Image Segmentation

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF SCIENCE

IN

MATHEMATICS

Submitted by

Diksha Gupta

(2K21/MSCMAT/15)

Sushant Dhingra

(2K21/MSCMAT/52)

Under the supervision of

Dr. Dhirendra Kumar

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road , Delhi - 110042

May, 2023

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road , Delhi - 110042

DECLARATION

We , Diksha Gupta (2K21/MSCMAT/15) and Sushant Dhingra (2K21/MSCMAT/52) , students of M.Sc. Mathematics, We hereby affirm that the project dissertation "Image Segmentation" that we have submitted to the Department of Applied Mathematics at Delhi Technological University, Delhi, is our own work and has not been reused or otherwise altered in any way without our written permission.The awarding of a degree, diploma, associateship, fellowship, or any other title or honour of a similar nature has not historically been based on the work in question.

Place : Delhi — Diksha Gupta
Date : — Sushant Dhingra

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road , Delhi - 110042

CERTIFICATE

We hereby certify that the Project Dissertation titled "Image Segmentation" submitted by Diksha Gupta (2K21/MSCMAT/15) and Sushant Dhingra (2K21/MSCMAT/52) [Department of Applied mathematics] at Delhi Technological University, Delhi, is a record of the project work completed by the students under my supervision. This work hasn't, as far as I know, been submitted in whole or in part for a degree or diploma at this university or anywhere else.

Place : Delhi | **Dr. Dhirendra Kumar**

Date : | **SUPERVISOR**

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road , Delhi - 110042

ACKNOWLEDGEMENT

We want to sincerely thank Prof. S. Sivaprasad Kumar HOD, Head of the Department of Applied Mathematics. We would especially want to thank Dr. Dhirendra Kumar, our mentor, for his time and work throughout the course of the year. We truly appreciated your helpful recommendations and assistance as we finished the project. We will always be appreciative of you in this regard. We would want to highlight that we were the only ones who worked on this project altogether.

Signature	Signature
Diksha Gupta	Sushant Dhingra

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road , Delhi - 110042

ABSTRACT

One well-liked method for dividing up scientific photos is fuzzy C-Means (FCM).The use of intuitionistic fuzzy C-Means (IFCM), which is based on the idea of intuitionistic fuzzy sets (IFSs), is advocated in the literature as a way to deal with the ambiguity and uncertainty associated with real data.The hesitation and membership degrees are used to determine the objective function.However, FCM is used to achieve the approximate answer rather than analytically computing the objective function. Even though there are numerous variations of intuitionistic fuzzy set theory, all of them struggle with the issue of noise in images during the segmentation process.In order to address this issue, we have proposed using a picture fuzzy set theoretic approach, which improves the data's ability to be represented and aids in handling the noise structures present in the image In our proposed work,we have added algorithm of FCM , PFCM, and some applications..The method was applied to a fake image that had been ”Gaussian” and ”salt and pepper” distorted.Partition efficiency, average segmentation accuracy (ASA), and dice score (DS) were the performance metrics used. In order to determine the difference between two fuzzy sets or intuitionistic sets, we can use the distance measure and the dissimilarity between them, which are both employed in pattern recognition and image segmentation.

Keywords: Image Segmentation , Clustering, Intuitionistic fuzzy sets, Fuzzy c-means, Picture fuzzy clustering, Applications

Contents

List of Figures

List of Algorithms

Chapter 1

INTRODUCTION

Image analysis and pattern recognition begin with image segmentation. One of the hardest tasks in image processing, it is a crucial and necessary part of image analysis and/or pattern recognition systems and affects how well the analysis turns out in the end. The image segmentation problem is essentially one of psychophysical perception, hence a purely analytical solution is not possible, according to [2]. The methods for monochromatic picture segmentation have been extensively studied in both articles and surveys.

The fuzzy theory was introduced by Lotfi Zadeh [3], and the researchers put the fuzzy theory into clustering. The FCM algorithm is introduced by Dunn [4] and later, it is generalized by Bezdek [5] with (fuzzifier) and became very popular. The assignment of data objects to numerous clusters by fuzzy algorithms is partially possible. The distance between a data object and the cluster centres determines how much of it belongs in the fuzzy clusters.However, there are a number of drawbacks to the FCM algorithm. For instance, it struggles with data sets that have clusters of different sizes or densities, and it is susceptible to noise and outliers. Numerous FCM algorithm variations have been developed to address these FCM disadvantages.

A noise clustering (NC) variation of the FCM was proposed by Dave and Sen [6]. Noisy points are defined as data points whose distances from all cluster centers are greater than a specific threshold. It is not effected by the amount of clusters present in a particular data set, and it does not recognise noisy data points that are located between clusters.This technique discovers outliers with clarity and is unaffected by the number of clusters. Because it regularly spreads outliers across different clusters, this method successfully reduces the impact of outliers but is less noise-resistant.

Howerver, to overcome the poor performance of FCM caused by noisy data, possibilistic c-means (PCM) was proposed in [7]. In contrast to a fuzzy partition, PCM views the clustering as a possibilistic partition. To overcome these issues, PFCM clustering provides both the membership and typicality data.It is therefore a fusion of PCM and FCM. When the input data set has multiple outliers and different-sized clusters, PFCM performs ineffectively, but it still generates better clustering than FCM and PCM.

The FCM has been expanded to three algorithms: the EnFCM (Enhanced FCM), FCM_S1, and FCM_S2, to address the latter issue. When creating the membership function in FCM, more ambiguity appears. This ambiguity results from ignorance.

An innovative IFS introduced by [8] clustering technique for medical picture segmentation was created by Chaira. [9].In this method to maximize the good points in the class, a new objective function called intuitionistic fuzzy entropy is incorporated into the objective function of conventional FCM. Zhang et al. [10] proposed an intuitionistic fuzzy set clustering method. Xu et al.

In order to detect tumors in medical photos, Chaira and Anand [9] created a novel IFS technique. In order to remove undesirable regions from a clustered image, this method employs histogram thresholding. Moreover, the tumor's edge is removed. Cuong [11] has presented a Picture Fuzzy Set (PFS) which is a generalization of the traditional fuzzy set and IFS. PFS resolves issues that call for responses like "yes," "no," "refusal," and "abstaining". Thong and Son., [12] proposed a new Picture Fuzzy Clustering (PFC) proposed by [13]. The findings of the experiments show that PFC produces better clustering outcomes. This paper provides techniques of image segmentation with algorithm and some applications. We created an FCM method and implemented it on code to segment MRI brain pictures in this paper, which was motivated by the PFC's strong performance.

Chapter 2

Preliminaries and Related work

2.1 Image Segmentation

In segmentation, we want to colour our pixels in the image so that similar objects are coloured similarly. For example, in the image above, we can see that the input image contains a variety of different objects, such as a road, sidewalk, building, vehicle, etc. In the segmented image, we can see that similar objects are coloured similarly. The process of image segmentation divides a digital image into smaller groups, or "image segments," which simplifies the image and makes each segment easier to handle or analyse.

Segmenting images for identifying objects is a common application. Prior to processing a picture completely, it is customary to first utilise an image segmentation method to identify objects of interest in the image. When the segmentation procedure is finished, the object detector can use the bounding box it created. Accuracy and inference time can both be improved by pausing the detector from analysing the entire image.

