Homework Assignment 1

Quan Luo
Statistical Learning I - Fall 2021

October 12, 2021

Problem Quiz. 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).

To formulate this as a pattern recognition problem, we need to decide on an observation space. Here we will be using the space of 8×8 image blocks, i.e. we view each image as a collection of 8×8 blocks. For each block we compute the discrete cosine transform (function dct2 on MATLAB) and obtain an array of 8×8 frequency coefficients. We do this because the cheetah and the grass have different textures, with different frequency decompositions and the two classes should be better separated in the frequency domain. We then convert each 8×8 array into a 64 dimensional vector because it is easier to work with vectors than with arrays. The file Zig-Zag_Pattern.txt contains the position (in the 1D vector) of each coefficient in the 8×8 array. The file TrainingSamplesDCT_8.mat contains a training set of vectors obtained from a similar image (stored as a matrix, each row is a training vector) for each of the classes. There are two matrices, TrainsampleDCT_BG and TrainsampleDCT_FG for foreground and background samples respectively.

To make the task of estimating the class conditional densities easier, we are going to reduce each vector to a scalar. For this, for each vector, we compute the index (position within the vector) of the coefficient that has the 2nd largest energy value (absolute value). This is our observation or feature X. (The reason we do not use the coefficient with the largest energy is that it is always the so-called "DC" coefficient, which contains the mean of the block). By building an histogram of these indexes we obtain the class-conditionals for the two classes $P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$. The priors $P_Y(cheetah)$ and $P_Y(grass)$ should also be estimated from the training set.

(a) using the training data in *TraningSamplesDCT_8.mat*, what are reasonable estimates for the prior probabilities?

Sol. From the training data sample, we have 1053 background points and 250 foreground points. Thus the prior probabilities can be calculated as follow:

$$P_Y(cheetah) = \frac{250}{(1053 + 250)} = 0.1919$$

$$P_Y(grass) = \frac{1053}{(1053 + 250)} = 0.8081$$

□

(b) using the training data in *TraningSamplesDCT_8.mat*, compute and plot the index histograms $P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$.

Sol. Codes and result histogram graphs are as follow:

```
load('TrainingSamplesDCT_8.mat', 'TrainsampleDCT_FG')
load('TrainingSamplesDCT_8.mat', 'TrainsampleDCT_BG')

% FIRST CALCULATING P(X|CHEETAH)
bin_size = 1;[rc, cc] = size(TrainsampleDCT_FG);
histogram_cheetah = []; prob_cheetah = zeros([cc, 1]);
for i = 1:rc
    feature = TrainsampleDCT_FG(i, :);
    [feat, pos] = sort(feature, 'descend');
    histogram_cheetah = [histogram_cheetah pos(2)];
    prob_cheetah(ceil(pos(2) / bin_size)) = prob_cheetah(ceil(pos(2) / bin_size)
        ) + 1;
end
figure(1);histogram(histogram_cheetah, cc, 'Normalization', 'probability');
prob_cheetah = prob_cheetah / rc;

% FIRST CALCULATING P(X|GRASS)
[rg, cg] = size(TrainsampleDCT_BG);
histogram_grass = []; prob_grass = zeros([cg, 1]);
for i = 1:rg
    feature = TrainsampleDCT_BG(i, :);
    [feat, pos] = sort(feature, 'descend');
    histogram_grass = [histogram_grass pos(2)];
    prob_grass(ceil(pos(2) / bin_size)) = prob_grass(ceil(pos(2) / bin_size)) +
        1;
end
figure(2);histogram(histogram_grass, cc, 'Normalization', 'probability');
prob_grass = prob_grass / rg;
```

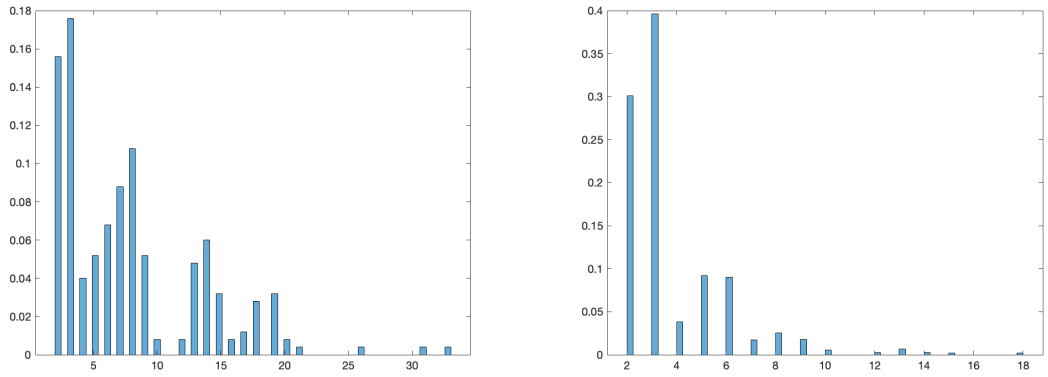

Figure 1: Left: $P_{X|Y}(x|cheetah)$, Right: $P_{X|Y}(x|grass)$

That's the most simple binning method, which is just use the 64 dimension as the bins. However, we can calculate the best binning method according to some criterion. Here we find the bin size using the following:

$$bin\_size = \underset{all\ bins}{argmax}(P_{X|Y}(x|cheetah) - P_{X|Y}(x|grass))^2$$

