Santosh Yadav Jada

16245773

Software used: MATLAB 2017

Mode 1:

clc

clear all

close all

% no of classes/subjects

class= 40;

% no of images in class

Total\_images\_in\_class = 10;

% no of training images

Trainingimages = 5;

% declaration of arrays

A = [];

C = [];

% for loop to retrieve data from the ATT\_faces

for i = 1:class

fn = strcat('ATT\_faces\s', num2str(i), '\');

for j = 1:Trainingimages

f = strcat(num2str(j) , '.pgm');

file = strcat(fn,f);

A = imread(file);

A = (A(:));

%horzcat for concatenation of the arrays

C = horzcat(C,A);

end

end

%% PCA

data=double(C);

[row,col]=size(data);

% mean of the data

m=mean(data')';

d=data-(repmat(m,1,col));

d=double(d);

% covariance

co=d\*d';

% Calculating eigen vectors and eigen values

[eigvector,eigvl] = eig(co);

eigvalue = diag(eigvl);

% sorting of eigen values in descending order

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

eigvalue = eigvalue(index);

eigvector = eigvector(:, index);

% 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

% We can use all the eigen vectors but this method will increase the

% computation time and complixity

%vec=eigvector(:,:);

% 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=eigvector(:,1:count1);

% Compute the feature matrix (the space that will use it to project the testing image on it)

x=vec'\*d;

%% test data

% Declaration of arrays

R = [];

T = [];

% reading the data

for i = 1:class

fn = strcat('\\kc.umkc.edu\kc-users\home\s\sjzkt\My Documents\MATLAB\ATT\_faces\s', num2str(i) , '\');

for j = 6:10

f = strcat(num2str(j), '.pgm');

file1 = strcat(fn,f);

image = imread(file1);

R = image;

R = (R(:));

% concatenation of arrays

T= horzcat(T,R);

end

end

disp(size(T));

T=double(T);

T=(T - m);

T = vec'\*T;

% using calssifier

D = pdist2(T',x','Euclidean');

% creating 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

% using ezroc3 function

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

function [roc,EER,area,EERthr,ALLthr,d,gen,imp]=ezroc3(H,T,plot\_stat,headding,printInfo)%,rbst

t1=min(min(min(H)));

t2=max(max(max(H)));

num\_subj=size(H,1);

stp=(t2-t1)/500; %step size here is 0.2% of threshold span, can be adjusted

if stp==0 %if all inputs are the same...

stp=0.01; %Token value

end

ALLthr=(t1-stp):stp:(t2+stp);

if (nargin==1 || (nargin==3 && isempty(T))||(nargin==2 && isempty(T))||(nargin==4 && isempty(T))||(nargin==5 && isempty(T))) %Using only H, multi-class case, and maybe 3D or no plot

GAR=zeros(503,size(H,3)); %initialize for accumulation in case of multiple H (on 3rd dim of H)

FAR=zeros(503,size(H,3));

gen=[]; %genuine scores place holder (diagonal of H), for claculation of d'

imp=[]; %impostor scores place holder (non-diagonal elements of H), for claculation of d'

for setnum=1:size(H,3); %multiple H measurements (across 3rd dim, where 2D H's stack up)

gen=[gen; diag(H(:,:,setnum))]; %digonal scores

imp=[imp; H(find(not(eye(size(H,2)))))]; %off-diagonal scores, with off-diagonal indices being listed by find(not(eye(size(H,2))))

for t=(t1-stp):stp:(t2+stp), %Note that same threshold is used for all H's, and we increase the limits by a smidgeon to get a full curve

ind=round((t-t1)/stp+2); %current loop index, +2 to start from 1

id=H(:,:,setnum)>t;

True\_Accept=trace(id); %TP

False\_Reject=num\_subj-True\_Accept; %FN

% In the following, id-diag(diag(id)) simply zeros out the diagonal of id

True\_Reject=sum( sum( (id-diag(diag(id)))==0 ) )-size(id,1); %TN, number of off-diag zeros. We need to subtract out the number of diagonals, as 'id-diag(diag(id))' introduces those many extra zeros into the sum