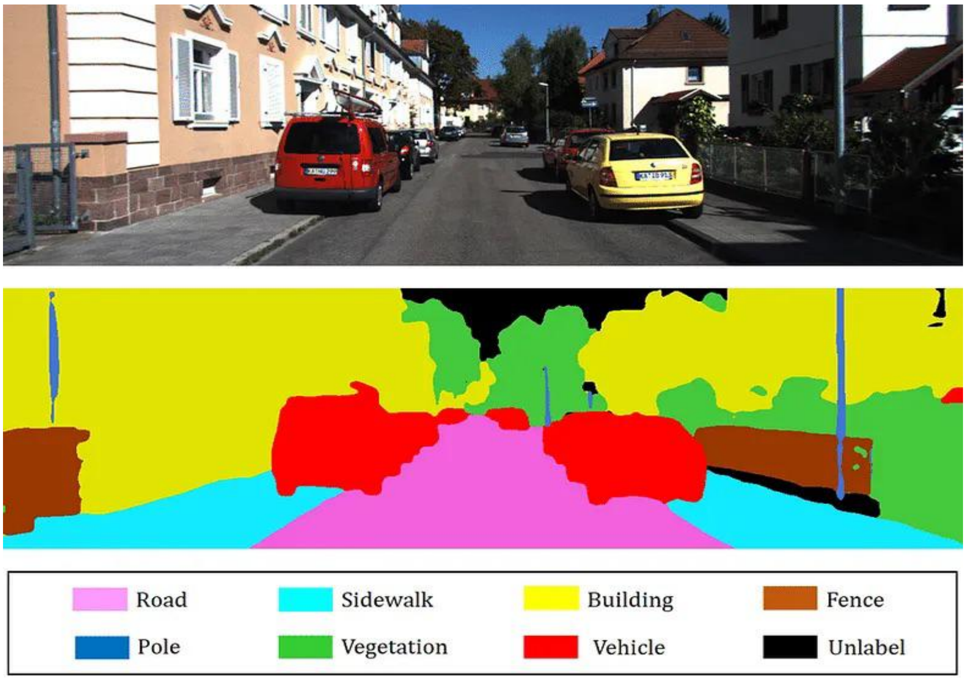


Figure 2.1: Image Segmentation

2.1.1 Why do we need Image Segmentation?

Cancer has historically been a fatal disease. Cancer can be devastating even in today's technologically advanced day if it isn't detected at an early stage. Millions of lives could be saved by early cancer detection. The severity of the cancer is significantly influenced by the morphology of the malignant cells. It's possible that you placed the puzzle pieces together; object detection won't be very helpful in this case. We won't be able to determine the shape of the cells because we will just create bounding boxes.

Image segmentation methods have a MASSIVE influence in this situation. They enable us to take a more detailed approach to the issue at hand and produce more fruitful outcomes. a situation where everyone in the healthcare sector benefits.

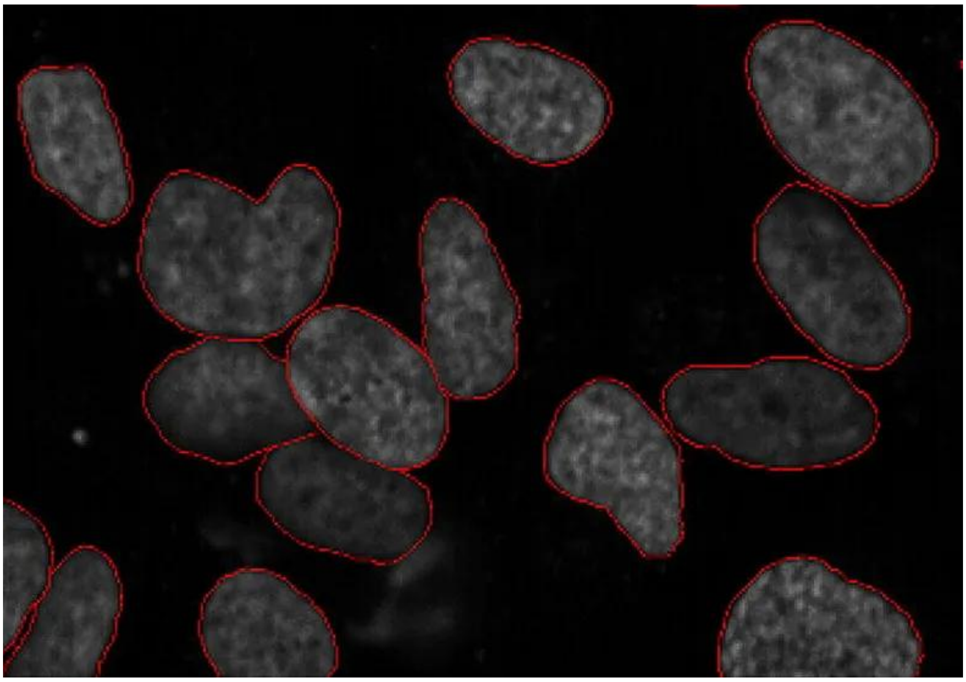


Figure 2.2: Cancer Cells

Here, all of the malignant cells' forms are easily visible. There are numerous other uses for image segmentation that are revolutionising industries:

1. Traffic control systems

2.Self driving cars

3.Locating objects in satellite images

2.1.2 Image segmentation types

An image can be segmented in many ways. Here are a few of the principal methods:

Semantic Segmentation

The pixels in an image are organised into semantic classes during semantic segmentation. The segmentation model does not make use of any additional context or data and each pixel in this model belongs to a single class.

A mask that categorises all tree types into one category (tree) and all vehicle types, such as buses, cars, and bicycles into one category (vehicles), for example, will be produced by semantic segmentation of an image containing several trees and vehicles.

Figure 2.3: Semantic Segmentation

When utilising this approach, the problem description can often be nebulous, particularly when several instances are combined into one class. For instance, the entire crowd in a photo of a crowded street might be categorised under the "people" class.

Instance Segmentation

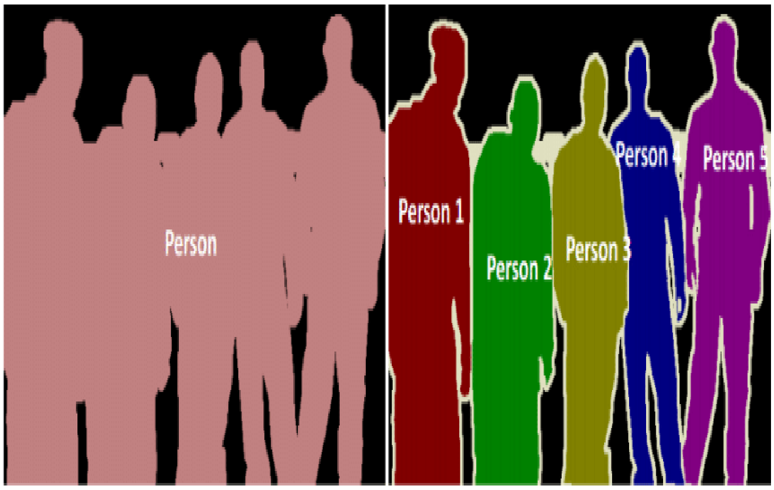
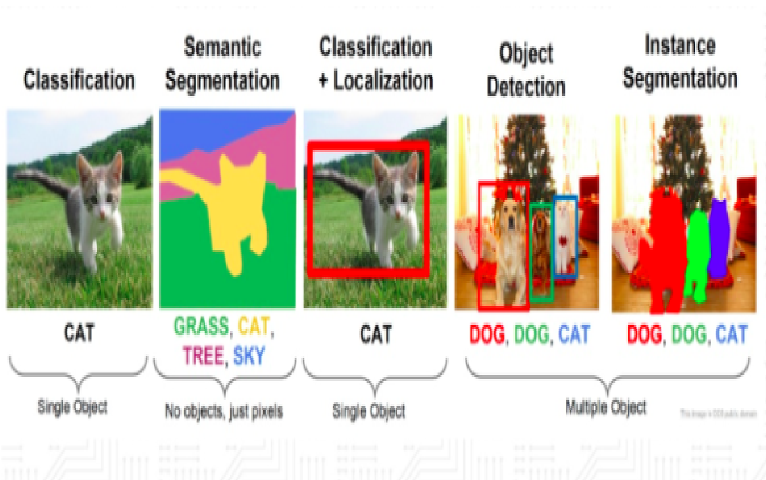


Figure 2.4: Instance Segmentation

Instance segmentation divides pixels into groups based on specific instances of an item. Instance segmentation techniques instead divide comparable or overlapping regions based on the boundaries of objects rather than knowing to which class each region belongs.

