TRAINING

%% clearing

clc;

clear all;

warning off all

close all;

%% Deletion of previously system generated files

delete net\_our\_data.mat

delete svmStruct\_our\_data\_A.mat

delete svmStruct\_our\_data\_S.mat

delete svmStruct\_our\_data\_H.mat

delete S\_our\_feature.mat % sad

delete A\_our\_feature.mat % angry

delete H\_our\_feature.mat % Happy

delete our\_total\_set.mat

delete total\_features\_list\_our\_data.xlsx

%% Exctracting Features for SAD emotion and storing the values.

Files1=dir('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\sad');%loading all the audioclips in database for sad emotion

num\_files\_Stype=size(Files1,1); % initialising number of images

x1(num\_files\_Stype,288)=0; % creating a matrix of zero for number of images vs features

count=1; % initialising count

for i=3:num\_files\_Stype %Change loop, according to number of images in the folder%

str = strcat('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\sad\', Files1(i).name); %extracting files from folder

disp(str)%For displaying the path of image in command window

[sig\_limited,fs]=audioread(str);

sig\_1000=fs/1000;% maximum speech Fx at 1000Hz

sig\_50=fs/50;% minimum speech Fx at 50Hz

sig\_500=fs/500;% maximum speech Fx at 500Hz

t=(0:length(sig\_limited)-1)/fs;

Y=fft(sig\_limited.\*hamming(length(sig\_limited))); % taking FFT

hz5000=5000\*length(Y)/fs;

f=(0:hz5000)\*fs/length(Y);

C=fft(log(abs(Y)+eps));% cepstrum is DFT of log spectrum

q=(sig\_1000:sig\_50)/fs; % plotting between 1ms (1000Hz) and 20ms (50Hz)

[c,fx]=max(abs(C(sig\_1000:sig\_50)));

r\_corre=xcorr(sig\_limited,sig\_50,'coeff'); % calculate autocorrelation

d\_timw=(-sig\_50:sig\_50)/fs;

r\_corre=r\_corre(sig\_50+1:2\*sig\_50+1);

[rmax,tx]=max(r\_corre(sig\_500:sig\_50));

x=resample(sig\_limited,10000,fs);

fs=10000;

lpf=2+fs/1000; % rule

a=lpc(x,lpf);

[h,f]=freqz(1,a,512,fs);

r=roots(a); % find roots of polynomial a

r=r(imag(r)>0.01); % only look for roots >0Hz up to fs/2

ffreq=sort(atan2(imag(r),real(r))\*fs/(2\*pi));

Features = stFeatureExtraction(sig\_limited, fs, 0.020, 0.020);

F = Features(3,:);

timeFeature = 0.010:0.020:length(x)/fs;

time = 0:1/fs:length(x)/fs-1/fs;

MIN1 = min([length(F);length(timeFeature)]);

timeFeature = timeFeature(1:MIN1);

F = F(1:MIN1);

MIN2 = min([length(x);length(time)]);

time = time(1:MIN2);

x = x(1:MIN2);

% [Frt] = spFormantsLpc(sig\_limited, fs);

Ft=Features';

[mp,np]=size(Ft);

Ft1=Ft(1:round(mp/2),:);

Ft2=Ft(round(mp/2):mp,:);

A1=max(Ft1);

A2=min(Ft1);

A3=mean(Ft1);

A4=std(Ft1);

A7=median(Ft1);

A012=entropy(Ft1);

A10=max(Ft2);

A20=min(Ft2);

A30=mean(Ft2);

A40=std(Ft2);

A70=median(Ft2);

A120=entropy(Ft2);

A201=max(Ft1);

A202=min(Ft1);

A203=mean(Ft1);

A204=std(Ft1);

A207=median(Ft1);

Final\_feat(count,:)=[A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207 ];

count=count+1;

clear A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207

clear Ft1 Ft2 mp np Ft

end

save S\_our\_feature Final\_feat

clear all

%% Exctracting Features for ANGRY emotion and storing the values.

