ECE 271A: Statistical Learning I

Homework 1

Name	:	Sanchit Gupta
PID	:	A59010276
Date of Submission	:	10^{th} October 2022

Solution for Question 5

a) The prior probabilities are calculated by taking a ratio of the length of foreground and background training data with the total length of the training data respectively. Foreground corresponds to cheetah and background corresponds to grass. The prior probabilities obtained are as follows:

- P_Y (cheetah) = 0.1919
- P_Y (grass) = 0.8081

The code snippet written to calculate the prior probabilities is shown below.

```
% Part a: Calculation of Prior Probabilities
length_TrainSampleFG    = length(TrainsampleDCT_FG);
length_TrainSampleBG    = length(TrainsampleDCT_BG);

P_cheetah               = length_TrainSampleFG / (length_TrainSampleFG +
length_TrainSampleBG);
P_grass                 = length_TrainSampleBG / (length_TrainSampleFG +
length_TrainSampleBG);
```

b) The steps followed for computation and plotting of index histograms $P_{X|Y}$ (x | cheetah) and $P_{X|Y}$ (x | grass) are as follows:

- For each row in the provided training data for foreground and background, the index of the 2nd largest energy value is found.
- These indexes are stored in arrays, one each for foreground and background. The length of these arrays is equal to the length of training data for foreground and background.
- The histogram of these arrays of indexes is created which gives the total number of instances for each index. The index histograms for foreground and background are shown in Figures 1 and 2 respectively.

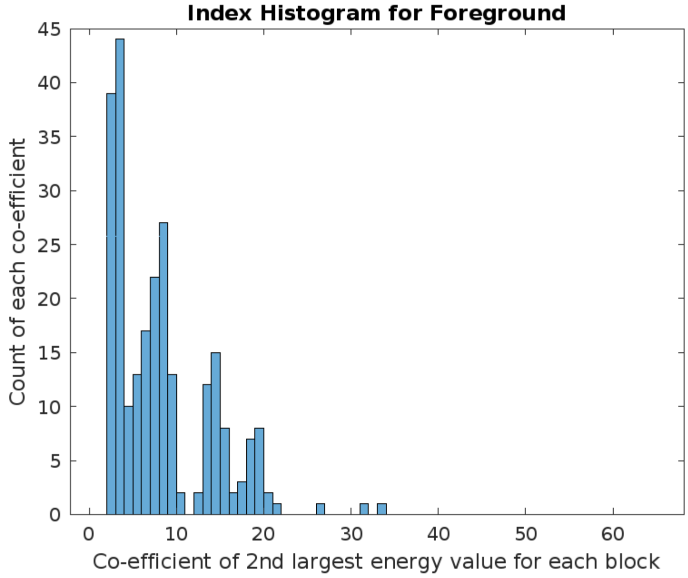

Figure 1: Index histogram for foreground using the training data provided

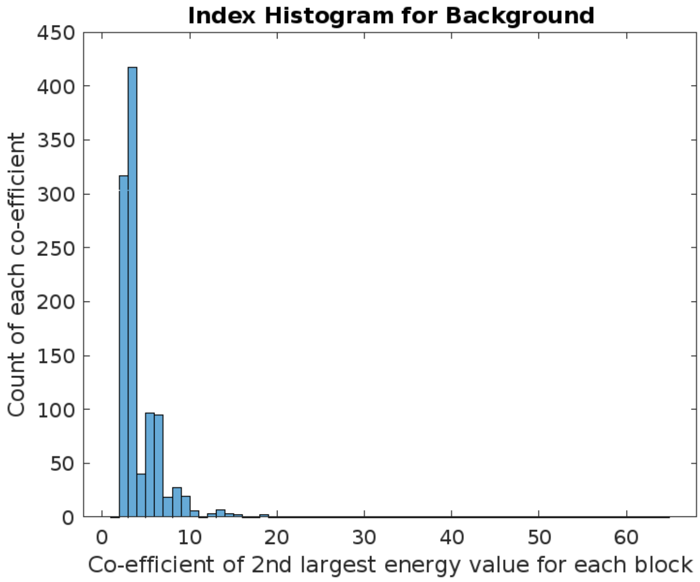

Figure 2: Index histogram for background using the training data provided

- Once the histogram is plotted, the class conditional probabilities $P_{X|Y}$ (x | cheetah) and $P_{X|Y}$ (x | grass) for each index in foreground and background is calculated by taking a ratio of number of instances of a particular index with the total length of the training data provided. This process is followed individually for index histograms of foreground and background. The conditional probability chart for foreground and background are shown in Figures 3 and 4 respectively.

