APPENDIX
SOFTWARE CODE

A.1 Feature Extraction using (MFCC ,LPC)approaches MATLAB program

i) Allah(moufakham,mouraqeq) for Three readers (sherif,umahamed,umhajer)

```
% Performs testing one 1 sample of the speech database.
clc
clear all
close all
%% Load training and test sets
cAlpha = [{'Large'},{'Small'}];
% Large = ????? ???????
% Small = ????? ???????

load Sherif_All.mat
load Umahmed_All.mat
load Umhajer_All.mat

%% Set variables
NoOfSamples = 20;
Readers = {'Sherif','Umahmed','Umhajer'};
Cases = {'L','S'};

%%Set variable number of coffieceints
%% MFCC parameters
MFCCNo=12;
mfcc = zeros(MFCCNo,size(Readers,2)*NoOfSamples,size(Cases,2));
fs=44100;
%% MFCC Training for all letters
for ii = 1:size(Cases,2);
    for jj = 1:size(Readers,2);
        for kk = 1:NoOfSamples
            file_name = strcat(Readers(jj),'_',Cases(ii),'_',int2str(kk));
            Samples = eval(char(file_name));
            Samples=find(Samples);
            [cepstra,aspectrum,pspectrum] = melfcc(Samples, fs);
            cepstra1=cepstra(2:13,:);
            cepstra1=mean(cepstra1,2);
            mfcc(:,(jj-1)*NoOfSamples+kk,ii)=cepstra1;
        end
    end
end


%% LPC parameters
NoOfLPCFilters = 10;
lpccoeff = zeros(NoOfLPCFilters,size(Readers,2)*NoOfSamples,size(Cases,2));
%% LPC Training for all letters
% Training for all letters
for ii = 1:size(Cases,2);
```

```
    for jj = 1:size(Readers,2);
        for kk = 1:NoOfSamples
            file_name = strcat(Readers(jj),'_',Cases(ii),'_',int2str(kk));
            Samples = eval(char(file_name));
            zz = find(Samples) < max(Samples/3);%Threshold speech regions
            Samples(zz) = 0;
            zz = find(Samples);
            Speech_Region = Samples(zz)/norm(Samples(zz));
%Pre-processing for the training data
             lpccoeff1 = lpc(Speech_Region,NoOfLPCFilters);
            lpccoeff(:,(jj-1)*NoOfSamples+kk,ii) = lpccoeff1(1,2:end)';
        end
    end
end

save('MFCC_allah_L&S.mat','mfcc');
save('LPC_allah_L&S.mat','lpccoeff');
```

ii) Alsudis reader for (moonyلام ,sunnyلام)

```
clc
clear
inpath1= 'C:\Users\user\Desktop\last\files and video projects\1-
9\al_sudes\FE\sample\A\moon\training';
inpath = fullfile(inpath1);
wavefiles = dir(fullfile(inpath1,'*.wav'));
filenum=size(wavefiles,1);
fetrs=[];
char file_names =[];
numm=filenum;
MFCCNo=12;
mfcc = zeros(MFCCNo,numm);

%% MFCC Training for all letters
for ii = 1:numm;
            f2=[];
            filename=fullfile(inpath,wavefiles(ii).name);
            [myfile,fs]=wavread(filename);
            Samples=myfile(:,1);
            zz=find(Samples);
            Samples=Samples(zz);
            [cepstra,aspectrum,pspectrum] = melfcc(Samples, fs);
            cepstra1=cepstra(2:13,:);
            cepstra1=mean(cepstra1,2);
     mfcc(:,ii)=cepstra1;
end
%% LPC parameters
NoOfLPCFilters = 10;
lpccoeff = zeros(NoOfLPCFilters,numm);
%% LPC Training for all letters
% Training for all letters
for ii = 1:numm;
           filename=fullfile(inpath,wavefiles(ii).name);
```

```
           [myfile,fs]=wavread(filename);
           Samples=myfile(:,1);
            zz = find(Samples) < max(Samples/3);%Threshold speech regions
            Samples(zz) = 0;
            zz = find(Samples);
            Speech_Region = Samples(zz)/norm(Samples(zz));
%Pre-processing for the training data
             lpccoeff1 = lpc(Speech_Region,NoOfLPCFilters);
            lpccoeff(:,ii) = lpccoeff1(1,2:end)';
end
%% FFT parameters

FFT_coeff=zeros(32,numm);
%% FFT Training for all letters
% Training for all letters
for ii = 1:numm
filename=fullfile(inpath,wavefiles(ii).name);
[myfile,fs]=wavread(filename);
Samples=myfile(:,1);
zz=find(Samples);
Samples=Samples(zz);
window=500;
noverlap=250;
nfft=64;
h = spectrum.welch('Hamming',window,100*noverlap/window);
hpsd = psd(h,Samples,'NFFT',nfft,'Fs',fs);
Pw = hpsd.Data;
Fw = hpsd.Frequencies;
FFT_coeff(:,ii)=Pw(1:32);
end
FFT_coeff=real(log10(FFT_coeff));
save('Sudis_A_moon.mat','mfcc','lpccoeff','FFT_coeff')
%%
inpath2= 'C:\Users\user\Desktop\last\files and video projects\1-
9\al_sudes\FE\sample\A\sun\training';
inpath2 = fullfile(inpath2);
wavefiles = dir(fullfile(inpath2,'*.wav'));
filenum=size(wavefiles,1);
fetrs=[];
char file_names =[];
numm=filenum;
MFCCNo=12;
mfcc2 = zeros(MFCCNo,numm);

%% MFCC Training for all letters
for ii = 1:numm;
            f2=[];
            filename=fullfile(inpath2,wavefiles(ii).name);
            [myfile,fs]=wavread(filename);
            Samples=myfile(:,1);
            zz=find(Samples);
            Samples=Samples(zz);
            [cepstra,aspectrum,pspectrum] = melfcc(Samples, fs);
            cepstra1=cepstra(2:13,:);
            cepstra1=mean(cepstra1,2);
            mfcc2(:,ii)=cepstra1;
```

```
end
%% LPC parameters
NoOfLPCFilters = 10;
lpccoeff2 = zeros(NoOfLPCFilters,numm);
%% LPC Training for all letters
% Training for all letters
for ii = 1:numm;
           filename=fullfile(inpath2,wavefiles(ii).name);
           [myfile,fs]=wavread(filename);
           Samples=myfile(:,1);
            zz = find(Samples) < max(Samples/3);%Threshold speech regions
            Samples(zz) = 0;
            zz = find(Samples);
            Speech_Region = Samples(zz)/norm(Samples(zz));
%Pre-processing for the training data
             lpccoeff1 = lpc(Speech_Region,NoOfLPCFilters);
            lpccoeff2(:,ii) = lpccoeff1(1,2:end)';
end
%% FFT parameters

FFT_coeff2=zeros(32,numm);
%% FFT Training for all letters
% Training for all letters
for ii = 1:numm
filename=fullfile(inpath2,wavefiles(ii).name);
[myfile,fs]=wavread(filename);
Samples=myfile(:,1);
zz=find(Samples);
Samples=Samples(zz);
 window=500;
noverlap=250;
nfft=64;
h = spectrum.welch('Hamming',window,100*noverlap/window);
hpsd = psd(h,Samples,'NFFT',nfft,'Fs',fs);
Pw = hpsd.Data;
Fw = hpsd.Frequencies;
FFT_coeff2(:,ii)=Pw(1:32);
end
FFT_coeff2=real(log10(FFT_coeff2));
save('Sudis_A_sun.mat','mfcc2','lpccoeff2','FFT_coeff2')
```