Consider processing an image of a busy street using an instance segmentation model. It should ideally be able to pinpoint certain objects within the crowd while counting the number of occurrences. The region or item (i.e., a "person") for each instance cannot be predicted, though.

Here, we see the difference between the Semantic and Instance Segmentation and its combination called Panoptic Segmentation



2.1.3 Techniques for Image Segmentation

Edge-Based Segmentation

Figure 2.5: Edge-Based Segmentation

Edge-based segmentation is a term used to describe a popular technique for processing images that detects the edges of various objects in a picture. It aids in identifying features of connected objects in the image by using information from the edges. Edge detection reduces the size of photographs and facilitates analysis by deleting unnecessary information.

Edge-based segmentation algorithms identify edges based on variations in contrast, texture, colour, and saturation. Edge chains, which are constructed from the individual edges, can be used to accurately represent the borders of objects in an image.

Threshold-Based Segmentation

Figure 2.6: Threshold-Based Segmentation

Thresholding, which separates pixels based on how intense they are in relation to a predetermined value or threshold, is the most straightforward method for segmenting images. It is suitable for dividing up objects with a stronger contrast than background or other objects.

The threshold value T can act as a constant in low-noise images. In some cases, flexible thresholds are a possibility. Thresholding divides a grayscale image into two parts based on how closely they relate to T, producing a binary image.

Watershed Segmentation

In a grayscale image, watersheds are changes. The elevation (height) of an image is determined by the pixel brightness in watershed segmentation algorithms, which treat images like topographic maps. Using this method, the regions between the watershed lines are marked by lines forming ridges and basins. One of the main applications for the watershed method is medical image processing. Being able to identify lighter and darker areas in an MRI image, for example, can help with diagnosis.

2.2 Cluster based image segmentation

Image segmentation involves dividing an image into meaningful regions or objects, and cluster-based methods provide a powerful approach for achieving this. Cluster-based methods play a significant role in image segmentation due to their ability to group similar pixels or regions, adaptability to different image content, computational efficiency, and flexibility in parameter control. They provide a powerful and versatile approach for extracting meaningful regions from images, enabling numerous applications in fields such as computer vision, medical imaging, object recognition, and more.

2.3 Clustering

A technique for assembling several clusters of related data points from the data points.

i.e., the potentially similar object stays in a group that shares few to no similarities with another group.

2.4 Different types of clustering

- Partitional Clustering VS Hierarchical clustering

  (a) a division of data objects into non-overlapping subsets in which each object can be found inside just one subset

(b) a group of stacked clusters organised in a form like a tree

• Exclusive VS non-exclusive

(a) Points may belong to more than one cluster in non-exclusive clustering.

(b) represents many types or ”boundary” points

• Fuzzy VS non-fuzzy

(a) A point belongs to each cluster in fuzzy clustering where the weight ranges from 0 to 1.

(b) Probabilistic clustering has similar characteristics

• Heterogeneous VS Homogeneous

(a) Cluster of widely different sizes, shapes, and densities

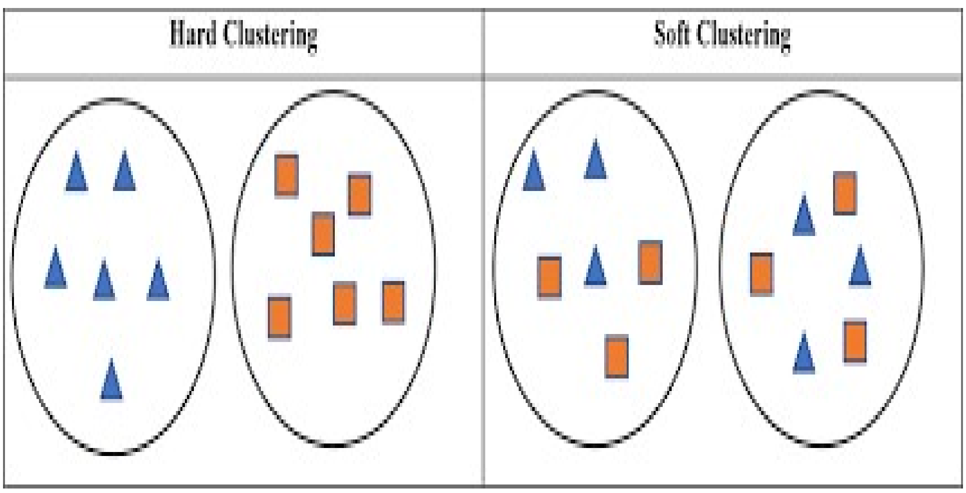
2.5 Clustering Analysis

By clustering, we mean dividing a data universe X into n data samples and counting the number of subclasses of c clusters within that universe.$(2 \leq c < n)$

Note: $c = 1$ denotes rejection of the hypothesis that there are clusters in the data, where as $c = n$ denotes the trivial case where each sample is in a cluster by itself.

There are two types of data C-partitions:

1. HARD (or crisp)
2. SOFT (or fuzzy)



One of the techniques used in hard clustering, where each data item is grouped so that it can only belong to one cluster, is K-Means. Soft clustering, which groups the data items in a way that allows for the existence of a single item in multiple clusters, as exemplified by fuzzy C-Means (FCM).

Now, the question arises that how to calculate the degree of similarity between data and how to assess the partitions after they have been created.

The distance between two vector pairs is among the simplest metrics for comparing similarity. i.e., the distance between points within a cluster should be smaller than the distance between points outside of that cluster.

Chapter 3

Fuzzy Clustering

With a membership weight ranging from zero to one, a person or thing can be a member of every cluster.

Membership weight zero means the person/object absolutely does not belong to the cluster.

Membership weight one means the person/object absolutely belong to the cluster

3.1 Hard C-Means

HCM is used to clearly categorise data. This refers to the assignment of each data point to one single data cluster. These clusters can also be referred to as data partitions in this sense.
Let's define the family of sets $\{A_i, \;\; i = 1, 2, 3, ...c\}$ as hard c-partitions of X, when the following characteristics ought to be met.

1. $\bigcup_{i=1}^{c} A_i = X$
2. $A_i \cap A_j = \emptyset \quad \forall\, i \neq j$
3. $\emptyset \subset A_i \subset X \quad \forall\, i = 1, 2, 3, ...c$

$X = \{x_1, x_2, ...x_n\}$ is a constrained set space consisting of all data samples, together with the number of classes, divisions, or clusters we intend to create for the data into is c. where the defining fn is $\chi_{A_i}\ (x_k)$ is defined as:

$$\chi_{A_i}(x_k) = \begin{cases} 1 & x_k \in A_i \\ 0 & x_k \notin A_i \end{cases}$$

Notations:

- $\chi_{ij} = \chi_{A_i}\ (x_j)$ i.e, membership assignment of the j^{th} data point in the i^{th} cluster or class.
- U is a matrix comprising elements $\chi_{ij}\ (i = 1, 2, ...c;\ j = 1, 2, ...n)$ Consequently, it has a matrix of c-rows and n-columns.

Thus, for X, the following matrix set is defined as a hard c-partition space:

$$\mathcal{M}_c = \left\{ U \ \mid\ \chi_{ij} \in \{0,1\}\ \ ,\ \ \sum_{i=1}^{c} \chi_{ik} = 1 \right\} \tag{3.1}$$

and

$$0 \ < \ \sum_{k=1}^{n} \chi_{ik} \ < \ n$$

any matrix $U \in \mathcal{M}_c$ is a hard-partition.

