Indian Institute of Technology, Guwahati

Department of Computer Science and Engineering

Project report

On

"Speaking Calculator"

Based on

Speech recognition system

Course: CS566 Speech Processing

Submitted to
Prof. P. K. Das

Submitted by:
Prosenjit Biswas(214101038)

Souvik Gorai(214101055)

TABLE OF CONTENT

ABSTRACT

The Speech is most prominent & one of the natural forms of communication among of human being. The speech is a signal of infinite information. There are different aspects related to speech like speech recognition, speech verification, speech synthesis, speaker recognition, speaker identification etc. The speaking calculator is a promising implementation of speech-recognition technology, which is designed to recognise numbers and symbols to perform mathematical operations. The development includes an extensive study of Hidden Markov Model (HMM) which is currently the state of art in the field of speech recognition. This calculator might be used for the people with functional disability.

INTRODUCTION

Speech recognition is a topic that is very useful in many applications and environments in our daily life. In this report we concentrate on the speech recognition programs that are human-computer interactive. When software evaluators observe humans testing such software programs, they gain valuable insights into technological problems and barriers that they may never witness otherwise. . Testing speech recognition products for universal usability is an important step before considering the product to be a viable solution for its customers later. This document concerns Speech Recognition accuracy in recognising symbols and digits, which is a critical factor in the development of hands-free human- machine interactive devices. There are two separate issues that we want to test: word recognition accuracy and software friendliness.Major factors that impede recognition accuracy in the environment noise sources and system noise**.**

But, what is speech recognition?

Speech recognition works like this. You speak into a microphone and the computer transforms the sound of your words into text to be used by your word processor or other applications available on your computer. The computer may repeat what you just said or it may give you a prompt for what you are expected to say next. This is the central promise of interactive speech recognition. You also had to correct any errors virtually as soon as they happened, which means that you had to concentrate so hard on the software that you often forgot what you were trying to say.

The new voice recognition systems are certainly much easier to use. You can speak at a normal pace without leaving distinct pauses between words. However, you cannot really use "*natural speech*" as claimed by the manufacturers. You must speak clearly, as you do when you speak to a Dictaphone or when you leave someone a telephone message. Remember, the computer is relying solely on your spoken words. It cannot interpret your tone or inflection, and it cannot interpret your gestures and facial expressions, which are part of everyday human communication. Some of the systems also look at whole phrases, not just the individual words you speak. They try to get information from the context of your speech, to help work out the correct interpretation

The goal of this project is to define a set of evaluation criteria and test methods for the interactive voice recognition systems for searching symbols and digits and retrieving corresponding details for successful search .

PROPOSED METHODOLOGY

Basic requirements to develop this project are as follows:

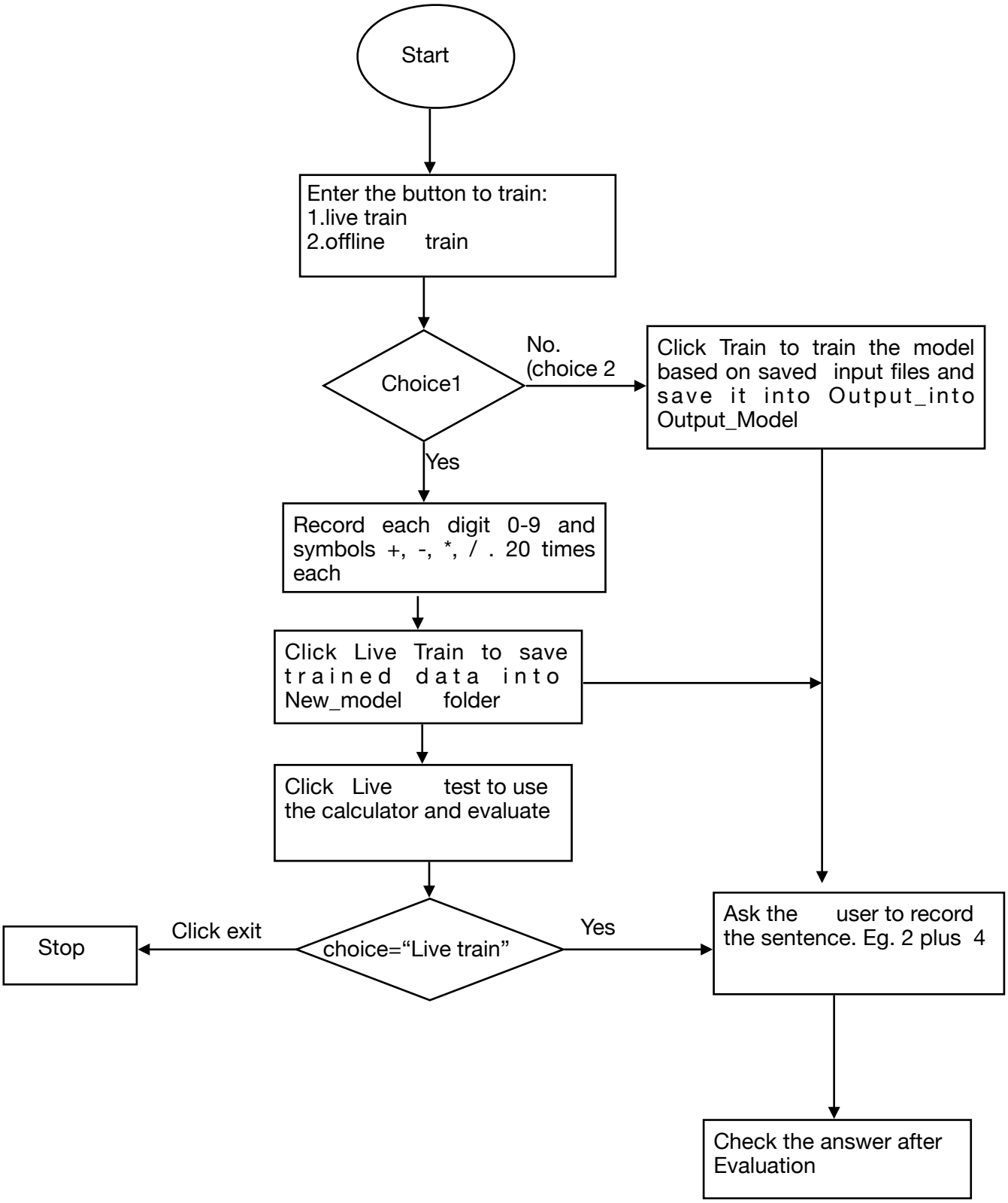
- Windows OS
- Microsoft Visual Studio 2010
- C++ 11 integrated with VS2010
- Recording Module

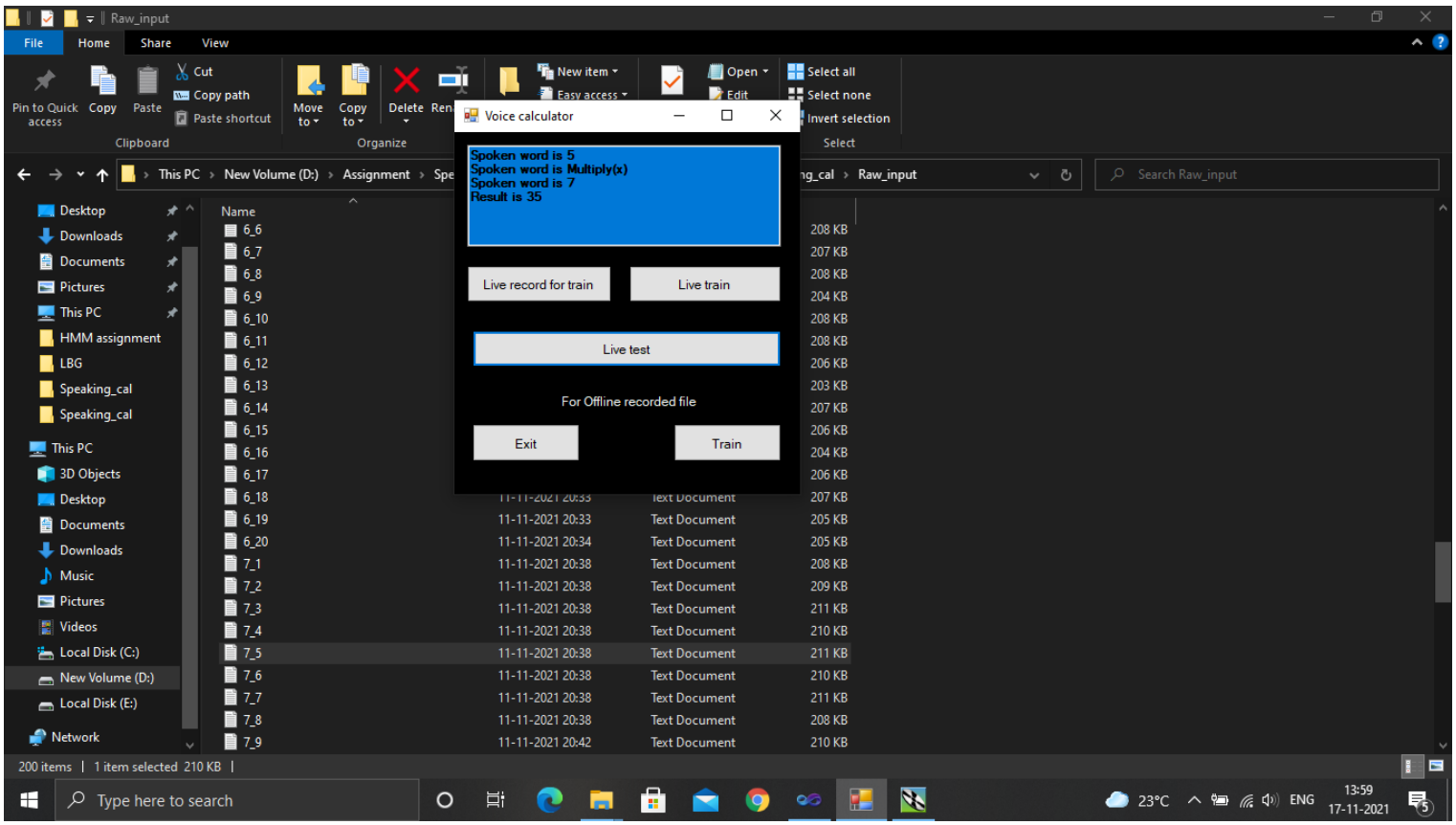
With the availability of above software, we further proceed in modelling the logic. The prerequisites of this project are

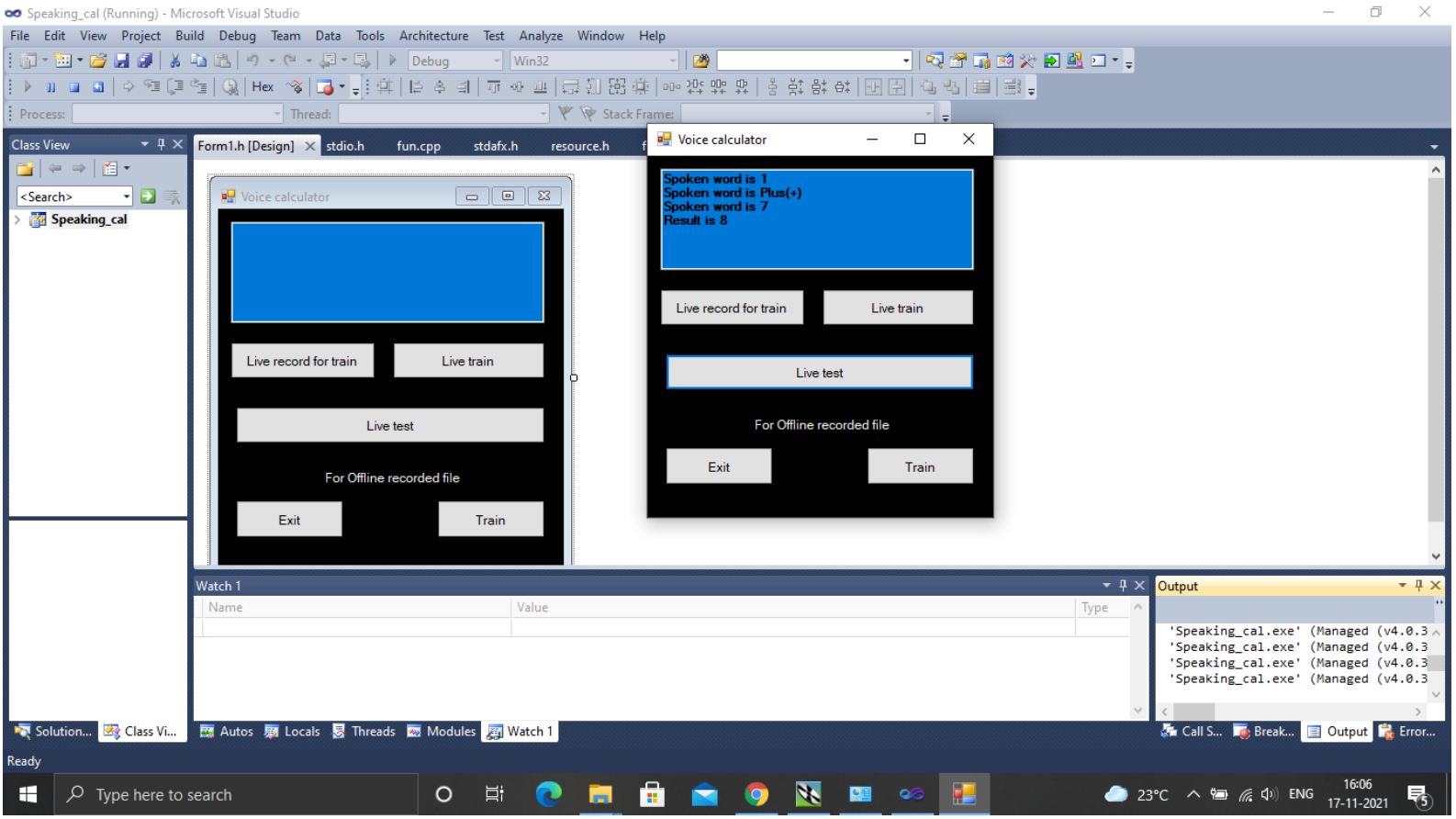
- Basic i/o operations on file
- Pre-processing of speech data
- Feature extraction
- Modelling of extracted feature
- Enhancing model

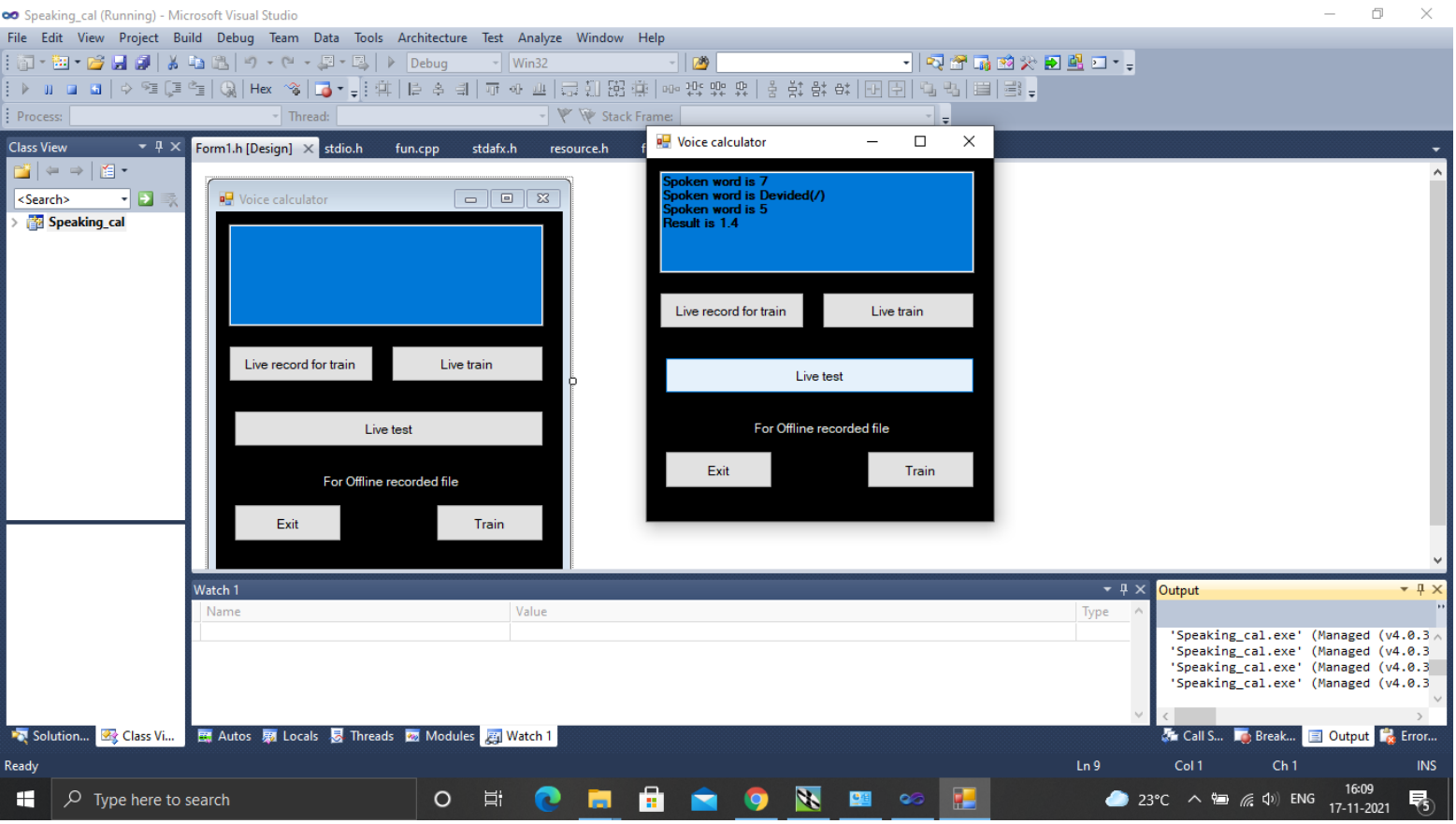
Above discussed topics are broadly elaborated in experimental setup section.