File_seg

```
% Performs testing one 1 sample of the speech database.
clc
clear all
close all
inpath= uigetdir('start');
inpath = fullfile(inpath);
wavefiles = dir(fullfile(inpath,'*.wav'));
filenum=size(wavefiles,1);
fetrs=[];
char file_names =[];
MFCCNo=12;
fs=44100;
```

```
file_files=zeros(30000,filenum);
for i=1:length(wavefiles)
    f2=[];
    filename=fullfile(inpath,wavefiles(i).name);
    [myfile,fs]=audioread(filename);
     myfile1=myfile(1:30000);
     file_files(:,i)=myfile1;
end

save('Umhajer_A_sun.mat','file_files')
```

Trimming tool

```
function varargout = trimming_tool(varargin)
% TRIMMING_TOOL MATLAB code for trimming_tool.fig
%      TRIMMING_TOOL, by itself, creates a new TRIMMING_TOOL or raises the
existing
%      singleton*.
%
%      H = TRIMMING_TOOL returns the handle to a new TRIMMING_TOOL or the
handle to
%      the existing singleton*.
%
%      TRIMMING_TOOL('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in TRIMMING_TOOL.M with the given input
arguments.
%
%      TRIMMING_TOOL('Property','Value',...) creates a new TRIMMING_TOOL or
raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before trimming_tool_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to trimming_tool_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help trimming_tool

% Last Modified by GUIDE v2.5 12-Apr-2016 18:48:03

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @trimming_tool_OpeningFcn, ...
                   'gui_OutputFcn',  @trimming_tool_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
```

```
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before trimming_tool is made visible.
function trimming_tool_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to trimming_tool (see VARARGIN)

% Choose default command line output for trimming_tool
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes trimming_tool wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = trimming_tool_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global file
global fs
global outname
global name
global foldername
[filename, foldername] = uigetfile({'*.*'}, 'Select file');
if filename ~= 0
[pathstr,name,ext] = fileparts([foldername filename]);
    FileName=fullfile(foldername,filename);
    [file,fs]=audioread(FileName);
    set(handles.edit1,'String',name)
    file=file(:,1);
```

```
    set(handles.edit2,'String',num2str(length(file)));
    set(handles.edit3,'String',num2str(fs));
    outname=fullfile(foldername,name);
    outname=fullfile([outname,'.wav']);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a
double



% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end




function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a
double



% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
 function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a
double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global file
global fs
sound(file,fs)



function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4 as a
double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a
double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global strt
global endd
global file
global fs
global FileName
strt=str2num(get(handles.edit4,'String'));
endd=str2num(get(handles.edit5,'String'));
sound(file(strt:endd),fs)



% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global strt
global endd
global file
global fs
```

```
global outname
file1=file(strt:endd);
audiowrite( outname,file1,fs);
```

