Histogram Clustering with Expectation-Maximization on SAR Images

KK Feng

1 Problem

1.1 Description

Image segmentation[1] problem for synthetic aperture radar (SAR) images[2].

1.2 Data

- **histograms.bin**: the histograms extracted from an 800 by 800 grayscale image per following procedure
  1. Select a subset of pixels.
  2. Place a rectangle of fixed radius around the site pixel.
  3. Select all pixels within the rectangle and sort their intensity values into a histogram.

1.3 Idea

- Extract the histograms from the image
  - The histograms were drawn at the nodes of a 4-by-4 pixel grid, therefore there are 200 by 200 = 40000 histograms.
  - Each histogram was drawn within a rectangle of edge length 11 pixels, so each histogram contains 11 by 11 = 121 values.
- Apply the Expectation-Maximization algorithm and a finite mixture of multinomial distributions.

2 Solution

2.1 Plot

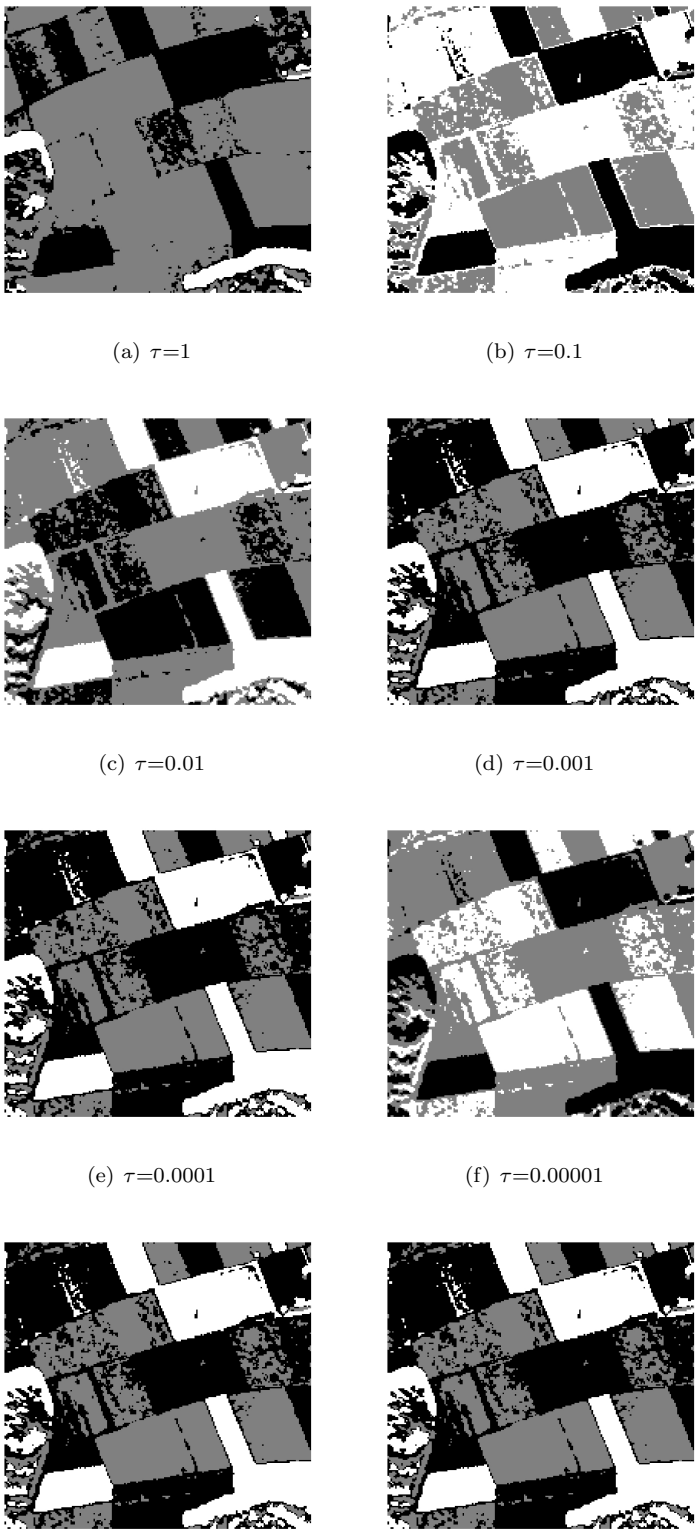
We have an integer K which specifies the number of clusters and a threshold parameter τ which is used for the termination of the iteration. For $K = 3, 4, 5$, we tested τ for following values

$$1,\ 0.1,\ 0.01,\ 0.001,\ 0.0001,\ 0.00001, 0.000001, 0.0000001 \tag{2.1}$$

Then we visualize the clustering results, which are the numbers of clusters assigned to the histograms, as an image. Note that when use *image* function in R, we need to rotate the axes to get the images to the right position. From the figures below, we can see that the larger K is, the smaller τ the algorithm requires to get a convergent solution.