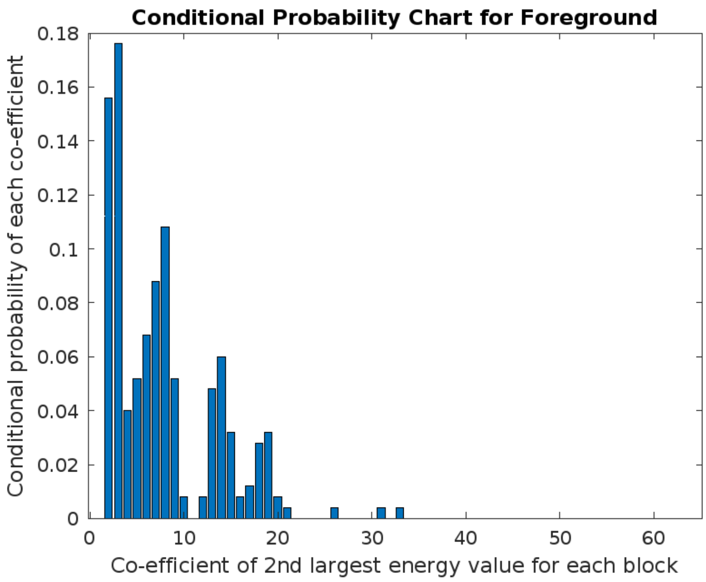

Figure 3: Conditional probability chart for foreground

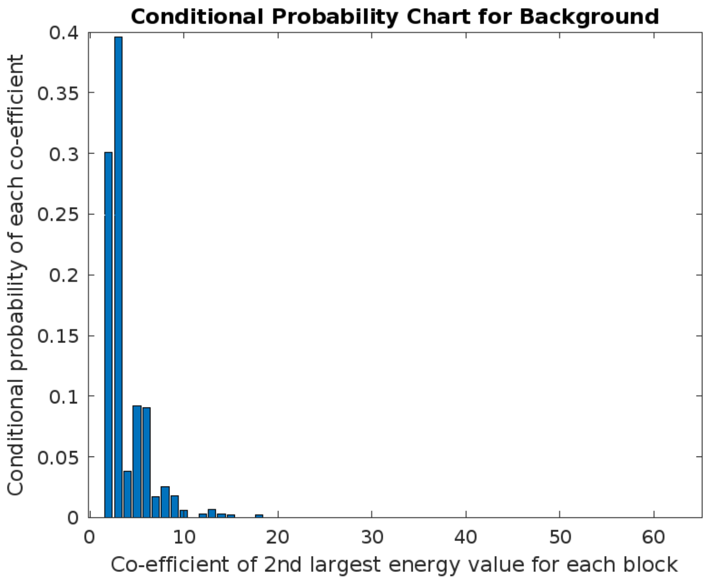

Figure 4: Conditional probability chart for background

The code snippet written for this part is shown below.

```
% Part b: Computation and plot of index histograms

index_List_BG            = zeros(length_TrainSampleBG,1);
index_List_FG            = zeros(length_TrainSampleFG,1);
class_Conditionals_BG    = zeros(64,1);
class_Conditionals_FG    = zeros(64,1);

for i = 1:length_TrainSampleBG
    row                     = abs(TrainsampleDCT_BG(i,:));
    temp_array              = sort( row , 'descend' );
    sec_largest_eng_val     = temp_array(2);
    ind_sec_largest_eng_val = find(row == sec_largest_eng_val);
    index_List_BG(i)        = ind_sec_largest_eng_val;
end

for i = 1:length_TrainSampleFG
    row                     = abs(TrainsampleDCT_FG(i,:));
    temp_array              = sort( row , 'descend' );
    sec_largest_eng_val     = temp_array(2);
    ind_sec_largest_eng_val = find(row == sec_largest_eng_val);
    index_List_FG(i)        = ind_sec_largest_eng_val;
end

figure;
hist_BG = histogram(index_List_BG, 1:65);
title('Index Histogram for Background')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Count of each co-efficient')
saveas(gcf,'Index Histogram_BG.png')

figure;
```

```
hist_FG = histogram(index_List_FG, 1:65);
title('Index Histogram for Foreground')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Count of each co-efficient')
saveas(gcf,'Index Histogram_FG.png')

% Computation of class conditional probabilities

sum_BG  = sum(hist_BG.Values);
sum_FG  = sum(hist_FG.Values);

for i = 1:64
    class_Conditionals_BG(i) = hist_BG.Values(i) / sum_BG;
    class_Conditionals_FG(i) = hist_FG.Values(i) / sum_FG;
end

figure;
hist_BG = bar(class_Conditionals_BG);
title('Conditional Probability Chart for Background')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Conditional probability of each co-efficient')