A.2 Neural network(NN) approaches MATLAB program

A.2.1 Input/Output MFCC_Allah.(sheriff,umahmed,umhajer)

```
%% two NN ,one for man allah and one for woman
%  each NN for detection allah large or small
%

 load('MFCC_allah_L&S.mat')                                  %load coefficients
matrix
 mfcc_man_input_allah=zeros(12,40);
 mfcc_woman_input_allah=zeros(12,80);
 %% rearrange coefficients
 temp=mfcc(:,1:20,1);
 mfcc_man_input_allah(:,1:20)=temp;
 mfcc_man_input_allah(:,21:40)=mfcc(:,1:20,2);
 mfcc_woman_input_allah(:,1:40)=mfcc(:,21:60,1);
 mfcc_woman_input_allah(:,41:80)=mfcc(:,21:60,2);
 mfcc_man_output_allah=zeros(40,2);
 mfcc_woman_output_allah=zeros(80,2);
 mfcc_man_output_allah(1:20,1)=1;
 mfcc_man_output_allah(21:40,2)=1;
 mfcc_woman_output_allah(1:40,1)=1;
 mfcc_woman_output_allah(41:80,2)=1;

save('mfcc_man_input_allah.mat','mfcc_man_input_allah');
save('mfcc_woman_input_allah.mat','mfcc_woman_input_allah');
save('mfcc_man_output_allah.mat','mfcc_man_output_allah');
save('mfcc_woman_output_allah.mat','mfcc_woman_output_allah');
```

A.2.2 Input/Output LPC_Allah.(sheriff,umahmed,umhajer)

```
%% two NN ,one for man allah and one for woman
%  each NN for detection allah large or small
%

 load('LPC_allah_L&S.mat')                                  %load coefficients
matrix
 lpc_man_input_allah=zeros(10,40);
 lpc_woman_input_allah=zeros(10,80);
 %% rearrange coefficients
 temp=lpccoeff(:,1:20,1);
 lpc_man_input_allah(:,1:20)=temp;
 lpc_man_input_allah(:,21:40)=lpccoeff(:,1:20,2);
 lpc_woman_input_allah(:,1:40)=lpccoeff(:,21:60,1);
 lpc_woman_input_allah(:,41:80)=lpccoeff(:,21:60,2);
 lpc_man_output_allah=zeros(40,2);
 lpc_woman_output_allah=zeros(80,2);
 lpc_man_output_allah(1:20,1)=1;
```

```
 lpc_man_output_allah(21:40,2)=1;
 lpc_woman_output_allah(1:40,1)=1;
 lpc_woman_output_allah(41:80,2)=1;

save('lpc_man_input_allah.mat','lpc_man_input_allah');
save('lpc_woman_input_allah.mat','lpc_woman_input_allah');
save('lpc_man_output_allah.mat','lpc_man_output_allah');
save('lpc_woman_output_allah.mat','lpc_woman_output_allah');
```

A.2.3 Input/Output LPC_Optmization.(sheriff,umahmed,umhajer)

```
clc
load('lpc_man_input_A.mat')
load('lpc_woman_input_A.mat')
lpc_man_input_A_tmp=zeros(size(lpc_man_input_A));
lpc_woman_input_A_tmp=zeros(size(lpc_woman_input_A));
lpc_man_input_A1=log10(lpc_man_input_A(2:end,:));
lpc_man_input_A1=real(lpc_man_input_A1);
lpc_woman_input_A2=log10(lpc_woman_input_A(2:end,:));
lpc_woman_input_A2=real(lpc_woman_input_A2);
lpc_man_input_A_tmp(1,:)=lpc_man_input_A(1,:);
lpc_man_input_A_tmp(2:end,:)=lpc_man_input_A1;
lpc_man_input_A=lpc_man_input_A_tmp;
lpc_woman_input_A_tmp(1,:)=lpc_woman_input_A(1,:);
lpc_woman_input_A_tmp(2:end,:)=lpc_woman_input_A2;
lpc_woman_input_A=lpc_woman_input_A_tmp;
save('lpc_man_input_A.mat','lpc_man_input_A');
save('lpc_woman_input_A.mat','lpc_woman_input_A')
```

A.2.4 training code : (sheriff,umahmed,umhajer)
Train Function : Train Scg/ Perform Function : mse

```
% This script assumes these variables are defined:
%   input - input data.
%   target - target data.

load('lpc_woman_input_allah.mat');
load('lpc_woman_output_allah.mat');

inputs = lpc_woman_input_allah';
targets = transpose(lpc_woman_input_allah);

% Create a   Network
hiddenLayerSize = 25;
net_allah_woman_LPC = newff(inputs,targets,hiddenLayerSize);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net_allah_woman_LPC.inputs{1}.processFcns =
{'removeconstantrows','mapminmax'};
```

```
net_allah_woman_LPC.outputs{2}.processFcns =
{'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
net_allah_woman_LPC.divideFcn = 'dividerand';  % Divide data randomly
net_allah_woman_LPC.divideMode = 'sample';  % Divide up every sample
net_allah_woman_LPC.divideParam.trainRatio = 80/100;
net_allah_woman_LPC.divideParam.testRatio = 20/100;

net_allah_woman_LPC.trainFcn = 'trainscg';


net_allah_woman_LPC.performFcn = 'mse';

% Choose Plot Functions

net_allah_woman_LPC.plotFcns = {'plotperform','plottrainstate','ploterrhist',
...
  'plotregression', 'plotfit'};

net_allah_woman_LPC.trainParam.max_fail = 6;
net_allah_woman_LPC.trainParam.min_grad=1e-5;
net_allah_woman_LPC.trainParam.show=10;
net_allah_woman_LPC.trainParam.lr=0.9;
net_allah_woman_LPC.trainParam.epochs=1000;
net_allah_woman_LPC.trainParam.goal=0.00;


% Train the Network
net_allah_woman_LPC = train(net_allah_woman_LPC,inputs,targets);

% Test the Network
outputs = net_allah_woman_LPC(inputs);
errors = gsubtract(targets,outputs);
performance = perform(net_allah_woman_LPC,targets,outputs);

y1 = sim(net_allah_woman_LPC,inputs);
y1=abs(y1);
y1=round(y1);

save ('net_allah_woman_LPC.mat','net_allah_woman_LPC');
```