Files1=dir('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\angry');%loading all the audioclips in database for sad emotion

num\_files\_Atype=size(Files1,1); % initialising number of images

x1(num\_files\_Atype,288)=0; % creating a matrix of zero for number of images vs features

count=1; % initialising count

for i=3:num\_files\_Atype %Change loop, according to number of images in the folder%

str = strcat('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\angry\', Files1(i).name); %extracting files from folder

disp(str)%For displaying the path of image in command window

[sig\_limited,fs]=audioread(str);

sig\_1000=fs/1000;% maximum speech Fx at 1000Hz

sig\_50=fs/50;% minimum speech Fx at 50Hz

sig\_500=fs/500;% maximum speech Fx at 500Hz

t=(0:length(sig\_limited)-1)/fs;

Y=fft(sig\_limited.\*hamming(length(sig\_limited))); % taking FFT

hz5000=5000\*length(Y)/fs;

f=(0:hz5000)\*fs/length(Y);

C=fft(log(abs(Y)+eps));% cepstrum is DFT of log spectrum

q=(sig\_1000:sig\_50)/fs; % plotting between 1ms (1000Hz) and 20ms (50Hz)

[c,fx]=max(abs(C(sig\_1000:sig\_50)));

r\_corre=xcorr(sig\_limited,sig\_50,'coeff'); % calculate autocorrelation

d\_timw=(-sig\_50:sig\_50)/fs;

r\_corre=r\_corre(sig\_50+1:2\*sig\_50+1);

[rmax,tx]=max(r\_corre(sig\_500:sig\_50));

x=resample(sig\_limited,10000,fs);

fs=10000;

lpf=2+fs/1000; % rule

a=lpc(x,lpf);

[h,f]=freqz(1,a,512,fs);

r=roots(a); % find roots of polynomial a

r=r(imag(r)>0.01); % only look for roots >0Hz up to fs/2

ffreq=sort(atan2(imag(r),real(r))\*fs/(2\*pi));

Features = stFeatureExtraction(sig\_limited, fs, 0.020, 0.020);

F = Features(3,:);

timeFeature = 0.010:0.020:length(x)/fs;

time = 0:1/fs:length(x)/fs-1/fs;

MIN1 = min([length(F);length(timeFeature)]);

timeFeature = timeFeature(1:MIN1);

F = F(1:MIN1);

MIN2 = min([length(x);length(time)]);

time = time(1:MIN2);

x = x(1:MIN2);

% [Frt] = spFormantsLpc(sig\_limited, fs);

Ft=Features';

[mp,np]=size(Ft);

Ft1=Ft(1:round(mp/2),:);

Ft2=Ft(round(mp/2):mp,:);

A1=max(Ft1);

A2=min(Ft1);

A3=mean(Ft1);

A4=std(Ft1);

A7=median(Ft1);

A012=entropy(Ft1);

A10=max(Ft2);

A20=min(Ft2);

A30=mean(Ft2);

A40=std(Ft2);

A70=median(Ft2);

A120=entropy(Ft2);

A201=max(Ft1);

A202=min(Ft1);

A203=mean(Ft1);

A204=std(Ft1);

A207=median(Ft1);

Final\_feat(count,:)=[A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207 ];

count=count+1;

clear A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207

clear Ft1 Ft2 mp np Ft

end

save A\_our\_feature Final\_feat

clear all

%% Exctracting Features for HAPPY emotion and storing the values.

