ECE 271A: Quiz #1

Due on October 16, 2023 at 11:59pm

Professor Vasconcelos

Ray Tsai

A16848188

Part A

Using the training data in `TrainingSamplesDCT_8.mat`, what are reasonable estimates for the prior probabilities?

Solution

We assume that `TrainingSampleDCT_FG` and `TrainingSampleDCT_BG` represents a set of complete images collectively. Then, given that there are 250 rows in `TrainingSampleDCT_FG` and 1053 rows in `TrainingSampleDCT_BG`, we use the ratio of the size of samples to give an estimate for the prior probabilities, namely

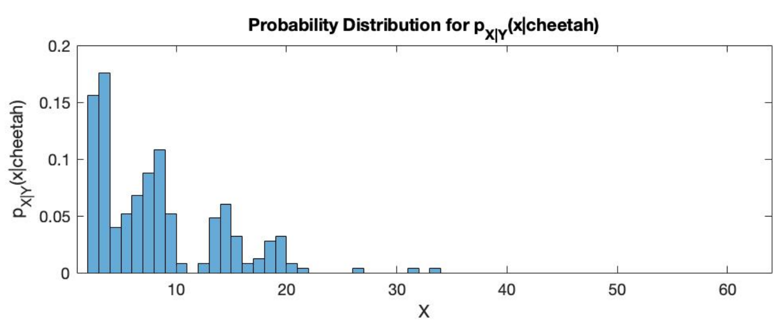
$$\mathbb{P}_Y(cheetah) = \frac{250}{1053 + 250} \approx 19\%,$$
$$\mathbb{P}_Y(grass) = \frac{1053}{1053 + 250} \approx 81\%.$$

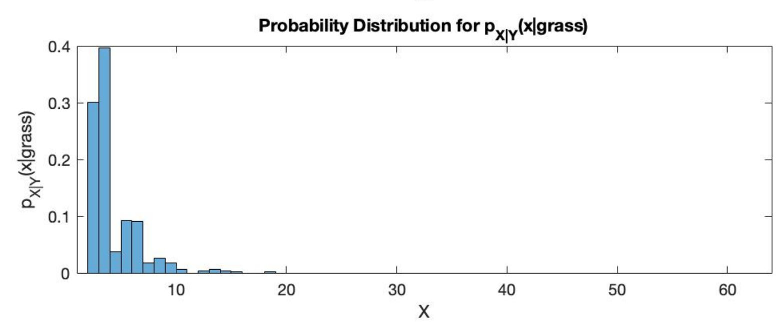
Part B

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

We collect feature X, the position of the coefficient of the second largest magnitude, from each row vector and normalize it to obtained the index histogram for $P_{X|Y}(x|cheetah)$ and $P_{X|Y}(x|grass)$:





Part C

For each block in the image `cheetah.bmp`, compute the feature X (index of the DCT coefficient with 2nd greatest energy). Compute the state variable Y using the minimum probability of error rule based on the probabilities obtained in part 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

Let $g(x)$ be our decision function. Assume our loss function $L[g(x), y] = \begin{cases} 1, & g(x) \neq y \\ 0, & g(x) = y \end{cases}$. By the maximum a-posteriori probability rule, the optimal decision function

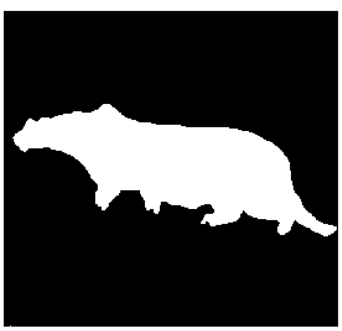
$$\begin{aligned} g^*(x) &= \arg\max_i \mathbb{P}_{Y|X}(i|x) \\ &= \arg\max_i \mathbb{P}_{X|Y}(x|i)\mathbb{P}_Y(i) \\ &= \begin{cases} cheetah, & \frac{\mathbb{P}_{X|Y}(grass|x)}{\mathbb{P}_{X|Y}(cheetah|x)} < \frac{\mathbb{P}_Y(cheetah)}{\mathbb{P}_Y(grass)} \\ grass, & \text{otherwise} \end{cases} \\ &= \begin{cases} cheetah, & \frac{\mathbb{P}_{X|Y}(grass|x)}{\mathbb{P}_{X|Y}(cheetah|x)} < 0.24 \\ grass, & \text{otherwise,} \end{cases} \end{aligned}$$

for the 0-1 loss function. We then use function $g*(x)$ to assign a state to each pixel and obtian our pridiction mask array A.

