Jiaming Lai
A53314574

# Assignment #1

ECE 271A
Fall 2019

## Computer Problem Solution

a) Using the training data in **TrainingSamplesDCT8.mat**, what are reasonable estimates for the prior probabilities?

**Solution**:
Two priors probabilities, $P_Y(cheetah)$ and $P_Y(grass)$, could be estimated based on the number of vectors in the training set. The estimation of $P_Y(cheetah)$ and $P_Y(grass)$ are:

$$P_Y(cheetah) = N_{FG}/(N_{FG} + N_{BG}) = 0.1919 \tag{1}$$

$$P_Y(grass) = N_{BG}/(N_{FG} + N_{BG}) = 0.8081 \tag{2}$$

where

$N_{BG}$ is the number of vectors in matrix **TrainsampleDCT_BG**

$N_{FG}$ is the number of vectors in matrix **TrainsampleDCT_FG**

b) Using the training data in **TrainingSamplesDCT8.mat**, compute and plot the index histograms $P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$.

**Solution**:
According to training data, the frequency histograms is the following picture:

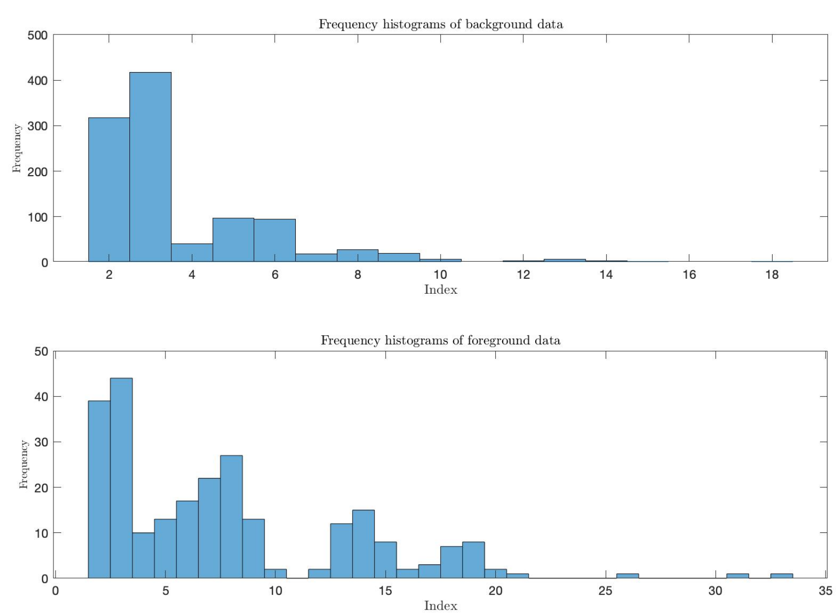


Figure 1: Frequency histograms

The index histograms of $P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$ is showed as following:

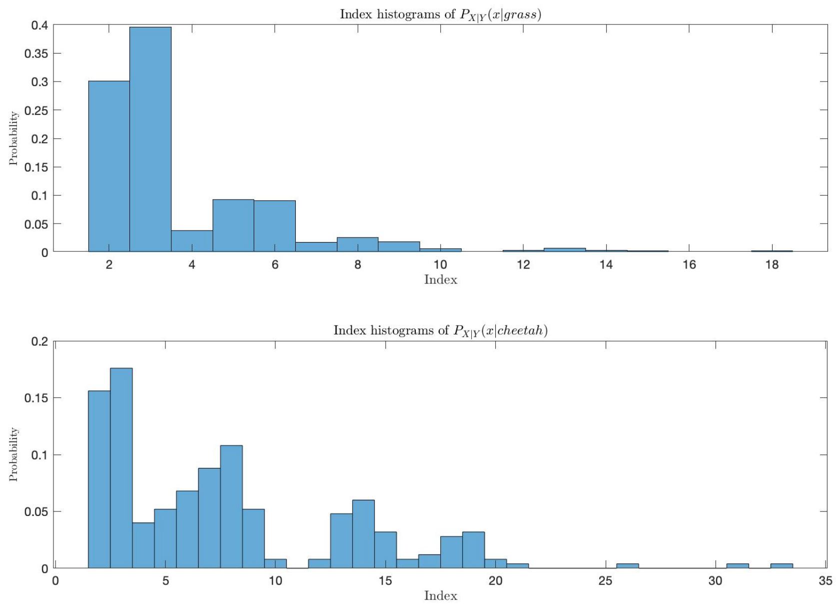


Figure 2: Index histograms

c) For each block in the image **cheetah.bmp**, compute the feature X (index of the DCT coefficient with $2^{nd}$ greatest energy). Compute the state variable Y using the minimum probability of error rule based on the probabilities obtained in a) and b). Store the state in an array A. Using the commands imagesc and colormap(gray(255)) create a picture of that array.