Files1=dir('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\happy');%loading all the audioclips in database for sad emotion

num\_files\_Htype=size(Files1,1); % initialising number of images

x1(num\_files\_Htype,288)=0; % creating a matrix of zero for number of images vs features

count=1; % initialising count

for i=3:num\_files\_Htype %Change loop, according to number of images in the folder%

str = strcat('C:\fina year project\Emotion detection\Emo\_Det\Training\_Audio\_Clips\happy\', Files1(i).name); %extracting files from folder

disp(str)%For displaying the path of image in command window

[sig\_limited,fs]=audioread(str);

sig\_1000=fs/1000;% maximum speech Fx at 1000Hz

sig\_50=fs/50;% minimum speech Fx at 50Hz

sig\_500=fs/500;% maximum speech Fx at 500Hz

t=(0:length(sig\_limited)-1)/fs;

Y=fft(sig\_limited.\*hamming(length(sig\_limited))); % taking FFT

hz5000=5000\*length(Y)/fs;

f=(0:hz5000)\*fs/length(Y);

C=fft(log(abs(Y)+eps));% cepstrum is DFT of log spectrum

q=(sig\_1000:sig\_50)/fs; % plotting between 1ms (1000Hz) and 20ms (50Hz)

[c,fx]=max(abs(C(sig\_1000:sig\_50)));

r\_corre=xcorr(sig\_limited,sig\_50,'coeff'); % calculate autocorrelation

d\_timw=(-sig\_50:sig\_50)/fs;

r\_corre=r\_corre(sig\_50+1:2\*sig\_50+1);

[rmax,tx]=max(r\_corre(sig\_500:sig\_50));

x=resample(sig\_limited,10000,fs);

fs=10000;

lpf=2+fs/1000; % rule

a=lpc(x,lpf);

[h,f]=freqz(1,a,512,fs);

r=roots(a); % find roots of polynomial a

r=r(imag(r)>0.01); % only look for roots >0Hz up to fs/2

ffreq=sort(atan2(imag(r),real(r))\*fs/(2\*pi));

Features = stFeatureExtraction(sig\_limited, fs, 0.020, 0.020);

F = Features(3,:);

timeFeature = 0.010:0.020:length(x)/fs;

time = 0:1/fs:length(x)/fs-1/fs;

MIN1 = min([length(F);length(timeFeature)]);

timeFeature = timeFeature(1:MIN1);

F = F(1:MIN1);

MIN2 = min([length(x);length(time)]);

time = time(1:MIN2);

x = x(1:MIN2);

% [Frt] = spFormantsLpc(sig\_limited, fs);

Ft=Features';

[mp,np]=size(Ft);

Ft1=Ft(1:round(mp/2),:);

Ft2=Ft(round(mp/2):mp,:);

A1=max(Ft1);

A2=min(Ft1);

A3=mean(Ft1);

A4=std(Ft1);

A7=median(Ft1);

A012=entropy(Ft1);

A10=max(Ft2);

A20=min(Ft2);

A30=mean(Ft2);

A40=std(Ft2);

A70=median(Ft2);

A120=entropy(Ft2);

A201=max(Ft1);

A202=min(Ft1);

A203=mean(Ft1);

A204=std(Ft1);

A207=median(Ft1);

Final\_feat(count,:)=[A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207 ];

count=count+1;

clear A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207

clear Ft1 Ft2 mp np Ft

end

save H\_our\_feature Final\_feat

clear all

%% ANGRY

%% LOADING ALL FEATURE SET

A=load('A\_our\_feature');

S=load('S\_our\_feature');

H=load('H\_our\_feature');

total\_set\_A=[A.Final\_feat;S.Final\_feat;H.Final\_feat];

%total\_set = total\_set/norm(total\_set);

xlswrite('total\_features\_list\_our\_data\_A.xlsx',total\_set\_A,'A1')

save our\_all\_feat\_A.mat total\_set\_A

%% Labelling data

[A\_a\_lab,~]=size(A.Final\_feat);

[A\_b\_lab,~]=size(total\_set\_A);

Size\_A=A\_a\_lab;

size\_rem\_A=A\_b\_lab-A\_a\_lab;