Since each cluster centre needs m coordinates to indicate its location in the same space as the data sample it represents, each data sample needs m coordinates to explain its placement in the space.

Therefore, the i^{th} cluster centre is a vector of length m

$$v_i = \{v_{i1}, v_{i2}, ..., v_{im}\}$$

where the j^{th} coordinates is calculated by

$$v_{ij} = \frac{\sum\limits_{k=1}^{n} \chi_{ik} x_{kj}}{\sum\limits_{k=1}^{n} \chi_{ik}} \tag{3.2}$$

and

$$d_{ik} = d\left(x_k - v_i\right) = ||x_k - v_i|| = \left[\sum_{j=1}^{m} \left(x_{kj} - v_{ij}\right)^2 \right]^{\frac{1}{2}} \tag{3.3}$$

where the objective function is specified as follows and d_{ik} is the euclidean distance between the k^{th} data sample x_k and the ith cluster centre v_i:

$$J(U,V) = \sum_{k=1}^{n}\sum_{i=1}^{c} \chi_{ik} (d_{ik})^2 \tag{3.4}$$

where U is the partition matrix and $enquoteV$ is a vector representing the cluster centre. In order to find the best partition, U^* must be the partition that results in the function J's minimum value.

$$J(U^*,V^*) = \min_{U \in M_c} J(U,V) \tag{3.5}$$

3.2 Algorithm

Algorithm 1 FCM Algorithm

1. Fix c $(2 \leq c < n)$ and reload the U matrix

   $$U^{(0)} \in \mathcal{M}_c$$

   then do r=0,1,2,...

2. Calculate the centre vectors for c $\left\{ v_i^{(r)} \ \text{with} \ U^{(r)} \right\}$

3. Utilising the updated characteristic function, determine the value for each i,k $U^{(r)}$.

   $$\chi_{ik}^{(r+1)} = \begin{cases} 1 & d_{ik}^{(r)} = min\left\{ d_{jk}^{(r)} \right\} \quad \forall j \in c \\ 0 & otherwise \end{cases}$$

4. If $||U^{(r+1)} - U^{(r)}|| \leq \epsilon$ tolerence level then stop; otherwise set $r = r + 1$ and return to step 2

Chapter 4

CODE

we used the `optimize.linprog` command from `Scipy` module of `Python`

```
import numpy as np
import pandas as pd
import cv2

from google.colab.patches import cv2_imshow # for image display
from skimage import io
from PIL import Image
import matplotlib.pylab as plt



def InitMem(x,y,c):
  U = np.random.rand(x,y,c)
  Rsum = U.sum(axis=2)
  NU = np.divide(U,Rsum[:,:, None])
  return NU
def UpdateMem(Dist_Mat,m):
  Temp = Dist_Mat**(-1/(m-1))
  SumTemp = Temp.sum(axis=2)
  UpdatedMem = np.divide(Temp,SumTemp[:,:,None])
  return UpdatedMem
def UpdateCen(mf,data,c):
```

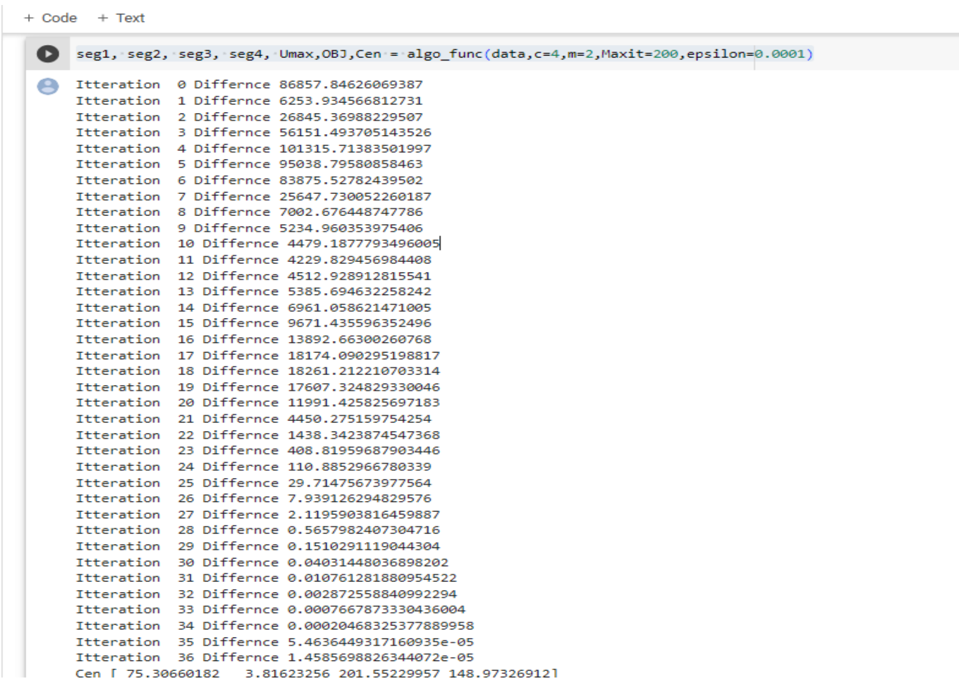
```
  data = np.repeat(data[:, :, None], repeats = c, axis=2)
  mfsum = mf.sum(axis = 1)
  mfsum = mfsum.sum(axis = 0)
  mfdata = np.multiply(mf,data)
  mfdatasum = mfdata.sum(axis = 1)
  mfdatasum = mfdatasum.sum(axis = 0)
  Cen = np.divide(mfdatasum,mfsum)
  # Cen = Cen.reshape(c,1)
  return Cen
def Distance_Mat(data,Cen):
  x,y = data.shape
  c = Cen.shape[0]
  data =  np.repeat(data[:, :, None], repeats = c, axis=2)
  dist = np.zeros((x,y,c))
  repetitions = x*y
  repeats_Cen = np.tile(Cen, (repetitions, 1))
  repeats_Cen = np.reshape(repeats_Cen,(x,y,c))
  dist = (data-repeats_Cen)**2
  return dist
def Objective_Fun(mf,Dist_Mat):
  obj = np.multiply(mf,Dist_Mat)
  objvalue = obj.sum()
  return objvalue
def FCM(Data, c, m, MaxItter,epsilon):
  # c = 4
  x,y = Data.shape
  # n, d = Data.shape
  Uinit = InitMem(x,y,c) # InitMem is updated below
  U = Uinit
  OldDif = 0.0
  OBJ=[]
  for i in range(MaxItter):
```

```
    mf = U**(m)
    Cen = UpdateCen(mf,Data,c)
    Dist_Mat = Distance_Mat(Data,Cen)
    obj = Objective_Fun(mf,Dist_Mat)
    OBJ.append(obj)
    UNew = UpdateMem(Dist_Mat,m)
    Dif = (((np.absolute(U-UNew)).sum(axis=2)).sum(axis=1)).sum()
    print("Itteration ", i, "Differnce", Dif )
    if (np.absolute(OldDif-Dif)<epsilon):
      break
    OldDif = Dif
    U = UNew
  return U,Cen,OBJ
def algo_func(data,c,m,Maxit,epsilon):
  # x,y = image.shape
  # numpydata = asarray(image)
  data = data.astype(float)
  Maxit = 200
  m=2
  c = 4
  U, Cen,OBJ = FCM(data, c, m, Maxit,epsilon)
  print('Cen', Cen)
  Umax = (U == U.max(axis=2)[:,:,None]).astype(float)
  seg1 = Umax[:,:,0]
  seg2 = Umax[:,:,1]
  seg3 = Umax[:,:,2]
  seg4 = Umax[:,:,3]
  plt.imshow(data,'gray')
  figure, axis = plt.subplots(2, 2)
  axis[0,0].imshow(seg1,'gray')
  axis[0, 1].imshow(seg2,'gray')
  axis[1, 0].imshow(seg3,'gray')
```