saveas(gcf,'Cond_Prob_BG.png')

figure;
hist_FG = bar(class_Conditionals_FG);
title('Conditional Probability Chart for Foreground')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Conditional probability of each co-efficient')
saveas(gcf,'Cond_Prob_FG.png')
```

c) The steps followed to compute feature X, state variable Y and mask array A are as follows:

- The given input image 'cheetah.bmp' and the text file 'Zig-Zag Pattern.txt' are read. The input image is of size 255 x 270.
- The given image is broken into 8x8 overlapping blocks. DCT is computed for each block.
- Using the provided zig-zag pattern, the DCT data for each block is transformed into a vector of length 64. From this vector, the index of 2^{nd} highest energy value is found.
- The index with 2^{nd} highest energy value is considered for the top-left corner pixel of the 8x8 block (*which has indices (1,1)*). The feature X matrix is computed following this process.
- The histogram of feature X matrix gives the total number of instances for each index. The state variable Y for foreground and background is calculated using the minimum probability of error rule based on the prior probabilities and conditional probabilities calculated in part a and b. Bayes Decision Rule (BDR) is followed for the minimum probability of error rule.

- Once the state variable Y for foreground and background are calculated, they are compared for each pixel. The pixel for which foreground has a higher probability is written as 1 and for which background has a higher probability is written as 0 in the mask array A.
- After computing mask array A, imagesc and colormap(gray(255)) functions are used to create the mask of given input image. The mask obtained is shown in Figure 5.

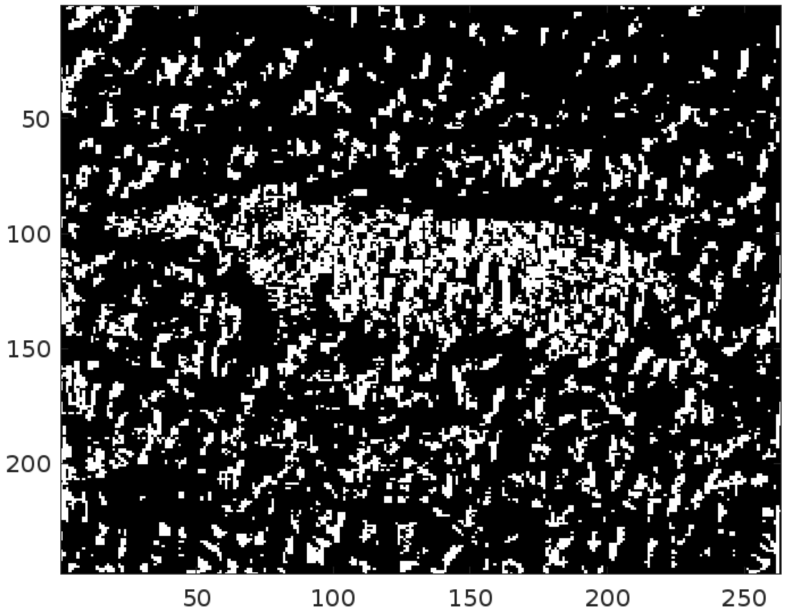

Figure 5: Mask obtained for the given input image 'cheetah.bmp'

