Payam Khorramshahi

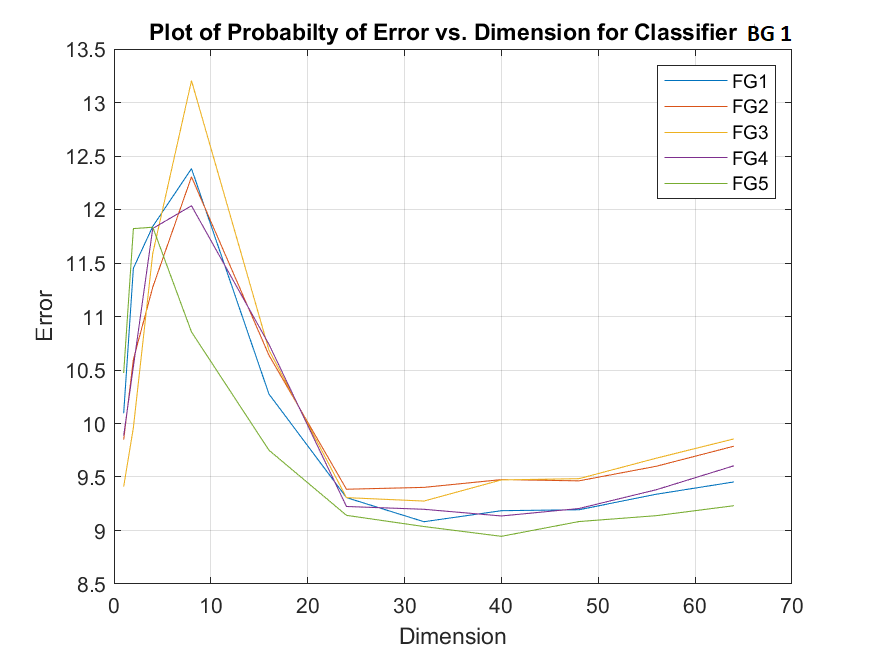
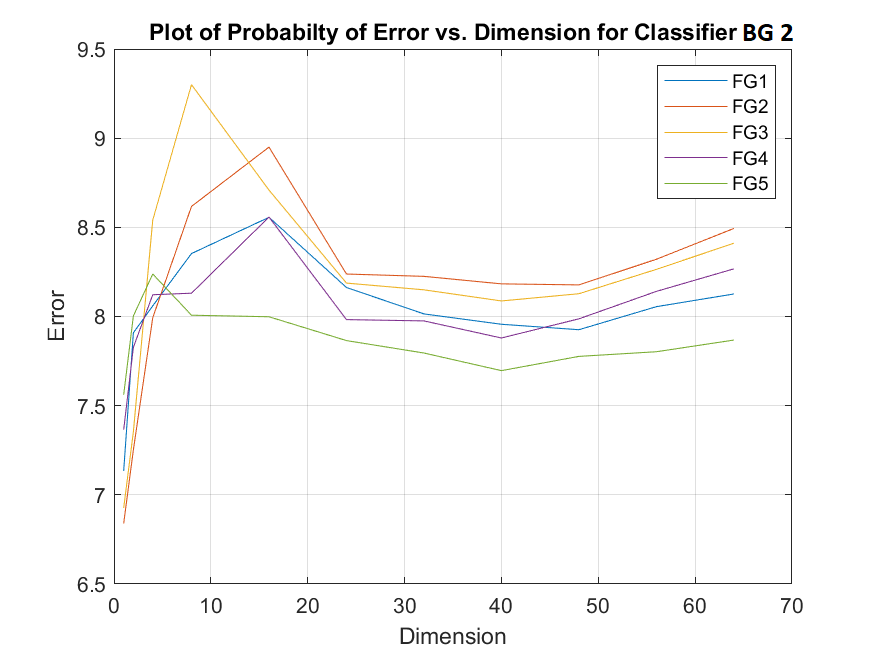
Homework #5

