ECE 271A - Homework assignment

Taruj Goyal

December 7, 2019

SOLUTION

Part a)

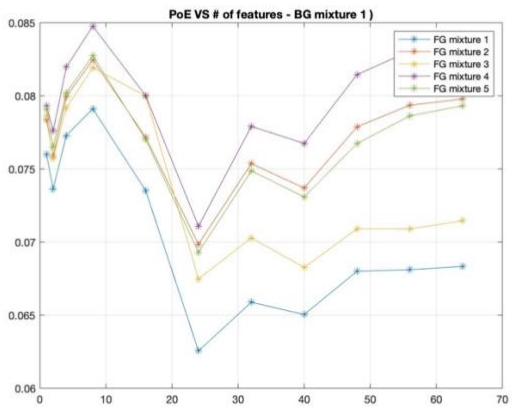
For each class, we learn five mixtures of C = 8 components, using a random initialization.

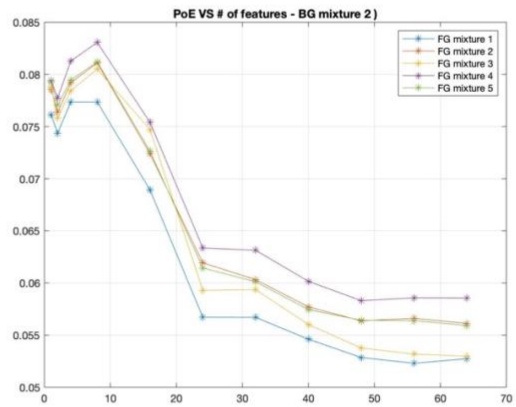
Plots for the probability of error vs. dimension for each of the 25 classifiers obtained with all possible mixture pairs.

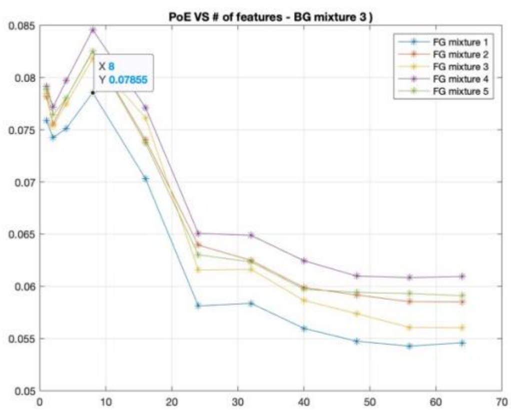
Comment the dependence of the probability of error on the initialization.
Based on the above plots we notice that there is a significant variation on all the plots.
The only difference is the initialization for all of them. Since we give the parameters random values initially, we observe that each time the algorithm has a different set of initial parameters for the each of the Gaussian mixture component. Therefore, we can state that the probability of error is highly dependent on the initial values of all the parameters.

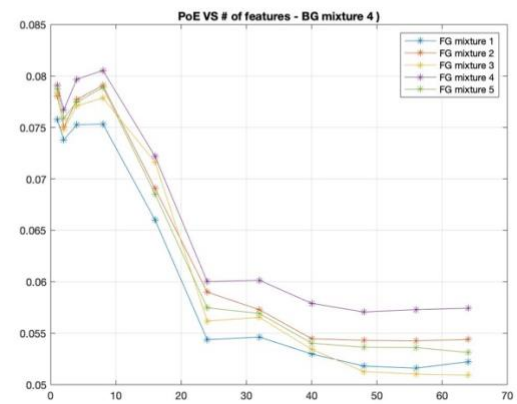
The reason for this is fundamental to the Expectation Maximization algorithm. This algorithm is basically two continuous steps taken. The first step is the E step where we build a lower bound for the Likelihood function which is then maximized in the M step.
The E-step will give different lower bounds depending on how we initialize the values.
Therefore, based on the different values we get a different probability of error for each value.
Furthermore, we can also notice the fact that for Gaussians with smaller dimensionality the Probability of error is roughly similar. But as we move on to more complex (higher dimensionality) of gaussians there is a change in the error probability. The likelihood function in this scenario may have a chance of converging in local optimum as the surface for the likelihood function in higher dimensional spaces would be much more complex.

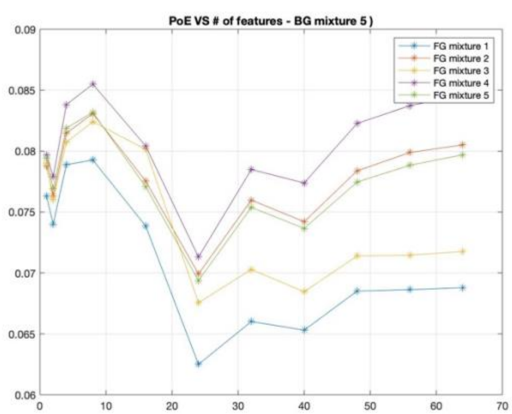
We also observe that for lower number of dimensions, the PoE is in general very similar and becomes more dissimilar as the number of dimensions increase. This can be reasoned out as the fact that the likelihood has many more maxima and minima as it belongs to such a high dimensional space and EM can get stuck on local optima instead of a global one.