A.2.5 patternet MFCC_man_input_allah (Alsudies reader)

```
clc
clear all
%%%%%% Net for man%%%%%%%%%%%%%
load('mfcc_man_input_allah.mat');
load('mfcc_man_output_allah.mat');
```

```
inputs2 = mfcc_man_input_allah;
targets2 =mfcc_man_output_allah';
net2 = patternnet(15);
perf2=1;
while perf2>0.05
net2 = train(net2,inputs2,targets2);
y2 = net2(inputs2);
perf2 = perform(net2,targets2,y2)
classes = vec2ind(y2);
end
[cc2,i2]=max(y2)
net_allah_man_MFCC=net2;
save ('net1_allah_man_MFCC.mat','net_allah_man_MFCC');
```

A.2.6 patternet Lpc_man_input_allah (Alsudies reader)

```
clc
clear all
load('lpc_man_input_allah.mat');
load('lpc_man_output_allah.mat');
inputs = lpc_man_input_allah;
targets =lpc_man_output_allah';
net1 = patternnet(10);
perf=1;
while perf>0.28
net1 = train(net1,inputs,targets);
y = net1(inputs);
perf = perform(net1,targets,y)
classes = vec2ind(y);
end
[cc,i]=max(y);
net_allah_man_LPC=net1;
save ('net1_allah_man_LPC.mat','net_allah_man_LPC');
```

A.2.7 patternet net_allah_ LPC(man,woman)

```
clc
clear all
load('lpc_man_input_allah.mat');
load('lpc_man_output_allah.mat');
inputs = lpc_man_input_allah;
targets =lpc_man_output_allah';
net1 = patternnet(15);
perf=1;
while perf>0.25
net1 = train(net1,inputs,targets);
y = net1(inputs);
perf = perform(net1,targets,y)
classes = vec2ind(y);
end
[cc,i]=max(y);
net_allah_man_LPC=net1;
```

```
save ('net_allah_man_LPC.mat','net_allah_man_LPC');
%%%%%%net for woman%%%%%%%%%%%%
load('lpc_woman_input_allah.mat');
load('lpc_woman_output_allah.mat');
inputs2 = lpc_woman_input_allah;
targets2 =lpc_woman_output_allah';
net2= patternnet(15);
perf2=1;
while perf2>0.25
net2 = train(net2,inputs2,targets2);
y2 = net2(inputs2);
perf2 = perform(net2,targets2,y2)
classes = vec2ind(y2);
end
[cc2,i2]=max(y2);
net_allah_woman_LPC=net2;
save ('net_allah_woman_LPC.mat','net_allah_woman_LPC');
```

A.2.8 patternet MFCC_man_input_A (Alhuzafi reader)

```
clc
clear all
%%%%%% Net for man%%%%%%%%%%%%%
load('mfcc_man_input_A.mat');
load('mfcc_man_output_A.mat');
inputs2 = mfcc_man_input_allah;
targets2 =mfcc_man_output_allah';
net2 = patternnet(15);
perf2=1;
while perf2>0.05
net2 = train(net2,inputs2,targets2);
y2 = net2(inputs2);
perf2 = perform(net2,targets2,y2)
classes = vec2ind(y2);
end
[cc2,i2]=max(y2)
net_A_man_MFCC=net2;
save ('net2_A_man_MFCC.mat','net_A_man_MFCC');
```

A.2.9 patternet Lpc_man_A (Alhuzafi reader)

```
clear all
load('lpc_man_input_A.mat');
load('lpc_man_output_A.mat');
inputs = lpc_man_input_allah;
targets =lpc_man_output_allah';
net1 = patternnet(10);
perf=1;
while perf>0.3
net1 = train(net1,inputs,targets);
y = net1(inputs);
```

```
perf = perform(net1,targets,y)
classes = vec2ind(y);
end
[cc,i]=max(y);
net_A_man_LPC=net1;
save ('net2_A_man_LPC.mat','net_A_man_LPC');
```

A.2.10 Huzaifi_A_moon_testing

```
clc
load('net2_A_man_LPC.mat')
load ('Huzaifi_A_moon_testing.mat')
hh1=zeros(10,1);
for k=1:10
    inputs=lpccoeff(:,k);
    y1 = net_A_man_LPC(inputs);
    y1=abs(y1);
    [c,i]=max(y1);
    hh1(k,1)=i;
end
 accuracy1=((numel(find(hh1==1)))/10)*100
load ('Huzaifi_A_sun_testing.mat')
hh2=zeros(10,1);
for k=1:10
    inputs=lpccoeff2(:,k);
    y1 = net_A_man_LPC(inputs);
    y1=abs(y1);
    [c,i]=max(y1);
    hh2(k,1)=i;
end
   accuracy2=((numel(find(hh2==2)))/10)*100
```

A.2.11 Sharif_allah_Large _testing

```
clc
load('net_allah_man_MFCC.mat')
load ('Sharif_allah_Large.mat')
hh=zeros(10,2);
for k=1:10
    inputs=lpccoeff(:,k);
    y1 = net_allah_man_MFCC(inputs);
    y1=abs(y1);
    hh(k,:)=y1;
end
   hh
```