ECE 271A

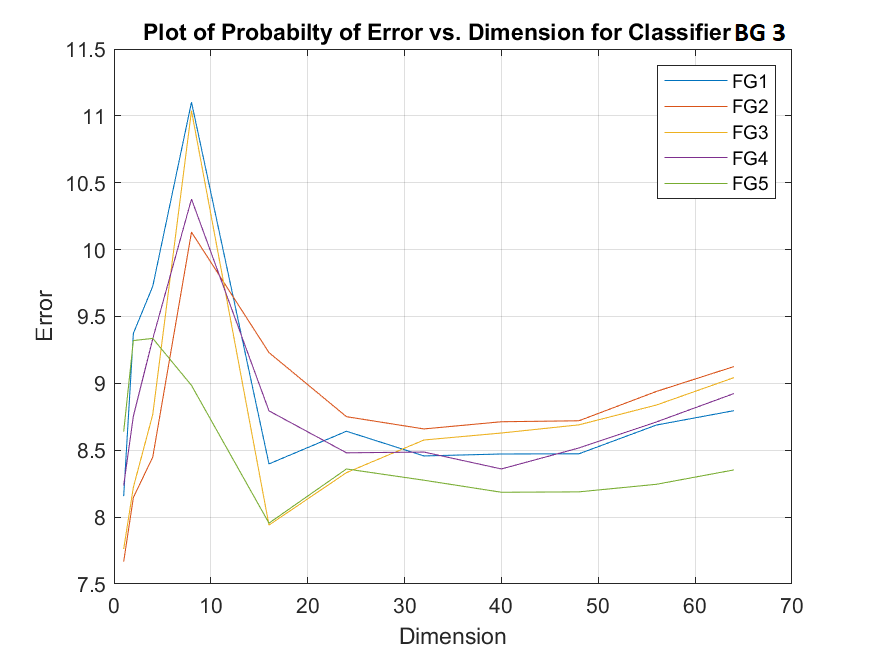
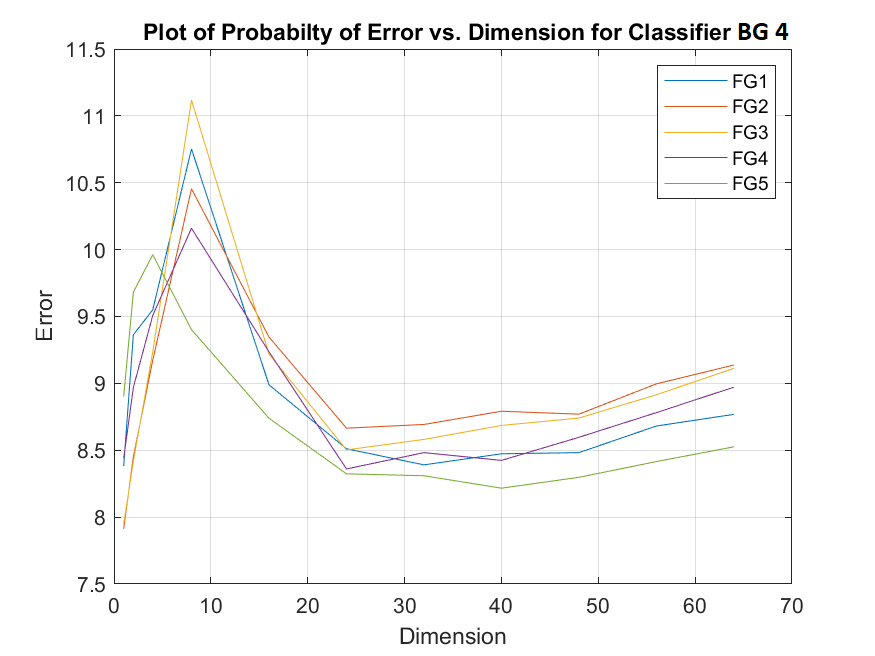
December 6, 2019

