Name: Neil Chen October 8, 2022
PID: A59013393

ECE 271A Homework 1

1. The goal of this problem is to segment the "'cheetah"' image (shown below in the left) into its two components, cheetah (foreground) and grass (background).

   (a) Using the training data in **TrainingSamplesDCT 8.mat**, what are reasonable estimates for the prior probabilities?

   > *Solution.*
   > We can assume that the probability of being cheetah for some x_i is higher than that of being grass, that is $P_{X|Y}(x_i \,|\, cheetah) > P_{X|Y}(x_i \,|\, grass)$, and vice versa.

   (b) Using the training data in TrainingSamplesDCT 8.mat, compute and plot the index histograms $P_{X|Y}(x \,|\, cheetah)$ and $P_{X|Y}(x \,|\, grass)$.

   > *Solution.*
   > The following are my codes.
   > If we look at the histogram of training sample data sets (see Table 1), we can tell that the likelihood of being **grass** is higher than being **cheetah** for x_i for $i = 2, 3, \ldots, 6$; whereas the likelihood of being **cheetah** is higher than being **grass** for x_i for $x \geq 7$.

Matlab Code

```
% we first deal with the training data sets
load("TrainingSamplesDCT_8.mat");   % load the training data sets
M_cheetah = CreateHisto(TrainsampleDCT_FG); % the training data set of cheetah
M_grass = CreateHisto(TrainsampleDCT_BG); % the training data set of grass
[m_cheetah, ~] = size(TrainsampleDCT_FG);   % the size of the training data set of cheetah
[m_grass, ~] = size(TrainsampleDCT_BG); % the size of the training data set of cheetah
P_cheetah = m_cheetah / (m_cheetah + m_grass);  % the probability of cheetah
P_grass = m_grass / (m_cheetah + m_grass); % the probability of grass
H_cheetah = histogram(M_cheetah, "Normalization","probability");    % create a histogram of cheetah
hold on
posMax = 25;    % there is no data after x = 25
Y_cheetah = zeros(1, posMax);   % store in a vector
Y_cheetah(1, 2 : end) = H_cheetah.Values(1, 1 : posMax - 1);
P_X_cheetah = Y_cheetah * P_cheetah;    % P(x, cheetah)
H_grass = histogram(M_grass, "Normalization","probability");    % create a histogram of grass
title("Probability of Cheetah vs. Probability of Grass");
Y_grass = zeros(1, posMax); % store in a vector
Y_grass(1, 2 : 18) = H_grass.Values(1, 1 : end);
P_X_grass = Y_grass * P_grass;  % P(x, grass)
```

ProbabilityOfCheetahVsProbabilityOfGrass.jpg

Table 1: A histogram of the probability of cheetah vs. the probability of grass

(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. The following are my code.
And my result is shown at Figure 1.
My code yield an error rate of 18.01%.

Matlab Code

```
clear
clc

% we first deal with the training data sets
load("TrainingSamplesDCT_8.mat");    % load the training data sets
M_cheetah = CreateHisto(TrainsampleDCT_FG); % the training data set of cheetah
M_grass = CreateHisto(TrainsampleDCT_BG); % the training data set of grass
[m_cheetah, ~] = size(TrainsampleDCT_FG);    % the size of the training data set of cheetah
[m_grass, ~] = size(TrainsampleDCT_BG); % the size of the training data set of cheetah
P_cheetah = m_cheetah / (m_cheetah + m_grass);   % the probability of cheetah
P_grass = m_grass / (m_cheetah + m_grass); % the probability of grass
H_cheetah = histogram(M_cheetah, "Normalization","probability");     % create a histogram of cheetah
hold on
posMax = 25;     % there is no data after x = 25
Y_cheetah = zeros(1, posMax);    % store in a vector
Y_cheetah(1, 2 : end) = H_cheetah.Values(1, 1 : posMax - 1);
P_X_cheetah = Y_cheetah * P_cheetah;     % P(x, cheetah)
H_grass = histogram(M_grass, "Normalization","probability");     % create a histogram of grass
title("Probability of Cheetah vs. Probability of Grass");
Y_grass = zeros(1, posMax); % store in a vector
Y_grass(1, 2 : 18) = H_grass.Values(1, 1 : end);
P_X_grass = Y_grass * P_grass;  % P(x, grass)


maskTable = zeros(1, posMax);

for i = 1 : posMax
    if P_X_cheetah(i) > P_X_grass(i)
```

```
            maskTable(i) = 1;
        else
            maskTable(i) = 0;
        end
end

img = imread("cheetah.bmp");
img = im2double(img);
[m, n] = size(img); % row = 255, column = 270
img_answer = imread("cheetah_mask.bmp");
img_answer = im2gray(img_answer);

% padding the image
I = zeros(m + 7, n + 7);
I(1:m, 1:n) = img;

% the zigzag map
ZigZagM = [0    1    5    6   14   15   27   28;
           2    4    7   13   16   26   29   42;
           3    8   12   17   25   30   41   43;
           9   11   18   24   31   40   44   53;
          10   19   23   32   39   45   52   54;
          20   22   33   38   46   51   55   60;
          21   34   37   47   50   56   59   61;
          35   36   48   49   57   58   62   63];
ZigZagV = ZigZagM(:);

% use sliding window method to perform dct2
X = zeros(m, n);

for i = 1 : m
    for j = 1 : n
        Block = I(i : i + 7, j : j + 7);
        Block_DCT = dct2(Block, 8, 8);
        V = abs(Block_DCT(:));
        V_descend = sort(V, "descend");
        val = V_descend(2);
        index = find(V == val);
        x = ZigZagV(index);
```

```
            X(i, j) = x(1);
        end
end
H_X = histogram(X, "Normalization","probability");

mask = zeros(m, n);
for i = 1 : m
    for j = 1 : n
        pos = X(i, j);
        if 1 <= pos && pos <= 25
            mask(i, j) = maskTable(pos);
        end
    end
end

img_mask = mat2gray(mask);

% compare the result with the answer
subplot(1, 2, 1), imshow(img_answer), title('Idea Mask');
subplot(1, 2, 2), imshow(img_mask), title('My Result');

% change the answer to 0s and 1s
for i = 1 : m
    for j = 1 : n
        bit = img_answer(i, j);
        if bit == 255
            img_answer(i, j) = 1;
        end
    end
end

error = 0;
for i = 1 : m
    for j = 1 : n
        a = img_answer(i, j);
        b = img_mask(i, j);
        if a ~= b
            error = error + 1;
        end
```

```
        end
end
e = error / (m * n)

% analysize training data sets function
function H = CreateHisto(M)
        [m, ~] = size(M);
        H = zeros(m, 1);
        for i = 1 : m
                v = abs(M(i, :));
                v_descend = sort(v, "descend");
                val = v_descend(2);
                index = find(v == val);
                H(i) = index;
        end
end
```

Homework1-Result.jpg

Figure 1: The ideal result compares with mine