[1] Image segmentation refers to the task of dividing an input image into regions of pixels that belong together.

[2] This type of image is well-suited for segmentation by histogram clustering, because the local intensity distributions provide distinctive information about the segments.



(a) τ=1

(b) τ=0.1

(c) τ=0.01

(d) τ=0.001

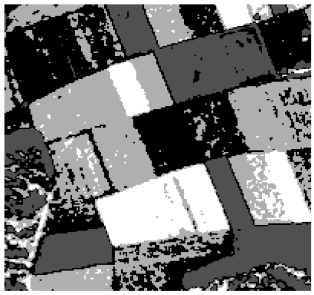
(e) τ=0.0001

(f) τ=0.00001

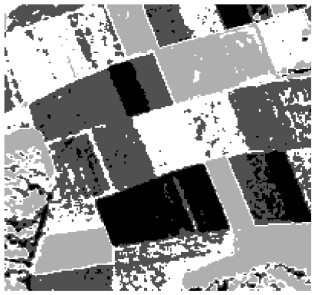
(g) τ=0.000001

(h) τ=0.0000001

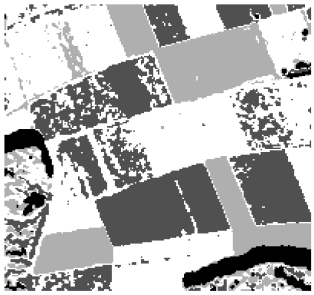
Figure 1: K=3



(a) τ=1

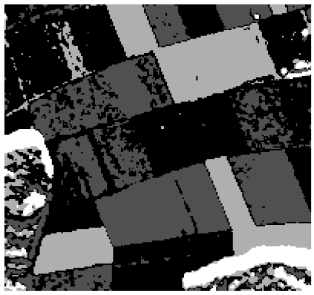


(b) τ=0.1

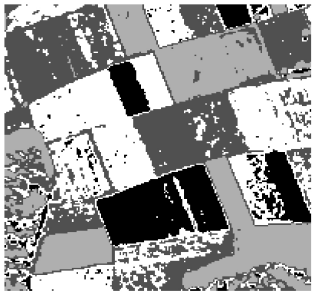


(c) τ=0.01

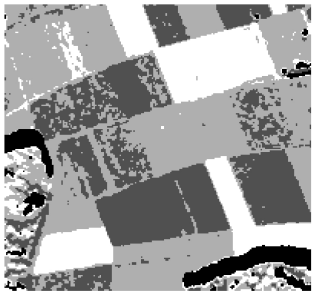
(d) τ=0.001



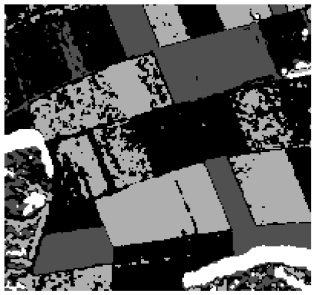
(e) τ=0.0001



(f) τ=0.00001

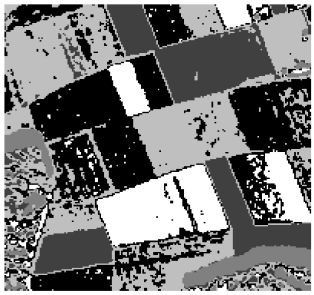


(g) τ=0.000001

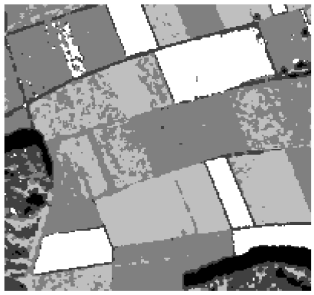


(h) τ=0.0000001

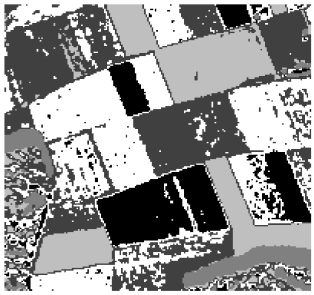
Figure 2: K=4



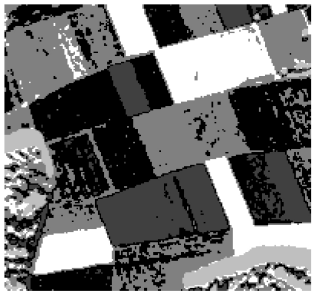
(a) τ=1



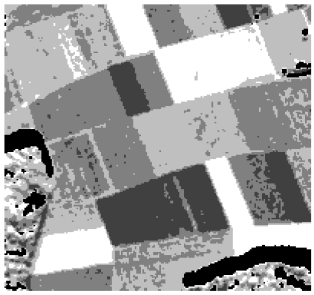
(b) τ=0.1



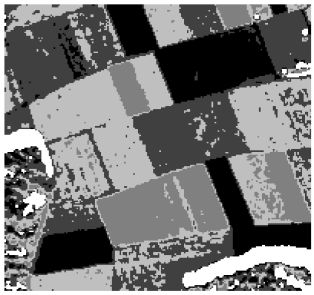
(c) τ=0.01



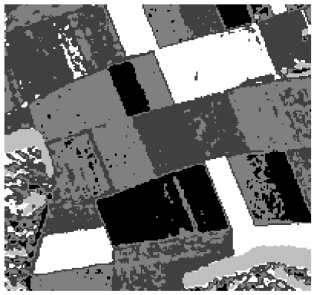
(d) τ=0.001



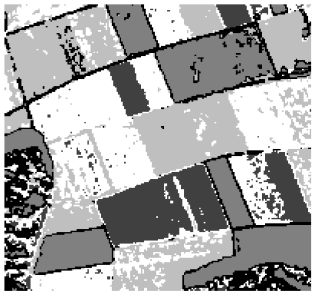
(e) τ=0.0001



(f) τ=0.00001



(g) τ=0.000001



(h) τ=0.0000001

Figure 3: K=5

2.2 Code

```
1
2  ###### Histogram Clustering with Expectation-Maximization on SAR Images ######
3
4  H <- matrix(readBin("histograms.bin", "double", 640000), 40000, 16)
5  H[,] <- H[,] + 0.01  # add a small constant (such as 0.01) to the input histograms.
6
7  # Implement the EM algorithm
8  MultinomialEM <- function(H, K, tau){
9    n <- dim(H)[1]
10   d <- dim(H)[2]