PID: 12719367

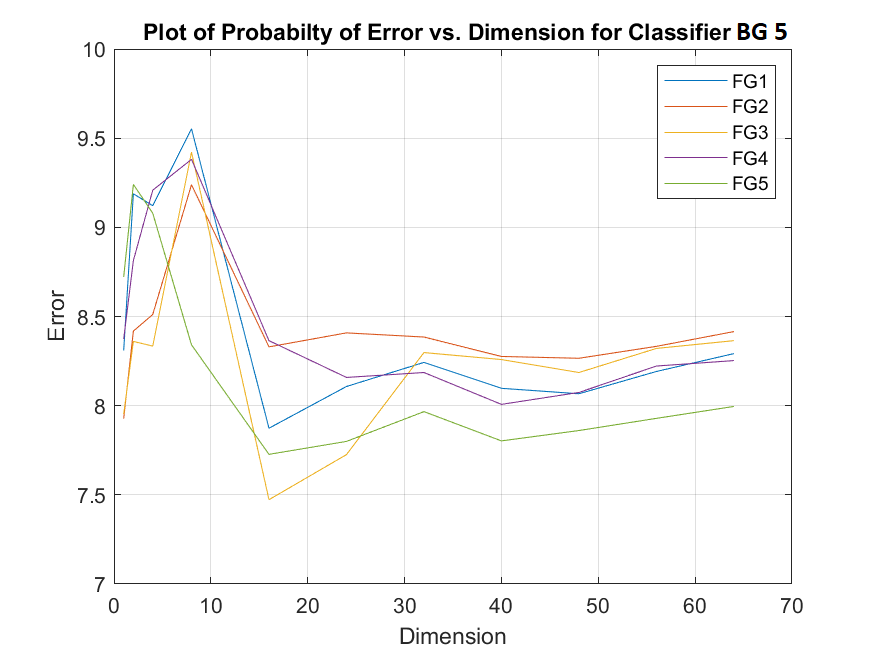
Plots

1. b)

c) d)



e)

Figure 1) The plots above represent the relation between probability of error vs. the dimension where each plot is shown a single BG mixture with all FG mixtures.

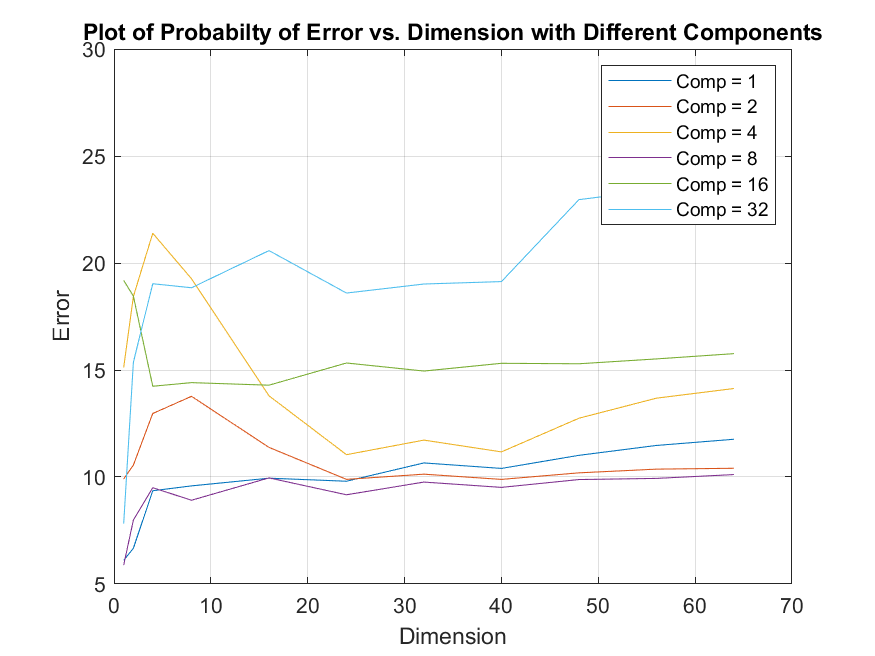


Figure2) The plot above represents the relation between probability of error vs. the dimension for mixtures of different components.

Discussion

The plots shown above are the result of estimation using Bayes Decision Rule (BDR) where the posterior probabilities are estimated using Expectation Maximization and optimized through 100 iteration.

From the plots shown in (figure 1), we conclude that as the number of dimension increases, the error decreases. This shows the significant role of the features meaning most of the feature provide important information about the two classes. There is also a sudden peak in error in the range of [5-10] dimension providing irrelevant information. Although we can observe small fluctuation through the plot which represents the irrelevance a few features, the overall tendency of the plots is downward toward decreasing error.