A.3 hmm_quantize_trainingdata

```
clc
clear
states=[3 4 5 6 7 8 9 10];
addpath(genpath('C:\Program Files\MATLAB\R2014b\toolbox\HMMall'))
load('Huzaifi_allah_large_training.mat')
load('Huzaifi_allah_small_training.mat')
```

```
%%%%%%HMM FOR allah man larg %%%%%%%%%
for ll=1:numel(states)
    stat=states(ll);
    disp(['&&&&&&&&&& states = ',num2str(stat),'  &&&&&&&&&&'])
data1=mfcc;
all_data1=[mfcc mfcc2];
 [codebook1]=codebook_gen(all_data1);
 save('Allah_sudes_codebook','codebook1')
 quantized_data1=zeros(size(data1));
 for h=1:size(data1,2)
     datavec=data1(:,h)';
     [quantizedvec]=vectore_quant(codebook1,datavec);
     quantized_data1(:,h)=quantizedvec';
 end
num_codebook1=length(codebook1);
quantized_data1=flipud(quantized_data1);
hmmprior1 = zeros(1,stat);
hmmprior1(1,1)=1;
hmmtransmat1 = rand(stat,stat); % 5 by 5 transition matrix
hmmtransmat1=triu(hmmtransmat1);

for j=1:stat
    for k=1:stat
        if ((k-j)>1)
            hmmtransmat1(j,k)=0;
        end
    end
end
hmmtransmat1 = mk_stochastic(hmmtransmat1);
hmmobsmat1 = rand(stat,num_codebook1); % # of states * # of observation
hmmobsmat1 = mk_stochastic(hmmobsmat1);
[LL0, hmmprior_allah_sudes_large_MFCC, hmmtransmat_allah_sudes_large_MFCC,
hmmobsmat_allah_sudes_large_MFCC] = dhmm_em(quantized_data1',hmmprior1,
hmmtransmat1, hmmobsmat1, 'max_iter', 20);
save('HMM_allah_sudes_large_MFCC.mat','hmmprior_allah_sudes_large_MFCC'
,'hmmtransmat_allah_sudes_large_MFCC', 'hmmobsmat_allah_sudes_large_MFCC' )

% disp(['######evaluation proccess###### '])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for dt=1:size(quantized_data1,2)
    loglike0 = dhmm_logprob(quantized_data1(:,dt)',
hmmprior_allah_sudes_large_MFCC,hmmtransmat_allah_sudes_large_MFCC,hmmobsmat_
allah_sudes_large_MFCC);
   % disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path0 = zeros(size(quantized_data1));
for dtt =1:size(quantized_data1,2)
B =
multinomial_prob(quantized_data1(:,dtt)',hmmobsmat_allah_sudes_large_MFCC);
path0(:,dtt) =
viterbi_path(hmmprior_allah_sudes_large_MFCC,hmmtransmat_allah_sudes_large_MF
CC, B);
%disp(sprintf('%d', path0(:,dtt)));
end
```

```
%disp(['#######################################################################
##########'])
%%%%%%HMM FOR allah man SMALL %%%%%%%%%
 data2=mfcc2;
 quantized_data2=zeros(size(data2));
 for h=1:size(data2,2)
     datavec=data2(:,h)';
     [quantizedvec]=vectore_quant(codebook1,datavec);
     quantized_data2(:,h)=quantizedvec';
 end
quantized_data2=flipud(quantized_data2);
hmmprior2 = zeros(1,stat);
hmmprior2(1,1)=1;
hmmtransmat2 = rand(stat,stat); % 3 by 3 transition matrix
hmmtransmat2=triu(hmmtransmat2);

for j=1:stat
    for k=1:stat
        if ((k-j)>1)
            hmmtransmat2(j,k)=0;
        end
    end
end
hmmtransmat2 = mk_stochastic(hmmtransmat2);
hmmobsmat2 = rand(stat, num_codebook1); % # of states * # of observation
hmmobsmat2 = mk_stochastic(hmmobsmat2);
[LL0, hmmprior_allah_sudes_small_MFCC, hmmtransmat_allah_sudes_small_MFCC,
hmmobsmat_allah_sudes_small_MFCC] = dhmm_em(quantized_data2',hmmprior2,
hmmtransmat2, hmmobsmat2, 'max_iter', 20);
save('HMM_allah_sudes_small_MFCC.mat','hmmprior_allah_sudes_small_MFCC'
,'hmmtransmat_allah_sudes_small_MFCC', 'hmmobsmat_allah_sudes_small_MFCC' )
%disp(['#######evaluation proccess########## '])
%%%%%%%%%%%%%%%%%%##########################
for dt=1:size(quantized_data2,2)
    loglike0 = dhmm_logprob(quantized_data2(:,dt)',
hmmprior_allah_sudes_small_MFCC,hmmtransmat_allah_sudes_small_MFCC,hmmobsmat_
allah_sudes_small_MFCC);
   % disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path1 = zeros(size(quantized_data2));
for dtt =1:size(quantized_data2,2)
B =
multinomial_prob(quantized_data2(:,dtt)',hmmobsmat_allah_sudes_small_MFCC);
path1(:,dtt) =