which aims at maximizing the difference between all the FG and BG. Then our classifier can better distinguish these two kinds. It's just an intuitive guess and I would like to do a experiment on it. We wrote codes as follow to find the best bin size.

```
cum_p_grass = cumsum(prob_grass);
cum_p_cheetah = cumsum(prob_cheetah);

error = sum((prob_grass - prob_cheetah).^2);
best_bin_size = 1;
% I STANDS FOR BIN SIZE
for i = 2:64
    pg = [];
    pc = [];
    for j = 1:2:64
        if i + j <= 64
            pg = [pg (cum_p_grass(i + j) - cum_p_grass(i))];
            pc = [pc (cum_p_cheetah(i + j) - cum_p_cheetah(i))];
        else
            pg = [pg (cum_p_grass(64) - cum_p_grass(i))];
            pc = [pc (cum_p_cheetah(64) - cum_p_cheetah(i))];
        end
    end
    e = sum((pg - pc).^2) / i;
    if e > error
        error = e;
        best_bin_size = i;
    end
end

best_bin_size
```

According to this program, the best bin size is 3. □

(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.

Sol. Code and the result segmented picture is as follow.

```
% PREPROCESS
```

```
cheetah_original = imread('cheetah.bmp');
[r, c] = size(cheetah_original);
cheetah_original = im2double(cheetah_original);
cheetah = padarray(cheetah_original, [4, 4], 'replicate', 'both');

[rc, cc] = size(TrainsampleDCT_FG);
[rg, cg] = size(TrainsampleDCT_BG);
Pcheetah = rc / (rc + rg);
Pgrass = rg / (rc + rg);

res = [];
for i = 5:r+4
    tmp = [];
    for j = 5:c+4
        area = cheetah([i - 3:i + 4], [j - 3:j + 4]);
        dct_res = abs(dct2(area));
        feature = zigzag(dct_res);
        [feat, pos] = sort(feature, 'descend');

        % Now Pos(2) is the result, calculate its probability
        p_fg = prob_cheetah(ceil(pos(2) / bin_size)) * Pcheetah;
        p_bg = prob_grass(ceil(pos(2) / bin_size)) * Pgrass;
        if p_fg > p_bg
            tmp = [tmp 255];
        else
            tmp = [tmp 0];
        end
    end
    res = [res; tmp];
end
img = imagesc(res);

mask = imread("cheetah_mask.bmp");
error = 0; total = r * c;
for i = 1:r
    for j = 1:c
        mask(i, j);
        if mask(i, j) ~= res(i, j)
            error = error + 1;
        end
    end
end

error_rate = error / total
```

The *zigzag* function in code maps from index value to zigzag value. Codes are as follow:

```
function output = zigzag(in)
    A = load('Zig-Zag Pattern.txt');
    output = [1:64];
    for i = 1:8
```

```
        for j = 1:8
            output(A(i, j) + 1) = in(i, j);
        end
    end
end
```

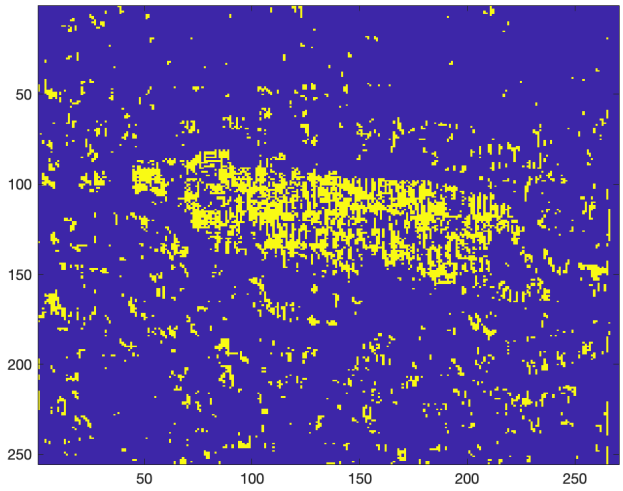

Figure 2: Segmented Result

Next since we've found the best bin size according to our calculation, we also got a segmented picture (just change variable bin_size for code, do not list codes again here) from the best bin size as follow:

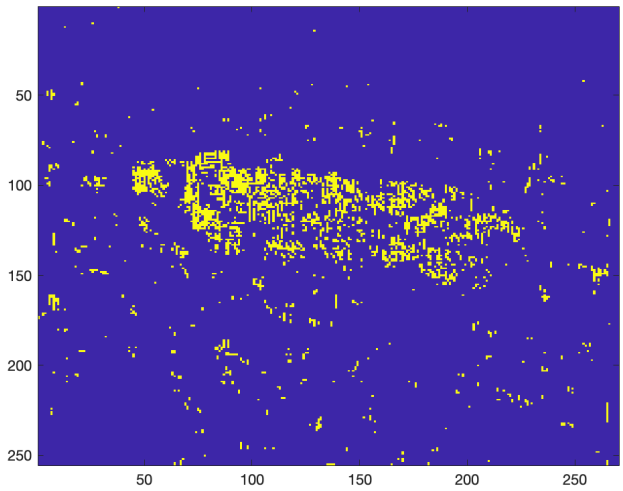

Figure 3: 3 Bin Size Segmented Result

□

(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.

Proof. The code is as the same as in (c). We output the error rate and in 64 binning histogram and probability, it's error rate is 17.35%.

```
>> segement_cheetah

error_rate =

    0.1735
```

Figure 4: Error Rate Output in MATLAB

For the best binning size we found, it didn't improve very significantly. The error rate decreased to 16.62% as follow.

```
>> segement_cheetah

error_rate =

    0.1662
```

Figure 5: Error Rate Output 2 in MATLAB

□