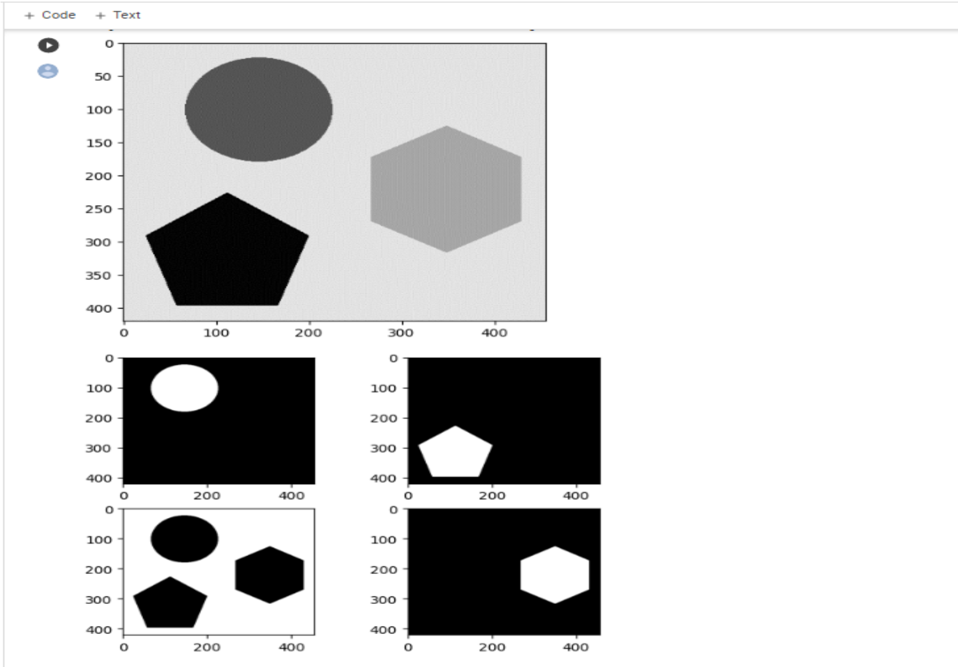
```
  axis[1, 1].imshow(seg4,'gray')
  return seg1, seg2, seg3, seg4, Umax,OBJ,Cen
def Plot(obj):
  y = obj
  y.pop(0)
  x = list(range(0,len(obj)))
  return plt.plot(x, y)
```

```
data = np.load('numpy_data.npy')
```

```
seg1, seg2, seg3, seg4, Umax,OBJ,Cen = algo_func(data,c=4,m=2,Maxit=200,epsil
```



```
+ Code  + Text
seg1, seg2, seg3, seg4, Umax,OBJ,Cen = algo_func(data,c=4,m=2,Maxit=200,epsilon=0.0001)
Itteration  0 Differnce 86857.84626069387
Itteration  1 Differnce 6253.934566812731
Itteration  2 Differnce 26845.36988229507
Itteration  3 Differnce 56151.493705143526
Itteration  4 Differnce 101315.71383501997
Itteration  5 Differnce 95038.79580858463
Itteration  6 Differnce 83875.52782439502
Itteration  7 Differnce 25647.730052260187
Itteration  8 Differnce 7002.676448747786
Itteration  9 Differnce 5234.960353975406
Itteration  10 Differnce 4479.1877793496005
Itteration  11 Differnce 4229.829456984408
Itteration  12 Differnce 4512.928912815541
Itteration  13 Differnce 5385.694632258242
Itteration  14 Differnce 6961.058621471005
Itteration  15 Differnce 9671.435596352496
Itteration  16 Differnce 13892.66300260768
Itteration  17 Differnce 18174.090295198817
Itteration  18 Differnce 18261.212210703314
Itteration  19 Differnce 17607.324829330046
Itteration  20 Differnce 11991.425825697183
Itteration  21 Differnce 4450.275159754254
Itteration  22 Differnce 1438.3423874547368
Itteration  23 Differnce 408.81959687903446
Itteration  24 Differnce 110.8852966780339
Itteration  25 Differnce 29.71475673977564
Itteration  26 Differnce 7.939126294829576
Itteration  27 Differnce 2.1195903816459887
Itteration  28 Differnce 0.5657982407304716
Itteration  29 Differnce 0.1510291119044304
Itteration  30 Differnce 0.04031448036898202
Itteration  31 Differnce 0.010761281880954522
Itteration  32 Differnce 0.002872558840992294
Itteration  33 Differnce 0.0007667873330436004
Itteration  34 Differnce 0.00020468325377889958
Itteration  35 Differnce 5.4636449317160935e-05
Itteration  36 Differnce 1.4585698826344072e-05
Cen [ 75.30660182   3.81623256 201.55229957 148.97326912]
```




```
    def DS(gt,data):
  gt = np.divide(gt,255)
  intersection = np.logical_and(gt,data)
  ds = (2*intersection.sum())/(gt.sum()+data.sum())
  return ds
def avg_intensity(image, seg1, seg2, seg3, seg4):
  avg1 = ((np.multiply(seg1,image)).sum())/(seg1.sum())
  avg2 = ((np.multiply(seg2,image)).sum())/(seg2.sum())
  avg3 = ((np.multiply(seg3,image)).sum())/(seg3.sum())
  avg4 = ((np.multiply(seg4,image)).sum())/(seg4.sum())
  avg = np.array([avg1,avg2,avg3,avg4])
  ind = np.argsort(avg)
  # print('ind',ind)
  return ind
def dicescore(gt,image,seg1,seg2,seg3,seg4,U):
  # Sorted Average Intensity of original image and noise images
  ind = avg_intensity(image,seg1, seg2, seg3, seg4)
  ds=[]
  for i in range(len(gt)):
    ds.append(DS(gt[i],U[:,:,ind[i]]))
```

```
  print('Dice Score:',ds)
  return ds
def ASA1(gt,data):
  gt = np.divide(gt,255)
  intersection = np.logical_and(gt,data)
  asa1 = intersection.sum()
  return asa1
def ASA2(gt):
  gt = np.divide(gt,255)
  return gt.sum()
def asa(gt,image,seg1,seg2,seg3,seg4,U):
  sum = 0
  num = 0
  ind = avg_intensity(image,seg1, seg2, seg3, seg4)
  for i in range(len(gt)):
    sum = sum + ASA2(gt[i])
    num = num + ASA1(gt[i],U[:,:,ind[i]])
  ASA = num/sum
  print('ASA:',ASA)
  return ASA
```

```
    gt_array = np.load('gt_array.npy')
```

```
    dice = dicescore(gt_array,data,seg1,seg2,seg3,seg4,Umax)
avgsegacc = asa(gt_array,data,seg1,seg2,seg3,seg4,Umax)
```

Dice Score: [1.0, 1.0, 1.0, 1.0] ASA: 1.0

```
    Plot(OBJ)
```

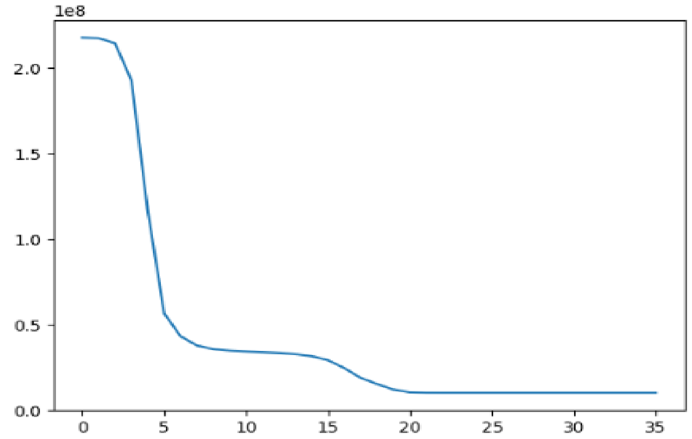
[¡matplotlib.lines.Line2D at 0x7fb94adb7dc0¿]

+ Code + Text

```
[ ]
Dice Score: [1.0, 1.0, 1.0, 1.0]
ASA: 1.0
```

```
Plot(OBJ)
```

```
[<matplotlib.lines.Line2D at 0x7fb94adb7dc0>]
```



Chapter 5

Fuzzy C-Means (FCM)

We define a fuzzy set $\{A_i \ : \ i = 1, 2, 3, ...c\}$ as a hazy c-partition on a set of X data points. The numerous data points in each fuzzy set (fuzzy class, fuzzy clusters) can now be assigned membership.