11   c <- rep(1,K) / K;
12   # Choose K of the histograms at random and normalize each. These are our initial centroids
         t1, ..., tK.
13   centroid = H[sample(1:n,K),]
14   for (i in (1:K)){
15     centroid[i,] <- centroid[i,] / sum(centroid[i,])
16   }
17   A <- matrix(0, nrow = n, ncol = K)
18   # Iterate
19   while (TRUE){
20     # E-step
21     Phi <- exp(H %*% t(log(centroid*121)))
22     A_old <- A
23     A <- (Phi * c) * matrix(rep(1/rowSums(Phi * c), K), nrow = n, ncol = K)
24     # M-step
25     c <- colSums(A) / n
26     b <- t(A) %*% H
27     centroid <- b * matrix(rep(1/rowSums(b), d), nrow = K, ncol = d)
28     # measure for terminating the iteration
29     if (norm(A-A_old, type="o")<tau){
30       break
31     }
32   }
33   # Turn the soft assignments into a vector m of hard assignments
34   m <- rep(0,n)
35   for (i in (1:n)){
36     m[i] <- which.max(A[i,])
37   }
38   return(m)
39 }
40
41 # Run for K=3,4,5. You may have to try different values of tau to obtain a reasonable result
       . Visualize the results as an image.
42 VisualizeMEM <- function(m, K, tau){
43   M <- matrix(m, nrow = 200, ncol = 200)
44   # image(x=1:200, y=1:200, M, col=grey((0:2^4)/2^4), xlab="row", ylab="col") # 4-bit image
45   # rotate the axes
46   img <- matrix(0, nrow = 200, ncol = 200)
47   for (i in (1:200)){
48     img[,i] <- M[,200+1-i]
49   }
50   image(x=1:200, y=1:200, img, col=grey((0:2^4)/2^4), xlab="", ylab="", axes = FALSE)
51 }
52
53 tau <- c(1,0.1,0.01,0.001,0.0001,0.00001,0.000001,0.0000001) # try different values of tau
54 for (K in (3:5)){
55   for (i in (1:length(tau))){
56     m <- MultinomialEM(H, K, tau[i])
57     VisualizeMEM(m)
58   }
59 }
```