With the availability of above tools, we further proceeded. Below is the flow chart for our project：


Start
Enter the button to train:
1.live train
2.offline train
Choice1
No.
(choice 2
Click Train to train the model based on saved input files and save it into Output_into Output_Model
Yes
Record each digit 0-9 and symbols +, -, *, / . 20 times each
Click Live Train to save trained data into New_model folder
Click Live test to use the calculator and evaluate
choice="Live train"
Click exit
Stop
Yes
Ask the user to record the sentence. Eg. 2 plus 4
Check the answer after Evaluation


Raw_input
File Home Share View
Voice calculator
Spoken word is 5
Spoken word is Multiply(x)
Spoken word is 7
Result is 35
Live record for train
Live train
Live test
For Offline recorded file
Exit
Train
This PC › New Volume (D:) › Assignment
Search Raw_input
Name
6_6 208 KB
6_7 207 KB
6_8 208 KB
6_9 204 KB
6_10 208 KB
6_11 208 KB
6_12 206 KB
6_13 203 KB
6_14 207 KB
6_15 206 KB
6_16 204 KB
6_17 206 KB
6_18 207 KB
6_19 11-11-2021 20:33 Text Document 205 KB
6_20 11-11-2021 20:34 Text Document 205 KB
7_1 11-11-2021 20:38 Text Document 208 KB
7_2 11-11-2021 20:38 Text Document 209 KB
7_3 11-11-2021 20:38 Text Document 211 KB
7_4 11-11-2021 20:38 Text Document 210 KB
7_5 11-11-2021 20:38 Text Document 211 KB
7_6 11-11-2021 20:38 Text Document 210 KB
7_7 11-11-2021 20:38 Text Document 211 KB
7_8 11-11-2021 20:38 Text Document 208 KB
7_9 11-11-2021 20:42 Text Document 210 KB
200 items | 1 item selected 210 KB
Type here to search
23°C ENG 13:59 17-11-2021


Speaking_cal (Running) - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Architecture Test Analyze Window Help
Class View
Speaking_cal
Form1.h [Design] stdio.h fun.cpp stdafx.h resource.h
Voice calculator
Live record for train
Live train
Live test
For Offline recorded file
Exit
Train
Voice calculator
Spoken word is 1
Spoken word is Plus(+)
Spoken word is 7
Result is 8
Live record for train
Live train
Live test
For Offline recorded file
Exit
Train
Watch 1
Name Value Type
Output
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
Solution... Class Vi...
Autos Locals Threads Modules Watch 1
Call S... Break... Output Error...
Ready
Type here to search
23°C ENG 16:06 17-11-2021


Speaking_cal (Running) - Microsoft Visual Studio
File Edit View Project Build Debug Team Data Tools Architecture Test Analyze Window Help
Class View
Form1.h [Design]
stdio.h
fun.cpp
stdafx.h
resource.h
Speaking_cal
Voice calculator
Live record for train
Live train
Live test
For Offline recorded file
Exit
Train
Voice calculator
Spoken word is 7
Spoken word is Devided(/)
Spoken word is 5
Result is 1.4
Live record for train
Live train
Live test
For Offline recorded file
Exit
Train
Watch 1
Name
Value
Type
Output
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
'Speaking_cal.exe' (Managed (v4.0.3
Solution...
Class Vi...
Autos
Locals
Threads
Modules
Watch 1
Call S...
Break...
Output
Error...
Ready
Ln 9
Col 1
Ch 1
INS
Type here to search
23°C
ENG
16:09
17-11-2021

EXPERIMENTAL SETUP

This project is divided into following modules:

1. Training Module
2. Testing Module
3. Live Training Module

1. Training Module
   The flow for training over data is as follows:
   i. Record the data as 20 utterance of each word
   ii. Extract frames for every utterance
   iii. Using local distance analysis (in vector quantization) calculate the observation sequence.
   iv. Pass this observation sequence to HMM for model designing.
   v. Now enhance the model using HMM re-estimation algorithm.
2. Now reference model is ready for our project. The training of data is not integrated with GUI application. This is different module which will just evaluate reference model.
3. Testing Module
   System will give instruction what is going on and user is required to follow it. The flow of testing is as follows:
   i. Live recording of data is done when system instruct.
   ii. Testing the data with pre-trained models.
   iii. Verifying the contact name detected with user.
   iv. If verification is successful display contact details.
   v. If verification fails, record the input again

RESULT

We are getting result after the evaluation in speaking calculator with a good accuracy. Some of the result is shown above as picture.

Code

fun.h part

```
#pragma once
#define N 5
#define T 80
#define M 32
#define OB 20
#define pie 3.14
long extern double sample[500000],AvgEnergy[500000],r[13],ai[13]
[13],e[13],k[13],c[11][21][100][13];
long extern double Delta[T+1][N+1],pi[N+1],pi_Avg[OB+1]
[N+1],b[N+1][M+1],b_Avg[OB+1][N+1][M+1],a[N+1][N+1],a_Avg[OB+1]
[N+1][N+1],Alpha[T+1][N+1],Bita[T+1][N+1],Zie[N+1][N+1]
[T+1],Gamma[T+1][N+1],a_New[N+1][N+1],b_New[N+1][M+1],pi_New[N+1];
int extern Si[T+1][N+1],O[OB+1][T+1],q_Star_T,print[T+1];
long double Compute_P_Star(int obsNum);
void Compute_Alpha(int obsNum);
void Compute_Bita(int obsNum);
void Compute_Zie(int obsNum);
void Compute_Gamma(int obsNum);
long double Update_a_(int i,int j);
long double Update_b_(int i,int j,int obsNum);
void Update_pi_();
int LiveTesting();
int LiveTestingOP();
void LiveTrain();
int FileTesting(char namef[200]);
int FileTestingOP(char namef[200]);
void Train();
void LiveRecord();
```