There exists a small difference in plot of different features. This can also be related to the initialization of the parameters which is random.

The plot shown in (figure 2) provides the relationship between the probability of error vs. the number of dimensions based on the complexity of the mixture (different number of components). As shown in this figure, the plot representing 8 components led into the least error. This shows, the mixture is likely to be composed of approximately ~8 components.

Appendix

Part (A) 🡪 Saving Updated Parameters

clear all

clc

close all

load 'TrainingSamplesDCT\_8\_new.mat'

%% PART A

% Initially we have to setup the priors

py\_G = 1053/(1053+250);

py\_F = 250/(1053+250);

% loading the dataset for background and foreground

C = [1, 2, 4, 8, 16, 32];

BG = TrainsampleDCT\_BG;

FG = TrainsampleDCT\_FG;

% Solving for h and updating the parameters for 100 times for the Foreground.

mix\_comp\_all\_BG = {};

for comp = 1:length(C)

% initialize 8 parameters for background

pbg = 1-0.5\*rand(1,C(comp));

pbg = pbg/sum(pbg);

mbg = {};

sbg = {};

for i = 1:C(comp)

mbg{i} = 1 - 0.5\*rand(1,64);

sbg{i} = diag(1 - 0.5\*rand(1,64));

end

% Loop through iteration to optimize the parameters.

for i = 1:100

conditional\_BG = 0;

for j = 1:C(comp)

conditional\_BG = conditional\_BG + (mvnpdf(BG,mbg{j},sbg{j}))\*pbg(j);

end

Hij = [];

for j = 1:C(comp)

Hij = [Hij (mvnpdf(BG,mbg{j},sbg{j}))\*pbg(j)./conditional\_BG];

end

pbg = sum(Hij)/(i+1);

m = (Hij'\*BG)./sum(Hij)';

s =(Hij'\*((BG - mbg{j}).^2))./sum(Hij)';

for j = 1:C(comp)

mbg{j} = m(j,:);

sbg{j} = diag(s(j,:));

end

end

mix\_comp\_all\_BG{comp} = {pbg, mbg, sbg};

end

save('mix\_comp\_all\_BG.mat', 'mix\_comp\_all\_BG')

% Solving for h and updating the parameters for 100 times for the Foreground.

mix\_comp\_all\_FG = {};

for comp = 1:length(C)

% initialize 8 parameters for foreground

pfg = 1 - 0.5\*rand(1,C(comp));

pfg = pfg/sum(pfg);

mfg = {};

sfg = {};

for i = 1:C(comp)

mfg{i} = 1 - 0.5\*rand(1,64);

sfg{i} = diag(1 - 0.5\*rand(1,64));

end

% Loop through iteration to optimize the parameters.

for i = 1:100

conditional\_FG = 0;

for j = 1:C(comp)

conditional\_FG = conditional\_FG + (mvnpdf(FG,mfg{j},sfg{j}))\*pfg(j);

end

Hij = [];

for j = 1:C(comp)

Hij = [Hij (mvnpdf(FG,mfg{j},sfg{j}))\*pfg(j)./conditional\_FG];

end

pfg = sum(Hij)/(i+1);

m = (Hij'\*FG)./sum(Hij)';

s =(Hij'\*((FG - mfg{j}).^2))./sum(Hij)';

for j = 1:C(comp)

mfg{j} = m(j,:);

sfg{j} = diag(s(j,:));

end

end

mix\_comp\_all\_FG{comp} = {pfg, mfg, sfg};

end

save('mix\_comp\_all\_FG.mat', 'mix\_comp\_all\_FG')

Part (A) 🡪 Implementation

clc

clear all

close all

load 'mix\_comp\_all\_FG.mat'

load 'mix\_comp\_all\_BG.mat'

C = [1, 2, 4, 8, 16, 32];

% Priors

py\_G = 1053/(1053+250);

py\_F = 250/(1053+250);

% Dimensions

d = [1, 2, 4, 8, 16, 24, 32, 40, 48, 56, 64];

%% Loading the IMAGE and implementation

% Loading the data

image = imread('cheetah.bmp');

image = im2double(image);

image2 = padarray(image,[4 4],0,'both');

load 'Zig-Zag Pattern.txt'

