Now, we list the benefits and drawbacks of k-means methods.

Benefits:

1) Very fast algorithm (O (k . d . N), if we limit the number of iterations)
2) Convenient centroid vector for every cluster
3) Can be run multiple times to get different results

Limitations:

1) Difficult to choose the number of clusters, k
2) Cannot be used with arbitrary distances
3) Sensitive to scaling – requires careful preprocessing
4) Does not produce the same result every time
5) Sensitive to outliers (squared errors emphasize outliers)
6) Cluster sizes can be quite unbalanced (e.g., one-element outlier clusters)

Try an exercise.

E6) What are advantages and disadvantages of k -means clustering methods?

Now let us summaries what we have learnt in this unit.

13.8 SUMMARY

We have discussed the following points:

1) Concept of clustering.
2) Various distance measures.
3) Various clustering methods.
3) Analyzed various Hierarchical clustering algorithms in detail.
4) Analyzed various Partitional clustering and k -nn clustering algorithms.

13.9 SOLUTION/ ANSWERS

E1) Different formula in defining the distance between two data points can lead to different classification results. Domain knowledge must be used to guide the formulation of a suitable distance measure for each particular application. For high dimensional data, a popular measure is the Minkowski

$$d(x_i, x_j) = \left(\sum_{k=1}^{d} | x_{i,k} - x_{j,k} |^p \right)^{\frac{1}{p}}$$ Metric:

Where d is the dimensionality of the data.
Special Cases:

• p=2: Euclidean distance

• p=1: Manhattan distance

The commonly used Euclidean distance between two objects is achieved when $p = 2$.

$$d_{ij} = ((x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{id} - x_{jd})^2)^{\frac{1}{2}}$$

Another well-known measure is the Manhattan distance which is defined when $p = 1$.

$$d_{ij} = |(x_{i1} - x_{j1}) + (x_{i2} - x_{j2}) + \cdots + (x_{id} - x_{jd})|$$

The Mahalanobis distance is another very important distance measure used in statistics that measures the statistical distance between two populations of Gaussian mixtures having mean μ_i and μ_j and a common covariance matrix $\sum_{ij}$. This measure is given by

$$d_{ij} = (\mu_i - \mu_j)^T \sum_{ij} (\mu_i - \mu_j).$$

E2) i) Exclusive clustering

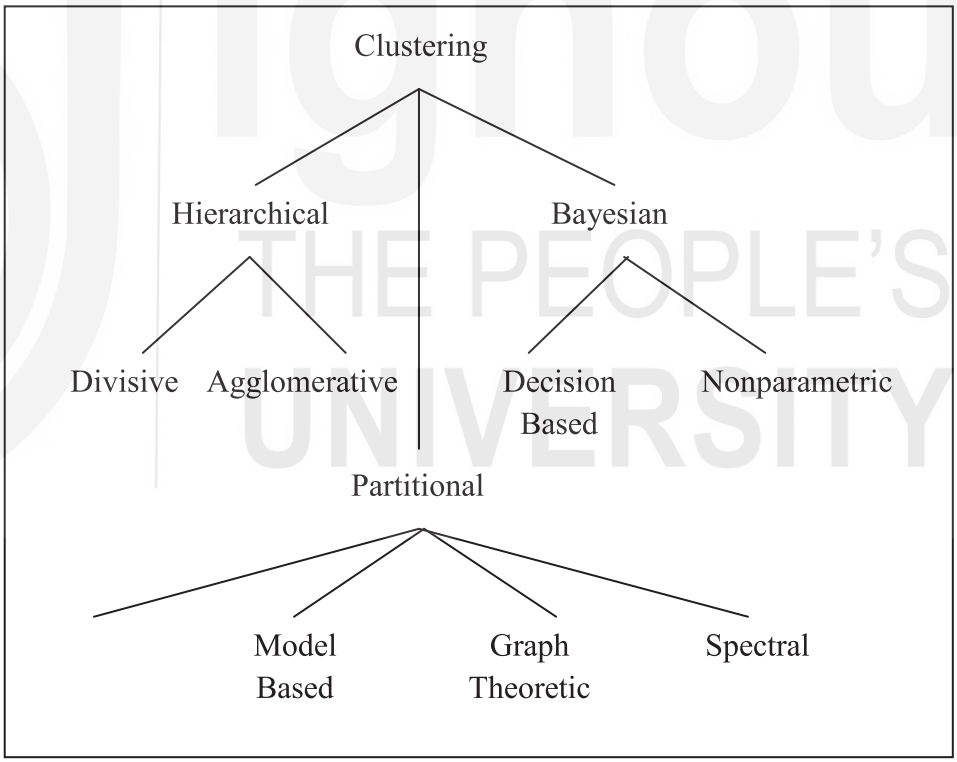
ii) Overlapping clustering

iii) Agglomerative clustering

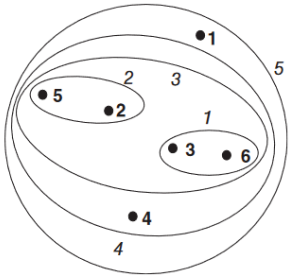
iv) Divisive clustering

v) Probabilistic clustering

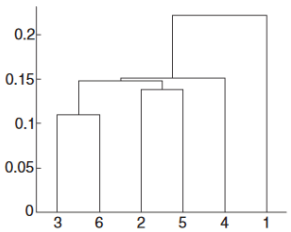
E3)