Main part
Fun.cpp

```
#include "stdafx.h"
#include "stdio.h"
#include<stdlib.h>
#include<math.h>
#include <string.h>
#include<float.h>
#include<ctype.h>
#include <windows.h>
#include "fun.h"

long double sample[500000],AvgEnergy[500000],r[13],ai[13]
[13],e[13],k[13],c[11][21][100][13];
long double Delta[T+1][N+1],pi[N+1],pi_Avg[OB+1][N+1],b[N+1]
[M+1],b_Avg[OB+1][N+1][M+1],a[N+1][N+1],a_Avg[OB+1][N+1]
[N+1],Alpha[T+1][N+1],Bita[T+1][N+1],Zie[N+1][N+1][T+1],Gamma[T+1]
[N+1],a_New[N+1][N+1],b_New[N+1][M+1],pi_New[N+1];
int Si[T+1][N+1],O[OB+1][T+1],q_Star_T,print[T+1];
long double Compute_P_Star(int obsNum){
    int t,i,j,maxpi;
    long double maxp,P_Star;
    for(i=1;i<=N;i++){                  //Initialization
        Delta[1][i]=pi_New[i]*b_New[i][O[obsNum][1]];
        Si[1][i]=1;
    }
    for(t=2;t<=T;t++){                  //Recursion
        for(j=1;j<=N;j++){
            maxp=Delta[t-1][1]*a_New[1][j];
            maxpi=1;
            for(i=2;i<=N;i++){
                if(maxp<Delta[t-1][i]*a_New[i][j]){
                    maxp=Delta[t-1][i]*a_New[i][j];
                    maxpi=i;
                }
            }
            Delta[t][j]=maxp*b_New[j][O[obsNum][t]];
            Si[t][j]=maxpi;
        }
    }
    P_Star=Delta[T][1];             //Calculation P* and q*T
    q_Star_T=1;
    for(i=2;i<=N;i++){
        if(P_Star<Delta[T][i]){
            P_Star=Delta[T][i];
            q_Star_T=i;
        }
    }
```

```
    return P_Star;
}
void Compute_Alpha(int obsNum){
    int i,j,t;
    long double temp;
    for(i=1;i<=N;i++){                  //Initialization
        Alpha[1][i]=pi_New[i]*b_New[i][O[obsNum][1]];
    }
    for(t=1;t<=T-1;t++){
        for(j=1;j<=N;j++){
            temp=0;
            for(i=1;i<=N;i++){
                temp=temp+Alpha[t][i]*a_New[i][j];
            }
            Alpha[t+1][j]=temp*b_New[j][O[obsNum][t+1]];
            //Alpha[t+1][j]=temp;
        }
    }
}
void Compute_Bita(int obsNum){
    int i,j,t;
    long double temp;
    for(i=1;i<=N;i++){
        Bita[T][i]=1;
    }
    for(t=T-1;t>=1;t--){
        for(i=1;i<=N;i++){
            temp=0;
            for(j=1;j<=N;j++){
                temp=temp+a_New[i][j]*b_New[j][O[obsNum]
[t+1]]*Bita[t+1][j];
            }
            Bita[t][i]=temp;
        }
    }
}
void Compute_Zie(int obsNum){
    int i,j,t,p,q;
    long double temp1,temp2;
    for(t=1;t<=T-1;t++){
        temp2=0;
        for(p=1;p<=N;p++){
            for(q=1;q<=N;q++){
                temp2+=Alpha[t][p]*a_New[p][q]*b_New[q][O[obsNum]
[t+1]]*Bita[t+1][q];
            }
            //temp2+=Alpha[T][p];
        }
        for(i=1;i<=N;i++){
            for(j=1;j<=N;j++){
```

```
                temp1=Alpha[t][i]*a_New[i][j]*b_New[j][O[obsNum]
[t+1]]*Bita[t+1][j];
                Zie[i][j][t]=temp1/temp2;
            }
        }

    }
}
void Compute_Gamma(int obsNum){
    int i,j,t;
    long double temp;
    for(t=1;t<=T-1;t++){
        for(i=1;i<=N;i++){
            temp=0;
            for(j=1;j<=N;j++){
                temp=temp+Zie[i][j][t];
            }
            Gamma[t][i]=temp;
            //printf("%Le ",Gamma[t][i]);
        }
        //printf("\n");
    }
}
long double Update_a_(int i,int j){
    int t;
    long double temp1,temp2;
    temp1=0;
    for(t=1;t<=T-1;t++){
        temp1=temp1+Zie[i][j][t];
    }
    temp2=0;
    for(t=1;t<=T-1;t++){
        temp2=temp2+Gamma[t][i];
    }
    temp1=temp1/temp2;
    return temp1;
}
long double Update_b_(int i,int j,int obsNum){
    int t;
    long double temp1,temp2;
    temp1=0;
    for(t=1;t<=T-1;t++){
        if(j==O[obsNum][t]){
            temp1=temp1+Gamma[t][i];
        }
    }
    temp2=0;
    for(t=1;t<=T-1;t++){
        temp2=temp2+Gamma[t][i];
    }
    temp1=temp1/temp2;
```

```
    return temp1;
}
void Update_pi_(){
    int i;
    for(i=1;i<=N;i++){
        pi_New[i]=Gamma[1][i];
    }
}
int LiveTesting(){
            int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
            char buffer[1024],record[3000],*line,file[100];
            long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
            FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
            system("Recording_Module.exe 3 Live_test_input/
input_file.wav Live_test_input/input_fileinput_file.txt");
            if(1){
            char t[100];
            speech=fopen("Live_test_input/
input_fileinput_file.txt","r");          //Opening file which are
in 214101055_vowel_number this formate
            d=1;
            dn=1;
            if(speech!=NULL){
                for(i=0;i<5;i++){
                    fgets(t,sizeof(t),speech);           //Skip
first 5 lines as it header
                }
                i=1;
                MaxVal=0;
                while(!feof(speech)){
                    fscanf(speech,"%lf",&sample[i]); //Scaning
from file
                    sample[i]=sample[i]+DC_shift;
                    if(abs(sample[i])>MaxVal){
                        MaxVal=abs(sample[i]);
                    }
                    i++;
                }
                fclose(speech);
                NumberOfValue=i-1;
                TotalSegment=(NumberOfValue-320)/80;
                for(i=1;i<=NumberOfValue;i++)
{                   //Doing normalised
                    sample[i]=(sample[i]-DC_shift)*NormalisedAmpl/
MaxVal;
                }
```

```
                Threshold=0;
                l=Segment;
                for(i=1;i<=TotalSegment;i++)
{                   //Calculating per segment energy
                    l=80*(i-1);
                    temp=0;
                    j=1;
                    while(j<=Segment){
                        temp=temp+(sample[l+j]*sample[l+j]);
//Calculating total energy per segment
                        j++;
                    }
                    AvgEnergy[i]=temp/Segment;
    //Calculate average energy per segment
                }
                Threshold=0;
                for(i=1;i<=5;i++){
                    Threshold+=AvgEnergy[i];
                }
                Threshold/=5;
                Threshold*=10;
                fstart=0;
                for(i=2;i<TotalSegment-2;i++)
{                   //Take hight energy fream as Steady point
                    if(!fstart && AvgEnergy[i]>Threshold &&
AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                        start=i;
                        fstart=1;
                        break;
                    }
                    //if(!fstart && AvgEnergy[i]<Threshold &&
AvgEnergy[i+1]<Threshold && AvgEnergy[i+2]<Threshold){
                    //  end=i;

                    //}
                }
                end=start+80;
                //SteadyPoint=SteadyPoint-1;
                for(fream=start;fream<=end;fream++){
                    //Calculating Ri
                    for(i=0;i<=12;i++){                         //
Calculation of Ri
                        r[i]=0;
                        for(j=0;j<=319-i;j++){
                            r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                        }
                    }
                    //Calculation complete for Ri
                    //Calculation start for Ai
                    e[0]=r[0];
```

```
                    for(i=1;i<=12;i++){
                        sum=0;
                        for(j=1;j<=i-1;j++){
                            sum=sum+ai[i-1][j]*r[i-j];
                        }
                        if(i==1){
                            k[i]=r[1]/r[0];
                        }
                        else{
                            k[i]=(r[i]-sum)/e[i-1];
                        }
                        ai[i][i]=k[i];
                        for(j=1;j<=i-1;j++){
                            ai[i][j]=ai[i-1][j]-k[i]*ai[i-1][i-
j];
                        }
                        e[i]=(1-k[i]*k[i])*e[i-1];
                    }
                    //Calculation complete for Ai
                    //Calculation start for Ci
                    c[d][dn][fream-start+1][0]=2*log(r[0]);
                    for(i=1;i<=12;i++){
                        temp=0;
                        for(j=1;j<=i-1;j++)
{                           //To calculate Ci,we taken a 4D array in
fornamt C[vowel][VowelFile][Fream][i]
                            temp=temp+(j*c[d][dn][fream-start+1]
[j]*(ai[12][i-j]))/i;
                        }
                        c[d][dn][fream-start+1][i]=(ai[12][i])
+temp;
                    }
                    //Calculation complete for Ci
                    //Apply Raised sine window
                    for(m=1;m<=12;m++){
                        c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
                        //fprintf(fc,"%f\t",c[d][dn][fream-
start+1][m]);
                    }
                    //fprintf(fc,"\n");
                }
                //c[d][dn][0][0]=end-start+1;
            }
            //fclose(fc);
            char buffer[1024],record[50],*line;

            double A[33][13];
            f=fopen("codebook.txt","r");
            if(f!=NULL){
                int b;
```

```
                i=1;
                while(!feof(f)){
                    fgets(buffer,sizeof(buffer),f);       //Take
line in buffer
                    record[0]='\0';
                    l=1;
                    b=0;
                    for(j=0;buffer[j]!='\0';j++){         //For
each semecolon we divided the string
                        if(buffer[j]=='\t'){
                            record[b]='\0';
                            A[i]
[l]=strtod(record,&line);         //String to double convert
                            record[0]='\0';
                            l++;
                            b=0;
                        }
                        else{
                            record[b]=buffer[j];          //Untill
semecolon is encounter we store the string
                            b++;
                        }
                    }
                    record[b]='\0';
                    A[i][l]=strtod(record,&line);
                    record[0]='\0';
                    l++;
                    b=0;
                    i++;
                }
                fclose(f);
                double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
                int index=0,CBindex;
                f=fopen("Live_test/Live_Test_obs_seq.txt","w");
                if(f!=NULL){
                    fprintf(f,"-----------------------------\n");
                    for(fream=1;fream<=80;fream++){
                        dmin=DBL_MAX;
                        for(j=1;j<=32;j++){
                            sum=0;
                            for(i=1;i<=12;i++){
                                x=A[j][i]-c[d][dn][fream][i];
                                sum=sum+w[i]*x*x;             //
Applying the formula W[i] given in question
                            }
                            if(dmin>sum){
                                dmin=sum;
                                CBindex=j;
                            }
```

```
                        }
                        fprintf(f,"%d ",CBindex);
                    }
                    fprintf(f,"\n");
                }
                fclose(f);
            }
            fo=fopen("Live_test/Live_Test_obs_seq.txt","r");
            i=1;
            skip=1;
            while(!feof(fo)){
                fgets(buffer, sizeof(buffer), fo);
                if(skip%2==1){
                    skip=skip+1;
                    continue;
                }
                skip=skip+1;
                //fscanf(fa,"%s",buffer);          //Take first line
in buffer as string
                record[0]='\0';
                l=1;
                s=0;
                for(j=0;buffer[j]!='\0';j++){          //Take string
and seperate for each ;
                    if(buffer[j]==' '){
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);      //
String to double convert
                        record[0]='\0';
                        l++;
                        s=0;
                    }
                    else{
                        record[s]=buffer[j];          //Untill tab
is encounter we store string in record
                        s++;
                    }
                }
                record[s]='\0';
                O[i][l]=strtol(record,&line,10);
                record[0]='\0';
                l++;
                s=0;
                i++;
            }
            fclose(fo);
            count=0;
                minp=0;
                for(input=0;input<=9;input++){
                    if(input==0){
```

```
                        fa=fopen("Output_Model/Digit0/
A_0.txt","r");
                        fb=fopen("Output_Model/Digit0/
B_0.txt","r");
                        fpi=fopen("Output_Model/Digit0/
Pi_0.txt","r");
                    }
                    else if(input==1){
                        fa=fopen("Output_Model/Digit1/
A_1.txt","r");
                        fb=fopen("Output_Model/Digit1/
B_1.txt","r");
                        fpi=fopen("Output_Model/Digit1/
Pi_1.txt","r");
                    }
                    else if(input==2){
                        fa=fopen("Output_Model/Digit2/
A_2.txt","r");
                        fb=fopen("Output_Model/Digit2/
B_2.txt","r");
                        fpi=fopen("Output_Model/Digit2/
Pi_2.txt","r");
                    }
                    else if(input==3){
                        fa=fopen("Output_Model/Digit3/
A_3.txt","r");
                        fb=fopen("Output_Model/Digit3/
B_3.txt","r");
                        fpi=fopen("Output_Model/Digit3/
Pi_3.txt","r");
                    }
                    else if(input==4){
                        fa=fopen("Output_Model/Digit4/
A_4.txt","r");
                        fb=fopen("Output_Model/Digit4/
B_4.txt","r");
                        fpi=fopen("Output_Model/Digit4/
Pi_4.txt","r");
                    }
                    else if(input==5){
                        fa=fopen("Output_Model/Digit5/
A_5.txt","r");
                        fb=fopen("Output_Model/Digit5/
B_5.txt","r");
                        fpi=fopen("Output_Model/Digit5/
Pi_5.txt","r");
                    }
                    else if(input==6){
                        fa=fopen("Output_Model/Digit6/
A_6.txt","r");
```

```
                        fb=fopen("Output_Model/Digit6/
B_6.txt","r");
                        fpi=fopen("Output_Model/Digit6/
Pi_6.txt","r");
                    }
                    else if(input==7){
                        fa=fopen("Output_Model/Digit7/
A_7.txt","r");
                        fb=fopen("Output_Model/Digit7/
B_7.txt","r");
                        fpi=fopen("Output_Model/Digit7/
Pi_7.txt","r");
                    }
                    else if(input==8){
                        fa=fopen("Output_Model/Digit8/
A_8.txt","r");
                        fb=fopen("Output_Model/Digit8/
B_8.txt","r");
                        fpi=fopen("Output_Model/Digit8/
Pi_8.txt","r");
                    }
                    else if(input==9){
                        fa=fopen("Output_Model/Digit9/
A_9.txt","r");
                        fb=fopen("Output_Model/Digit9/
B_9.txt","r");
                        fpi=fopen("Output_Model/Digit9/
Pi_9.txt","r");
                    }
                    else if(input==10){
                        fa=fopen("Output_Model/DigitP/
A_P.txt","r");
                        fb=fopen("Output_Model/DigitP/
B_P.txt","r");
                        fpi=fopen("Output_Model/DigitP/
Pi_P.txt","r");
                    }
                    else if(input==11){
                        fa=fopen("Output_Model/DigitM/
A_M.txt","r");
                        fb=fopen("Output_Model/DigitM/
B_M.txt","r");
                        fpi=fopen("Output_Model/DigitM/
Pi_M.txt","r");
                    }
                    else if(input==12){
                        fa=fopen("Output_Model/DigitD/
A_D.txt","r");
                        fb=fopen("Output_Model/DigitD/
B_D.txt","r");
```

```
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","r");
                    }
                    else if(input==13){
                        fa=fopen("Output_Model/DigitG/
A_G.txt","r");
                        fb=fopen("Output_Model/DigitG/
B_G.txt","r");
                        fpi=fopen("Output_Model/DigitG/
Pi_G.txt","r");
                    }
                    if(fa!=NULL && fb!=NULL && fpi!=NULL){
                        //Read A matrix
                        i=1;
                        while(!feof(fa)){
                            fgets(buffer, sizeof(buffer), fa);
                            //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                            record[0]='\0';
                            l=1;
                            s=0;
                            for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                if(buffer[j]==' '){
                                    record[s]='\0';
                                    a[i][l]=strtod(record,&line);
    //String to double convert
                                    record[0]='\0';
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
                            record[s]='\0';
                            a[i][l]=strtod(record,&line);
                            record[0]='\0';
                            l++;
                            s=0;
                            i++;
                        }
                        fclose(fa);
                        //-----------------------------------
--
                        //Read B matrix
                        i=1;
                        while(!feof(fb)){
```

```
                                fgets(buffer, sizeof(buffer), fb);
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        b[i][l]=strtod(record,&line);
    //String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];            //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                b[i][l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fb);
                            //-----------------------------------------
------
                            //Read observation matrix

                            //-----------------------------------------
-----------
                            //Read Pi matrix
                            i=1;
                            skip=1;
                            while(!feof(fpi)){
                                fgets(buffer, sizeof(buffer), fpi);
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
```

```
                                        pi[l]=strtod(record,&line);
//String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];            //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                pi[l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fpi);
                            for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    a_New[i][j]=a[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    b_New[i][j]=b[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                pi_New[i]=pi[i];
                            }
                            Compute_Alpha(1);
                            temp=0;
                            for(i=1;i<=N;i++){
                                temp=temp+Alpha[T][i];
                            }
                            if(minp<temp){
                                minp=temp;
                                index=input;
                            }
                        }
                        //printf("P(O/Lamda)=%g for digits
%d\n",temp,input);
                    }
                    if(d==index){
                        count++;
                    }
                    //printf("Prediction is %d\n",index);
```

```
                //printf("%d\n",s);
            //printf("Accuracy is %d\n",count*10);
            //printf("**************************\n");
        }
        return index;
}
int LiveTestingOP(){
            int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
            char buffer[1024],record[3000],*line,file[100];
            long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
            FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
            system("Recording_Module.exe 3 Live_test_input/
input_file.wav Live_test_input/input_fileinput_file.txt");
            if(1){
            char t[100];
            speech=fopen("Live_test_input/
input_fileinput_file.txt","r");          //Opening file which are
in 214101055_vowel_number this formate
            d=1;
            dn=1;
            if(speech!=NULL){
                for(i=0;i<5;i++){
                    fgets(t,sizeof(t),speech);           //Skip
first 5 lines as it header
                }
                i=1;
                MaxVal=0;
                while(!feof(speech)){
                    fscanf(speech,"%lf",&sample[i]); //Scaning
from file
                    sample[i]=sample[i]+DC_shift;
                    if(abs(sample[i])>MaxVal){
                        MaxVal=abs(sample[i]);
                    }
                    i++;
                }
                fclose(speech);
                NumberOfValue=i-1;
                TotalSegment=(NumberOfValue-320)/80;
                for(i=1;i<=NumberOfValue;i++)
{                   //Doing normalised
                    sample[i]=(sample[i]-DC_shift)*NormalisedAmpl/
MaxVal;
                }
                Threshold=0;
                l=Segment;
```

```
                for(i=1;i<=TotalSegment;i++)
{                   //Calculating per segment energy
                    l=80*(i-1);
                    temp=0;
                    j=1;
                    while(j<=Segment){
                        temp=temp+(sample[l+j]*sample[l+j]);
//Calculating total energy per segment
                        j++;
                    }
                    AvgEnergy[i]=temp/Segment;
    //Calculate average energy per segment
                }
                Threshold=0;
                for(i=1;i<=5;i++){
                    Threshold+=AvgEnergy[i];
                }
                Threshold/=5;
                Threshold*=10;
                fstart=0;
                for(i=2;i<TotalSegment-2;i++)
{                   //Take hight energy fream as Steady point
                    if(!fstart && AvgEnergy[i]>Threshold &&
AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                        start=i;
                        fstart=1;
                        break;
                    }
                    //if(!fstart && AvgEnergy[i]<Threshold &&
AvgEnergy[i+1]<Threshold && AvgEnergy[i+2]<Threshold){
                    //  end=i;

                    //}
                }
                end=start+80;
                //SteadyPoint=SteadyPoint-1;
                for(fream=start;fream<=end;fream++){
                    //Calculating Ri
                    for(i=0;i<=12;i++){                         //
Calculation of Ri
                        r[i]=0;
                        for(j=0;j<=319-i;j++){
                            r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                        }
                    }
                    //Calculation complete for Ri
                    //Calculation start for Ai
                    e[0]=r[0];
                    for(i=1;i<=12;i++){
                        sum=0;
```

```
                        for(j=1;j<=i-1;j++){
                            sum=sum+ai[i-1][j]*r[i-j];
                        }
                        if(i==1){
                            k[i]=r[1]/r[0];
                        }
                        else{
                            k[i]=(r[i]-sum)/e[i-1];
                        }
                        ai[i][i]=k[i];
                        for(j=1;j<=i-1;j++){
                            ai[i][j]=ai[i-1][j]-k[i]*ai[i-1][i-
j];
                        }
                        e[i]=(1-k[i]*k[i])*e[i-1];
                    }
                    //Calculation complete for Ai
                    //Calculation start for Ci
                    c[d][dn][fream-start+1][0]=2*log(r[0]);
                    for(i=1;i<=12;i++){
                        temp=0;
                        for(j=1;j<=i-1;j++)
{                       //To calculate Ci,we taken a 4D array in
fornamt C[vowel][VowelFile][Fream][i]
                            temp=temp+(j*c[d][dn][fream-start+1]
[j]*(ai[12][i-j]))/i;
                        }