% Implementation on the image.

rows = size(image,1);

columns = size(image,2);

pad = 4;

batch\_size = 8;

for iii = 1:length(mix\_comp\_all\_BG)

P = [];

pbg = mix\_comp\_all\_BG{iii}{1};

mbg = mix\_comp\_all\_BG{iii}{2};

sbg = mix\_comp\_all\_BG{iii}{3};

pfg = mix\_comp\_all\_FG{iii}{1};

mfg = mix\_comp\_all\_FG{iii}{2};

sfg = mix\_comp\_all\_FG{iii}{3};

for T = 1:length(d)

A = zeros(rows,columns);

for r = 5 : 5+rows -1

col = 5;

while col <= columns

block = image2([(r-pad):((r-pad) + batch\_size - 1)],[(col-pad):((col-pad) + batch\_size -1)]);

vec = dct2(block);

new\_vec(Zig\_Zag\_Pattern(:)+1) = vec(:);

new\_vec = new\_vec(1:d(T));

conditional\_BG = 0;

for j = 1:C

conditional\_BG = conditional\_BG + (mvnpdf(new\_vec,mbg{j}(1:d(T)),sbg{j}(1:d(T),1:d(T))))\*pbg(j);

end

Gprob = py\_G\*conditional\_BG;

conditional\_FG = 0;

for j = 1:C

conditional\_FG = conditional\_FG + (mvnpdf(new\_vec,mfg{j}(1:d(T)),sfg{j}(1:d(T),1:d(T))))\*pfg(j);

end

Fprob = py\_F\*conditional\_FG;

if(Fprob>Gprob)

A(r-pad, col-pad) = 255;

end

col = col + 1;

end

end

compare = imread('cheetah\_mask.bmp');

total\_param = size(compare,1)\*size(compare,2);

P\_error = ((total\_param - sum(sum( A == compare)))/total\_param)\* 100;

P = [P P\_error];

end

figure(1)

plot(d,P)

hold on

title(sprintf('Plot of Probabilty of Error vs. Dimension with Different Components'))

xlabel('Dimension')

ylabel('Error')

legend('Comp = 1','Comp = 2','Comp = 4','Comp = 8','Comp = 16','Comp = 32')

grid on

box on

end

Part (B) 🡪 Saving Updated Parameters

clear all

clc

close all

load 'TrainingSamplesDCT\_8\_new.mat'

%% PART A

% Initially we have to setup the priors

py\_G = 1053/(1053+250);

py\_F = 250/(1053+250);

% loading the dataset for background and foreground

C = [1, 2, 4, 8, 16, 32];

BG = TrainsampleDCT\_BG;

FG = TrainsampleDCT\_FG;

% Solving for h and updating the parameters for 100 times for the Foreground.

mix\_comp\_all\_BG = {};

for comp = 1:length(C)

% initialize 8 parameters for background

pbg = 1-0.5\*rand(1,C(comp));

pbg = pbg/sum(pbg);

mbg = {};

sbg = {};

for i = 1:C(comp)

mbg{i} = 1 - 0.5\*rand(1,64);

sbg{i} = diag(1 - 0.5\*rand(1,64));

end

% Loop through iteration to optimize the parameters.

for i = 1:100

conditional\_BG = 0;

for j = 1:C(comp)

conditional\_BG = conditional\_BG + (mvnpdf(BG,mbg{j},sbg{j}))\*pbg(j);

end

Hij = [];

for j = 1:C(comp)

Hij = [Hij (mvnpdf(BG,mbg{j},sbg{j}))\*pbg(j)./conditional\_BG];

end

pbg = sum(Hij)/(i+1);

m = (Hij'\*BG)./sum(Hij)';

s =(Hij'\*((BG - mbg{j}).^2))./sum(Hij)';

for j = 1:C(comp)

mbg{j} = m(j,:);

sbg{j} = diag(s(j,:));

end

end

mix\_comp\_all\_BG{comp} = {pbg, mbg, sbg};

end

save('mix\_comp\_all\_BG.mat', 'mix\_comp\_all\_BG')

% Solving for h and updating the parameters for 100 times for the Foreground.

