ECE 271A - Homework assignment

Taruj Goyal

October 17, 2019

1 Problem Formulation

- **Note:** Homework implemented in MATLAB 2019b.
- **Goal:** Segment the image into the cheetah (foreground) and grass (background).

Figure 1: Cheetah in the Savannah

- **Problem Setup as Pattern Recognition:**
  - **Observation Space:** Images as a collection of 8×8 blocks.
  - **Feature Computation:**
    - For each block compute the 2D Discrete Cosine Transform.
    - Reorder the 8×8 frequency decomposition into Zig-Zag pattern.

* Resize the 8×8 array into 64D vector.
* Compute the index of the coefficient with 2nd largest energy value.
* *This is done as the cheetah and the grass have different textures and using their frequency decomposition they're better separated in the frequency domain.*

- **Probability Computation: Part (a) and (b)**
  - **Class Probability:** $\Pr(cheetah)$ & $\Pr(grass)$
    * These values are estimated from the TrainingSamplesDCT_8.mat dataset.
    * The training dataset is taken from an image similar to the test dataset.
    * This means that the relative distribution of the grass versus the cheetah is based on their relative presence in the training date.
    * Hence, the relative number of values for each class is a measure of the chance of an image block belonging to that class.
    * This can be computed using the dimensions TrainingSamplesCD_8.mat
    * The number of training examples for the Cheetah and Grass are 250 and 1053, respectively.
    * **Based on these dimensions** $\Pr(cheetah) = \mathbf{0.1919}$ **&** $\Pr(grass) = \mathbf{0.8081}$**.**
  - **Class Conditional Densities:** $\Pr(x \mid cheetah)$ & $\Pr(x \mid grass)$
    * *For each block present in the image:*
    * The index of the second highest DCT component is used as the scalar feature.
    * The indexes for all the training set vectors are used to compute the histogram.
    * The histogram is normalized to obtain conditional distribution.
    * This done directly using **Histogram** function in MATLAB.