                        c[d][dn][fream-start+1][i]=(ai[12][i])
+temp;
                    }
                    //Calculation complete for Ci
                    //Apply Raised sine window
                    for(m=1;m<=12;m++){
                        c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
                        //fprintf(fc,"%f\t",c[d][dn][fream-
start+1][m]);
                    }
                    //fprintf(fc,"\n");
                }
                //c[d][dn][0][0]=end-start+1;
            }
            //fclose(fc);
            char buffer[1024],record[50],*line;

            double A[33][13];
            f=fopen("codebook.txt","r");
            if(f!=NULL){
                int b;
                i=1;
                while(!feof(f)){
```

```
                    fgets(buffer,sizeof(buffer),f);      //Take
line in buffer
                    record[0]='\0';
                    l=1;
                    b=0;
                    for(j=0;buffer[j]!='\0';j++){        //For
each semecolon we divided the string
                        if(buffer[j]=='\t'){
                            record[b]='\0';
                            A[i]
[l]=strtod(record,&line);        //String to double convert
                            record[0]='\0';
                            l++;
                            b=0;
                        }
                        else{
                            record[b]=buffer[j];         //Untill
semecolon is encounter we store the string
                            b++;
                        }
                    }
                    record[b]='\0';
                    A[i][l]=strtod(record,&line);
                    record[0]='\0';
                    l++;
                    b=0;
                    i++;
                }
                fclose(f);
                double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
                int index=0,CBindex;
                f=fopen("Live_test/Live_Test_obs_seq.txt","w");
                if(f!=NULL){
                    fprintf(f,"-------------------------\n");
                    for(fream=1;fream<=80;fream++){
                        dmin=DBL_MAX;
                        for(j=1;j<=32;j++){
                            sum=0;
                            for(i=1;i<=12;i++){
                                x=A[j][i]-c[d][dn][fream][i];
                                sum=sum+w[i]*x*x;            //
Applying the formula W[i] given in question
                            }
                            if(dmin>sum){
                                dmin=sum;
                                CBindex=j;
                            }
                        }
                        fprintf(f,"%d ",CBindex);
```

```
                    }
                    fprintf(f,"\n");
                }
                fclose(f);
            }
            fo=fopen("Live_test/Live_Test_obs_seq.txt","r");
            i=1;
            skip=1;
            while(!feof(fo)){
                fgets(buffer, sizeof(buffer), fo);
                if(skip%2==1){
                    skip=skip+1;
                    continue;
                }
                skip=skip+1;
                //fscanf(fa,"%s",buffer);          //Take first line
in buffer as string
                record[0]='\0';
                l=1;
                s=0;
                for(j=0;buffer[j]!='\0';j++){          //Take string
and seperate for each ;
                    if(buffer[j]==' '){
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);      //
String to double convert
                        record[0]='\0';
                        l++;
                        s=0;
                    }
                    else{
                        record[s]=buffer[j];          //Untill tab
is encounter we store string in record
                        s++;
                    }
                }
                record[s]='\0';
                O[i][l]=strtol(record,&line,10);
                record[0]='\0';
                l++;
                s=0;
                i++;
            }
            fclose(fo);
            count=0;
                minp=0;
                index=10;
                for(input=10;input<=13;input++){
                    if(input==0){
                        fa=fopen("Output_Model/Digit0/
A_0.txt","r");
```

```
                        fb=fopen("Output_Model/Digit0/
B_0.txt","r");
                        fpi=fopen("Output_Model/Digit0/
Pi_0.txt","r");
                    }
                    else if(input==1){
                        fa=fopen("Output_Model/Digit1/
A_1.txt","r");
                        fb=fopen("Output_Model/Digit1/
B_1.txt","r");
                        fpi=fopen("Output_Model/Digit1/
Pi_1.txt","r");
                    }
                    else if(input==2){
                        fa=fopen("Output_Model/Digit2/
A_2.txt","r");
                        fb=fopen("Output_Model/Digit2/
B_2.txt","r");
                        fpi=fopen("Output_Model/Digit2/
Pi_2.txt","r");
                    }
                    else if(input==3){
                        fa=fopen("Output_Model/Digit3/
A_3.txt","r");
                        fb=fopen("Output_Model/Digit3/
B_3.txt","r");
                        fpi=fopen("Output_Model/Digit3/
Pi_3.txt","r");
                    }
                    else if(input==4){
                        fa=fopen("Output_Model/Digit4/
A_4.txt","r");
                        fb=fopen("Output_Model/Digit4/
B_4.txt","r");
                        fpi=fopen("Output_Model/Digit4/
Pi_4.txt","r");
                    }
                    else if(input==5){
                        fa=fopen("Output_Model/Digit5/
A_5.txt","r");
                        fb=fopen("Output_Model/Digit5/
B_5.txt","r");
                        fpi=fopen("Output_Model/Digit5/
Pi_5.txt","r");
                    }
                    else if(input==6){
                        fa=fopen("Output_Model/Digit6/
A_6.txt","r");
                        fb=fopen("Output_Model/Digit6/
B_6.txt","r");
```

```
                        fpi=fopen("Output_Model/Digit6/
Pi_6.txt","r");
                    }
                    else if(input==7){
                        fa=fopen("Output_Model/Digit7/
A_7.txt","r");
                        fb=fopen("Output_Model/Digit7/
B_7.txt","r");
                        fpi=fopen("Output_Model/Digit7/
Pi_7.txt","r");
                    }
                    else if(input==8){
                        fa=fopen("Output_Model/Digit8/
A_8.txt","r");
                        fb=fopen("Output_Model/Digit8/
B_8.txt","r");
                        fpi=fopen("Output_Model/Digit8/
Pi_8.txt","r");
                    }
                    else if(input==9){
                        fa=fopen("Output_Model/Digit9/
A_9.txt","r");
                        fb=fopen("Output_Model/Digit9/
B_9.txt","r");
                        fpi=fopen("Output_Model/Digit9/
Pi_9.txt","r");
                    }
                    else if(input==10){
                        fa=fopen("Output_Model/DigitP/
A_P.txt","r");
                        fb=fopen("Output_Model/DigitP/
B_P.txt","r");
                        fpi=fopen("Output_Model/DigitP/
Pi_P.txt","r");
                    }
                    else if(input==11){
                        fa=fopen("Output_Model/DigitM/
A_M.txt","r");
                        fb=fopen("Output_Model/DigitM/
B_M.txt","r");
                        fpi=fopen("Output_Model/DigitM/
Pi_M.txt","r");
                    }
                    else if(input==12){
                        fa=fopen("Output_Model/DigitD/
A_D.txt","r");
                        fb=fopen("Output_Model/DigitD/
B_D.txt","r");
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","r");
                    }
```

```
                    else if(input==13){
                        fa=fopen("Output_Model/DigitG/
A_G.txt","r");
                        fb=fopen("Output_Model/DigitG/
B_G.txt","r");
                        fpi=fopen("Output_Model/DigitG/
Pi_G.txt","r");
                    }
                    if(fa!=NULL && fb!=NULL && fpi!=NULL){
                        //Read A matrix
                        i=1;
                        while(!feof(fa)){
                            fgets(buffer, sizeof(buffer), fa);
                            //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                            record[0]='\0';
                            l=1;
                            s=0;
                            for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                if(buffer[j]==' '){
                                    record[s]='\0';
                                    a[i][l]=strtod(record,&line);
    //String to double convert
                                    record[0]='\0';
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
                            record[s]='\0';
                            a[i][l]=strtod(record,&line);
                            record[0]='\0';
                            l++;
                            s=0;
                            i++;
                        }
                        fclose(fa);
                        //-----------------------------------------
--
                        //Read B matrix
                        i=1;
                        while(!feof(fb)){
                            fgets(buffer, sizeof(buffer), fb);
                            //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
```

```
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        b[i][l]=strtod(record,&line);
    //String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];            //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                b[i][l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fb);
                            //-----------------------------------------
------
                            //Read observation matrix

                            //-----------------------------------------
-----------
                            //Read Pi matrix
                            i=1;
                            skip=1;
                            while(!feof(fpi)){
                                fgets(buffer, sizeof(buffer), fpi);
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        pi[l]=strtod(record,&line);
//String to double convert
                                        record[0]='\0';
```

```
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
                            record[s]='\0';
                            pi[l]=strtod(record,&line);
                            record[0]='\0';
                            l++;
                            s=0;
                            i++;
                        }
                        fclose(fpi);
                        for(i=1;i<=N;i++){
                            for(j=1;j<=N;j++){
                                a_New[i][j]=a[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            for(j=1;j<=M;j++){
                                b_New[i][j]=b[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            pi_New[i]=pi[i];
                        }
                        Compute_Alpha(1);
                        temp=0;
                        for(i=1;i<=N;i++){
                            temp=temp+Alpha[T][i];
                        }
                        if(minp<temp){
                            minp=temp;
                            index=input;
                        }
                    }
                    //printf("P(O/Lamda)=%g for digits
%d\n",temp,input);
                }
                if(d==index){
                    count++;
                }
                //printf("Prediction is %d\n",index);
                //printf("%d\n",s);
            //printf("Accuracy is %d\n",count*10);
            //printf("**************************\n");
```

```
        }
        return index;
}
void Train(){
    int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
    char buffer[1024],record[3000],*line,file[100];
    long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
    FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
    if(1){
        char t[100];
            for(d=0;d<=13;d++){      //We have 5 vowel, to take 5
vowel we are put a loop
                file[0]='\0';
                strcat(file,"Raw_input/");
                if(d==0){
                    file[10]='0';
                    file[11]='_';
                }
                else if(d==1){
                    file[10]='1';
                    file[11]='_';
                }
                else if(d==2){
                    file[10]='2';
                    file[11]='_';
                }
                else if(d==3){
                    file[10]='3';
                    file[11]='_';
                }
                else if(d==4){
                    file[10]='4';
                    file[11]='_';
                }
                else if(d==5){
                    file[10]='5';
                    file[11]='_';
                }
                else if(d==6){
                    file[10]='6';
                    file[11]='_';
                }
                else if(d==7){
                    file[10]='7';
                    file[11]='_';
                }
```

```
                else if(d==8){
                    file[10]='8';
                    file[11]='_';
                }
                else if(d==9){
                    file[10]='9';
                    file[11]='_';
                }
                else if(d==10){
                    file[10]='P';
                    file[11]='_';
                }
                else if(d==11){
                    file[10]='M';
                    file[11]='_';
                }
                else if(d==12){
                    file[10]='D';
                    file[11]='_';
                }
                else if(d==13){
                    file[10]='G';
                    file[11]='_';
                }
                for(dn=1;dn<=20;dn++){
                    if(dn>=10){
                        file[12]=dn/10+'0';
                        file[13]=dn%10+'0';
                        file[14]='\0';
                    }
                    else{
                        file[12]=dn+'0';
                        file[13]='\0';
                    }
                    strcat(file,".txt");
                    speech=fopen(file,"r");          //Opening
file which are in 214101055_vowel_number this formate
                    if(speech!=NULL){
                        for(i=0;i<5;i++){

fgets(t,sizeof(t),speech);          //Skip first 5 lines as it
header
                        }
                        i=1;
                        MaxVal=0;
                        while(!feof(speech)){
                            fscanf(speech,"%lf",&sample[i]); //
Scaning from file
                            sample[i]=sample[i]+DC_shift;
                            if(abs(sample[i])>MaxVal){
                                MaxVal=abs(sample[i]);
```

```
                                }
                                i++;
                            }
                            fclose(speech);
                            NumberOfValue=i-1;
                            TotalSegment=(NumberOfValue-320)/80;
                            for(i=1;i<=NumberOfValue;i++){
    //Doing normalised
                                sample[i]=(sample[i]-
DC_shift)*NormalisedAmpl/MaxVal;
                            }
                            Threshold=0;
                            l=Segment;
                            for(i=1;i<=TotalSegment;i++){
    //Calculating per segment energy
                                l=80*(i-1);
                                temp=0;
                                j=1;
                                while(j<=Segment){
                                    temp=temp+
(sample[l+j]*sample[l+j]);            //Calculating total energy
per segment
                                    j++;
                                }
                                AvgEnergy[i]=temp/Segment;
            //Calculate average energy per segment
                            }
                            Threshold=0;
                            for(i=1;i<=5;i++){
                                Threshold+=AvgEnergy[i];
                            }
                            Threshold/=5;
                            Threshold*=10;
                            fstart=0;
                            for(i=2;i<TotalSegment-2;i++){
    //Take hight energy fream as Steady point
                                if(!fstart && AvgEnergy[i]>Threshold
&& AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                                    start=i;
                                    fstart=1;
                                    break;
                                }
                                //if(!fstart &&
AvgEnergy[i]<Threshold && AvgEnergy[i+1]<Threshold &&
AvgEnergy[i+2]<Threshold){
                                //  end=i;

                                //}
                            }
                            end=start+80;
                            //SteadyPoint=SteadyPoint-1;
```

```
                        for(fream=start;fream<=end;fream++){
                            //Calculating Ri
                            for(i=0;i<=12;i++){
//Calculation of Ri
                                r[i]=0;
                                for(j=0;j<=319-i;j++){
                                    r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                                }
                            }
                            //Calculation complete for Ri
                            //Calculation start for Ai
                            e[0]=r[0];
                            for(i=1;i<=12;i++){
                                sum=0;
                                for(j=1;j<=i-1;j++){
                                    sum=sum+ai[i-1][j]*r[i-j];
                                }
                                if(i==1){
                                    k[i]=r[1]/r[0];
                                }
                                else{
                                    k[i]=(r[i]-sum)/e[i-1];
                                }
                                ai[i][i]=k[i];
                                for(j=1;j<=i-1;j++){
                                    ai[i][j]=ai[i-1][j]-
k[i]*ai[i-1][i-j];
                                }
                                e[i]=(1-k[i]*k[i])*e[i-1];
                            }
                            //Calculation complete for Ai
                            //Calculation start for Ci
                            c[d][dn][fream-start+1]
[0]=2*log(r[0]);
                            for(i=1;i<=12;i++){
                                temp=0;
                                for(j=1;j<=i-1;j++){
    //To calculate Ci,we taken a 4D array in fornamt C[vowel]
[VowelFile][Fream][i]
                                    temp=temp+(j*c[d][dn][fream-
start+1][j]*(ai[12][i-j]))/i;
                                }
                                c[d][dn][fream-start+1]
[i]=(ai[12][i])+temp;
                            }
                            //Calculation complete for Ci
                            //Apply Raised sine window
                            for(m=1;m<=12;m++){
                                c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
```

```
                                    //fprintf(fc,"%f\t",c[d][dn]
[fream-start+1][m]);
                                }
                                //fprintf(fc,"\n");
                            }
                            //c[d][dn][0][0]=end-start+1;
                        }

                    }
                }
                //fclose(fc);
                char buffer[1024],record[50],*line;

                double A[33][13];
                f=fopen("codebook.txt","r");
                if(f!=NULL){
                    int b;
                    i=1;
                    while(!feof(f)){
                        fgets(buffer,sizeof(buffer),f);       //Take
line in buffer
                        record[0]='\0';
                        l=1;
                        b=0;
                        for(j=0;buffer[j]!='\0';j++){         //For
each semecolon we divided the string
                            if(buffer[j]=='\t'){
                                record[b]='\0';
                                A[i]
[l]=strtod(record,&line);         //String to double convert
                                record[0]='\0';
                                l++;
                                b=0;
                            }
                            else{
                                record[b]=buffer[j];          //Untill
semecolon is encounter we store the string
                                b++;
                            }
                        }
                        record[b]='\0';
                        A[i][l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        b=0;
                        i++;
                    }
                    fclose(f);
                    double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
```

```
                int index=0,CBindex;
                for(d=0;d<=13;d++){
                    file[0]='\0';
                    strcat(file,"Input/obs_seq_");
                    if(d==0){
                        file[14]='0';
                    }
                    else if(d==1){
                        file[14]='1';
                    }
                    else if(d==2){
                        file[14]='2';
                    }
                    else if(d==3){
                        file[14]='3';
                    }
                    else if(d==4){
                        file[14]='4';
                    }
                    else if(d==5){
                        file[14]='5';
                    }
                    else if(d==6){
                        file[14]='6';
                    }
                    else if(d==7){
                        file[14]='7';
                    }
                    else if(d==8){
                        file[14]='8';
                    }
                    else if(d==9){
                        file[14]='9';
                    }
                    else if(d==10){
                        file[14]='P';
                    }
                    else if(d==11){
                        file[14]='M';
                    }
                    else if(d==12){
                        file[14]='D';
                    }
                    else if(d==13){
                        file[14]='G';
                    }
                    file[15]='\0';
                    strcat(file,".txt");
                    f=fopen(file,"w");
                    for(dn=1;dn<=20;dn++){
                        if(f!=NULL){
```

```
fprintf(f,"-------------------------\n");
                        for(fream=1;fream<=80;fream++){
                            dmin=DBL_MAX;
                            for(j=1;j<=32;j++){
                                sum=0;
                                for(i=1;i<=12;i++){
                                    x=A[j][i]-c[d][dn][fream]
[i];
                                    sum=sum+w[i]*x*x;
//Applying the formula W[i] given in question
                                }
                                if(dmin>sum){
                                    dmin=sum;
                                    CBindex=j;
                                }
                            }
                            fprintf(f,"%d ",CBindex);
                        }
                        fprintf(f,"\n");
                    }
                }
                fclose(f);
                printf("Save obs sequence %d...\n",d);
            }
        }
        //
**********************************************************************
        printf("Training all 9 model\n");
        //scanf("%d",&input);
        for(input=0;input<=13;input++){
            printf("************************\n");
            printf("Training model %d....\n",input);
            if(input==0){
                //fa=fopen("A_1.txt","r");
                //fb=fopen("B_1.txt","r");
                fo=fopen("Input/obs_seq_0.txt","r");
                //fpi=fopen("Pi_1.txt","r");
            }
            else if(input==1){