viterbi_path(hmmprior_allah_sudes_small_MFCC,hmmtransmat_allah_sudes_small_MF
CC, B);
%disp(sprintf('%d', path1(:,dtt)));
end
%disp(['#######################################################################
##########'])

%%%%%%HMM FOR allah LPC %%%%%%%%%
%MAN LPC LARGE
data3=lpccoeff;
all_data2=[lpccoeff lpccoeff2];
```

```
[codebook2]=codebook_gen(all_data2);
 save('Allah_sudes_codebook_LPC.mat','codebook2')
quantized_data3=zeros(size(data3));
 for h=1:size(data3,2)
     datavec=data3(:,h)';
     [quantizedvec]=vectore_quant(codebook2,datavec);
     quantized_data3(:,h)=quantizedvec';
 end
quantized_data3=flipud(quantized_data3);
num_codebook2=length(codebook2);
hmmprior3 = zeros(1,stat);
hmmprior3(1,1)=1;
hmmtransmat3 = rand(stat,stat); % 3 by 3 transition matrix
hmmtransmat3=triu(hmmtransmat3);

for j=1:stat
    for k=1:stat
        if ((k-j)>1)
            hmmtransmat3(j,k)=0;
        end
    end
end
hmmtransmat3 = mk_stochastic(hmmtransmat3);
hmmobsmat3 = rand(stat, num_codebook2); % # of states * # of observation
hmmobsmat3 = mk_stochastic(hmmobsmat3);
[LL0, hmmprior_allah_sudes_large_LPC, hmmtransmat_allah_sudes_large_LPC,
hmmobsmat_allah_sudes_large_LPC] = dhmm_em(quantized_data3',hmmprior3,
hmmtransmat3, hmmobsmat3, 'max_iter', 20);
save('HMM_allah_sudes_large_LPC.mat','hmmprior_allah_sudes_large_LPC'
,'hmmtransmat_allah_sudes_large_LPC', 'hmmobsmat_allah_sudes_large_LPC' )
%disp(['############evaluation proccess###########'])
%%%%%%%%%%%%%%%%%%%%############################
for dt=1:size(quantized_data3,2)
    loglike0 = dhmm_logprob(quantized_data3(:,dt)',
hmmprior_allah_sudes_large_LPC,hmmtransmat_allah_sudes_large_LPC,hmmobsmat_al
lah_sudes_large_LPC);
    %disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path2 = zeros(size(quantized_data3));
for dtt =1:size(quantized_data3,2)
B =
multinomial_prob(quantized_data3(:,dtt)',hmmobsmat_allah_sudes_large_LPC);
path2(:,dtt) =
viterbi_path(hmmprior_allah_sudes_large_LPC,hmmtransmat_allah_sudes_large_LPC
, B);
%disp(sprintf('%d', path2(:,dtt)));
end
%
disp(['#########################################################################
#########'])

%%%%%%HMM FOR allah man Smal LPC %%%%%%%%%
 data4=lpccoeff2;
 quantized_data4=zeros(size(data4));
 for h=1:size(data4,2)
```

```
     datavec=data4(:,h)';
     [quantizedvec]=vectore_quant(codebook2,datavec);
     quantized_data4(:,h)=quantizedvec';
 end
quantized_data4=flipud(quantized_data4);
hmmprior4 = zeros(1,stat);
hmmprior4(1,1)=1;
hmmtransmat4 = rand(stat,stat); % 3 by 3 transition matrixhmmtransmat4 =
mk_stochastic(hmmtransmat4);
hmmtransmat4=triu(hmmtransmat4);
for j=1:stat
    for k=1:stat
        if ((k-j)>1)
            hmmtransmat4(j,k)=0;
        end
    end
end
hmmobsmat4 = rand(stat, num_codebook2); % # of states * # of observation
hmmobsmat4 = mk_stochastic(hmmobsmat4);
[LL0, hmmprior_allah_sudes_small_LPC, hmmtransmat_allah_sudes_small_LPC,
hmmobsmat_allah_sudes_small_LPC] = dhmm_em(quantized_data4',hmmprior4,
hmmtransmat4, hmmobsmat4, 'max_iter', 20);
save('HMM_allah_sudes_small_LPC.mat','hmmprior_allah_sudes_small_LPC'
,'hmmtransmat_allah_sudes_small_LPC', 'hmmobsmat_allah_sudes_small_LPC' )

% disp(['################################################'])
% disp(['#######evaluation proccess######## '])
%%%%%%%%%%%%%%%%%%#########################
for dt=1:size(quantized_data4,2)
    loglike0 = dhmm_logprob(quantized_data4(:,dt)',
hmmprior_allah_sudes_small_LPC,hmmtransmat_allah_sudes_small_LPC,hmmobsmat_al
lah_sudes_small_LPC);
   % disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path3 = zeros(size(quantized_data4));
for dtt =1:size(quantized_data4,2)
B =
multinomial_prob(quantized_data4(:,dtt)',hmmobsmat_allah_sudes_small_LPC);
path3(:,dtt) =