Consequently, a single point can be partially a member of more than one class. $\mu_{ik} = \mu_{Ai}(x_k) \in [0, 1]$ signifies the membership value of the kth data point in the ith class, and the sum of all the membership values for one of the data points across all classes must be 1. i.e,

$$\sum_{i=1}^{c} \mu_{ik} = 1 \qquad \forall \ k = 1, 2....n$$

and

$$0 \ < \ \sum_{i=1}^{n} \mu_{ik} \ < \ n$$

For the classification involving c classes and ndata points, we may now create a set of fuzzy partition matrices, M_{fc}.

$$M_{fc} = \left\{ \underset{\sim}{U} \ \mid \ \mu_{ik} \in [0, 1]; \sum_{i=1}^{c} \mu_{ik} = 1; \ \ 0 < \sum_{k=1}^{n} \mu_{ik} < n \right\} \tag{5.1}$$

where $i = 1, 2, ...c$ and $k = 1, 2, ...n$

Any $\underset{\sim}{U} \in M_{fc}$ is a fuzzy c-partition.

Given that v_i is the ith cluster centre, which is represented by m features (m coordinates), v_i can be arranged in a vector form, i.e. $v_i \ = \{v_{i1}, v_{i2}, ... \mathrm{v}_{im}$. Similar to the procedure in the crisp case, one can compute each cluster coordinate for each class:-

$$v_{ij} = \frac{\sum\limits_{k=1}^{n} \mu_{ik}^{m'} .x_{ik}}{\sum\limits_{k=1}^{n} \mu_{ik}^{m'}} \tag{5.2}$$

where the feature space's variable j is j=1,2,...m

We construct an objective J_m for a fuzzy c-partition, which is, to determine the fuzzy c-partition matrix $\underset{\sim}{U}$ for classifying a collection of n data sets into c classes.

$$J_m \left(\underset{\sim}{U}, V \right) = \sum_{k=1}^{n} \sum_{i=1}^{c} \left(\mu_{ik} \right)^{m'} \left(d_{ik} \right)^2 \tag{5.3}$$

and where ,

$$d_{ik} = d \left(x_k - v_i \right) = \left[\sum_{j=1}^{m} \left(x_{kj} - v_{ij} \right)^2 \right]^{\frac{1}{2}} \tag{5.4}$$

5.1 Algorithm

Algorithm 2 PFCM Algorithm

1. Fix c $(2 \leq c < n)$ and select a value for parameter m'. Initialize the partition matrix $\underset{\sim}{U}^{(0)}$. Each step in this algorithm will be labeled r, where r = 0,1,2,...
2. Calculate the centres for c $\left\{ v_i^{(r)} \right\}$ for each step.
3. Update the partition matrix for r^{th} step, $\underset{\sim}{U}^{(r)}$ as follows:

$$\mu_{ik}^{(r+1)} = \left[\sum_{j=1}^{c} \left(\frac{d_{ik}^{(r)}}{d_{jk}^{(r)}} \right)^{\frac{2}{m'-1}} \right]^{-1}$$

4. If $||\underset{\sim}{U}^{(r+1)} - \underset{\sim}{U}^{(r)}|| \leq \epsilon$ stop; otherwise set r=r+1 and return to step 2.

5.2 Example

U^* as the initial fuzzy partition, $U^{(0)}$ and assuming a weighting factor of $m' = 2$ and a criterion for convergence of $\epsilon = 0.01$ that's

$$\underset{i,k}{max} |\mu_{ik}^{(r+1)} - \mu_{ik}^{(r)}| \leq 0.01$$

We seek to identify the ideal fuzzy 2-partition. $\underset{\sim}{U}$

First, the fuzzy initial partition is

$$U^{(0)} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The initial cluster centres' calculation follows:

$$v_{ij} = \frac{\sum_{k=1}^{n} \mu_{ik}^2 . x_{kj}}{\sum_{k=1}^{n} \mu_{ik}^2}$$

where for c=1

$$v_{1j} = \frac{\mu_1^2 x_{1j} + \mu_2^2 x_{2j} + \mu_3^2 x_{3j} + \mu_4^2 x_{4j}}{\mu_1^2 + \mu_2^2 + \mu_3^2 + \mu_4^2}$$

$$v_{11} = \frac{1 + 1.5 + 1.3}{3} = 1.26$$

$$v_{12} = \frac{3 + 3.2 + 2.8}{3} = 3.0$$

then $v_1 = \{1.26, 3\}$

and similarly for c=2

$$v_{21} = \frac{3}{1} = 3$$

$$v_{22} = \frac{1}{1} = 1$$

then $v_2 = \{3, 1\}$

The distances between each data point and each cluster centre are now calculated as follows:

$$d_{11} = \sqrt{(1 - 1.26)^2 + (3 - 3)^2} = 0.26$$

$$d_{21} = \sqrt{(1 - 3)^2 + (3 - 1)^2} = 2.82$$

$$d_{12} = \sqrt{(1.5 - 1.26)^2 + (3.2 - 3)^2} = 0.31$$

$$d_{13} = \sqrt{(1.3 - 1.26)^2 + (2.8 - 3)^2} = 0.20$$

$$d_{14} = \sqrt{(3 - 1.26)^2 + (1 - 3)^2} = 2.65$$

similarly we get $d_{22} = 2.66$; $d_{23} = 2.47$; $d_{24} = 0$

$$\mu_{12} = \left[\left(\frac{d_{12}}{d_{12}} \right)^2 + \left(\frac{d_{12}}{d_{22}} \right)^2 \right]^{-1} = 0.986$$

We are now able to update using the distance measurements $\underset{\sim}{U}$ for $m' = 2$

$$\mu_{ik}^{(r+1)} = \left[\sum_{j=1}^{c}\left(\frac{d_{ik}^{(r)}}{d_{jk}^{(r)}}\right)^2\right]^{-1}$$

$$\mu_{11} = \left[\sum_{j=1}^{c}\left(\frac{d_{11}}{d_{j1}}\right)^2\right]^{-1} = \left[\left(\frac{d_{11}}{d_{11}}\right)^2 + \left(\frac{d_{11}}{d_{21}}\right)^2\right]^{-1}$$

$$= \left[\left(\frac{0.26}{0.26}\right)^2 + \left(\frac{0.26}{2.82}\right)^2\right]^{-1}$$

$$= 0.991$$

$$\mu_{13} = \left[\left(\frac{d_{13}}{d_{13}}\right)^2 + \left(\frac{d_{13}}{d_{23}}\right)^2\right]^{-1} = 0.993$$

$$\mu_{14} = \left[\left(\frac{d_{14}}{d_{14}}\right)^2 + \left(\frac{d_{14}}{d_{24}}\right)^2\right]^{-1} = 0$$

Hence we obtain

$$\underset{\sim}{U}^{(1)} = \begin{bmatrix} 0.991 & 0.986 & 0.993 & 0 \\ 0.009 & 0.014 & 0.007 & 1 \end{bmatrix}$$

We select a matrix norm || such as the highest pairwise comparison value of each value in the matrix to determine whether or not we have achieved convergence $\underset{\sim}{U}^{(0)}$ and $\underset{\sim}{U}^{(1)}$

i.e,

$$\max_{i,k}|\mu_{ik}^{(1)} - \mu_{ik}^{(0)}| = 0.014 > 0.01$$

This result indicates that the procedure needs to be iterated upon in order to determine whether our convergence conditions have been met.