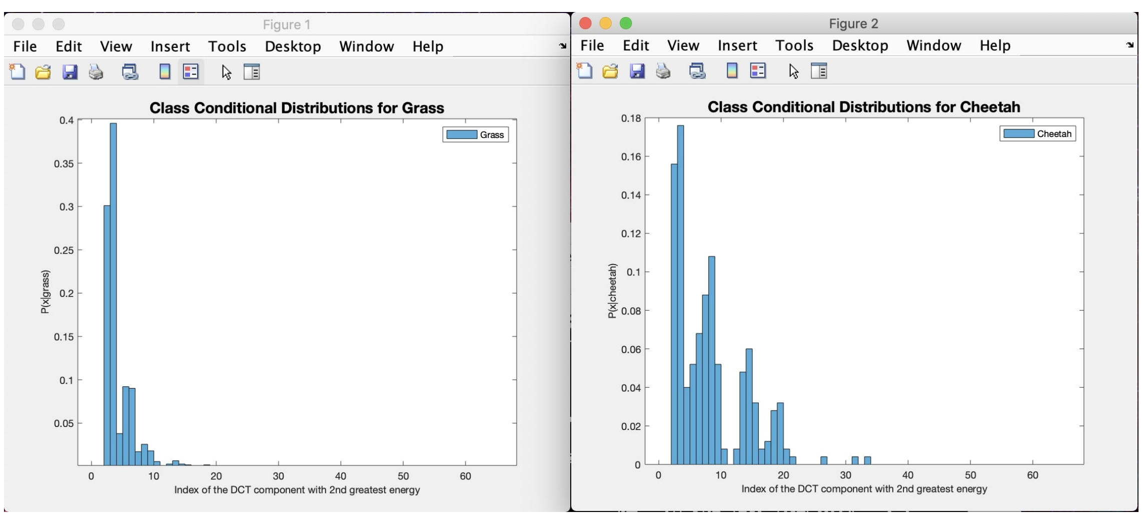

Figure 2: Class Conditional Densities: $\Pr(x \mid y)$

- **Assumptions for Inferencing:**
  - The input image is zeropadded to obtain a segmentation map with same dimensions as input image, i.e., 255×270.
  - The image is traversed in the form of a 8×8 sliding window.
  - Feature x (index of the DCT coefficient with 2nd greatest energy) computed after ensuring that the coefficients are in the zig-zag order.
  - Each pixel prediction is based on the feature computation with the 8×8 starting with the being the Top-Left pixel of the window.
  - For each pixel then the problem boils down to a Binary Classification problem.
  - Assuming there is no loss associated with the correct decision and the misclassification loss is same for both classes.
- **Posterior Probability:** Part (c) - $\Pr(cheetah \mid x)$ & $\Pr(grass \mid x)$
  - For each 8×8 block in the **cheetah.bmp** we compute the feature x.
  - $\Pr(cheetah \mid x) = \frac{\Pr(x \mid cheetah) \Pr(cheetah)}{\Pr(x)}$
  - $\Pr(grass \mid x) = \frac{\Pr(x \mid grass) \Pr(grass)}{\Pr(x)}$
  - We compute the ratio $\frac{\Pr(cheetah \mid x)}{\Pr(grass \mid x)} = \frac{\Pr(x \mid cheetah) \Pr(cheetah)}{\Pr(x \mid grass) \Pr(grass)}$
  - If the ratio is greater than the threshold, i.e., 1 then we classify it as a the *Cheetah* (foreground) else we classify as it *Grass* (background).
  - Basically for each feature x we look up the value in the class conditional distribution and multiply it with the prior and compare with the threshold.

2 Results

- **Computing the Probability of Error**
  - By comparing the pixel-wise mismatches between the ground truth and the generated segmentation map the error is computed.
  - Average probability of error is $R^* = \int P_{X,Y}(x, y \neq g^*(x))dx$.
  - Let cheetah = 1 and grass = 0,
  - The above equation boils down to $P_{error} = P(y \neq g(x))$
  - $P_{error} = P(y = 1)P(g(x) = 0|y = 1) + P(y = 0)P(g(x) = 1|y = 0)$.
  - This is basically the number of pixels mismatching between the predicted output and ground truth divided by the total number of pixels.
  - **The probability of error is 0.1726**