False\_Accept=sum( sum( id-diag(diag(id)) ) ); %FP, number of off-diagonal ones

GAR(ind,setnum)=GAR(ind,setnum)+True\_Accept/(True\_Accept+False\_Reject); %1-FRR, Denum: all the positives (correctly IDed+incorrectly IDed)

FAR(ind,setnum)=FAR(ind,setnum)+False\_Accept/(True\_Reject+False\_Accept); %1-GRR, Denum: all the negatives (correctly IDed+incorrectly IDed)

end

end

GAR=sum(GAR,2)/size(H,3); %average across multiple H's

FAR=sum(FAR,2)/size(H,3);

elseif (nargin==2 || nargin==3 || nargin == 4 || nargin == 5), %Regular, 1-class-vs-rest ROC, and maybe 3D or no plot

gen=H(find(T)); %genuine scores

imp=H(find(not(T))); %impostor scores

for t=(t1-stp):stp:(t2+stp), %span the limits by a smidgeon to get a full curve

ind=round((t-t1)/stp+2); %current loop index, +2 to start from 1

id=H>t;

True\_Accept=sum(and(id,T)); %TP

False\_Reject=sum(and(not(id),T)); %FN

True\_Reject=sum(and(not(id),not(T))); %TN

False\_Accept=sum(and(id,not(T))); %FP

GAR2(ind)=True\_Accept/(True\_Accept+False\_Reject); %1-FRR, Denum: all the positives (correctly IDed+incorrectly IDed)

FAR2(ind)=False\_Accept/(True\_Reject+False\_Accept); %1-GRR, Denum: all the negatives (correctly IDed+incorrectly IDed)

end

GAR=GAR2';

FAR=FAR2';

end

roc=[GAR';FAR'];

FRR=1-GAR;

[e ind]=min(abs(FRR'-FAR')); %This is Approx w/ error e. Fix by linear inerpolation of neigborhood and intersecting w/ y=x

EER=(FRR(ind)+FAR(ind))/2;

area=abs(trapz(roc(2,:),roc(1,:)));

EERthr=t1+(ind-1)\*stp;%EER threshold

d=abs(mean(gen)-mean(imp))/(sqrt(0.5\*(var(gen)+var(imp)))); %Decidability or d'

if (nargin==1 || nargin==2 || nargin==3 || nargin == 4 || nargin == 5)

if plot\_stat == 2

if printInfo == 1

figure, plot(roc(2,:),roc(1,:),'LineWidth',3),axis([-0.002 1 0 1.002]),title(['ROC Curve: ' headding ' EER=' num2str(EER) ', Area=' num2str(area) ', Decidability=' num2str(d)]),xlabel('FAR'),ylabel('GAR');

elseif printInfo == 0

figure, plot(roc(2,:),roc(1,:),'LineWidth',3),axis([-0.002 1 0 1.002]),title(['ROC Curve: ' headding ' ']),xlabel('FAR'),ylabel('GAR');

end

elseif plot\_stat == 3

if printInfo == 1

figure, plot3(roc(2,:),roc(1,:),ALLthr,'LineWidth',3),axis([0 1 0 1 (t1-stp) (t2+stp)]),title(['3D ROC Curve: ' headding ' EER=' num2str(EER) ', Area=' num2str(area) ', Decidability=' num2str(d)]),xlabel('FAR'),ylabel('GAR'),zlabel('Threshold'),grid on,axis square;

elseif printInfo == 0

figure, plot3(roc(2,:),roc(1,:),ALLthr,'LineWidth',3),axis([0 1 0 1 (t1-stp) (t2+stp)]),title(['3D ROC Curve: ' headding ' ']),xlabel('FAR'),ylabel('GAR'),zlabel('Threshold'),grid on,axis square;

end

else

%else it must be 0, i.e. no plot

end

end

end

Mode 2:

clc

clear all

close all

% no of class/subjects

class = 25;

% training images

Total\_images\_in\_class = 10;

% declaration of arrays

A = [];

C = [];

% for loop to read the data

for i = 1:class

fn = strcat('\\kc.umkc.edu\kc-users\home\s\sjzkt\My Documents\MATLAB\ATT\_faces\s', num2str(i), '\');

for j = 1:Total\_images\_in\_class

f = strcat(num2str(j) , '.pgm');

file = strcat(fn,f);