%% Setting f to 1 and others to zero

lab\_A=ones( Size\_A,1);

lab\_rem\_A=zeros(size\_rem\_A,1);

svm\_label\_A=[lab\_A;lab\_rem\_A]; % for svm

save our\_svm\_label\_A.mat svm\_label\_A

%% SVM CLASSIFIER

A\_k=total\_set\_A(:,2:3);

figure, svmStruct\_A = svmtrain(A\_k,svm\_label\_A,'kernel\_function','quadratic','showplot',true);

figure, svmStructA2 = svmtrain(A\_k,svm\_label\_A,'kernel\_function','linear','showplot',true);

figure, svmStructA3 = svmtrain(A\_k,svm\_label\_A,'kernel\_function','polynomial','showplot',true);

save svmStruct\_our\_data\_A.mat svmStruct\_A

%% HAPPY

%% LOADING ALL FEATURE SET

A=load('A\_our\_feature');

S=load('S\_our\_feature');

H=load('H\_our\_feature');

total\_set\_H=[H.Final\_feat;A.Final\_feat;S.Final\_feat];

%total\_set = total\_set/norm(total\_set);

xlswrite('total\_features\_list\_our\_data\_H.xlsx',total\_set\_H,'A1')

save our\_all\_feat\_H.mat total\_set\_H

%% Labelling data

[H\_a\_lab,~]=size(S.Final\_feat);

[H\_b\_lab,~]=size(total\_set\_H);

Size\_H=H\_a\_lab;

size\_rem\_H=H\_b\_lab-H\_a\_lab;

%% Setting f to 1 and others to zero

lab\_H=ones( Size\_H,1);

lab\_rem\_H=zeros(size\_rem\_H,1);

svm\_label\_H=[lab\_H;lab\_rem\_H]; % for svm

save our\_svm\_label\_H.mat svm\_label\_H

%% SVM CLASSIFIER

H\_k=total\_set\_H(:,2:3);

figure, svmStruct\_H = svmtrain(H\_k,svm\_label\_H,'kernel\_function','quadratic','showplot',true);

figure, svmStructH2 = svmtrain(H\_k,svm\_label\_H,'kernel\_function','linear','showplot',true);

figure, svmStructH3 = svmtrain(H\_k,svm\_label\_H,'kernel\_function','polynomial','showplot',true);

save svmStruct\_our\_data\_H.mat svmStruct\_H

%% SAD

%% LOADING ALL FEATURE SET

A=load('A\_our\_feature');

S=load('S\_our\_feature');

H=load('H\_our\_feature');

total\_set\_S=[S.Final\_feat;A.Final\_feat;H.Final\_feat];

%total\_set = total\_set/norm(total\_set);

xlswrite('total\_features\_list\_our\_data\_S.xlsx',total\_set\_S,'A1')

save our\_all\_feat\_S.mat total\_set\_S

%% Labelling data

[S\_a\_lab,~]=size(S.Final\_feat);

[S\_b\_lab,~]=size(total\_set\_S);

Size\_S=S\_a\_lab;

size\_rem\_S=S\_b\_lab-S\_a\_lab;

%% Setting f to 1 and others to zero

lab\_S=ones( Size\_S,1);

lab\_rem\_S=zeros(size\_rem\_S,1);

svm\_label\_S=[lab\_S;lab\_rem\_S]; % for svm

save our\_svm\_label\_S.mat svm\_label\_S

%% SVM CLASSIFIER

S\_k=total\_set\_S(:,2:3);

figure, svmStruct\_S = svmtrain(S\_k,svm\_label\_S,'kernel\_function','quadratic','showplot',true);

figure, svmStructS2 = svmtrain(S\_k,svm\_label\_S,'kernel\_function','linear','showplot',true);

figure, svmStructS3 = svmtrain(S\_k,svm\_label\_S,'kernel\_function','polynomial','showplot',true);

save svmStruct\_our\_data\_S.mat svmStruct\_S

TESTING

%% --- LOAD DATA FOR TESTING ----

clc;

clear all

close all

%load net\_our\_data.mat

load svmStruct\_our\_data\_A.mat

load svmStruct\_our\_data\_S.mat

load svmStruct\_our\_data\_H.mat

delete svmStruct1.mat

[path,file]=uigetfile('\*.wav','Load the speech signal');

full\_path=strcat(file,path);