Figure 3: Resulting Segmentation Map

Figure 4: Ground Truth Segmentation Map

3 Appendix

Listing 1: MATLAB Solution Script

```matlab
%% Clear workspace and close windows

clc
clear
close all

%% Load training dataset

    DCT = load('TrainingSamplesDCT_8.mat');
    Train_DCT_FG = DCT.TrainsampleDCT_FG;
    Train_DCT_BG = DCT.TrainsampleDCT_BG;

%% The class priors for FG and BG are computed
%  Section A

    % Assuming prior distribution is (# of samples of A)/total training ...
        samples
    TotalNumberOfTraining_samples = size(Train_DCT_FG,1) + ...
        size(Train_DCT_BG,1);
    FG_prior = size(Train_DCT_FG,1)/TotalNumberOfTraining_samples
    BG_prior = size(Train_DCT_BG,1)/TotalNumberOfTraining_samples

%% Training data used for computing and plotting index histograms.
%  P(x|cheetah) and P(x|grass) are the class conditionals for FG and BG.
%  The feature is the index of DCT component with second largest magnitude.
%  The feature is the index of the DCT component with 2nd greatest energy.
    % Feature computation for BG sections
        X_BG = zeros(size(Train_DCT_BG,1),1);
        for idx = 1:size(Train_DCT_BG,1)
            X_BG(idx) = Index2ndLargest(Train_DCT_BG(idx,:));
        end

    % Feature computation for FG sections
        X_FG = zeros(size(Train_DCT_FG,1),1);
        for idx = 1:size(Train_DCT_FG,1)
            X_FG(idx) = Index2ndLargest(Train_DCT_FG(idx,:));
        end

    % Histogram plotting for both BG and FG
    % The bins for the histograms
        edges = 1:65;
        fontSize = 10;

        figure(1)

        h_BG = histogram(X_BG,'BinEdges',edges,'normalization', ...
            'pdf','DisplayName','BG');
        BG_CCD = histcounts(X_BG,'BinEdges',edges, 'Normalization', ...
            'probability');

        title('Class Conditional Distributions for Grass', 'FontSize', ...
            1.5*fontSize);
```

```
        xlabel('Index of the DCT component with 2nd greatest energy', ...
            'FontSize', fontSize);
        ylabel('P(x|grass)', 'FontSize', fontSize);
        legend('Grass')

        figure(2)

        h_FG = histogram(X_FG,'BinEdges',edges,'normalization', ...
            'pdf','DisplayName','FG');
        FG_CCD = histcounts(X_FG,'BinEdges',edges, 'Normalization', ...
            'probability');

        title('Class Conditional Distributions for Cheetah', ...
            'FontSize', 1.5*fontSize);
        xlabel('Index of the DCT component with 2nd greatest energy', ...
            'FontSize', fontSize);
        ylabel('P(x|cheetah)', 'FontSize', fontSize);
        legend('Cheetah')

%% For each block in the image cheetah.bmp, compute the feature X and ...
    state variable Y.
%  For Y use the minimum probability of error rule based on the ...
    distributions obtained above.
%  Store the state in an array A and then convert to binary image using ...
    imagesc and colormap(gray(255))

    block_size = 8;

    % Read the ZigZag pattern and convert to array and index from 1
        ZigZagPattern = table2array(readtable('Zig-Zag Pattern.txt'))+1;

    % Read image and convert to double
        image = im2double(imread('cheetah.bmp'));

    % Reading image dimensions
        [height,width] = size(image);
        [h_8,w_8] = deal(8*(ceil(height/8)+1),8*(ceil(width/8)+1));

    % Zeropad the image and convert to 8x8
        zeropad = zeros(h_8,w_8);
        zeropad(1:height,1:width) = image;

    % Create a blank array X
        X = zeros(height,width);
        dctZigZag = zeros(1,block_size*block_size);

    % Iterating through the image in 8x8 blocks through a sliding window
    for h = 1:height
        for w = 1:width

            dctBlock = dct2(zeropad(h:h+block_size-1,w:w+block_size-1));

            % Rearranging the DCT Components in ZigZag Patterned Vector
            for i = 1:block_size
```

```
                for j = 1:block_size
                    dctZigZag(ZigZagPattern(i,j)) = dctBlock(i,j);
                end
            end

            % Computing the feature for the block
            X(h,w) = Index2ndLargest(dctZigZag);

        end
    end

    % Computing the Posteriori distributions for generating predictions
    BG_posteriori = BG_CCD(X)*BG_prior;
    FG_posteriori = FG_CCD(X)*FG_prior;
    A = FG_posteriori > BG_posteriori;

    figure(3)
    imagesc(A);
    colormap(gray(255));
    imwrite(A, 'result.bmp');
    title('Predicted Segmentation based on Bayesian Decision Theory', ...
        'FontSize', 1.5*fontSize);


%% Compare the ground truth in image cheetah_mask.bmp and compute the ...
    probability of error

    % Read the ground truth image
        ground_truth = im2double(imread('cheetah_mask.bmp'));

    % Probability of error for Cheetah pixels misclassified as Grass
        probability_error_cheetah = sum(ground_truth & ¬...
            A,'all')/sum(ground_truth,'all');
    % Probability of error for Grass pixels misclassified as Cheetah
        probability_error_grass = sum(¬ground_truth & ...
            A,'all')/sum(¬ground_truth,'all');

        probability_error = (FG_prior*probability_error_cheetah) + ...
            (BG_prior*probability_error_grass)

%% UTILITY FUNCTIONS
    % 1. Index2ndLargest
    % Find Index of the second coefficient with second largest magnitude
    % Much faster than sorting and finding value at second position

    function [ind2] = Index2ndLargest(FeatureVector) % i is the ...
        x-largest value
        absFeatureVector = abs(FeatureVector);
        [¬,ind1] = max(absFeatureVector);
        absFeatureVector(ind1) = -Inf;
        [¬,ind2] = max(absFeatureVector);
    end
```