                //fa=fopen("A_7.txt","r");
                //fb=fopen("B_7.txt","r");
                fo=fopen("Input/obs_seq_1.txt","r");
                //fpi=fopen("Pi_7.txt","r");
            }
            else if(input==2){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_2.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
```

```
            else if(input==3){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_3.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==4){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_4.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==5){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_5.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==6){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_6.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==7){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_7.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==8){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_8.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==9){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_9.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==10){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("Input/obs_seq_P.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==11){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
```

```
                    fo=fopen("Input/obs_seq_M.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                else if(input==12){
                    //fa=fopen("A_8.txt","r");
                    //fb=fopen("B_8.txt","r");
                    fo=fopen("Input/obs_seq_D.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                else if(input==13){
                    //fa=fopen("A_8.txt","r");
                    //fb=fopen("B_8.txt","r");
                    fo=fopen("Input/obs_seq_G.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                if(fo!=NULL){
                    //Read A matrix
                    /*i=1;
                    while(!feof(fa)){
                        fgets(buffer, sizeof(buffer), fa);
                        //fscanf(fa,"%s",buffer);        //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]=='\t'){
                                record[s]='\0';
                                a[i][l]=strtod(record,&line);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        a[i][l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fa);
                    //-----------------------------------------
                    //Read B matrix
```

```
                    i=1;
                    while(!feof(fb)){
                        fgets(buffer, sizeof(buffer), fb);
                        //fscanf(fa,"%s",buffer);         //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]=='\t'){
                                record[s]='\0';
                                b[i][l]=strtod(record,&line);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        b[i][l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fb);*/
                    //-----------------------------------------
--
                    //Read observation matrix
                    i=1;
                    skip=1;
                    while(!feof(fo)){
                        fgets(buffer, sizeof(buffer), fo);
                        if(skip%2==1){
                            skip=skip+1;
                            continue;
                        }
                        skip=skip+1;
                        //fscanf(fa,"%s",buffer);         //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
```

```
                            if(buffer[j]==' '){
                                record[s]='\0';
                                O[i][l]=strtol(record,&line,10);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fo);
                    //-----------------------------------------
-------
                    //Read Pi matrix
                    /*i=1;
                    skip=1;
                    while(!feof(fpi)){
                        fgets(buffer, sizeof(buffer), fpi);
                        //fscanf(fa,"%s",buffer);         //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]==' '){
                                record[s]='\0';

pi[l]=strtod(record,&line);       //String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
```

```
            pi[l]=strtod(record,&line);
            record[0]='\0';
            l++;
            s=0;
            i++;
        }
        fclose(fpi);*/
        //------------Pi-----------------------
        for(i=1;i<=N;i++){
            if(i==1){
                pi[i]=1;
                //pi_temp[i]=1;
            }
            else{
                pi[i]=0;
                //pi_temp[i]=0;
            }
        }
        for(i=1;i<=N;i++){
            for(j=1;j<=N;j++){
                if(i==j && i==N){
                    a[i][j]=1;
                    //a_temp[i][j]=1;
                }
                else if(i==j){
                    a[i][j]=0.8;
                    //a_temp[i][j]=0.8;
                    j++;
                    a[i][j]=0.2;
                    //a_temp[i][j]=0.2;
                }
            }
        }
        for(i=1;i<=N;i++){
            for(j=1;j<=M;j++){
                b[i][j]=0.03125;
                //b_temp[i][j]=0.03125;
            }
        }
        //fw=fopen("Output.txt","w");
        //-------------------------------------
        int Overite=0;
        while(Overite!=4){
            Overite++;
            for(obsNum=1;obsNum<=OB;obsNum++){
                for(i=1;i<=N;i++){
                    for(j=1;j<=N;j++){
                        a_New[i][j]=a[i][j];
                    }
                }
                for(i=1;i<=N;i++){
```

```
                                for(j=1;j<=M;j++){
                                    b_New[i][j]=b[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                pi_New[i]=pi[i];
                            }
                            //fprintf(fw,"----------
Observation%d----------\n",obsNum);
                            //fprintf(fw,"Total iteration is
200\n");
                            //P_Star=Compute_P_Star(obsNum);
                            /*for(i=T;i>=1;i++){
                                q_Star_T=Si[t+1][
                            }*/
                            //printf("For observation sequence
%d, P* is %Le \n",obsNum,P_Star);
                            //printf("Sequence are : %d
",q_Star_T);

                            //Compute_Alpha(obsNum);
                            /*temp=0;
                            for(i=1;i<=N;i++){
                                temp=temp+Alpha[T][i];
                            }*/
                            //printf("For observation sequence
%d, P(O/Lamda)= %Le using forward\n",obsNum,temp);
                            //Compute_Bita(obsNum);
                            /*temp=0;
                            for(i=1;i<=N;i++){
                                temp=temp+pi[i]*b_New[i]
[O[obsNum][1]]*Bita[1][i];
                            }*/
                            //printf("For observation sequence
%d, P(O/Lamda)= %Le using backward\n\n\n\n",obsNum,temp);
                            /*for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    a_Old[i][j]=a[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    b_Old[i][j]=b[i][j];
                                }
                            }*/
                            //---------------Zei
computing-----------------
                            /*Compute_Zie(obsNum);
                            //-------------
Complete-------------------------
                            //printf("Gamma matrix is :\n");
```

```
                                Compute_Gamma(obsNum);
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
                                        temp=Update_a_(i,j);
                                        if(temp==0){
                                            a_New[i][j]=0;
                                        }
                                        else{
                                            a_New[i][j]=temp;
                                        }
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=M;j++){
                                        temp=Update_b_(i,j,obsNum);
                                        if(temp==0){
                                            b_New[i][j]=1e-30;
                                        }
                                        else{
                                            b_New[i][j]=temp;
                                        }
                                    }
                                }
                                Update_pi_();
                                //--------------
adjust----------------
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=a_New[i][1];
                                    index=1;
                                    for(j=1;j<=N;j++){
                                        sum+=a_New[i][j];
                                        if(maxval<a_New[i][j]){
                                            maxval=a_New[i][j];
                                            index=j;
                                        }
                                    }
                                    if(sum>=1){
                                        a_New[i][index]=a_New[i]
[index]-(sum-1);
                                    }
                                    else{
                                        a_New[i][index]=a_New[i]
[index]+(1-sum);
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=b_New[i][1];
                                    index=1;
                                    for(j=1;j<=M;j++){
```

```
                                    sum+=b_New[i][j];
                                    if(maxval<b_New[i][j]){
                                        maxval=b_New[i][j];
                                        index=j;
                                    }
                                }
                                if(sum>=1){
                                    b_New[i][index]=b_New[i]
[index]-(sum-1);
                                }
                                else{
                                    b_New[i][index]=b_New[i]
[index]+(1-sum);
                                }
                            }
                            sum=0;
                            maxval=pi_New[1];
                            index=1;
                            for(i=1;i<=N;i++){
                                sum+=pi_New[i];
                                if(maxval<pi_New[i]){
                                    maxval=pi_New[i];
                                    index=i;
                                }
                            }
                            if(sum>=1){
                                pi_New[index]=pi_New[index]-
(sum-1);
                            }
                            else{
                                pi_New[index]=pi_New[index]+(1-
sum);
                            }
                            P_Star_Old=P_Star;*/
                            P_Star=Compute_P_Star(obsNum);
                            iet=0;
                            while(iet!=5){
                                iet++;
                                //printf("hello");
                                Compute_Alpha(obsNum);
                                Compute_Bita(obsNum);
                                //---------------Zei
computing-----------------
                                Compute_Zie(obsNum);
                                //-------------
Complete------------------------
                                Compute_Gamma(obsNum);
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
                                        temp=Update_a_(i,j);
                                        if(temp==0){
```

```
                                        a_New[i][j]=0;
                                    }
                                    else{
                                        a_New[i][j]=temp;
                                    }
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
temp=Update_b_(i,j,obsNum);
                                    if(temp==0){
                                        b_New[i][j]=1e-30;
                                    }
                                    else{
                                        b_New[i][j]=temp;
                                    }
                                }
                            }
                            Update_pi_();
                            //---------------
adjust----------------
                            for(i=1;i<=N;i++){
                                sum=0;
                                maxval=a_New[i][1];
                                index=1;
                                for(j=1;j<=N;j++){
                                    sum+=a_New[i][j];
                                    if(maxval<a_New[i][j]){
                                        maxval=a_New[i][j];
                                        index=j;
                                    }
                                }
                                if(sum>=1){
                                    a_New[i][index]=a_New[i]
[index]-(sum-1);
                                }
                                else{
                                    a_New[i][index]=a_New[i]
[index]+(1-sum);
                                }
                            }
                            for(i=1;i<=N;i++){
                                sum=0;
                                maxval=b_New[i][1];
                                index=1;
                                for(j=1;j<=M;j++){
                                    sum+=b_New[i][j];
                                    if(maxval<b_New[i][j]){
                                        maxval=b_New[i][j];
                                        index=j;
```

```
                                        }
                                    }
                                    if(sum>=1){
                                        b_New[i][index]=b_New[i]
[index]-(sum-1);
                                    }
                                    else{
                                        b_New[i][index]=b_New[i]
[index]+(1-sum);
                                    }
                                }
                                sum=0;
                                maxval=pi_New[1];
                                index=1;
                                for(i=1;i<=N;i++){
                                    sum+=pi_New[i];
                                    if(maxval<pi_New[i]){
                                        maxval=pi_New[i];
                                        index=i;
                                    }
                                }
                                if(sum>=1){
                                    pi_New[index]=pi_New[index]-
(sum-1);
                                }
                                else{
                                    pi_New[index]=pi_New[index]
+(1-sum);
                                }
                                P_Star_Old=P_Star;
                                P_Star=Compute_P_Star(obsNum);
                                //printf("For observation
sequence %d,Optimal P* is %Le \n",obsNum,P_Star);
                            }
                            //printf("For observation sequence
%d,Optimal P* is %Le \n",obsNum,P_Star);
                            //fprintf(fw,"Optimal
P*=%Le\n",P_Star);
                            //printf("Final state sequence
is :");
                            //fprintf(fw,"Finalstate sequence
is :");
                            print[T]=q_Star_T;
                            for(i=T-1;i>=1;i--){
                                q_Star_T=Si[i][q_Star_T];
                                print[i]=q_Star_T;
                                //printf(", %d",q_Star_T);
                            }
                            /*for(i=1;i<=T;i++){
                                if(i!=T){
                                    printf("%d ,",print[i]);
```

```
                                    fprintf(fw,"%d ,",print[i]);
                                }
                                else{
                                    printf("%d",print[i]);
                                    fprintf(fw,"%d\n",print[i]);
                                }
                            }
                            printf("\n");
                            fprintf(fw,"Matrix a:\n");
                            for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    fprintf(fw,"%f ",a_New[i]
[j]);
                                }
                                fprintf(fw,"\n");
                            }
                            fprintf(fw,"\n");
                            fprintf(fw,"Matrix b:\n");
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    fprintf(fw,"%Le ",b_New[i]
[j]);
                                }
                                fprintf(fw,"\n");
                            }*/
                            for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    a_Avg[obsNum][i][j]=a_New[i]
[j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    b_Avg[obsNum][i][j]=b_New[i]
[j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                pi_Avg[obsNum][i]=pi_New[i];
                            }
                            /*printf("Matrix is :\n");
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    printf("%Le ",b_New[i][j]);
                                }
                                printf("\n");
                            }*/
                        }

                        for(i=1;i<=N;i++){
                            for(j=1;j<=N;j++){
```

```
                                    temp=0;
                                    for(obsNum=1;obsNum<=OB;obsNum++)
{
                                        temp+=a_Avg[obsNum][i][j];
                                    }
                                    a[i][j]=temp/OB;
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    temp=0;
                                    for(obsNum=1;obsNum<=OB;obsNum++)
{
                                        temp+=b_Avg[obsNum][i][j];
                                    }
                                    b[i][j]=temp/OB;
                                }
                            }
                            for(i=1;i<=N;i++){
                                temp=0;
                                for(obsNum=1;obsNum<=OB;obsNum++){
                                    temp+=pi_Avg[obsNum][i];
                                }
                                pi[i]=temp/OB;
                            }
                            printf("Model updating %d....\n",Overite);
                        }
                        //fclose(fw);
                        if(input==0){
                            fa=fopen("Output_Model/Digit0/
A_0.txt","w");
                            fb=fopen("Output_Model/Digit0/
B_0.txt","w");
                            fpi=fopen("Output_Model/Digit0/
Pi_0.txt","w");
                        }
                        else if(input==1){
                            fa=fopen("Output_Model/Digit1/
A_1.txt","w");
                            fb=fopen("Output_Model/Digit1/
B_1.txt","w");
                            fpi=fopen("Output_Model/Digit1/
Pi_1.txt","w");
                        }
                        else if(input==2){
                            fa=fopen("Output_Model/Digit2/
A_2.txt","w");
                            fb=fopen("Output_Model/Digit2/
B_2.txt","w");
                            fpi=fopen("Output_Model/Digit2/
Pi_2.txt","w");
```

```
                    }
                    else if(input==3){
                        fa=fopen("Output_Model/Digit3/
A_3.txt","w");
                        fb=fopen("Output_Model/Digit3/
B_3.txt","w");
                        fpi=fopen("Output_Model/Digit3/
Pi_3.txt","w");
                    }
                    else if(input==4){
                        fa=fopen("Output_Model/Digit4/
A_4.txt","w");
                        fb=fopen("Output_Model/Digit4/
B_4.txt","w");
                        fpi=fopen("Output_Model/Digit4/
Pi_4.txt","w");
                    }
                    else if(input==5){
                        fa=fopen("Output_Model/Digit5/
A_5.txt","w");
                        fb=fopen("Output_Model/Digit5/
B_5.txt","w");
                        fpi=fopen("Output_Model/Digit5/
Pi_5.txt","w");
                    }
                    else if(input==6){
                        fa=fopen("Output_Model/Digit6/
A_6.txt","w");
                        fb=fopen("Output_Model/Digit6/
B_6.txt","w");
                        fpi=fopen("Output_Model/Digit6/
Pi_6.txt","w");
                    }
                    else if(input==7){
                        fa=fopen("Output_Model/Digit7/
A_7.txt","w");
                        fb=fopen("Output_Model/Digit7/
B_7.txt","w");
                        fpi=fopen("Output_Model/Digit7/
Pi_7.txt","w");
                    }
                    else if(input==8){
                        fa=fopen("Output_Model/Digit8/
A_8.txt","w");
                        fb=fopen("Output_Model/Digit8/
B_8.txt","w");
                        fpi=fopen("Output_Model/Digit8/
Pi_8.txt","w");
                    }
                    else if(input==9){
```

```
                        fa=fopen("Output_Model/Digit9/
A_9.txt","w");
                        fb=fopen("Output_Model/Digit9/
B_9.txt","w");
                        fpi=fopen("Output_Model/Digit9/
Pi_9.txt","w");
                    }
                    else if(input==10){
                        fa=fopen("Output_Model/DigitP/
A_P.txt","w");
                        fb=fopen("Output_Model/DigitP/
B_P.txt","w");
                        fpi=fopen("Output_Model/DigitP/
Pi_P.txt","w");
                    }
                    else if(input==11){
                        fa=fopen("Output_Model/DigitM/
A_M.txt","w");
                        fb=fopen("Output_Model/DigitM/
B_M.txt","w");
                        fpi=fopen("Output_Model/DigitM/
Pi_M.txt","w");
                    }
                    else if(input==12){
                        fa=fopen("Output_Model/DigitD/
A_D.txt","w");
                        fb=fopen("Output_Model/DigitD/
B_D.txt","w");
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","w");
                    }
                    else if(input==13){
                        fa=fopen("Output_Model/DigitG/
A_G.txt","w");
                        fb=fopen("Output_Model/DigitG/
B_G.txt","w");
                        fpi=fopen("Output_Model/DigitG/
Pi_G.txt","w");
                    }
                    for(i=1;i<=N;i++){
                        for(j=1;j<=N;j++){
                            fprintf(fa,"%f ",a[i][j]);
                        }
                        fprintf(fa,"\n");
                    }
                    fclose(fa);

                    for(i=1;i<=N;i++){
                        for(j=1;j<=M;j++){
                            fprintf(fb,"%Le ",b[i][j]);
                        }
```

```
                        fprintf(fb,"\n");
                    }
                    fclose(fb);

                    for(i=1;i<=N;i++){
                        fprintf(fpi,"%f ",pi[i]);
                    }
                    fclose(fpi);
                    printf("Save model in Output filder\n");
                }
            }
    }
}
void LiveRecord(){
    char file[100];
    int d,dn;
    char path[200]="Recording_Module.exe 3 New_input/
input_file.wav ";
            for(d=0;d<=13;d++){
                file[0]='\0';
                strcat(file,"New_input/");
                if(d==0){
                    file[10]='0';
                    file[11]='_';
                }
                else if(d==1){
                    file[10]='1';
                    file[11]='_';
                }
                else if(d==2){
                    file[10]='2';
                    file[11]='_';
                }
                else if(d==3){
                    file[10]='3';
                    file[11]='_';
                }
                else if(d==4){
                    file[10]='4';
                    file[11]='_';
                }