PoE VS # of features - BG mixture 1)
FG mixture 1
FG mixture 2
FG mixture 3
FG mixture 4
FG mixture 5


PoE VS # of features - BG mixture 2)
FG mixture 1
FG mixture 2
FG mixture 3
FG mixture 4
FG mixture 5

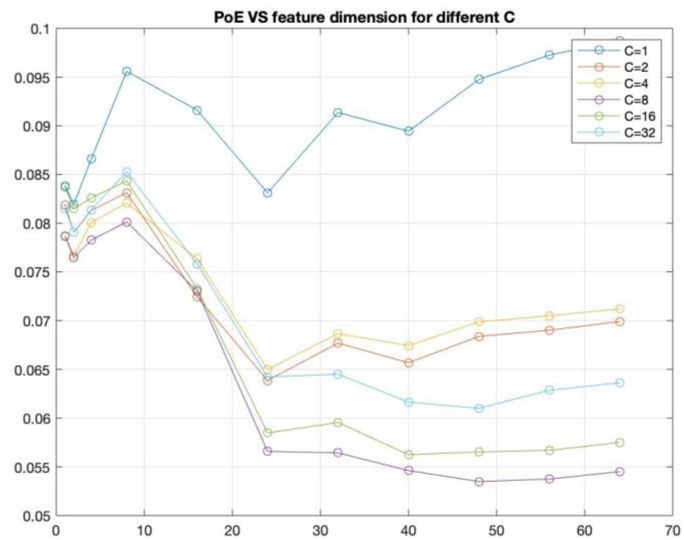

PoE VS # of features - BG mixture 3)
X 8
Y 0.07855
FG mixture 1
FG mixture 2
FG mixture 3
FG mixture 4
FG mixture 5


PoE VS # of features - BG mixture 4)
FG mixture 1
FG mixture 2
FG mixture 3
FG mixture 4
FG mixture 5


PoE VS # of features - BG mixture 5)
FG mixture 1
FG mixture 2
FG mixture 3
FG mixture 4
FG mixture 5

Part b)

For each class, we learn a range of mixtures with C ∈ {1, 2, 4, 8, 16, 32}.
Plotting the probability of error for different dimensions with varying number of mixture components.



What is the effect of the number of mixture components on the probability of error?

In this scenario we notice that there is a varying effect on the number of mixture components on the probability of error.

The probability of error generally tends to decrease as we increase the number of clusters. But after a certain point we notice that when the number of components tends to very high, there seems to be some kind of overfitting for the Expectation maximization algorithm. This leads to a much higher Probability of Error. But if the number of clusters is very less, there is an issue with the Expectation maximization algorithm. Since the algorithm is not as expressive in this case and cannot contain all the changes of the data.

Also, in the above case the Mixture with just one component has a very high error as it cannot explain the in the data.
For the higher dimensions there could be less data available and this could possibly be attributed to the curse of dimensionality.

