**Project Assignment**

**Face Recognition using Linear Discriminant Analysis (LDA)**

1. **Face Recognition using Linear Discriminant Analysis (LDA)**

**MATLAB Code:**

%data extraction using imageset

Data = imageSet('faces\_att','recursive');

train\_data=cell(1,200);

test\_data=cell(1,200);

a = 1;

for j=1:40

for i=1:5 %first five images of all 40 subjects(classes) for training

X= read(Data(j),i);

X=reshape(X,prod(size(X)),1);

X=double(X);

train\_data{a} = X;

a = a + 1;

end;

end;

a=1;

for j=1:40 %40 subjects

for i=6:10 %last five images (6 to 10) of all 40 subjects for testing

X= read(Data(j),i);

X=reshape(X,prod(size(X)),1);

X=double(X);

test\_data{a} = X;

a = a + 1;

end;

end;

%%converting the cellarray to ordinary array or matrix

train\_data=cell2mat(train\_data);

test\_data=cell2mat(test\_data);

m=mean(train\_data,2);

% Calculating Mean of Each Class%%

j=1;

for i=0:5:195

temp(:,j)=mean(train\_data(:,i+1:i+5),2);

m\_class(:,i+1:i+5)=repmat(temp(:,j),[1,5]); %% Calculating Mean of Each Class%%

j=j+1;

end;

% Calculate the within class scatter (SW)

temp1=zeros(10304,10304);

wsca=zeros(10304,10304);

for i =0:5:195

temp1=(train\_data(:,i+1:i+5)-m\_class(:,i+1:i+5))\*((train\_data(:,i+1:i+5)-m\_class(:,i+1:i+5))');

wsca=temp1+wsca; %%calculating with in scatter matrix%%

end;

v=pinv(wsca); % Calculate the within class scatter (SW)

%%Calculating between scatter matrix%%

temp2=zeros(10304,10304);

bsca=zeros(10304,10304);

for i=1:40

temp2=(temp(:,i)-m)\*((temp(:,i)-m)');

bsca=temp2+bsca; %%Calculating between scatter matrix%%

end;

% Subtract the mean from each image [Centering the data]

d=train\_data-repmat(m,1,200); %for the training set

test\_data=test\_data-repmat(mean(test\_data,2),1,200);% performing the mean of the test matrix and subtracting the mean from each image(centering the data)

% find eigen values and eigen vectors of the (v)

[evec,eval]=eig(v\*bsca);

% Sort the eigen vectors according to the eigen values

eigvalue = diag(eval);

[junk, index] = sort(eigvalue,'descend');

% Compute the number of eigen values that greater than zero (you can select any threshold)

count1=0;

for i=1:size(eigvalue,1)

if(eigvalue(i)>0)

count1=count1+1;

end

end

% And also we can use the eigen vectors that the corresponding eigen values is greater than zero(Threshold) and this method will decrease the

% computation time and complixity

vec=evec(:,index(1:40)); %Number of principal components used

%% Fisher projection

tr\_pro=vec'\*d; %train projection

ts\_pro=vec'\*test\_data; %test projection

%Use Euclidean distance as distance metrics.

D=pdist2(tr\_pro',ts\_pro','Euclidean');

%labels

labels=zeros(200,200);

for i=1:200

for j=1:200

if(fix((i-1)/5)==fix((j-1)/5))

labels(i,j)=0;

else

labels(i,j)=1;

end

end

end

%performance evaluation plotting

ezroc3(D,labels,2,'',1);



1. **Comparison with Principal Component Analysis (PCA) based Face Recognition**





PCA (from Project1) VS LDA

**Results:**

* 40 Principal Components used**.**
* LDA outperforms PCA since it directly deals with class discrimination while PCA does not pay attention to the underlying class structure**.**
* We see that the accuracy obtained using LDA is 98.5% compared to using PCA which is 95.18%.
* LDA outperforms PCA when the sample is large
* It might not be the same when the data set is small, PCA might overperform better than LDA.
* Eigenfaces and Fisher faces work well only under controlled illumination, pose, etc. conditions, because they depend heavily on an almost pixel-by-pixel correspondence between the face images to be matched.