A = imread(file);

A = (A(:));

%horzcat for concatenation of the arrays

C = horzcat(C,A);

end

end

%% PCA

data=C;

[row,col]=size(data);

% Compute the mean of the data matrix "The mean of each row"

m=mean(data')';

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

d=data- uint8(repmat(m,1,col));

d=double(d);

% Compute the covariance matrix (co)

co=d\*d';

% Compute the eigen values and eigen vectors of the covariance matrix

[eigvector,eigvl] = eig(co);

% Sort the eigen vectors according to the eigen values

eigvalue = diag(eigvl);

% sorting them accoridng to descending order

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

eigvalue = eigvalue(index);

eigvector = eigvector(:, index);

% 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

% We can use all the eigen vectors but this method will increase the

% computation time and complixity

%vec=eigvector(:,:);

% 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=eigvector(:,1);

%% training data

class = 15;

Images\_per\_class = 10;

% decalration of arrays

R = [];

T = [];

% reading the data from the database

for i = 26: 40

fn = strcat('\\kc.umkc.edu\kc-users\home\s\sjzkt\My Documents\MATLAB\ATT\_faces\s', num2str(i) , '\');

for j = 1:5

f = strcat(num2str(j), '.pgm');

file1 = strcat(fn,f);

image = imread(file1);

R = image;

R = (R(:));

% concatenation of arrays

T = horzcat(T,R);

end

end

disp(size(T));

T=double(T);

T=(T- m);

T = vec'\*T;

%% test data

% reading the data

% declaration of arrays

X = [];

Z= [];

for i = 26: 40

fn = strcat('\\kc.umkc.edu\kc-users\home\s\sjzkt\My Documents\MATLAB\ATT\_faces\s', num2str(i) , '\');

Y= [];

for j = 6:10

f = strcat(num2str(j), '.pgm');

filename1 = strcat(fn,f);

img = imread(filename1);

X= img;

X = (X(:));

Z = horzcat(Z,X);

end

end

disp(size(Z));

Z=double(Z);

Z=(Z - m);

Z = vec'\*Z;

% using the classifier

D = pdist2(Z',T','Euclidean');

% creation of labels

labels =zeros(75,75);

for i=1:75

for j=1:75

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

labels(i,j)=0;

else

labels(i,j)=1;

end

end

end

% using ezroc3 function

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

function [roc,EER,area,EERthr,ALLthr,d,gen,imp]=ezroc3(H,T,plot\_stat,headding,printInfo)%,rbst

t1=min(min(min(H)));

t2=max(max(max(H)));

num\_subj=size(H,1);

stp=(t2-t1)/500; %step size here is 0.2% of threshold span, can be adjusted

if stp==0 %if all inputs are the same...

stp=0.01; %Token value

end

ALLthr=(t1-stp):stp:(t2+stp);

if (nargin==1 || (nargin==3 && isempty(T))||(nargin==2 && isempty(T))||(nargin==4 && isempty(T))||(nargin==5 && isempty(T))) %Using only H, multi-class case, and maybe 3D or no plot

GAR=zeros(503,size(H,3)); %initialize for accumulation in case of multiple H (on 3rd dim of H)

FAR=zeros(503,size(H,3));

gen=[]; %genuine scores place holder (diagonal of H), for claculation of d'

imp=[]; %impostor scores place holder (non-diagonal elements of H), for claculation of d'

for setnum=1:size(H,3); %multiple H measurements (across 3rd dim, where 2D H's stack up)

gen=[gen; diag(H(:,:,setnum))]; %digonal scores

imp=[imp; H(find(not(eye(size(H,2)))))]; %off-diagonal scores, with off-diagonal indices being listed by find(not(eye(size(H,2))))

for t=(t1-stp):stp:(t2+stp), %Note that same threshold is used for all H's, and we increase the limits by a smidgeon to get a full curve

ind=round((t-t1)/stp+2); %current loop index, +2 to start from 1

id=H(:,:,setnum)>t;