We then calculate cluster centres once more for the following iteration using values from the most recent fuzzy partition. $\underset{\sim}{U}^{(1)}$ for C=1

$$v_{11} = \frac{(0.991)^2\,.1 + (0.986)^2\,.\,(1.5) + (0.993)^2\,.\,(1.3)}{(0.991)^2 + (0.986)^2 + (0.993)^2}$$

$$v_{11} = \frac{3.719}{2.94} = 1.26$$

Similarly,

$$v_{12} = \frac{8.816}{2.94} = 3.0$$

Therefore,

$$v_1 = \{1.26, 3\}$$

Now for C=2

$$v_{21} = \frac{(0.009)^2 .1 + (0.014)^2 . (1.5) + (0.007)^2 . (1.3) + (1) .3}{(0.009)^2 + (0.014)^2 + (0.007)^2}$$

$$v_{21} = 3.0$$

$$v_{22} = 1.0$$

then

$$v_2 = \{3, 1\}$$

we observe that these two cluster centers are identical to those from the first step, at least to with in the stated accuracy of (0.01)

Hence the final partiton $\underset{\sim}{U}^{(2)}$ result as:

$$\underset{\sim}{U}^{(2)} = \begin{bmatrix} 0.991 & 0.986 & 0.993 & 0 \\ 0.009 & 0.014 & 0.007 & 1 \end{bmatrix}$$

Chapter 6

Fuzzy set and it's extensions

6.1 Fuzzy set

A Fuzzy set is a set in which each member element will have the fractional membership via a membership function

$$\mu_A : X ß[0, 1] \tag{6.1}$$

which gives its degree of belongingness [14]. It is possible to represent A, a fuzzy set said over a set X, as follows:

$$A = (x, \mu A(x)) : xX \tag{6.2}$$

6.2 Intuitionistic fuzzy set

A fuzzy set over X that is expressed as follows is an extension of an intuitionistic fuzzy set B.

$$B = (x, \mu B(x), \nu B(x)) : xX \; and \; 0\mu B(x) + \nu B(x)1 \tag{6.3}$$

where

$$\mu B : X \longrightarrow [0, 1], \; \nu B : X \longrightarrow [0, 1] \tag{6.4}$$

are the functions of an element x's membership and non-membership in set B. When $\mu B(x) + \nu B(x) = 1$ for every x in B, the IFS B becomes FS B.

An ordered triple set serves as the representation of A in an intuitionistic fuzzy set.

$(x, \mu A(x), \nu A(x))$,

where,

$\mu A(x)$ is the degree at which x is a member of A, and $\nu A(x)$ is the degree at which x is not a member of A. The degree of hesitation, hA(x), is a mathematical constant.

6.3 Hesitation Degree

hA(x) = 1 - $\mu A(x) - \nu A(x)$

When compared to membership and non-membership degrees, the hesitation degree reflects how uncertain or ambiguous those degrees are. The total of the membership and non-membership degrees divided by 1 is used to calculate it.

6.4 Picture Fuzzy Set

[11] proposed a Picture Fuzzy Set (PFS), which is generalization of conventional fuzzy set and intuitionistic fuzzy set. A PFS is a non empty set X given by

$$A = hx, uA(x), \eta A(x), \gamma A(x)i|xX \tag{6.5}$$

where µA(x) is the value of each element's positive membership, $\eta A(x)$ is its neutral membership degree, and $\gamma A(x)$ is its negative membership degree that satisfies the constraints.

$$0 \leq \mu A(x) + \eta A(x) + \gamma A(x) \leq 1 \tag{6.6}$$

An element's refusal degree is computed as follows:

$$\xi A(x) = 1 - (\mu A(x) + \eta A(x) + \gamma A(x)$$

In case $\xi A(x) = 0$ PFS returns Intuitionistic fuzzy set.

If $\xi A(x) = \eta A(x) = 0$ PFS returns to fuzzy set.

6.5 Picture Fuzzy set representation of Image

The fuzzy complement generator developed by Yager is used to create the fuzzy image. Take a look at the image X = x1, x2,.....xNi, which consists of N pixels with intensities ranging from 0 to L - 1. The image's PFS representation can be specified as follows:

I = (xij, $\mu I(x_{ij})$, $\eta I(x_{ij})$, $\gamma I(x_{ij})$, $\xi I(x_{ij})$)

where I represents the refusal degree of the pixel, I represents the neutral membership value, I represents the negative membership value, and uI represents the positive membership value. Each pixel in an image has a corresponding intensity value. We compute the normalised intensity level for each pixel to translate the intensity data into membership values. i.e:

$$\mu I(x_{ij}) = \frac{x_{ij}}{L-1} \tag{6.7}$$

In this study, we used Yager's fuzzy complement generator to calculate the negative membership value. The fuzzy complement generator by Yager is described as follows:

$$\gamma I(x_{ij}) = (1(\mu I(x_{ij}) + \nu I(x_{ij}))^{\alpha})^{\frac{1}{\alpha}} \tag{6.8}$$

Following the use of Yager's fuzzy complement generator, the PFS picture is thus:

$$I_{\alpha}^{PFS} = (x_{ij}, \mu I(x_{ij}), \nu I(x_{ij}), (1(\mu I(x_{ij}) + \nu I(x_{ij}))^{)^{\frac{1}{\alpha}}, \xi I(x_{ij}))}(6.9)$$

The refusal degree of the pixel is calculated as:

$$\xi I(x_{ij}) = 1(\mu I(x_{ij}) + \nu I(x_{ij}))(1(\mu I(x_{ij}) + \nu I(x_{ij})^{\alpha}))^{\frac{1}{\alpha}} \tag{6.10}$$

where is exponent, the value varies between 0 and 1.

6.5.1 Picture Fuzzy Clustering

The Picture Fuzzy Clustering (PFC) method for segmenting MRI brain pictures is shown in this section. This method clusters the image by searching for local minima of the subsequent objective function:

$$J = \sum_{i=1}^{N}\sum_{J=1}^{C}(u_{ij}(2\xi_{ij}))^m||x_i - v_j|| + \sum_{i=1}^{N}\sum_{J=1}^{C}\nu ij((log\nu_{ij} + \xi_{ij}) \quad (6.11)$$

When the element fits the following characteristics and $_{ij}, \nu_{ij}, and\xi_{ij} arethepositive, neutral, and$

$$\mu_{ij} + \nu_{ij} + \xi_{ij}1 \quad (6.12)$$

$$\sum_{j=1}^{C}(\mu_{ij}(2 - \xi_{ij})) = 1 \quad (6.13)$$

$$\sum_{j=1}^{C}\nu_{ij} + \frac{\xi_{ij}}{c} = 1 \quad (6.14)$$

The Lagrangian approach is used to discover the optimal solutions of the objective function, and the optimal solutions of the systems for vj, $\mu_{ij}, \nu_{ij}, and\xi_{ij}\ are:$
$vj = \frac{\sum_{i=1}^{N}(\mu_{ij}(2-\xi_{ij}))^m x_i}{\sum_{i=1}^{N}(\mu_{ij}(2-\xi_{ij}))^m}$(6.15)

$$u_{ij} = \frac{1}{\sum_{i=1}^{N}(\mu_{ij}(2 - \xi_{ij})\left(\frac{||x_i-v_j||}{||x_i-v_k||}\right)^{\frac{2}{m-1}}} \quad (6.16)$$

$$\nu_{ij} = \frac{e^{-\xi_{ij}}}{\sum_{k=1}^{C}e^{-\xi_{ik}}}\left(1 - \frac{1}{c}\sum_{k=1}^{C}\xi_{ik}\right) \quad (6.17)$$

$$\xi_{ij} = 1 - (u_{ij} + \nu_{ij})(1(u_{ij} + \nu ij)^{\alpha})^{\frac{1}{\alpha}} \quad (6.18)$$

where i = 1.,,,,N, k = 1.,,,,,n, j = 1,.....c.