mix\_comp\_all\_FG = {};

for comp = 1:length(C)

% initialize 8 parameters for foreground

pfg = 1 - 0.5\*rand(1,C(comp));

pfg = pfg/sum(pfg);

mfg = {};

sfg = {};

for i = 1:C(comp)

mfg{i} = 1 - 0.5\*rand(1,64);

sfg{i} = diag(1 - 0.5\*rand(1,64));

end

% Loop through iteration to optimize the parameters.

for i = 1:100

conditional\_FG = 0;

for j = 1:C(comp)

conditional\_FG = conditional\_FG + (mvnpdf(FG,mfg{j},sfg{j}))\*pfg(j);

end

Hij = [];

for j = 1:C(comp)

Hij = [Hij (mvnpdf(FG,mfg{j},sfg{j}))\*pfg(j)./conditional\_FG];

end

pfg = sum(Hij)/(i+1);

m = (Hij'\*FG)./sum(Hij)';

s =(Hij'\*((FG - mfg{j}).^2))./sum(Hij)';

for j = 1:C(comp)

mfg{j} = m(j,:);

sfg{j} = diag(s(j,:));

end

end

mix\_comp\_all\_FG{comp} = {pfg, mfg, sfg};

end

save('mix\_comp\_all\_FG.mat', 'mix\_comp\_all\_FG')

Part (B) 🡪 Implementation

clc

clear all

close all

load 'mix\_comp\_all\_FG.mat'

load 'mix\_comp\_all\_BG.mat'

C = [1, 2, 4, 8, 16, 32];

% Priors

py\_G = 1053/(1053+250);

py\_F = 250/(1053+250);

% Dimensions

d = [1, 2, 4, 8, 16, 24, 32, 40, 48, 56, 64];

%% Loading the IMAGE and implementation

% Loading the data

image = imread('cheetah.bmp');

image = im2double(image);

image2 = padarray(image,[4 4],0,'both');

load 'Zig-Zag Pattern.txt'

% Implementation on the image.

rows = size(image,1);

columns = size(image,2);

pad = 4;

batch\_size = 8;

for iii = 1:length(mix\_comp\_all\_BG)

P = [];

pbg = mix\_comp\_all\_BG{iii}{1};

mbg = mix\_comp\_all\_BG{iii}{2};

sbg = mix\_comp\_all\_BG{iii}{3};

pfg = mix\_comp\_all\_FG{iii}{1};

mfg = mix\_comp\_all\_FG{iii}{2};

sfg = mix\_comp\_all\_FG{iii}{3};

for T = 1:length(d)

A = zeros(rows,columns);

for r = 5 : 5+rows -1

col = 5;

while col <= columns

block = image2([(r-pad):((r-pad) + batch\_size - 1)],[(col-pad):((col-pad) + batch\_size -1)]);

vec = dct2(block);

new\_vec(Zig\_Zag\_Pattern(:)+1) = vec(:);

new\_vec = new\_vec(1:d(T));

conditional\_BG = 0;

for j = 1:C

conditional\_BG = conditional\_BG + (mvnpdf(new\_vec,mbg{j}(1:d(T)),sbg{j}(1:d(T),1:d(T))))\*pbg(j);

end

Gprob = py\_G\*conditional\_BG;

conditional\_FG = 0;

for j = 1:C

conditional\_FG = conditional\_FG + (mvnpdf(new\_vec,mfg{j}(1:d(T)),sfg{j}(1:d(T),1:d(T))))\*pfg(j);

end

Fprob = py\_F\*conditional\_FG;

if(Fprob>Gprob)

A(r-pad, col-pad) = 255;

end

col = col + 1;

end

end

compare = imread('cheetah\_mask.bmp');

total\_param = size(compare,1)\*size(compare,2);

P\_error = ((total\_param - sum(sum( A == compare)))/total\_param)\* 100;

P = [P P\_error];

end

figure(1)

plot(d,P)

hold on

title(sprintf('Plot of Probabilty of Error vs. Dimension with Different Components'))

xlabel('Dimension')

ylabel('Error')

legend('Comp = 1','Comp = 2','Comp = 4','Comp = 8','Comp = 16','Comp = 32')

grid on

box on

end