True\_Accept=trace(id); %TP

False\_Reject=num\_subj-True\_Accept; %FN

% In the following, id-diag(diag(id)) simply zeros out the diagonal of id

True\_Reject=sum( sum( (id-diag(diag(id)))==0 ) )-size(id,1); %TN, number of off-diag zeros. We need to subtract out the number of diagonals, as 'id-diag(diag(id))' introduces those many extra zeros into the sum

False\_Accept=sum( sum( id-diag(diag(id)) ) ); %FP, number of off-diagonal ones

GAR(ind,setnum)=GAR(ind,setnum)+True\_Accept/(True\_Accept+False\_Reject); %1-FRR, Denum: all the positives (correctly IDed+incorrectly IDed)

FAR(ind,setnum)=FAR(ind,setnum)+False\_Accept/(True\_Reject+False\_Accept); %1-GRR, Denum: all the negatives (correctly IDed+incorrectly IDed)

end

end

GAR=sum(GAR,2)/size(H,3); %average across multiple H's

FAR=sum(FAR,2)/size(H,3);

elseif (nargin==2 || nargin==3 || nargin == 4 || nargin == 5), %Regular, 1-class-vs-rest ROC, and maybe 3D or no plot

gen=H(find(T)); %genuine scores

imp=H(find(not(T))); %impostor scores

for t=(t1-stp):stp:(t2+stp), %span the limits by a smidgeon to get a full curve

ind=round((t-t1)/stp+2); %current loop index, +2 to start from 1

id=H>t;

True\_Accept=sum(and(id,T)); %TP

False\_Reject=sum(and(not(id),T)); %FN

True\_Reject=sum(and(not(id),not(T))); %TN

False\_Accept=sum(and(id,not(T))); %FP

GAR2(ind)=True\_Accept/(True\_Accept+False\_Reject); %1-FRR, Denum: all the positives (correctly IDed+incorrectly IDed)

FAR2(ind)=False\_Accept/(True\_Reject+False\_Accept); %1-GRR, Denum: all the negatives (correctly IDed+incorrectly IDed)

end

GAR=GAR2';

FAR=FAR2';

end

roc=[GAR';FAR'];

FRR=1-GAR;

[e ind]=min(abs(FRR'-FAR')); %This is Approx w/ error e. Fix by linear inerpolation of neigborhood and intersecting w/ y=x

EER=(FRR(ind)+FAR(ind))/2;

area=abs(trapz(roc(2,:),roc(1,:)));

EERthr=t1+(ind-1)\*stp;%EER threshold

d=abs(mean(gen)-mean(imp))/(sqrt(0.5\*(var(gen)+var(imp)))); %Decidability or d'

if (nargin==1 || nargin==2 || nargin==3 || nargin == 4 || nargin == 5)

if plot\_stat == 2

if printInfo == 1

figure, plot(roc(2,:),roc(1,:),'LineWidth',3),axis([-0.002 1 0 1.002]),title(['ROC Curve: ' headding ' EER=' num2str(EER) ', Area=' num2str(area) ', Decidability=' num2str(d)]),xlabel('FAR'),ylabel('GAR');

elseif printInfo == 0

figure, plot(roc(2,:),roc(1,:),'LineWidth',3),axis([-0.002 1 0 1.002]),title(['ROC Curve: ' headding ' ']),xlabel('FAR'),ylabel('GAR');

end

elseif plot\_stat == 3

if printInfo == 1

figure, plot3(roc(2,:),roc(1,:),ALLthr,'LineWidth',3),axis([0 1 0 1 (t1-stp) (t2+stp)]),title(['3D ROC Curve: ' headding ' EER=' num2str(EER) ', Area=' num2str(area) ', Decidability=' num2str(d)]),xlabel('FAR'),ylabel('GAR'),zlabel('Threshold'),grid on,axis square;

elseif printInfo == 0

figure, plot3(roc(2,:),roc(1,:),ALLthr,'LineWidth',3),axis([0 1 0 1 (t1-stp) (t2+stp)]),title(['3D ROC Curve: ' headding ' ']),xlabel('FAR'),ylabel('GAR'),zlabel('Threshold'),grid on,axis square;

end

else

%else it must be 0, i.e. no plot

end

end

end