                else if(d==5){
                    file[10]='5';
                    file[11]='_';
                }
                else if(d==6){
                    file[10]='6';
                    file[11]='_';
                }
                else if(d==7){
                    file[10]='7';
```

```
                    file[11]='_';
                }
                else if(d==8){
                    file[10]='8';
                    file[11]='_';
                }
                else if(d==9){
                    file[10]='9';
                    file[11]='_';
                }
                else if(d==10){
                    file[10]='P';
                    file[11]='_';
                }
                else if(d==11){
                    file[10]='M';
                    file[11]='_';
                }
                else if(d==12){
                    file[10]='D';
                    file[11]='_';
                }
                else if(d==13){
                    file[10]='G';
                    file[11]='_';
                }
                for(dn=1;dn<=20;dn++){
                    if(dn>=10){
                        file[12]=dn/10+'0';
                        file[13]=dn%10+'0';
                        file[14]='\0';
                    }
                    else{
                        file[12]=dn+'0';
                        file[13]='\0';
                    }
                    strcat(file,".txt");
                    strcat(path,file);
                    system(path);
                }
            }
}
void LiveTrain(){
        int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
        char buffer[1024],record[3000],*line,file[100];
        long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
```

```
        FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
        if(1){
            char t[100];
            for(d=0;d<=13;d++){      //We have 5 vowel, to take 5
vowel we are put a loop
                file[0]='\0';
                strcat(file,"New_input/");
                if(d==0){
                    file[10]='0';
                    file[11]='_';
                }
                else if(d==1){
                    file[10]='1';
                    file[11]='_';
                }
                else if(d==2){
                    file[10]='2';
                    file[11]='_';
                }
                else if(d==3){
                    file[10]='3';
                    file[11]='_';
                }
                else if(d==4){
                    file[10]='4';
                    file[11]='_';
                }
                else if(d==5){
                    file[10]='5';
                    file[11]='_';
                }
                else if(d==6){
                    file[10]='6';
                    file[11]='_';
                }
                else if(d==7){
                    file[10]='7';
                    file[11]='_';
                }
                else if(d==8){
                    file[10]='8';
                    file[11]='_';
                }
                else if(d==9){
                    file[10]='9';
                    file[11]='_';
                }
                else if(d==10){
                    file[10]='P';
                    file[11]='_';
                }
```

```
                else if(d==11){
                    file[10]='M';
                    file[11]='_';
                }
                else if(d==12){
                    file[10]='D';
                    file[11]='_';
                }
                else if(d==13){
                    file[10]='G';
                    file[11]='_';
                }
                for(dn=1;dn<=20;dn++){
                    if(dn>=10){
                        file[12]=dn/10+'0';
                        file[13]=dn%10+'0';
                        file[14]='\0';
                    }
                    else{
                        file[12]=dn+'0';
                        file[13]='\0';
                    }
                    strcat(file,".txt");
                    speech=fopen(file,"r");          //Opening
file which are in 214101055_vowel_number this formate
                    if(speech!=NULL){
                        for(i=0;i<5;i++){

fgets(t,sizeof(t),speech);          //Skip first 5 lines as it
header
                        }
                        i=1;
                        MaxVal=0;
                        while(!feof(speech)){
                            fscanf(speech,"%lf",&sample[i]); //
Scaning from file
                            sample[i]=sample[i]+DC_shift;
                            if(abs(sample[i])>MaxVal){
                                MaxVal=abs(sample[i]);
                            }
                            i++;
                        }
                        fclose(speech);
                        NumberOfValue=i-1;
                        TotalSegment=(NumberOfValue-320)/80;
                        for(i=1;i<=NumberOfValue;i++){
    //Doing normalised
                            sample[i]=(sample[i]-
DC_shift)*NormalisedAmpl/MaxVal;
                        }
                        Threshold=0;
```

```
                        l=Segment;
                        for(i=1;i<=TotalSegment;i++){
    //Calculating per segment energy
                            l=80*(i-1);
                            temp=0;
                            j=1;
                            while(j<=Segment){
                                temp=temp+
(sample[l+j]*sample[l+j]);          //Calculating total energy
per segment
                                j++;
                            }
                            AvgEnergy[i]=temp/Segment;
            //Calculate average energy per segment
                        }
                        Threshold=0;
                        for(i=1;i<=5;i++){
                            Threshold+=AvgEnergy[i];
                        }
                        Threshold/=5;
                        Threshold*=10;
                        fstart=0;
                        for(i=2;i<TotalSegment-2;i++){
    //Take hight energy fream as Steady point
                            if(!fstart && AvgEnergy[i]>Threshold
&& AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                                start=i;
                                fstart=1;
                                break;
                            }
                            //if(!fstart &&
AvgEnergy[i]<Threshold && AvgEnergy[i+1]<Threshold &&
AvgEnergy[i+2]<Threshold){
                            //  end=i;

                            //}
                        }
                        end=start+80;
                        //SteadyPoint=SteadyPoint-1;
                        for(fream=start;fream<=end;fream++){
                            //Calculating Ri
                            for(i=0;i<=12;i++){
//Calculation of Ri
                                r[i]=0;
                                for(j=0;j<=319-i;j++){
                                    r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                                }
                            }
                            //Calculation complete for Ri
                            //Calculation start for Ai
```

```
                                e[0]=r[0];
                                for(i=1;i<=12;i++){
                                    sum=0;
                                    for(j=1;j<=i-1;j++){
                                        sum=sum+ai[i-1][j]*r[i-j];
                                    }
                                    if(i==1){
                                        k[i]=r[1]/r[0];
                                    }
                                    else{
                                        k[i]=(r[i]-sum)/e[i-1];
                                    }
                                    ai[i][i]=k[i];
                                    for(j=1;j<=i-1;j++){
                                        ai[i][j]=ai[i-1][j]-
k[i]*ai[i-1][i-j];
                                    }
                                    e[i]=(1-k[i]*k[i])*e[i-1];
                                }
                                //Calculation complete for Ai
                                //Calculation start for Ci
                                c[d][dn][fream-start+1]
[0]=2*log(r[0]);
                                for(i=1;i<=12;i++){
                                    temp=0;
                                    for(j=1;j<=i-1;j++){
    //To calculate Ci,we taken a 4D array in fornamt C[vowel]
[VowelFile][Fream][i]
                                        temp=temp+(j*c[d][dn][fream-
start+1][j]*(ai[12][i-j]))/i;
                                    }
                                    c[d][dn][fream-start+1]
[i]=(ai[12][i])+temp;
                                }
                                //Calculation complete for Ci
                                //Apply Raised sine window
                                for(m=1;m<=12;m++){
                                    c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
                                    //fprintf(fc,"%f\t",c[d][dn]
[fream-start+1][m]);
                                }
                                //fprintf(fc,"\n");
                            }
                            //c[d][dn][0][0]=end-start+1;
                        }

                    }
                }
                //fclose(fc);
                char buffer[1024],record[50],*line;
```

```
            double A[33][13];
            f=fopen("codebook.txt","r");
            if(f!=NULL){
                int b;
                i=1;
                while(!feof(f)){
                    fgets(buffer,sizeof(buffer),f);      //Take
line in buffer
                    record[0]='\0';
                    l=1;
                    b=0;
                    for(j=0;buffer[j]!='\0';j++){        //For
each semecolon we divided the string
                        if(buffer[j]=='\t'){
                            record[b]='\0';
                            A[i]
[l]=strtod(record,&line);        //String to double convert
                            record[0]='\0';
                            l++;
                            b=0;
                        }
                        else{
                            record[b]=buffer[j];         //Untill
semecolon is encounter we store the string
                            b++;
                        }
                    }
                    record[b]='\0';
                    A[i][l]=strtod(record,&line);
                    record[0]='\0';
                    l++;
                    b=0;
                    i++;
                }
                fclose(f);
                double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
                int index=0,CBindex;
                for(d=0;d<=13;d++){
                    file[0]='\0';
                    strcat(file,"LiObs/obs_seq_");
                    if(d==0){
                        file[14]='0';
                    }
                    else if(d==1){
                        file[14]='1';
                    }
                    else if(d==2){
                        file[14]='2';
```

```
                    }
                    else if(d==3){
                        file[14]='3';
                    }
                    else if(d==4){
                        file[14]='4';
                    }
                    else if(d==5){
                        file[14]='5';
                    }
                    else if(d==6){
                        file[14]='6';
                    }
                    else if(d==7){
                        file[14]='7';
                    }
                    else if(d==8){
                        file[14]='8';
                    }
                    else if(d==9){
                        file[14]='9';
                    }
                    else if(d==10){
                        file[14]='P';
                    }
                    else if(d==11){
                        file[14]='M';
                    }
                    else if(d==12){
                        file[14]='D';
                    }
                    else if(d==13){
                        file[14]='G';
                    }
                    file[15]='\0';
                    strcat(file,".txt");
                    f=fopen(file,"w");
                    for(dn=1;dn<=20;dn++){
                        if(f!=NULL){

fprintf(f,"----------------------------\n");
                            for(fream=1;fream<=80;fream++){
                                dmin=DBL_MAX;
                                for(j=1;j<=32;j++){
                                    sum=0;
                                    for(i=1;i<=12;i++){
                                        x=A[j][i]-c[d][dn][fream]
[i];
                                        sum=sum+w[i]*x*x;
//Applying the formula W[i] given in question
                                    }
```

```
                                    if(dmin>sum){
                                        dmin=sum;
                                        CBindex=j;
                                    }
                                }
                                fprintf(f,"%d ",CBindex);
                            }
                            fprintf(f,"\n");
                        }
                    }
                    fclose(f);
                    printf("Save obs sequence %d...\n",d);
                }
            }
            //
**********************************************************************
            printf("Training all 9 model\n");
            //scanf("%d",&input);
            for(input=0;input<=13;input++){
                printf("************************\n");
                printf("Training model %d....\n",input);
                if(input==0){
                    //fa=fopen("A_1.txt","r");
                    //fb=fopen("B_1.txt","r");
                    fo=fopen("LiObs/obs_seq_0.txt","r");
                    //fpi=fopen("Pi_1.txt","r");
                }
                else if(input==1){
                    //fa=fopen("A_7.txt","r");
                    //fb=fopen("B_7.txt","r");
                    fo=fopen("LiObs/obs_seq_1.txt","r");
                    //fpi=fopen("Pi_7.txt","r");
                }
                else if(input==2){
                    //fa=fopen("A_8.txt","r");
                    //fb=fopen("B_8.txt","r");
                    fo=fopen("LiObs/obs_seq_2.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                else if(input==3){
                    //fa=fopen("A_8.txt","r");
                    //fb=fopen("B_8.txt","r");
                    fo=fopen("LiObs/obs_seq_3.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                else if(input==4){
                    //fa=fopen("A_8.txt","r");
                    //fb=fopen("B_8.txt","r");
                    fo=fopen("LiObs/obs_seq_4.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
```

```
            else if(input==5){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_5.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==6){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_6.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==7){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_7.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==8){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_8.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==9){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_9.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==10){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_P.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==11){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_M.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==12){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
                fo=fopen("LiObs/obs_seq_D.txt","r");
                //fpi=fopen("Pi_8.txt","r");
            }
            else if(input==13){
                //fa=fopen("A_8.txt","r");
                //fb=fopen("B_8.txt","r");
```

```
                    fo=fopen("LiObs/obs_seq_G.txt","r");
                    //fpi=fopen("Pi_8.txt","r");
                }
                if(fo!=NULL){
                    //Read A matrix
                    /*i=1;
                    while(!feof(fa)){
                        fgets(buffer, sizeof(buffer), fa);
                        //fscanf(fa,"%s",buffer);        //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]=='\t'){
                                record[s]='\0';
                                a[i][l]=strtod(record,&line);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        a[i][l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fa);
                    //-----------------------------------------
                    //Read B matrix
                    i=1;
                    while(!feof(fb)){
                        fgets(buffer, sizeof(buffer), fb);
                        //fscanf(fa,"%s",buffer);        //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]=='\t'){
                                record[s]='\0';
```

```
                                b[i][l]=strtod(record,&line);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        b[i][l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fb);*/
                    //-----------------------------------------
--
                    //Read observation matrix
                    i=1;
                    skip=1;
                    while(!feof(fo)){
                        fgets(buffer, sizeof(buffer), fo);
                        if(skip%2==1){
                            skip=skip+1;
                            continue;
                        }
                        skip=skip+1;
                        //fscanf(fa,"%s",buffer);         //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]==' '){
                                record[s]='\0';
                                O[i][l]=strtol(record,&line,10);
//String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
```

```
                            }
                        }
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fo);
                    //-----------------------------------------
-------
                    //Read Pi matrix
                    /*i=1;
                    skip=1;
                    while(!feof(fpi)){
                        fgets(buffer, sizeof(buffer), fpi);
                        //fscanf(fa,"%s",buffer);        //Take
first line in buffer as string
                        record[0]='\0';
                        l=1;
                        s=0;
                        for(j=0;buffer[j]!='\0';j++){         //
Take string and seperate for each ;
                            if(buffer[j]==' '){
                                record[s]='\0';

pi[l]=strtod(record,&line);      //String to double convert
                                record[0]='\0';
                                l++;
                                s=0;
                            }
                            else{
                                record[s]=buffer[j];          //
Untill tab is encounter we store string in record
                                s++;
                            }
                        }
                        record[s]='\0';
                        pi[l]=strtod(record,&line);
                        record[0]='\0';
                        l++;
                        s=0;
                        i++;
                    }
                    fclose(fpi);*/
                    //------------Pi----------------------
                    for(i=1;i<=N;i++){
                        if(i==1){
                            pi[i]=1;
                            //pi_temp[i]=1;
```

```
                    }
                    else{
                        pi[i]=0;
                        //pi_temp[i]=0;
                    }
                }
                for(i=1;i<=N;i++){
                    for(j=1;j<=N;j++){
                        if(i==j && i==N){
                            a[i][j]=1;
                            //a_temp[i][j]=1;
                        }
                        else if(i==j){
                            a[i][j]=0.8;
                            //a_temp[i][j]=0.8;
                            j++;
                            a[i][j]=0.2;
                            //a_temp[i][j]=0.2;
                        }
                    }
                }
                for(i=1;i<=N;i++){
                    for(j=1;j<=M;j++){
                        b[i][j]=0.03125;
                        //b_temp[i][j]=0.03125;
                    }
                }
                //fw=fopen("Output.txt","w");
                //-------------------------------------
                int Overite=0;
                while(Overite!=4){
                    Overite++;
                    for(obsNum=1;obsNum<=OB;obsNum++){
                        for(i=1;i<=N;i++){
                            for(j=1;j<=N;j++){
                                a_New[i][j]=a[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            for(j=1;j<=M;j++){
                                b_New[i][j]=b[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            pi_New[i]=pi[i];
                        }
                        //fprintf(fw,"---------
Observation%d----------\n",obsNum);
                        //fprintf(fw,"Total iteration is
200\n");
                        //P_Star=Compute_P_Star(obsNum);
```

```
                                /*for(i=T;i>=1;i++){
                                    q_Star_T=Si[t+1][
                                }*/
                                //printf("For observation sequence
%d, P* is %Le \n",obsNum,P_Star);
                                //printf("Sequence are : %d
",q_Star_T);

                                //Compute_Alpha(obsNum);
                                /*temp=0;
                                for(i=1;i<=N;i++){
                                    temp=temp+Alpha[T][i];
                                }*/
                                //printf("For observation sequence
%d, P(O/Lamda)= %Le using forward\n",obsNum,temp);
                                //Compute_Bita(obsNum);
                                /*temp=0;
                                for(i=1;i<=N;i++){
                                    temp=temp+pi[i]*b_New[i]
[O[obsNum][1]]*Bita[1][i];
                                }*/
                                //printf("For observation sequence
%d, P(O/Lamda)= %Le using backward\n\n\n\n",obsNum,temp);
                                /*for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
                                        a_Old[i][j]=a[i][j];
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=M;j++){
                                        b_Old[i][j]=b[i][j];
                                    }
                                }*/
                                //---------------Zei
computing------------------
                                /*Compute_Zie(obsNum);
                                //-------------
Complete-------------------------
                                //printf("Gamma matrix is :\n");
                                Compute_Gamma(obsNum);
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
                                        temp=Update_a_(i,j);
                                        if(temp==0){
                                            a_New[i][j]=0;
                                        }
                                        else{
                                            a_New[i][j]=temp;
                                        }
                                    }
                                }
```

```
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=M;j++){
                                        temp=Update_b_(i,j,obsNum);
                                        if(temp==0){
                                            b_New[i][j]=1e-30;
                                        }
                                        else{
                                            b_New[i][j]=temp;
                                        }
                                    }
                                }
                                Update_pi_();