The code snippet written for this part is shown below.

```
% Part c: Creation of mask for the given image

inputImg     = imread("cheetah.bmp");
inputImg     = im2double(inputImg);
img_Size     = size(inputImg);
img_Width    = img_Size(1);
img_Height   = img_Size(2);
winSize      = 8;
feature_X    = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
state_Y_BG   = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
state_Y_FG   = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
A            = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
fileID       = fopen('Zig-Zag Pattern.txt','r');
global zigzag
zigzag       = fscanf(fileID, '%d');

for j = 1:img_Height - winSize + 1
    for i = 1:img_Width - winSize + 1
        block                   = inputImg(i:i+winSize-1, j:j+winSize-1);
        block_DCT               = dct2(block);
        dct_Vector              = matrix_to_zigzag_vector(block_DCT);
        temp_array              = sort( dct_Vector , 'descend' );
```

```
        sec_largest_eng_val     = temp_array(2);
        ind_sec_largest_eng_val = find(dct_Vector == sec_largest_eng_val);
        feature_X(i,j)          = ind_sec_largest_eng_val;
    end
end
figure;
hist_X = histogram(feature_X, 1:65);
sum_X  = sum(hist_X.Values);

for j = 1:img_Height - winSize + 1
    for i = 1:img_Width - winSize + 1
        P_x            = hist_X.Values(feature_X(i,j)) / sum_X;
        state_Y_FG(i,j) =  ( class_Conditionals_FG(feature_X(i,j)) *
P_cheetah ) / P_x;
        state_Y_BG(i,j) =  ( class_Conditionals_BG(feature_X(i,j)) *
P_grass ) / P_x;

        if state_Y_FG(i,j) > state_Y_BG(i,j)
            A(i,j)  = 1;
        else
            A(i,j)  = 0;
        end
    end
end
figure;
imagesc(A)
colormap(gray(255))
saveas(gcf,'Mask.png')

function dct_vector = matrix_to_zigzag_vector(img_dct_block)
    dct_vector  = zeros(64,1);
    global zigzag
    for i = 1:8
        for j = 1:8
            index = zigzag( (i-1)*8 + j ) + 1;
            dct_vector(index) = img_dct_block(i,j);
        end
    end
end
```

d) The error is calculated by comparing mask array A with the ground truth mask image ‘cheetah_mask.bmp’. The total number of pixels which are different between the obtained mask and ground truth image are calculated. The error comes out to be **0.1874** or **18.74%**. The code snippet written for this part is shown below.

```
% Part d: Calculation of error
ground_Truth_Mask    = imread("cheetah_mask.bmp");
ground_Truth_Mask    = im2double(ground_Truth_Mask);

error                = sum( abs(A - ground_Truth_Mask(img_Width - winSize +
1, img_Height - winSize + 1)), "all" );
error                = error / (img_Width * img_Height);
```