viterbi_path(hmmprior_allah_sudes_small_LPC,hmmtransmat_allah_sudes_small_LPC
, B);
%disp(sprintf('%d', path3(:,dtt)));
end
%% HMM for allah large man FFT
 data5=FFT_coeff;
 all_data3=[FFT_coeff  FFT_coeff2];
[codebook3]=codebook_gen(all_data3);
num_codebook3=length(codebook3);
 save('Allah_sudes_codebook_FFT','codebook3')
 quantized_data5=zeros(size(data5));
 for h=1:size(data5,2)
     datavec=data5(:,h)';
     [quantizedvec]=vectore_quant(codebook3,datavec);
     quantized_data5(:,h)=quantizedvec';
 end
quantized_data5=flipud(quantized_data5);
```

```
hmmprior5 = zeros(1,stat);
hmmprior5(1,1)=1;
hmmtransmat5 = rand(stat,stat); % 3 by 3 transition matrixhmmtransmat4 =
mk_stochastic(hmmtransmat4);
hmmtransmat5=triu(hmmtransmat5);

for j=1:stat
    for k=1:stat
        if ((k-j)>1)
            hmmtransmat5(j,k)=0;
        end
    end
end
hmmobsmat5 = rand(stat, num_codebook3); % # of states * # of observation
hmmobsmat5 = mk_stochastic(hmmobsmat5);
[LL0, hmmprior_allah_sudes_large_FFT, hmmtransmat_allah_sudes_large_FFT,
hmmobsmat_allah_sudes_large_FFT] = dhmm_em(quantized_data5',hmmprior5,
hmmtransmat5, hmmobsmat5, 'max_iter', 20);
save('HMM_allah_sudes_large_FFT.mat','hmmprior_allah_sudes_large_FFT'
,'hmmtransmat_allah_sudes_large_FFT', 'hmmobsmat_allah_sudes_large_FFT' )

%disp(['################################################'])
%disp(['#######evaluation proccess######## '])
%%%%%%%%%%%%%%%%%%#########################
for dt=1:size(quantized_data5,2)
    loglike0 = dhmm_logprob(quantized_data5(:,dt)',
hmmprior_allah_sudes_large_FFT,hmmtransmat_allah_sudes_large_FFT,hmmobsmat_al
lah_sudes_large_FFT);
   % disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path5 = zeros(size(quantized_data5));
for dtt =1:size(quantized_data5,2)
B =
multinomial_prob(quantized_data5(:,dtt)',hmmobsmat_allah_sudes_large_FFT);
path5(:,dtt) =
viterbi_path(hmmprior_allah_sudes_large_FFT,hmmtransmat_allah_sudes_large_FFT
, B);
%disp(sprintf('%d', path5(:,dtt)));
end
%% HMM for allah small man FFT
data6=FFT_coeff2;
 quantized_data6=zeros(size(data6));
 for h=1:size(data6,2)
     datavec=data6(:,h)';
     [quantizedvec]=vectore_quant(codebook3,datavec);
     quantized_data6(:,h)=quantizedvec';
 end
quantized_data6=flipud(quantized_data6);
hmmprior6 = zeros(1,stat);
hmmprior6(1,1)=1;
hmmtransmat6 = rand(stat,stat); % 5 by 5 transition
hmmtransmat6=triu(hmmtransmat6);

for j=1:stat
    for k=1:stat
```

```
        if ((k-j)>1)
            hmmtransmat6(j,k)=0;
        end
    end
end
hmmobsmat6 = rand(stat, num_codebook3); % # of states * # of observation
hmmobsmat6 = mk_stochastic(hmmobsmat6);
[LL0, hmmprior_allah_sudes_small_FFT, hmmtransmat_allah_sudes_small_FFT,
hmmobsmat_allah_sudes_small_FFT] = dhmm_em(quantized_data6',hmmprior6,
hmmtransmat6, hmmobsmat6, 'max_iter', 20);
save('HMM_allah_sudes_small_FFT.mat','hmmprior_allah_sudes_small_FFT'
,'hmmtransmat_allah_sudes_small_FFT', 'hmmobsmat_allah_sudes_small_FFT' )

%disp(['################################################'])
%disp(['#######evaluation proccess######## '])
%%%%%%%%%%%%%%%%%%%%############################
for dt=1:size(quantized_data6,2)
    loglike0 = dhmm_logprob(quantized_data6(:,dt)',
hmmprior_allah_sudes_small_FFT,hmmtransmat_allah_sudes_small_FFT,hmmobsmat_al
lah_sudes_small_FFT);
   % disp(['likehood for sample ',num2str(dt),' = ' ,num2str( loglike0)])
end
%%%Viterbi decoding%%%
path5 = zeros(size(quantized_data6));
for dtt =1:size(quantized_data6,2)
B =
multinomial_prob(quantized_data6(:,dtt)',hmmobsmat_allah_sudes_small_FFT);
path5(:,dtt) =
viterbi_path(hmmprior_allah_sudes_small_FFT,hmmtransmat_allah_sudes_small_FFT
, B);
%disp(sprintf('%d', path5(:,dtt)));
end
%%
hmm_quantize_testingdata_MFCC
hmm_quantize_testingdata_LPC
hmm_quantize_testingdata_FFT
end
```