                                //--------------
adjust----------------
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=a_New[i][1];
                                    index=1;
                                    for(j=1;j<=N;j++){
                                        sum+=a_New[i][j];
                                        if(maxval<a_New[i][j]){
                                            maxval=a_New[i][j];
                                            index=j;
                                        }
                                    }
                                    if(sum>=1){
                                        a_New[i][index]=a_New[i]
[index]-(sum-1);
                                    }
                                    else{
                                        a_New[i][index]=a_New[i]
[index]+(1-sum);
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=b_New[i][1];
                                    index=1;
                                    for(j=1;j<=M;j++){
                                        sum+=b_New[i][j];
                                        if(maxval<b_New[i][j]){
                                            maxval=b_New[i][j];
                                            index=j;
                                        }
                                    }
                                    if(sum>=1){
                                        b_New[i][index]=b_New[i]
[index]-(sum-1);
                                    }
                                    else{
```

```
                                    b_New[i][index]=b_New[i]
[index]+(1-sum);
                                }
                            }
                            sum=0;
                            maxval=pi_New[1];
                            index=1;
                            for(i=1;i<=N;i++){
                                sum+=pi_New[i];
                                if(maxval<pi_New[i]){
                                    maxval=pi_New[i];
                                    index=i;
                                }
                            }
                            if(sum>=1){
                                pi_New[index]=pi_New[index]-
(sum-1);
                            }
                            else{
                                pi_New[index]=pi_New[index]+(1-
sum);
                            }
                            P_Star_Old=P_Star;*/
                            P_Star=Compute_P_Star(obsNum);
                            iet=0;
                            while(iet!=5){
                                iet++;
                                //printf("hello");
                                Compute_Alpha(obsNum);
                                Compute_Bita(obsNum);
                                //---------------Zei
computing-----------------
                                Compute_Zie(obsNum);
                                //-------------
Complete-------------------------
                                Compute_Gamma(obsNum);
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
                                        temp=Update_a_(i,j);
                                        if(temp==0){
                                            a_New[i][j]=0;
                                        }
                                        else{
                                            a_New[i][j]=temp;
                                        }
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=M;j++){
temp=Update_b_(i,j,obsNum);
```

```
                                        if(temp==0){
                                            b_New[i][j]=1e-30;
                                        }
                                        else{
                                            b_New[i][j]=temp;
                                        }
                                    }
                                }
                                Update_pi_();
                                //--------------
adjust----------------
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=a_New[i][1];
                                    index=1;
                                    for(j=1;j<=N;j++){
                                        sum+=a_New[i][j];
                                        if(maxval<a_New[i][j]){
                                            maxval=a_New[i][j];
                                            index=j;
                                        }
                                    }
                                    if(sum>=1){
                                        a_New[i][index]=a_New[i]
[index]-(sum-1);
                                    }
                                    else{
                                        a_New[i][index]=a_New[i]
[index]+(1-sum);
                                    }
                                }
                                for(i=1;i<=N;i++){
                                    sum=0;
                                    maxval=b_New[i][1];
                                    index=1;
                                    for(j=1;j<=M;j++){
                                        sum+=b_New[i][j];
                                        if(maxval<b_New[i][j]){
                                            maxval=b_New[i][j];
                                            index=j;
                                        }
                                    }
                                    if(sum>=1){
                                        b_New[i][index]=b_New[i]
[index]-(sum-1);
                                    }
                                    else{
                                        b_New[i][index]=b_New[i]
[index]+(1-sum);
                                    }
                                }
```

```
                                    sum=0;
                                    maxval=pi_New[1];
                                    index=1;
                                    for(i=1;i<=N;i++){
                                        sum+=pi_New[i];
                                        if(maxval<pi_New[i]){
                                            maxval=pi_New[i];
                                            index=i;
                                        }
                                    }
                                    if(sum>=1){
                                        pi_New[index]=pi_New[index]-
(sum-1);
                                    }
                                    else{
                                        pi_New[index]=pi_New[index]
+(1-sum);
                                    }
                                    P_Star_Old=P_Star;
                                    P_Star=Compute_P_Star(obsNum);
                                    //printf("For observation
sequence %d,Optimal P* is %Le \n",obsNum,P_Star);
                                }
                                //printf("For observation sequence
%d,Optimal P* is %Le \n",obsNum,P_Star);
                                //fprintf(fw,"Optimal
P*=%Le\n",P_Star);
                                //printf("Final state sequence
is :");
                                //fprintf(fw,"Finalstate sequence
is :");
                                print[T]=q_Star_T;
                                for(i=T-1;i>=1;i--){
                                    q_Star_T=Si[i][q_Star_T];
                                    print[i]=q_Star_T;
                                    //printf(", %d",q_Star_T);
                                }
                                /*for(i=1;i<=T;i++){
                                    if(i!=T){
                                        printf("%d ,",print[i]);
                                        fprintf(fw,"%d ,",print[i]);
                                    }
                                    else{
                                        printf("%d",print[i]);
                                        fprintf(fw,"%d\n",print[i]);
                                    }
                                }
                                printf("\n");
                                fprintf(fw,"Matrix a:\n");
                                for(i=1;i<=N;i++){
                                    for(j=1;j<=N;j++){
```

```
                                    fprintf(fw,"%f ",a_New[i]
[j]);
                                }
                                fprintf(fw,"\n");
                            }
                            fprintf(fw,"\n");
                            fprintf(fw,"Matrix b:\n");
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    fprintf(fw,"%Le ",b_New[i]
[j]);
                                }
                                fprintf(fw,"\n");
                            }*/
                            for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    a_Avg[obsNum][i][j]=a_New[i]
[j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    b_Avg[obsNum][i][j]=b_New[i]
[j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                pi_Avg[obsNum][i]=pi_New[i];
                            }
                            /*printf("Matrix is :\n");
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    printf("%Le ",b_New[i][j]);
                                }
                                printf("\n");
                            }*/
                        }

                        for(i=1;i<=N;i++){
                            for(j=1;j<=N;j++){
                                temp=0;
                                for(obsNum=1;obsNum<=OB;obsNum++)
{
                                    temp+=a_Avg[obsNum][i][j];
                                }
                                a[i][j]=temp/OB;
                            }
                        }
                        for(i=1;i<=N;i++){
                            for(j=1;j<=M;j++){
                                temp=0;
```

```
                                for(obsNum=1;obsNum<=OB;obsNum++)
{
                                    temp+=b_Avg[obsNum][i][j];
                                }
                                b[i][j]=temp/OB;
                            }
                        }
                        for(i=1;i<=N;i++){
                            temp=0;
                            for(obsNum=1;obsNum<=OB;obsNum++){
                                temp+=pi_Avg[obsNum][i];
                            }
                            pi[i]=temp/OB;
                        }
                        printf("Model updating %d....\n",Overite);
                    }
                    //fclose(fw);
                    if(input==0){
                        fa=fopen("New_Model/Digit0/A_0.txt","w");
                        fb=fopen("New_Model/Digit0/B_0.txt","w");
                        fpi=fopen("New_Model/Digit0/
Pi_0.txt","w");
                    }
                    else if(input==1){
                        fa=fopen("New_Model/Digit1/A_1.txt","w");
                        fb=fopen("New_Model/Digit1/B_1.txt","w");
                        fpi=fopen("New_Model/Digit1/
Pi_1.txt","w");
                    }
                    else if(input==2){
                        fa=fopen("New_Model/Digit2/A_2.txt","w");
                        fb=fopen("New_Model/Digit2/B_2.txt","w");
                        fpi=fopen("New_Model/Digit2/
Pi_2.txt","w");
                    }
                    else if(input==3){
                        fa=fopen("New_Model/Digit3/A_3.txt","w");
                        fb=fopen("New_Model/Digit3/B_3.txt","w");
                        fpi=fopen("New_Model/Digit3/
Pi_3.txt","w");
                    }
                    else if(input==4){
                        fa=fopen("New_Model/Digit4/A_4.txt","w");
                        fb=fopen("New_Model/Digit4/B_4.txt","w");
                        fpi=fopen("New_Model/Digit4/
Pi_4.txt","w");
                    }
                    else if(input==5){
                        fa=fopen("New_Model/Digit5/A_5.txt","w");
                        fb=fopen("New_Model/Digit5/B_5.txt","w");
```

```
                        fpi=fopen("New_Model/Digit5/
Pi_5.txt","w");
                    }
                    else if(input==6){
                        fa=fopen("New_Model/Digit6/A_6.txt","w");
                        fb=fopen("New_Model/Digit6/B_6.txt","w");
                        fpi=fopen("New_Model/Digit6/
Pi_6.txt","w");
                    }
                    else if(input==7){
                        fa=fopen("New_Model/Digit7/A_7.txt","w");
                        fb=fopen("New_Model/Digit7/B_7.txt","w");
                        fpi=fopen("New_Model/Digit7/
Pi_7.txt","w");
                    }
                    else if(input==8){
                        fa=fopen("New_Model/Digit8/A_8.txt","w");
                        fb=fopen("New_Model/Digit8/B_8.txt","w");
                        fpi=fopen("New_Model/Digit8/
Pi_8.txt","w");
                    }
                    else if(input==9){
                        fa=fopen("New_Model/Digit9/A_9.txt","w");
                        fb=fopen("New_Model/Digit9/B_9.txt","w");
                        fpi=fopen("New_Model/Digit9/
Pi_9.txt","w");
                    }
                    else if(input==10){
                        fa=fopen("New_Model/DigitP/A_P.txt","w");
                        fb=fopen("New_Model/DigitP/B_P.txt","w");
                        fpi=fopen("New_Model/DigitP/
Pi_P.txt","w");
                    }
                    else if(input==11){
                        fa=fopen("New_Model/DigitM/A_M.txt","w");
                        fb=fopen("New_Model/DigitM/B_M.txt","w");
                        fpi=fopen("New_Model/DigitM/
Pi_M.txt","w");
                    }
                    else if(input==12){
                        fa=fopen("New_Model/DigitD/A_D.txt","w");
                        fb=fopen("New_Model/DigitD/B_D.txt","w");
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","w");
                    }
                    else if(input==13){
                        fa=fopen("New_Model/DigitG/A_G.txt","w");
                        fb=fopen("New_Model/DigitG/B_G.txt","w");
                        fpi=fopen("New_Model/DigitG/
Pi_G.txt","w");
                    }
```

```
                    for(i=1;i<=N;i++){
                        for(j=1;j<=N;j++){
                            fprintf(fa,"%f ",a[i][j]);
                        }
                        fprintf(fa,"\n");
                    }
                    fclose(fa);

                    for(i=1;i<=N;i++){
                        for(j=1;j<=M;j++){
                            fprintf(fb,"%Le ",b[i][j]);
                        }
                        fprintf(fb,"\n");
                    }
                    fclose(fb);

                    for(i=1;i<=N;i++){
                        fprintf(fpi,"%f ",pi[i]);
                    }
                    fclose(fpi);
                    printf("Save model in Output filder\n");
                }
            }

    }

}
int FileTesting(char namef[200]){
    int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
    char buffer[1024],record[3000],*line,file[100];
    long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
    FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
    if(1){
        char t[100];
            speech=fopen(namef,"r");          //Opening file which
are in 214101055_vowel_number this formate
            d=1;
            dn=1;
            if(speech!=NULL){
                for(i=0;i<5;i++){
                    fgets(t,sizeof(t),speech);            //Skip
first 5 lines as it header
                }
                i=1;
                MaxVal=0;
                while(!feof(speech)){
```

```
                    fscanf(speech,"%lf",&sample[i]); //Scaning
from file
                    sample[i]=sample[i]+DC_shift;
                    if(abs(sample[i])>MaxVal){
                        MaxVal=abs(sample[i]);
                    }
                    i++;
                }
                fclose(speech);
                NumberOfValue=i-1;
                TotalSegment=(NumberOfValue-320)/80;
                for(i=1;i<=NumberOfValue;i++)
{                   //Doing normalised
                    sample[i]=(sample[i]-DC_shift)*NormalisedAmpl/
MaxVal;
                }
                Threshold=0;
                l=Segment;
                for(i=1;i<=TotalSegment;i++)
{                   //Calculating per segment energy
                    l=80*(i-1);
                    temp=0;
                    j=1;
                    while(j<=Segment){
                        temp=temp+(sample[l+j]*sample[l+j]);
//Calculating total energy per segment
                        j++;
                    }
                    AvgEnergy[i]=temp/Segment;
    //Calculate average energy per segment
                }
                Threshold=0;
                for(i=1;i<=5;i++){
                    Threshold+=AvgEnergy[i];
                }
                Threshold/=5;
                Threshold*=10;
                fstart=0;
                for(i=2;i<TotalSegment-2;i++)
{                   //Take hight energy fream as Steady point
                    if(!fstart && AvgEnergy[i]>Threshold &&
AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                        start=i;
                        fstart=1;
                        break;
                    }
                    //if(!fstart && AvgEnergy[i]<Threshold &&
AvgEnergy[i+1]<Threshold && AvgEnergy[i+2]<Threshold){
                    //  end=i;

                    //}
```

```
                }
                end=start+80;
                //SteadyPoint=SteadyPoint-1;
                for(fream=start;fream<=end;fream++){
                    //Calculating Ri
                    for(i=0;i<=12;i++){                         //
Calculation of Ri
                        r[i]=0;
                        for(j=0;j<=319-i;j++){
                            r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                        }
                    }
                    //Calculation complete for Ri
                    //Calculation start for Ai
                    e[0]=r[0];
                    for(i=1;i<=12;i++){
                        sum=0;
                        for(j=1;j<=i-1;j++){
                            sum=sum+ai[i-1][j]*r[i-j];
                        }
                        if(i==1){
                            k[i]=r[1]/r[0];
                        }
                        else{
                            k[i]=(r[i]-sum)/e[i-1];
                        }
                        ai[i][i]=k[i];
                        for(j=1;j<=i-1;j++){
                            ai[i][j]=ai[i-1][j]-k[i]*ai[i-1][i-
j];
                        }
                        e[i]=(1-k[i]*k[i])*e[i-1];
                    }
                    //Calculation complete for Ai
                    //Calculation start for Ci
                    c[d][dn][fream-start+1][0]=2*log(r[0]);
                    for(i=1;i<=12;i++){
                        temp=0;
                        for(j=1;j<=i-1;j++)
{                           //To calculate Ci,we taken a 4D array in
fornamt C[vowel][VowelFile][Fream][i]
                            temp=temp+(j*c[d][dn][fream-start+1]
[j]*(ai[12][i-j]))/i;
                        }
                        c[d][dn][fream-start+1][i]=(ai[12][i])
+temp;
                    }
                    //Calculation complete for Ci
                    //Apply Raised sine window
                    for(m=1;m<=12;m++){
```

```
                        c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
                        //fprintf(fc,"%f\t",c[d][dn][fream-
start+1][m]);
                    }
                    //fprintf(fc,"\n");
                }
                //c[d][dn][0][0]=end-start+1;
            }
            //fclose(fc);
            char buffer[1024],record[50],*line;

            double A[33][13];
            f=fopen("codebook.txt","r");
            if(f!=NULL){
                int b;
                i=1;
                while(!feof(f)){
                    fgets(buffer,sizeof(buffer),f);       //Take
line in buffer
                    record[0]='\0';
                    l=1;
                    b=0;
                    for(j=0;buffer[j]!='\0';j++){         //For
each semecolon we divided the string
                        if(buffer[j]=='\t'){
                            record[b]='\0';
                            A[i]
[l]=strtod(record,&line);         //String to double convert
                            record[0]='\0';
                            l++;
                            b=0;
                        }
                        else{
                            record[b]=buffer[j];          //Untill
semecolon is encounter we store the string
                            b++;
                        }
                    }
                    record[b]='\0';
                    A[i][l]=strtod(record,&line);
                    record[0]='\0';
                    l++;
                    b=0;
                    i++;
                }
                fclose(f);
                double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
                int index=0,CBindex;
```

```
                f=fopen("Offline_test/Live_Test_obs_seq.txt","w");
                if(f!=NULL){
                    fprintf(f,"--------------------------\n");
                    for(fream=1;fream<=80;fream++){
                        dmin=DBL_MAX;
                        for(j=1;j<=32;j++){
                            sum=0;
                            for(i=1;i<=12;i++){
                                x=A[j][i]-c[d][dn][fream][i];
                                sum=sum+w[i]*x*x;            //
Applying the formula W[i] given in question
                            }
                            if(dmin>sum){
                                dmin=sum;
                                CBindex=j;
                            }
                        }
                        fprintf(f,"%d ",CBindex);
                    }
                    fprintf(f,"\n");
                }
                fclose(f);
            }
            fo=fopen("Live_test/Live_Test_obs_seq.txt","r");
            i=1;
            skip=1;
            while(!feof(fo)){
                fgets(buffer, sizeof(buffer), fo);
                if(skip%2==1){
                    skip=skip+1;
                    continue;
                }
                skip=skip+1;
                //fscanf(fa,"%s",buffer);        //Take first line
in buffer as string
                record[0]='\0';
                l=1;
                s=0;
                for(j=0;buffer[j]!='\0';j++){        //Take string
and seperate for each ;
                    if(buffer[j]==' '){
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);     //
String to double convert
                        record[0]='\0';
                        l++;
                        s=0;
                    }
                    else{
                        record[s]=buffer[j];         //Untill tab
is encounter we store string in record
```

```
                        s++;
                    }
                }
                record[s]='\0';
                O[i][l]=strtol(record,&line,10);
                record[0]='\0';
                l++;
                s=0;
                i++;
            }
            fclose(fo);
            count=0;
                minp=0;
                for(input=0;input<=9;input++){
                    if(input==0){
                        fa=fopen("Output_Model/Digit0/
A_0.txt","r");
                        fb=fopen("Output_Model/Digit0/
B_0.txt","r");
                        fpi=fopen("Output_Model/Digit0/
Pi_0.txt","r");
                    }
                    else if(input==1){
                        fa=fopen("Output_Model/Digit1/
A_1.txt","r");
                        fb=fopen("Output_Model/Digit1/
B_1.txt","r");
                        fpi=fopen("Output_Model/Digit1/
Pi_1.txt","r");
                    }
                    else if(input==2){
                        fa=fopen("Output_Model/Digit2/
A_2.txt","r");
                        fb=fopen("Output_Model/Digit2/
B_2.txt","r");
                        fpi=fopen("Output_Model/Digit2/
Pi_2.txt","r");
                    }
                    else if(input==3){