APPENDIX

```
%% Clear workspace and close windows

    clc
    clear
    close all

%% Feature Computation for 8x8 block in the image cheetah.bmp, compute the
feature X.

    block_size = 8;
    ZigZagPattern = table2array(readtable('Zig-Zag Pattern.txt'))+1;

    image = im2double(imread('cheetah.bmp'));
    [height,width] = size(image);
    image = padarray(image,[8,8],'symmetric');
    ground_truth = im2double(imread('cheetah_mask.bmp'));
    dctZigZag = zeros(height,width,block_size*block_size);

    % Iterating through the image in 8x8 blocks through a sliding window
    % Compute feature vectors for test image
    for h = 1:height
        for w = 1:width
            dctBlock = dct2(image(h+4:h+11,w+4:w+11));
            % Rearranging the DCT Components in ZigZag Patterned Vector
            for i = 1:block_size
                for j = 1:block_size
                    dctZigZag(h,w,ZigZagPattern(i,j)) = dctBlock(i,j);
                end
            end
        end
    end

    % Load training dataset
    DCT = load('TrainingSamplesDCT_8.mat');
    Train_FG = DCT.TrainsampleDCT_FG;
    Train_BG = DCT.TrainsampleDCT_BG;

    % Assuming prior distribution is (# of samples of A)/total training
samples
    N_BG = size(Train_BG,1);
    N_FG = size(Train_FG,1);

    % Based on dataset size
    prior_BG = N_BG/(N_BG+N_FG);
    prior_FG = N_FG/(N_BG+N_FG);

% % %% Generating the output image for the two features
    C = 8;
    epsilon = 1e-8;
    n_iter = 100;

    pi_fg = cell(5,1);
```

```
    pi_bg = cell(5,1);
    mu_fg= cell(5,1);
    mu_bg = cell(5,1);
    cov_fg = cell(5,1);
    cov_bg= cell(5,1);

    % Create 5 different mixtures based on different initializations
    for i =1:5
        [pi_fg{i},mu_fg{i},cov_fg{i}] =
compute_param(C,epsilon,n_iter,Train_FG);
        [pi_bg{i},mu_bg{i},cov_bg{i}] =
compute_param(C,epsilon,n_iter,Train_BG);
    end

    dim = [1,2,4,8,16,24,32,40,48,56,64];
    [~,dimlen] = size(dim);
    error = zeros(25,dimlen);

    for i = 1:5 %% BG
        figure
        for j = 1:5 %% FG
            for k = 1:dimlen
                error(5*(i-1)+j,k) =
prediction(dctZigZag,height,width,dim(k),C,pi_bg{i},pi_fg{j},mu_bg{i},mu_fg{j
},cov_bg{i},cov_fg{j},prior_BG,prior_FG,ground_truth);
            end
            plot(dim,error(5*(i-1)+j,:),'*-')
            hold on
        end
        grid on
        legend('FG mixture 1','FG mixture 2','FG mixture 3','FG mixture
4','FG mixture 5')
        title(['PoE vs # of features - BG mixture ',num2str(i)])
    end


%% Generating the output image for the two features
    epsilon = 1e-8;
    n_iter = 100;

    C = [1,2,4,8,16,32];
    [~,complen] = size(C);
    dim = [1,2,4,8,16,24,32,40,48,56,64];
    [~,dimlen] = size(dim);
    error_b = zeros(complen,dimlen);

    pi_fg = cell(complen,1);
    pi_bg = cell(complen,1);
    mu_fg= cell(complen,1);
    mu_bg = cell(complen,1);
    cov_fg = cell(complen,1);
    cov_bg= cell(complen,1);

        % Create 5 different mixtures based on different initializations
    for i =1:complen
```

```
        [pi_fg{i},mu_fg{i},cov_fg{i}] =
compute_param(C(i),epsilon,n_iter,Train_FG);
        [pi_bg{i},mu_bg{i},cov_bg{i}] =
compute_param(C(i),epsilon,n_iter,Train_BG);
    end

    figure
    for i = 1:complen
        for k = 1:dimlen
            error_b(i,k) =
prediction(dctZigZag,height,width,dim(k),C(i),pi_bg{i},pi_fg{i},mu_bg{i},mu_f
g{i},cov_bg{i},cov_fg{i},prior_BG,prior_FG,ground_truth);
        end
        plot(dim,error_b(i,:),'o-')
        hold on
    end
    grid on
    legend('C=1','C=2','C=4','C=8','C=16','C=32');
    title('PoE vs feature dimension for different C');


%% UTILITY FUNCTIONS

function [error] =
prediction(dctZigZag,height,width,dim,C,pi_bg,pi_fg,mu_bg,mu_fg,cov_bg,cov_fg
,prior_BG,prior_FG,ground_truth)

    g_grass = zeros(height*width,1);
    g_cheetah = zeros(height*width,1);

    dct_dimvec = reshape(dctZigZag(:,:,1:dim),[],dim);
    mu_fg_dim = mu_fg(:,1:dim);
    mu_bg_dim = mu_bg(:,1:dim);
    cov_fg_dim = cov_fg(:,1:dim,1:dim);
    cov_bg_dim = cov_bg(:,1:dim,1:dim);

    for j=1:C
        g_grass = g_grass +
mvnpdf(dct_dimvec,mu_fg_dim(j,:),squeeze(cov_fg_dim(j,:,:)))*pi_fg(j);
        g_cheetah = g_cheetah +
mvnpdf(dct_dimvec,mu_bg_dim(j,:),squeeze(cov_bg_dim(j,:,:)))*pi_bg(j);
    end

    X = prior_BG*g_grass > prior_FG*g_cheetah;
    X = reshape(X,height,width);

    error = error_computation(ground_truth,X,prior_FG,prior_BG);
end


function [pi,mu,cov] = compute_param(C,threshold,n_iter,TrainData)

    % Initialize mixture model parameters
        pi = (randi(1000,C,1)-1)/1000;
```

```
        pi = pi/sum(pi);
        mu = randn(C,64);
        cov = zeros(C,64,64);

        N_train = size(TrainData,1);

        for i=1:C
            cov(i,:,:) = diag(randn(64,1).^2 + 1);
        end

        for step=1:n_iter
            h_ij = zeros(N_train,C);
            for j=1:C
                h_ij(:,j) =
mvnpdf(TrainData,mu(j,:),squeeze(cov(j,:,:)))*pi(j);
            end

            h_ij = h_ij./(sum(h_ij,2));

            for j = 1:C
                diagonal = sum(h_ij(:,j).*((TrainData-
mu(j,:)).^2),1)./sum(h_ij(:,j),1) + threshold;
                cov(j,:,:) = diag(diagonal);
            end
            mu = (h_ij'*TrainData)./sum(h_ij,1)';
            pi = sum(h_ij,1)./N_train;
        end
end


function [probability_error] =
error_computation(ground_truth,prediction,FG_prior,BG_prior)

    probability_error_cheetah = sum(ground_truth &
~prediction,'all')/sum(ground_truth,'all');
    probability_error_grass = sum(~ground_truth &
prediction,'all')/sum(~ground_truth,'all');
    probability_error = (FG_prior*probability_error_cheetah) +
(BG_prior*probability_error_grass);
end
```