A.3.1 codebook_gen

```
function [codebook]=codebook_gen(data)

data2=reshape(data,[size(data,1)*size(data,2),1]);
minn=min(data2);
maxx=max(data2);
num_of_levels=300;
u=linspace(minn,maxx,num_of_levels);
u=fliplr(u);
 codebook=u;
end
```

A.3.2 vectore_quant

```
function[quantizedvec]=vectore_quant(codebook,datavec)
```

```
    partions=codebook(1:end-1);
    partions=sort(partions);
    [index,quantized] = quantiz(datavec,partions,codebook);
    for j=1:length(index)
        if index(j)==0
            index(j)=length(index);
     end
    quantizedvec=index;
    end
end
```

A.3.3 hmm_quantize_testingdata_MFCC

```
load('Huzaifi_allah_Large_testing.mat')
load('Huzaifi_allah_small_testing.mat')
load('Allah_sudes_codebook.mat')

 data=mfcc;
 quantized_data=zeros(size(data));
 for h=1:size(data,2)
     datavec=data(:,h)';
     [quantizedvec]=vectore_quant(codebook1,datavec);
     quantized_data(:,h)=quantizedvec';
 end
quantized_data=flipud(quantized_data);

%%%%%%%%%%%%%%%%%%%##########################
load('HMM_allah_sudes_large_MFCC.mat');
load('HMM_allah_sudes_small_MFCC.mat');
true=0;
true2=0;
disp(['%%%%% accuracy for MFCC %%%%%'])

for dt=1:size(quantized_data,2)
    loglike0 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_large_MFCC,hmmtransmat_allah_sudes_large_MFCC,hmmobsmat_
allah_sudes_large_MFCC);
    loglike2 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_small_MFCC,hmmtransmat_allah_sudes_small_MFCC,hmmobsmat_
allah_sudes_small_MFCC);
   %disp(['likehood for sample for large man ',num2str(dt),' = ' ,num2str(
loglike0)])
    %disp(['likehood for sample for small man',num2str(dt),' = ' ,num2str(
loglike2)])
    %disp(['####################'])
     if (loglike0>loglike2)
        true=true+1;
    end
end
accu=(true/dt)*100;
disp(['accuracy for large  = ' ,num2str( accu)])
```

```
data2=mfcc2;
 quantized_data=zeros(size(data2));
 for h=1:size(data2,2)
     datavec=data2(:,h)';
     [quantizedvec]=vectore_quant(codebook1,datavec);
     quantized_data(:,h)=quantizedvec';
 end
quantized_data=flipud(quantized_data);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for dt=1:size(quantized_data,2)
    loglike0 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_large_MFCC,hmmtransmat_allah_sudes_large_MFCC,hmmobsmat_
allah_sudes_large_MFCC);
    loglike2 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_small_MFCC,hmmtransmat_allah_sudes_small_MFCC,hmmobsmat_
allah_sudes_small_MFCC);
    %disp(['likehood for sample for large man ',num2str(dt),' = ' ,num2str(
loglike0)])
    %disp(['likehood for sample for small man',num2str(dt),' = ' ,num2str(
loglike2)])
    %disp(['####################'])
     if (loglike2>loglike0)
        true2=true2+1;
    end
end
accu2=(true2/dt)*100;
disp(['accuracy for small  = ' ,num2str( accu2)])
```

A.3.4 hmm_quantize_testingdata_LPC

```
load('Huzaifi_allah_Large_testing.mat')
load('Huzaifi_allah_small_testing.mat')
load('Allah_sudes_codebook_LPC.mat')
 data=lpccoeff;
 quantized_data=zeros(size(data));
 for h=1:size(data,2)
     datavec=data(:,h)';
     [quantizedvec]=vectore_quant(codebook2,datavec);
     quantized_data(:,h)=quantizedvec';
 end
quantized_data=flipud(quantized_data);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
load('HMM_allah_sudes_large_LPC.mat')
load('HMM_allah_sudes_small_LPC.mat')
true=0;
true2=0;
disp(['%%%%% accuracy for LPC %%%%%'])

for dt=1:size(quantized_data,2)
```

```
    loglike0 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_large_LPC,hmmtransmat_allah_sudes_large_LPC,hmmobsmat_al
lah_sudes_large_LPC);
    loglike2 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_small_LPC,hmmtransmat_allah_sudes_small_LPC,hmmobsmat_al
lah_sudes_small_LPC);
   % disp(['likehood for sample for large man ',num2str(dt),' = ' ,num2str(
loglike0)])
    %disp(['likehood for sample for small man',num2str(dt),' = ' ,num2str(
loglike2)])
    %disp(['####################'])
     if (loglike0>loglike2)
        true=true+1;
    end
end
accu=(true/dt)*100;
disp(['accuracy for large  = ' ,num2str( accu)])

data2=lpccoeff2;
 quantized_data=zeros(size(data2));
 for h=1:size(data2,2)
     datavec=data2(:,h)';
     [quantizedvec]=vectore_quant(codebook1,datavec);
     quantized_data(:,h)=quantizedvec';
 end
quantized_data=flipud(quantized_data);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for dt=1:size(quantized_data,2)
    loglike0 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_large_LPC,hmmtransmat_allah_sudes_large_LPC,hmmobsmat_al
lah_sudes_large_LPC);
    loglike2 = dhmm_logprob(quantized_data(:,dt)',
hmmprior_allah_sudes_small_LPC,hmmtransmat_allah_sudes_small_LPC,hmmobsmat_al
lah_sudes_small_LPC);
    %disp(['likehood for sample for small man',num2str(dt),' = ' ,num2str(
loglike2)])
    %disp(['####################'])
     if (loglike2>loglike0)
        true2=true2+1;
    end
end
accu2=(true2/dt)*100;
disp(['accuracy for small  = ' ,num2str( accu2)])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A.4 Test Code Search Allah(Moufakhum,mouraqeq)

```
[filename, foldername] = uigetfile({'*.*'}, 'Select file');
if filename ~= 0
[pathstr,name,ext] = fileparts([foldername filename]);
```

```
    FileName=fullfile(foldername,filename);
    [file,fs]=audioread(FileName);
    Speech=file(:,1);
else
    return
 end
file_lnth=length(Speech);
WindowSize=88200;
NoOfWindows=floor(file_lnth/WindowSize);
if NoOfWindows>1
shiftingratio=0.2;
NoOfSamples=(NoOfWindows-1)*(1/shiftingratio);
shiftingsize=floor(WindowSize*shiftingratio);
xx=zeros(WindowSize,NoOfSamples);
     for n= 0:NoOfSamples-1
                st=n*shiftingsize+1;
                bb = Speech(st:st+WindowSize-1);
                xx(:,n+1)=bb;
     end
else
    xx=Speech;
end
mfcc_for_allSamples=zeros(12,NoOfSamples);
for h=1:size(xx,2)
            Samples=find(xx(:,h));
            if isempty(Samples)
                cepstra1=ones(12,1);
            else
            [cepstra,aspectrum,pspectrum] = melfcc(Samples, fs);
            cepstra1=cepstra(2:13,:);
            cepstra1=mean(cepstra1,2);
             mfcc_for_allSamples(:,h)=cepstra1;
            end
end
```