                        fa=fopen("Output_Model/Digit3/
A_3.txt","r");
                        fb=fopen("Output_Model/Digit3/
B_3.txt","r");
                        fpi=fopen("Output_Model/Digit3/
Pi_3.txt","r");
                    }
                    else if(input==4){
                        fa=fopen("Output_Model/Digit4/
A_4.txt","r");
                        fb=fopen("Output_Model/Digit4/
B_4.txt","r");
```

```
                        fpi=fopen("Output_Model/Digit4/
Pi_4.txt","r");
                    }
                    else if(input==5){
                        fa=fopen("Output_Model/Digit5/
A_5.txt","r");
                        fb=fopen("Output_Model/Digit5/
B_5.txt","r");
                        fpi=fopen("Output_Model/Digit5/
Pi_5.txt","r");
                    }
                    else if(input==6){
                        fa=fopen("Output_Model/Digit6/
A_6.txt","r");
                        fb=fopen("Output_Model/Digit6/
B_6.txt","r");
                        fpi=fopen("Output_Model/Digit6/
Pi_6.txt","r");
                    }
                    else if(input==7){
                        fa=fopen("Output_Model/Digit7/
A_7.txt","r");
                        fb=fopen("Output_Model/Digit7/
B_7.txt","r");
                        fpi=fopen("Output_Model/Digit7/
Pi_7.txt","r");
                    }
                    else if(input==8){
                        fa=fopen("Output_Model/Digit8/
A_8.txt","r");
                        fb=fopen("Output_Model/Digit8/
B_8.txt","r");
                        fpi=fopen("Output_Model/Digit8/
Pi_8.txt","r");
                    }
                    else if(input==9){
                        fa=fopen("Output_Model/Digit9/
A_9.txt","r");
                        fb=fopen("Output_Model/Digit9/
B_9.txt","r");
                        fpi=fopen("Output_Model/Digit9/
Pi_9.txt","r");
                    }
                    else if(input==10){
                        fa=fopen("Output_Model/DigitP/
A_P.txt","r");
                        fb=fopen("Output_Model/DigitP/
B_P.txt","r");
                        fpi=fopen("Output_Model/DigitP/
Pi_P.txt","r");
                    }
```

```
                    else if(input==11){
                        fa=fopen("Output_Model/DigitM/
A_M.txt","r");
                        fb=fopen("Output_Model/DigitM/
B_M.txt","r");
                        fpi=fopen("Output_Model/DigitM/
Pi_M.txt","r");
                    }
                    else if(input==12){
                        fa=fopen("Output_Model/DigitD/
A_D.txt","r");
                        fb=fopen("Output_Model/DigitD/
B_D.txt","r");
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","r");
                    }
                    else if(input==13){
                        fa=fopen("Output_Model/DigitG/
A_G.txt","r");
                        fb=fopen("Output_Model/DigitG/
B_G.txt","r");
                        fpi=fopen("Output_Model/DigitG/
Pi_G.txt","r");
                    }
                    if(fa!=NULL && fb!=NULL && fpi!=NULL){
                        //Read A matrix
                        i=1;
                        while(!feof(fa)){
                            fgets(buffer, sizeof(buffer), fa);
                            //fscanf(fa,"%s",buffer);        //
Take first line in buffer as string
                            record[0]='\0';
                            l=1;
                            s=0;
                            for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                if(buffer[j]==' '){
                                    record[s]='\0';
                                    a[i][l]=strtod(record,&line);
    //String to double convert
                                    record[0]='\0';
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
```

```
                                record[s]='\0';
                                a[i][l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fa);
                            //-----------------------------------------
--
                            //Read B matrix
                            i=1;
                            while(!feof(fb)){
                                fgets(buffer, sizeof(buffer), fb);
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        b[i][l]=strtod(record,&line);
    //String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                b[i][l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fb);
                            //-----------------------------------------
------
                            //Read observation matrix

                            //-----------------------------------------
-----------
                            //Read Pi matrix
```

```
                        i=1;
                        skip=1;
                        while(!feof(fpi)){
                            fgets(buffer, sizeof(buffer), fpi);
                            //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                            record[0]='\0';
                            l=1;
                            s=0;
                            for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                if(buffer[j]==' '){
                                    record[s]='\0';
                                    pi[l]=strtod(record,&line);
//String to double convert
                                    record[0]='\0';
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
                            record[s]='\0';
                            pi[l]=strtod(record,&line);
                            record[0]='\0';
                            l++;
                            s=0;
                            i++;
                        }
                        fclose(fpi);
                        for(i=1;i<=N;i++){
                            for(j=1;j<=N;j++){
                                a_New[i][j]=a[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            for(j=1;j<=M;j++){
                                b_New[i][j]=b[i][j];
                            }
                        }
                        for(i=1;i<=N;i++){
                            pi_New[i]=pi[i];
                        }
                        Compute_Alpha(1);
                        temp=0;
                        for(i=1;i<=N;i++){
                            temp=temp+Alpha[T][i];
```

```
                        }
                        if(minp<temp){
                            minp=temp;
                            index=input;
                        }
                    }
                    //printf("P(O/Lamda)=%g for digits
%d\n",temp,input);
                }
                if(d==index){
                    count++;
                }
                //printf("Prediction is %d\n",index);
                //printf("%d\n",s);
            //printf("Accuracy is %d\n",count*10);
            //printf("**************************\n");
    }
    return index;
}
int FileTestingOP(char namef[200]){
    int
i,j,s,skip=1,obsNum,t,maxpi,input,p,q,index,iet,d,dn,m,count,testi
nput,NumberOfValue,TotalSegment,NormalisedAmpl=5000,Segment=320,St
eadyPoint,l,fream,start,end,fstart,fend;
    char buffer[1024],record[3000],*line,file[100];
    long double
maxp,P_Star,P_Star_Old,maxval,temp1,temp2,minp,DC_shift=0,sum=0,te
mp=0,MaxVal=0,Threshold=0;
    FILE *fa,*fb,*fpi,*fo,*fw,*speech,*f;
    if(1){
        char t[100];
            speech=fopen(namef,"r");          //Opening file which
are in 214101055_vowel_number this formate
            d=1;
            dn=1;
            if(speech!=NULL){
                for(i=0;i<5;i++){
                    fgets(t,sizeof(t),speech);            //Skip
first 5 lines as it header
                }
                i=1;
                MaxVal=0;
                while(!feof(speech)){
                    fscanf(speech,"%lf",&sample[i]); //Scaning
from file
                    sample[i]=sample[i]+DC_shift;
                    if(abs(sample[i])>MaxVal){
                        MaxVal=abs(sample[i]);
                    }
                    i++;
                }
```

```
                fclose(speech);
                NumberOfValue=i-1;
                TotalSegment=(NumberOfValue-320)/80;
                for(i=1;i<=NumberOfValue;i++)
{                   //Doing normalised
                    sample[i]=(sample[i]-DC_shift)*NormalisedAmpl/
MaxVal;
                }
                Threshold=0;
                l=Segment;
                for(i=1;i<=TotalSegment;i++)
{                   //Calculating per segment energy
                    l=80*(i-1);
                    temp=0;
                    j=1;
                    while(j<=Segment){
                        temp=temp+(sample[l+j]*sample[l+j]);
//Calculating total energy per segment
                        j++;
                    }
                    AvgEnergy[i]=temp/Segment;
    //Calculate average energy per segment
                }
                Threshold=0;
                for(i=1;i<=5;i++){
                    Threshold+=AvgEnergy[i];
                }
                Threshold/=5;
                Threshold*=10;
                fstart=0;
                for(i=2;i<TotalSegment-2;i++)
{                   //Take hight energy fream as Steady point
                    if(!fstart && AvgEnergy[i]>Threshold &&
AvgEnergy[i+1]>Threshold && AvgEnergy[i+2]>Threshold){
                        start=i;
                        fstart=1;
                        break;
                    }
                    //if(!fstart && AvgEnergy[i]<Threshold &&
AvgEnergy[i+1]<Threshold && AvgEnergy[i+2]<Threshold){
                    //  end=i;

                    //}
                }
                end=start+80;
                //SteadyPoint=SteadyPoint-1;
                for(fream=start;fream<=end;fream++){
                    //Calculating Ri
                    for(i=0;i<=12;i++){                         //
Calculation of Ri
                        r[i]=0;
```

```
                        for(j=0;j<=319-i;j++){
                            r[i]=r[i]
+sample[(fream-1)*80+j]*sample[(fream-1)*80+j+i];
                        }
                    }
                    //Calculation complete for Ri
                    //Calculation start for Ai
                    e[0]=r[0];
                    for(i=1;i<=12;i++){
                        sum=0;
                        for(j=1;j<=i-1;j++){
                            sum=sum+ai[i-1][j]*r[i-j];
                        }
                        if(i==1){
                            k[i]=r[1]/r[0];
                        }
                        else{
                            k[i]=(r[i]-sum)/e[i-1];
                        }
                        ai[i][i]=k[i];
                        for(j=1;j<=i-1;j++){
                            ai[i][j]=ai[i-1][j]-k[i]*ai[i-1][i-
j];
                        }
                        e[i]=(1-k[i]*k[i])*e[i-1];
                    }
                    //Calculation complete for Ai
                    //Calculation start for Ci
                    c[d][dn][fream-start+1][0]=2*log(r[0]);
                    for(i=1;i<=12;i++){
                        temp=0;
                        for(j=1;j<=i-1;j++)
{                       //To calculate Ci,we taken a 4D array in
fornamt C[vowel][VowelFile][Fream][i]
                            temp=temp+(j*c[d][dn][fream-start+1]
[j]*(ai[12][i-j]))/i;
                        }
                        c[d][dn][fream-start+1][i]=(ai[12][i])
+temp;
                    }
                    //Calculation complete for Ci
                    //Apply Raised sine window
                    for(m=1;m<=12;m++){
                        c[d][dn][fream-start+1]
[m]*=(1+6*sin((pie*m)/12));
                        //fprintf(fc,"%f\t",c[d][dn][fream-
start+1][m]);
                    }
                    //fprintf(fc,"\n");
                }
                //c[d][dn][0][0]=end-start+1;
```

```
            }
            //fclose(fc);
            char buffer[1024],record[50],*line;

            double A[33][13];
            f=fopen("codebook.txt","r");
            if(f!=NULL){
                int b;
                i=1;
                while(!feof(f)){
                    fgets(buffer,sizeof(buffer),f);      //Take
line in buffer
                    record[0]='\0';
                    l=1;
                    b=0;
                    for(j=0;buffer[j]!='\0';j++){        //For
each semecolon we divided the string
                        if(buffer[j]=='\t'){
                            record[b]='\0';
                            A[i]
[l]=strtod(record,&line);        //String to double convert
                            record[0]='\0';
                            l++;
                            b=0;
                        }
                        else{
                            record[b]=buffer[j];         //Untill
semecolon is encounter we store the string
                            b++;
                        }
                    }
                    record[b]='\0';
                    A[i][l]=strtod(record,&line);
                    record[0]='\0';
                    l++;
                    b=0;
                    i++;
                }
                fclose(f);
                double
w[13]={0,1.0,3.0,7.0,13.0,19.0,22.0,25.0,33.0,42.0,50.0,56.0,61.0}
,dmin=DBL_MIN,dis,x;
                int index=0,CBindex;
                f=fopen("Offline_test/Live_Test_obs_seq.txt","w");
                if(f!=NULL){
                    fprintf(f,"--------------------------\n");
                    for(fream=1;fream<=80;fream++){
                        dmin=DBL_MAX;
                        for(j=1;j<=32;j++){
                            sum=0;
                            for(i=1;i<=12;i++){
```

```
                                x=A[j][i]-c[d][dn][fream][i];
                                sum=sum+w[i]*x*x;            //
Applying the formula W[i] given in question
                            }
                            if(dmin>sum){
                                dmin=sum;
                                CBindex=j;
                            }
                        }
                        fprintf(f,"%d ",CBindex);
                    }
                    fprintf(f,"\n");
                }
                fclose(f);
            }
            fo=fopen("Live_test/Live_Test_obs_seq.txt","r");
            i=1;
            skip=1;
            while(!feof(fo)){
                fgets(buffer, sizeof(buffer), fo);
                if(skip%2==1){
                    skip=skip+1;
                    continue;
                }
                skip=skip+1;
                //fscanf(fa,"%s",buffer);        //Take first line
in buffer as string
                record[0]='\0';
                l=1;
                s=0;
                for(j=0;buffer[j]!='\0';j++){        //Take string
and seperate for each ;
                    if(buffer[j]==' '){
                        record[s]='\0';
                        O[i][l]=strtol(record,&line,10);     //
String to double convert
                        record[0]='\0';
                        l++;
                        s=0;
                    }
                    else{
                        record[s]=buffer[j];         //Untill tab
is encounter we store string in record
                        s++;
                    }
                }
                record[s]='\0';
                O[i][l]=strtol(record,&line,10);
                record[0]='\0';
                l++;
                s=0;
```

```
                i++;
            }
            fclose(fo);
            count=0;
                minp=0;
                index=10;
                for(input=10;input<=13;input++){
                    if(input==0){
                        fa=fopen("Output_Model/Digit0/
A_0.txt","r");
                        fb=fopen("Output_Model/Digit0/
B_0.txt","r");
                        fpi=fopen("Output_Model/Digit0/
Pi_0.txt","r");
                    }
                    else if(input==1){
                        fa=fopen("Output_Model/Digit1/
A_1.txt","r");
                        fb=fopen("Output_Model/Digit1/
B_1.txt","r");
                        fpi=fopen("Output_Model/Digit1/
Pi_1.txt","r");
                    }
                    else if(input==2){
                        fa=fopen("Output_Model/Digit2/
A_2.txt","r");
                        fb=fopen("Output_Model/Digit2/
B_2.txt","r");
                        fpi=fopen("Output_Model/Digit2/
Pi_2.txt","r");
                    }
                    else if(input==3){
                        fa=fopen("Output_Model/Digit3/
A_3.txt","r");
                        fb=fopen("Output_Model/Digit3/
B_3.txt","r");
                        fpi=fopen("Output_Model/Digit3/
Pi_3.txt","r");
                    }
                    else if(input==4){
                        fa=fopen("Output_Model/Digit4/
A_4.txt","r");
                        fb=fopen("Output_Model/Digit4/
B_4.txt","r");
                        fpi=fopen("Output_Model/Digit4/
Pi_4.txt","r");
                    }
                    else if(input==5){
                        fa=fopen("Output_Model/Digit5/
A_5.txt","r");
```

```
                        fb=fopen("Output_Model/Digit5/
B_5.txt","r");
                        fpi=fopen("Output_Model/Digit5/
Pi_5.txt","r");
                    }
                    else if(input==6){
                        fa=fopen("Output_Model/Digit6/
A_6.txt","r");
                        fb=fopen("Output_Model/Digit6/
B_6.txt","r");
                        fpi=fopen("Output_Model/Digit6/
Pi_6.txt","r");
                    }
                    else if(input==7){
                        fa=fopen("Output_Model/Digit7/
A_7.txt","r");
                        fb=fopen("Output_Model/Digit7/
B_7.txt","r");
                        fpi=fopen("Output_Model/Digit7/
Pi_7.txt","r");
                    }
                    else if(input==8){
                        fa=fopen("Output_Model/Digit8/
A_8.txt","r");
                        fb=fopen("Output_Model/Digit8/
B_8.txt","r");
                        fpi=fopen("Output_Model/Digit8/
Pi_8.txt","r");
                    }
                    else if(input==9){
                        fa=fopen("Output_Model/Digit9/
A_9.txt","r");
                        fb=fopen("Output_Model/Digit9/
B_9.txt","r");
                        fpi=fopen("Output_Model/Digit9/
Pi_9.txt","r");
                    }
                    else if(input==10){
                        fa=fopen("Output_Model/DigitP/
A_P.txt","r");
                        fb=fopen("Output_Model/DigitP/
B_P.txt","r");
                        fpi=fopen("Output_Model/DigitP/
Pi_P.txt","r");
                    }
                    else if(input==11){
                        fa=fopen("Output_Model/DigitM/
A_M.txt","r");
                        fb=fopen("Output_Model/DigitM/
B_M.txt","r");
```

```
                        fpi=fopen("Output_Model/DigitM/
Pi_M.txt","r");
                    }
                    else if(input==12){
                        fa=fopen("Output_Model/DigitD/
A_D.txt","r");
                        fb=fopen("Output_Model/DigitD/
B_D.txt","r");
                        fpi=fopen("Output_Model/DigitD/
Pi_D.txt","r");
                    }
                    else if(input==13){
                        fa=fopen("Output_Model/DigitG/
A_G.txt","r");
                        fb=fopen("Output_Model/DigitG/
B_G.txt","r");
                        fpi=fopen("Output_Model/DigitG/
Pi_G.txt","r");
                    }
                    if(fa!=NULL && fb!=NULL && fpi!=NULL){
                        //Read A matrix
                        i=1;
                        while(!feof(fa)){
                            fgets(buffer, sizeof(buffer), fa);
                            //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                            record[0]='\0';
                            l=1;
                            s=0;
                            for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                if(buffer[j]==' '){
                                    record[s]='\0';
                                    a[i][l]=strtod(record,&line);
    //String to double convert
                                    record[0]='\0';
                                    l++;
                                    s=0;
                                }
                                else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                    s++;
                                }
                            }
                            record[s]='\0';
                            a[i][l]=strtod(record,&line);
                            record[0]='\0';
                            l++;
                            s=0;
```

```
                                i++;
                            }
                            fclose(fa);
                            //-------------------------------------
--
                            //Read B matrix
                            i=1;
                            while(!feof(fb)){
                                fgets(buffer, sizeof(buffer), fb);
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        b[i][l]=strtod(record,&line);
    //String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                b[i][l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fb);
                            //-------------------------------------
------
                            //Read observation matrix

                            //-------------------------------------
-----------
                            //Read Pi matrix
                            i=1;
                            skip=1;
                            while(!feof(fpi)){
                                fgets(buffer, sizeof(buffer), fpi);
```

```
                                //fscanf(fa,"%s",buffer);         //
Take first line in buffer as string
                                record[0]='\0';
                                l=1;
                                s=0;
                                for(j=0;buffer[j]!='\0';j++)
{       //Take string and seperate for each ;
                                    if(buffer[j]==' '){
                                        record[s]='\0';
                                        pi[l]=strtod(record,&line);
//String to double convert
                                        record[0]='\0';
                                        l++;
                                        s=0;
                                    }
                                    else{

record[s]=buffer[j];          //Untill tab is encounter we store
string in record
                                        s++;
                                    }
                                }
                                record[s]='\0';
                                pi[l]=strtod(record,&line);
                                record[0]='\0';
                                l++;
                                s=0;
                                i++;
                            }
                            fclose(fpi);
                            for(i=1;i<=N;i++){
                                for(j=1;j<=N;j++){
                                    a_New[i][j]=a[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                for(j=1;j<=M;j++){
                                    b_New[i][j]=b[i][j];
                                }
                            }
                            for(i=1;i<=N;i++){
                                pi_New[i]=pi[i];
                            }
                            Compute_Alpha(1);
                            temp=0;
                            for(i=1;i<=N;i++){
                                temp=temp+Alpha[T][i];
                            }
                            if(minp<temp){
                                minp=temp;
                                index=input;
```

```
                            }
                        }
                        //printf("P(O/Lamda)=%g for digits
%d\n",temp,input);
                    }
                    if(d==index){
                        count++;
                    }
                    //printf("Prediction is %d\n",index);
                    //printf("%d\n",s);
                //printf("Accuracy is %d\n",count*10);
                //printf("**************************\n");
        }
        return index;
}
```