**Solution**:
Given a 8*8 block from the image **cheetah.bmp**, we can easily compute an array of 8*8 frequency coefficients by using funcition **dct2** on Matlab. Feature X would be index of the $2^{nd}$ greatest DCT coefficient. Given $X = x$ in one block, we can compute the $P_{Y|X}(cheetah|x)$ and $P_{Y|X}(grass|x)$ as following:

$$P_{Y|X}(cheetah|x) = \frac{P_{X|Y}(x|cheetah) * P_Y(cheetah)}{P_{X|Y}(x|cheetah) * P_Y(cheetah) + P_{X|Y}(x|grass) * P_Y(grass)} \tag{3}$$

$$P_{Y|X}(grass|x) = \frac{P_{X|Y}(x|grass) * P_Y(grass)}{P_{X|Y}(x|cheetah) * P_Y(cheetah) + P_{X|Y}(x|grass) * P_Y(grass)} \tag{4}$$

where

$P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$ are the estimation we get from training data.

$P_Y(cheetah)$ and $P_Y(grass)$ are the estimation we get from training data.

According to minimum probability of error rule, if $P_{Y|X}(cheetah|x) >= P_{Y|X}(grass|x)$, then we mask the top left corner of the 8*8 block as 1, regarding this pixel belongs to cheetah. Otherwise, we mask 0. By using a sliding window that moves by one pixel at each step, finally we get a array A containing the mask ndicates which blocks contain grass and which contain the cheetah.

d) The array A contains a mask that indicates which blocks contain grass and which contain the cheetah. Compare it with the ground truth provided in image **cheetah mask.bmp** (shown below on the right) and compute the probability of error of your algorithm.

**Solution**:
The comparision between ground truth and picture generated from array A is showed as following:

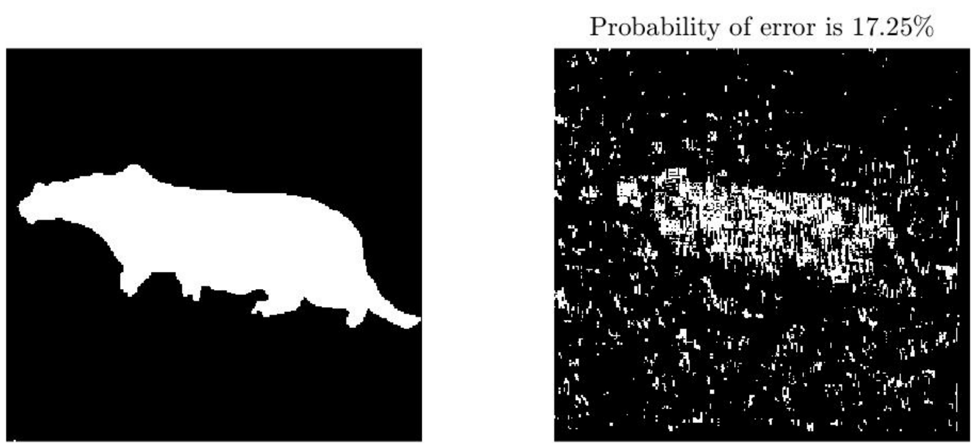


Figure 3: Comparision

The probabilities of error is 17.25%, as showed in the figure above.