[sig\_limited,fs]=audioread(full\_path);

sig\_1000=fs/1000; % maximum speech Fx at 1000Hz

sig\_50=fs/50; % minimum speech Fx at 50Hz

sig\_500=fs/500; % maximum speech Fx at 500Hz

t=(0:length(sig\_limited)-1)/fs;

Y=fft(sig\_limited.\*hamming(length(sig\_limited))); % taking FFT

hz5000=5000\*length(Y)/fs;

f=(0:hz5000)\*fs/length(Y);

C=fft(log(abs(Y)+eps));% cepstrum is DFT of log spectrum

q=(sig\_1000:sig\_50)/fs; % plotting between 1ms (1000Hz) and 20ms (50Hz)

[c,fx]=max(abs(C(sig\_1000:sig\_50)));

r\_corre=xcorr(sig\_limited,sig\_50,'coeff'); % calculate autocorrelation

d\_timw=(-sig\_50:sig\_50)/fs;

r\_corre=r\_corre(sig\_50+1:2\*sig\_50+1);

[rmax,tx]=max(r\_corre(sig\_500:sig\_50));

x=resample(sig\_limited,10000,fs);

fs=10000;

lpf=2+fs/1000; % rule

a=lpc(x,lpf);

[h,f]=freqz(1,a,512,fs);

r=roots(a); % find roots of polynomial a

r=r(imag(r)>0.01); % only look for roots >0Hz up to fs/2

ffreq=sort(atan2(imag(r),real(r))\*fs/(2\*pi));

Features = stFeatureExtraction(sig\_limited, fs, 0.020, 0.020);

F = Features(3,:);

timeFeature = 0.010:0.020:length(x)/fs;

time = 0:1/fs:length(x)/fs-1/fs;

MIN1 = min([length(F);length(timeFeature)]);

timeFeature = timeFeature(1:MIN1);

F = F(1:MIN1);

MIN2 = min([length(x);length(time)]);

time = time(1:MIN2);

x = x(1:MIN2);

%[Frt] = spFormantsLpc(sig\_limited, fs);

Ft=Features';

[mp,np]=size(Ft);

Ft1=Ft(1:round(mp/2),:);

Ft2=Ft(round(mp/2):mp,:);

A1=max(Ft1);

A2=min(Ft1);

A3=mean(Ft1);

A4=std(Ft1);

A7=median(Ft1);

A012=entropy(Ft1);

A10=max(Ft2);

A20=min(Ft2);

A30=mean(Ft2);

A40=std(Ft2);

A70=median(Ft2);

A120=entropy(Ft2);

A201=max(Ft1);

A202=min(Ft1);

A203=mean(Ft1);

A204=std(Ft1);

A207=median(Ft1);

Final\_feat\_test=[A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207 ];

% Final\_feat\_test = Final\_feat\_test/norm( Final\_feat\_test);

clear A1 A2 A3 A4 A7 A012 A10 A20 A30 A40 A70 A120 A201 A202 A203 A204 A207

clear Ft1 Ft2 mp np Ft

%% SVM

%load svmStruct\_our\_data

result\_class\_A = svmclassify(svmStruct\_A, Final\_feat\_test(:,2:3));

result\_class\_S = svmclassify(svmStruct\_S, Final\_feat\_test(:,2:3));

result\_class\_H = svmclassify(svmStruct\_H, Final\_feat\_test(:,2:3));

fprintf('The observation from svm is closely associated with\n')

if result\_class\_A==1

fprintf('\n ANGER! (\*\_\*) \n')

elseif result\_class\_S==1

fprintf('\n SAD! :( \n')

elseif result\_class\_H==1

fprintf('\n HAPPY! :)\n')

else

fprintf('\n NOT HAPPY, SAD OR ANGER\n')

end