Full code written to solve the quiz is shown below.

```
clc; clear; close all;
load("TrainingSamplesDCT_8.mat")

% Part a: Calculation of Prior Probabilities
length_TrainSampleFG    = length(TrainsampleDCT_FG);
length_TrainSampleBG    = length(TrainsampleDCT_BG);

P_cheetah               = length_TrainSampleFG / (length_TrainSampleFG +
length_TrainSampleBG);
P_grass                 = length_TrainSampleBG / (length_TrainSampleFG +
length_TrainSampleBG);

% Part b: Computation and plot of index histograms

index_List_BG           = zeros(length_TrainSampleBG,1);
index_List_FG           = zeros(length_TrainSampleFG,1);
class_Conditionals_BG   = zeros(64,1);
class_Conditionals_FG   = zeros(64,1);

for i = 1:length_TrainSampleBG
    row                     = abs(TrainsampleDCT_BG(i,:));
    temp_array              = sort( row , 'descend' );
    sec_largest_eng_val     = temp_array(2);
    ind_sec_largest_eng_val = find(row == sec_largest_eng_val);
    index_List_BG(i)        = ind_sec_largest_eng_val;
end

for i = 1:length_TrainSampleFG
    row                     = abs(TrainsampleDCT_FG(i,:));
    temp_array              = sort( row , 'descend' );
    sec_largest_eng_val     = temp_array(2);
    ind_sec_largest_eng_val = find(row == sec_largest_eng_val);
    index_List_FG(i)        = ind_sec_largest_eng_val;
end

figure;
hist_BG = histogram(index_List_BG, 1:65);
title('Index Histogram for Background')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Count of each co-efficient')
saveas(gcf,'Index Histogram_BG.png')

figure;
hist_FG = histogram(index_List_FG, 1:65);
title('Index Histogram for Foreground')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Count of each co-efficient')
saveas(gcf,'Index Histogram_FG.png')
```

```
% Computation of class conditional probabilities

sum_BG  = sum(hist_BG.Values);
sum_FG  = sum(hist_FG.Values);

for i = 1:64
    class_Conditionals_BG(i) = hist_BG.Values(i) / sum_BG;
    class_Conditionals_FG(i) = hist_FG.Values(i) / sum_FG;
end

figure;
hist_BG = bar(class_Conditionals_BG);
title('Conditional Probability Chart for Background')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Conditional probability of each co-efficient')
saveas(gcf,'Cond_Prob_BG.png')

figure;
hist_FG = bar(class_Conditionals_FG);
title('Conditional Probability Chart for Foreground')
xlabel('Co-efficient of 2nd largest energy value for each block')
ylabel('Conditional probability of each co-efficient')
saveas(gcf,'Cond_Prob_FG.png')

% Part c: Creation of mask for the given image

inputImg     = imread("cheetah.bmp");
inputImg     = im2double(inputImg);
img_Size     = size(inputImg);
img_Width    = img_Size(1);
img_Height   = img_Size(2);
winSize      = 8;
feature_X    = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
state_Y_BG   = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
state_Y_FG   = zeros(img_Width - winSize + 1, img_Height - winSize + 1);
A            = zeros(img_Width - winSize + 1, img_Height - winSize + 1);

fileID       = fopen('Zig-Zag Pattern.txt','r');
global zigzag
zigzag       = fscanf(fileID, '%d');

for j = 1:img_Height - winSize + 1
    for i = 1:img_Width - winSize + 1
        block                   = inputImg(i:i+winSize-1, j:j+winSize-1);
        block_DCT               = dct2(block);
        dct_Vector              = matrix_to_zigzag_vector(block_DCT);
        temp_array              = sort( dct_Vector , 'descend' );
        sec_largest_eng_val     = temp_array(2);
        ind_sec_largest_eng_val = find(dct_Vector == sec_largest_eng_val);
        feature_X(i,j)          = ind_sec_largest_eng_val;
    end
```

```
end

figure;
hist_X = histogram(feature_X, 1:65);
sum_X  = sum(hist_X.Values);

for j = 1:img_Height - winSize + 1
    for i = 1:img_Width - winSize + 1
        P_x             = hist_X.Values(feature_X(i,j)) / sum_X;
        state_Y_FG(i,j) =  ( class_Conditionals_FG(feature_X(i,j)) * P_cheetah ) /
P_x;
        state_Y_BG(i,j) =  ( class_Conditionals_BG(feature_X(i,j)) * P_grass ) /
P_x;

        if state_Y_FG(i,j) > state_Y_BG(i,j)
            A(i,j)  = 1;
        else
            A(i,j)  = 0;
        end
    end
end

figure;
imagesc(A)
colormap(gray(255))
saveas(gcf,'Mask.png')

% Part d: Calculation of error
ground_Truth_Mask   = imread("cheetah_mask.bmp");
ground_Truth_Mask   = im2double(ground_Truth_Mask);

error               = sum( abs(A - ground_Truth_Mask(img_Width - winSize + 1,
img_Height - winSize + 1)), "all" );
error               = error / (img_Width * img_Height);

function dct_vector = matrix_to_zigzag_vector(img_dct_block)
    dct_vector  = zeros(64,1);
    global zigzag
    for i = 1:8
        for j = 1:8
            index = zigzag( (i-1)*8 + j ) + 1;
            dct_vector(index) = img_dct_block(i,j);
        end
    end
end
```