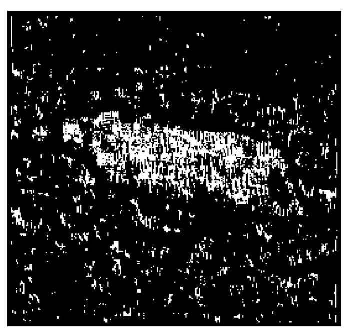
Part D

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



Truth



Prediction

Error Rate: 17.22%

MATLAB Code

```
load('../dataset/TrainingSamplesDCT_8.mat');
zigzag = load('../dataset/Zig-Zag Pattern.txt');
cheetah = imread('../dataset/cheetah.bmp');
cheetah_mask = imread('../dataset/cheetah_mask.bmp');
target = im2double(cheetah);
mask = im2double(cheetah_mask);

training_BG = TrainsampleDCT_BG;
training_FG = TrainsampleDCT_FG;
zigzag = zigzag + 1;

[row_BG, col_BG] = size(training_BG);
[row_FG, col_FG] = size(training_FG);
[row_TG, col_TG] = size(target);

padded_target = zeros(row_TG + 7, col_TG + 7);
padded_target(5:4 + row_TG, 5:4 + col_TG) = target;

% pick cheetah if (p(x | grass) / p(x | cheetah)) < threshold
prior_BG = row_BG / (row_BG + row_FG);
prior_FG = row_FG / (row_BG + row_FG);
threshold = prior_FG / prior_BG;

feature_BG = zeros(64, 1);
feature_FG = zeros(64, 1);

for r = 1:1:row_BG
    maxVal = max(training_BG(r));
    secVal = 0;
    secPos = 0;
    for c = 1:1:col_BG
        if (training_BG(r, c) < maxVal && training_BG(r, c) > secVal)
            secVal = training_BG(r, c);
            secPos = c;
        end
    end
    feature_BG(secPos) = feature_BG(secPos) + 1;
end

for r = 1:1:row_FG
    maxVal = max(training_FG(r));
    secVal = 0;
    secPos = 0;
    for c = 1:1:col_FG
        if (training_FG(r, c) < maxVal && training_FG(r, c) > secVal)
            secVal = training_FG(r, c);
            secPos = c;
```

```
        end
    end
    feature_FG(secPos) = feature_FG(secPos) + 1;
end

cprob_BG = feature_BG / sum(feature_BG);
cprob_FG = feature_FG / sum(feature_FG);
A = zeros(row_TG, col_TG);

for r = 1:row_TG
    for c = 1:col_TG
        block = padded_target(r:r + 7, c:c + 7);
        dctBlock = abs(dct2(block));
        maxVal = max(max(dctBlock));
        secVal = 0;
        x = 0;
        for i = 1:8
            for j = 1:8
                if dctBlock(i, j) < maxVal && dctBlock(i, j) > secVal
                    secVal = dctBlock(i, j);
                    x = zigzag(i, j);
                end
            end
        end
        A(r, c) = int8(cprob_BG(x)/cprob_FG(x) <= threshold);
    end
end

subplot(1, 2, 1);
imagesc(mask);
axis off
colormap(gray(255));
axis equal tight;

subplot(1, 2, 2);
imagesc(A);
axis off
colormap(gray(255));
axis equal tight;

error = 0;
for r = 1:row_TG
    for c = 1:col_TG
        if (A(r, c) ~= mask(r, c))
            error = error + 1;
        end
    end
end
error_rate = error / (row_TG * col_TG);
```