The membership value of fuzzy set-based clustering methods is influenced by the distance measure. Intensity values that are nearer to the cluster centre value indicate that the pixel has a higher membership value. In turn, this makes the membership value more noise-sensitive. Due of the noise and intensity inhomogeneity that the MRI brain pictures contain, the euclidean distance measure does not produce the desired segmentation results. The distance between the cluster centre and the pixel was calculated in this study using the image euclidean distance function, which took noise and intensity inhomogeneity into consideration. The following formula is used to determine picture euclidean distance:

$$d(x_i, v_j) = ((u(x_i)u(v_j)) + (\nu(x_i)\nu(vj)) + (\gamma(xi)\gamma(v_j)))^{\frac{1}{2}} \quad (6.19)$$

6.6 Distance between picture fuzzy sets [1]

Distances for two picture fuzzy sets A and B in X = $x_1, x_2, ..., x_n are$:

- The normalized Hamming distance dP(A,B)

$$dP(A,B) = \frac{1}{n}\sum_{i=1}^{n}(|\mu A(x_i)\mu B(x_i)| + |\eta A(x_i)\eta B(x_i)| + |\nu A(x_i)\nu B(x_i)| \tag{6.20}$$

- The normalized Euclidean distance eP(A,B)

$$eP(A,B) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}((\mu A(x_i))\mu B(x_i))^2 + (\eta A(x_i)\eta B(x_i))^2 + (\nu A(x_i)\nu B(x_i))^2)} \tag{6.21}$$

Chapter 7

Applications

7.1 Image Segmentation

There are many practical uses for image segmentation, which divides a picture into distinct regions or objects, across a variety of industries. Here are a few instances:

(a)Autonomous Vehicles = Image segmentation is crucial for scene comprehension and object detection in the context of autonomous cars. Autonomous vehicles can sense their environment and make wise decisions by segmenting various items such as pedestrians, vehicles, and road markings.

(b)Object Recognition and Tracking = Object recognition and tracking systems use image segmentation. In video surveillance, robotics, and augmented reality applications, segmentation aids in recognising and monitoring particular items of interest by separating them from the backdrop.

(c) Augmented Reality = Image segmentation is used in augmented reality applications to distinguish between foreground and background objects. This permits the accurate and realistic overlay of digital data or virtual objects onto the physical environment.

(d)Video Compression and Streaming = By detecting areas of interest and allocating resources appropriately, image segmentation helps with video compression and streaming. Compression algorithms can prioritise crucial parts and assign higher bitrates by segmenting the video frames into significant regions. This enhances video quality and bandwidth efficiency.

7.2 Intuitionistic fuzzy sets

Intuitionistic fuzzy sets (IFS) have been applied in various domains where uncertainty and ambiguity are prevalent. Here are some common applications of intuitionistic fuzzy sets:

(a)Decision Making = To deal with ambiguous and imperfect information, decision-making systems can use intuitionistic fuzzy sets. IFS can capture the hesitations and uncertainties in decision models by taking into account both membership and non-membership degrees, enabling more reliable and adaptable decision-making.

(b) Pattern recognition = To handle complicated and ambiguous data in pattern recognition tasks, intuitionistic fuzzy sets have been used. In comparison to conventional crisp or fuzzy sets, they may effectively capture ambiguous patterns and offer a more precise classification and identification framework.

(c)Image processing = Applications for IFS can be found in image processing tasks like object recognition, edge detection, and image segmentation. The accuracy and dependability of these activities are improved by having the ability to deal with ambiguity and uncertainty in picture data.

(d) Medical Diagnosis = In medical diagnosis, where imprecise and ambiguous data are frequently encountered, intuitionistic fuzzy sets have been used to describe symptoms, diseases, and diagnostic criteria. In addition to helping with uncertainty, they offer a more precise diagnosis.

7.3 Picture fuzzy set

As an extension of conventional fuzzy sets, picture fuzzy sets—also known as fuzzy sets with membership grades as images—have been presented. By employing pictures or images rather than numbers, they give a better visual representation of fuzzy membership ratings. Despite being a relatively recent idea, photo fuzzy sets offer a wide range of practical uses. Here are a few illustrations:

(a)Human Computer Interactions= Picture fuzzy sets can be used in human-computer interaction interfaces to record and decode user input and gesticulations. It enables a more natural and intuitive engagement with computers or other interactive systems by displaying fuzzy membership grades as visuals.

(b) Risk Assessment = Picture fuzzy sets can be used in risk assessment activities like analysing the susceptibility of vital infrastructure or the likelihood and consequences of natural disasters. It offers a more visual and thorough comprehension of risk levels by portraying ambiguous and imprecise risk indicators as image-based membership grades.

(c)Medical Imaging = Picture fuzzy sets may be used in the interpretation and analysis of MRI or CT scan pictures, among other applications. It aids in the diagnosis and treatment planning process by portraying ambiguous or uncertain picture elements as image-based membership ratings.

Bibliography

[1] D. J. Dubois, Fuzzy sets and systems: theory and applications, Vol. 144, Academic press, 1980.

[2] H.-D. Cheng, X. H. Jiang, Y. Sun, J. Wang, Color image segmentation: advances and prospects, Pattern recognition 34 (12) (2001) 2259–2281.

[3] H. Izakian, A. Abraham, Fuzzy c-means and fuzzy swarm for fuzzy clustering problem, Expert Systems with Applications 38 (3) (2011) 1835–1838.

[4] J. C. Dunn, A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters (1973).

[5] J. C. Bezdek, Pattern recognition with fuzzy objective function algorithms, Springer Science & Business Media, 2013.

[6] R. N. Davé, S. Sen, Generalized noise clustering as a robust fuzzy cm-estimators model, in: 1998 Conference of the North American Fuzzy Information Processing Society-NAFIPS (Cat. No. 98TH8353), IEEE, 1998, pp. 256–260.

[7] R. Krishnapuram, J. M. Keller, A possibilistic approach to clustering, IEEE transactions on fuzzy systems 1 (2) (1993) 98–110.

[8] K. T. Atanassov, K. T. Atanassov, Intuitionistic fuzzy sets, Springer, 1999.

[9] T. Chaira, A novel intuitionistic fuzzy c means clustering algorithm and its application to medical images, Applied soft computing 11 (2) (2011) 1711–1717.

[10] S. Zhang, R. Wang, X. Zhang, Identification of overlapping community structure in complex networks using fuzzy c-means clustering, Physica A. 374 (2007) 483–490.

[11] B. C. Cuong, V. Kreinovich, Picture fuzzy sets, Journal of Computer Science and Cybernetics 30 (4) (2014) 409–420.

[12] P. H. Thong, L. H. Son, Picture fuzzy clustering: a new computational intelligence method, Soft computing 20 (9) (2016) 3549–3562.

[13] S. A. Kumar, B. Harish, V. M. Aradhya, A picture fuzzy clustering approach for brain tumor segmentation, in: 2016 second International Conference on cognitive computing and information processing (CCIP), IEEE, 2016, pp. 1–6.

[14] L. A. Zadeh, Fuzzy sets, Information and control 8 (3) (1965) 338–353.