E4) (i) Single link clustering



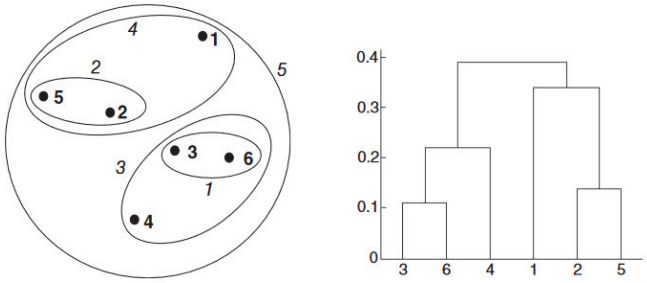
(a) Single Link Clustering



(b) Single Link Dendrogram

$$\text{dist}(\{3,6\},\{2,5\}) = \min(\text{dist}(3,2), \text{dist}(6,2), \text{dist}(3,5), \text{dist}(6,5))$$
$$= \min(0.15, 0.25, 0.28, 093)$$
$$= 0.15.$$

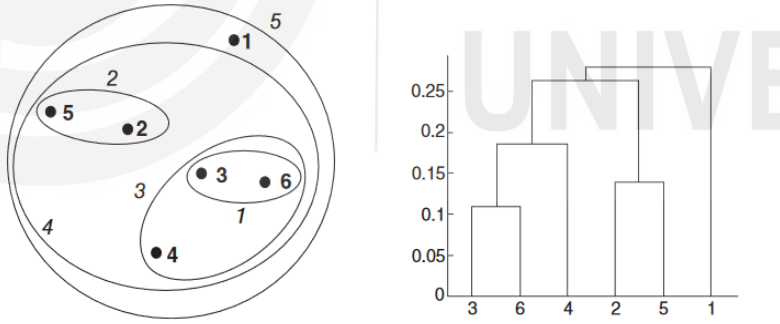
(ii) Complete link clustering



(a) Complete Link Clustering **(b) Complete Link Dendrogram**

$$\text{dist}(\{3,6\}),\{4\}) = \max(\text{dist}(3,4), \text{dist}(6,4))$$
$$= \max(0.15, 0.22)$$
$$= 0.22$$
$$\text{dist}(\{3,6\},\{2,5\} = \max(\text{dist}(3,2), \text{dist}(6,2), \text{dist}(3,5), \text{dist}(6,5))$$
$$= \max(0.15, 0.25, 0.28, 0.39)$$
$$= 0.39.$$
$$\text{dist}(\{3,6\},\{1\}) = \max(\text{dist}(3,1), \text{dist}(6,1))$$
$$= \max(0.22, 0.23)$$
$$= 0.23.$$

(iii) Average link clustering



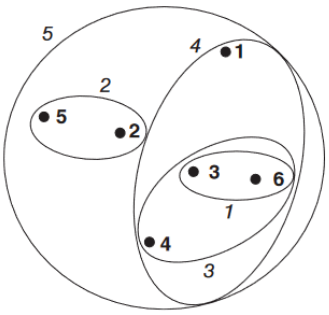
(a) Group Average Clustering **(b) Group Average Dendrogram**

$$\text{dist}(\{3,6,4\},\{1\}) = (0.22 + 0.37 + 0.23)/(3 * 1)$$
$$= 0.28$$
$$\text{dist}(\{2,5\},\{1\}) = (0.2357 + 0.3421)/(2 * 1).$$
$$= 0.2889$$

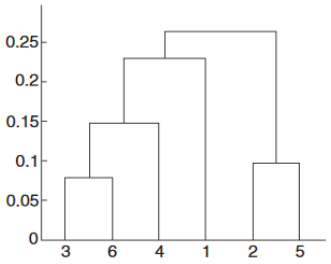
$$\text{dist}(\{3,6,4\},\{2,5\}) = (0.15 + 0.28 + 0.25 + 0.39 + 0.20 + 0.29)/(6 * 2)$$

$$= 0.26$$

iii) Clustering using Ward's method



(a) Ward's Clustering



(b) Ward's Dendrogram

E6) Benefits of k-nnalgorithm:

1) Very fast algorithm (O(k . d . N), if we limit the number of iterations)
2) Convenient centroid vector for every cluster
3) Can be run multiple times to get different results

Limitations of k-nn algorithm:

1) Difficult to choose the number of clusters, k
2) Cannot be used with arbitrary distances
3) Sensitive to scaling – requires careful preprocessing
4) Does not produce the same result every time
5) Sensitive to outliers (squared errors emphasize outliers)
6) Cluster sizes can be quite unbalanced (e.g., one-element outlier clusters)

MPDD/IGNOU/P.O. 0.2K/June, 2023

ISBN : 978-93-5568-870-5