# Appendix

The following is the Matlab code.

```
clear all
%%
%Training
%Read the TrainingSamplesDCT_8.mat file
load('dataset/TrainingSamplesDCT_8.mat');
%Save TrainsampleDCT_BG and TrainsampleDCT_FG in temporary value
train_BG = TrainsampleDCT_BG;
train_FG = TrainsampleDCT_FG;

%Find the second largest value in each row of matrix train_BG
[M_BG,N_BG] = max(train_BG,[],2);
train_BG(bsxfun(@eq, train_BG, M_BG)) = -1; % Set the largest value in each row to -inf
[M_BG,N_BG] = max(train_BG,[],2);

%Find the second largest value in each row of matrix train_FG
[M_FG,N_FG] = max(train_FG,[],2);
train_FG(bsxfun(@eq, train_FG, M_FG)) = -1; % Set the largest value in each row to -inF
[M_FG,N_FG] = max(train_FG,[],2);

% [M_BG,N_BG] = find(train_BG==max(train_BG(:))); % Find the largest coefficient
% train_BG(M_BG,N_BG) = -Inf; % Set the largest value as -Inf
% [M_BG,N_BG] = find(train_BG==max(train_BG(:))); % Find the second largest coefficient and its position
%
% [M_FG,N_FG] = find(train_FG==max(train_FG(:))); % Find the largest coefficient
% train_FG(M_BG,N_BG) = -Inf; % Set the largest value as -Inf
% [M_FG,N_FG] = find(train_FG==max(train_FG(:))); % Find the second largest coefficient and its position

%Plot the frequency histogram
subplot(2,1,1);
h1 = histogram(N_BG);
ylim([0, 500]);
ylabel('Frequency', 'interpreter', 'latex','FontSize', 10);
```

```
xlabel('Index', 'interpreter', 'latex');
title({['Frequency histograms of background data']},'Fontsize',12,
    'interpreter','latex');
subplot(2,1,2);
h2 = histogram(N_FG);
ylim([0, 50]);
ylabel('Frequency', 'interpreter', 'latex','FontSize', 10);
xlabel('Index', 'interpreter', 'latex');
title({['Frequency histograms of foreground data']},'Fontsize',12,
    'interpreter','latex');
%Save the statistic data
F_x_BG = zeros(1,64);
F_x_BG(min(N_BG):max(N_BG)) = h1.Values;
F_x_FG = zeros(1,64);
F_x_FG(min(N_FG):max(N_FG)) = h2.Values;
%Save the histogram figure
set(gcf,'Position',[400,100,900,600]);
saveas(gcf, ['Images/histograms1.jpg']);
close(gcf);

%Calculate the estimation of class-conditionals for two classes
    and priors probabilities
P_x_BG = F_x_BG ./ sum(F_x_BG);
P_x_FG = F_x_FG ./ sum(F_x_FG);
P_BG = size(train_BG,1) / (size(train_BG,1) + size(train_FG,1));
P_FG = size(train_FG,1) / (size(train_BG,1) + size(train_FG,1));

% %Plot the index histogram
subplot(2,1,1);
h1 = histogram(N_BG,'Normalization','pdf');
ylim([0, 0.4]);
ylabel('Probability', 'interpreter', 'latex','FontSize', 10);
xlabel('Index', 'interpreter', 'latex');
title({['Index histograms of $$P_{X|Y}(x|grass)$$']},'Fontsize'
    ,12,'interpreter','latex');
subplot(2,1,2);
h2 = histogram(N_FG,'Normalization','pdf');
ylim([0, 0.2]);
ylabel('Probability', 'interpreter', 'latex','FontSize', 10);
xlabel('Index', 'interpreter', 'latex');
title({['Index histograms of $$P_{X|Y}(x|cheetah)$$']},'Fontsize'
    ,12,'interpreter','latex');
```

```
%Save the histogram figure
set(gcf,'Position',[400,100,900,600]);
saveas(gcf, ['Images/histograms2.jpg']);
close(gcf);

%Read original image
I = imread('dataset/cheetah.bmp');
I = im2double(I);
%Define the loop numbers
loop_row = size(I,1) - 8 + 1;
loop_column = size(I,2) - 8 + 1;

mask = zeros(size(I));
position_ref = load('dataset/Zig-Zag Pattern.txt')
for i=1:1:loop_row
    for j=1:1:loop_column
        block = I(i:i+7,j:j+7);
        DCT_block = dct2(block);
        DCT_block = abs(DCT_block);
        [x,y] = find(DCT_block==max(DCT_block(:))); % Find the largest coefficient
        DCT_block(x,y) = -1; % Set the largest value as -1
        [x,y] = find(DCT_block==max(DCT_block(:))); % Find the second largest coefficient and its position
        feature = position_ref(x,y) + 1;
        %Decide the binary mask
        %Before decide the mask, we should caluate two class-conditionals
        P_FG_Decision = P_x_FG(1,feature) * P_FG / (P_x_FG(1,feature)* P_FG + P_x_BG(1,feature) * P_BG);
        P_BG_Decision = P_x_BG(1,feature) * P_BG / (P_x_FG(1,feature)* P_FG + P_x_BG(1,feature) * P_BG);
%         P_FG_Decision = P_x_FG(1,feature) / (P_x_FG(1,feature)*P_FG + P_x_BG(1,feature) * P_BG);
%         P_BG_Decision = P_x_BG(1,feature) / (P_x_FG(1,feature)*P_FG + P_x_BG(1,feature) * P_BG);
        if P_FG_Decision >= P_BG_Decision
            mask(i,j) = 1;
        end
    end
end
```

```
subplot(1,2,1)
I = imread('dataset/cheetah_mask.bmp');
I = im2double(I);
imshow(I);
subplot(1,2,2)
imshow(mask);
%Calculate the probability of error
error = length(find((mask-I)~=0)) / (size(I,1) * size(I,2));
title({['Probability of error is ',num2str(error*100,'%.2f'),'\%'
    ]},'Fontsize',12,'interpreter','latex');
%set(gcf,'Position',[900,600]);
saveas(gcf, ['Images/segmentation.jpg']);
close(gcf);
```