



河南理工大学
Henan Polytechnic University

教学上机实验报告

课程名称：____语音识别____
任课教师姓名：____晁浩____
学生学号：____311809000608____
学生姓名：____王荣胜____
学生专业班级：____计实验 1801____

2020 ~ 2021 学年 第 二 学期

河南理工大学

教学上机实验报告评价分值标准

序号	评价指标	分值	评价等级及参考分值					评价分
			优	良	中	合格	差	
1	实验报告内容完整充实	10	10	8	7	6	3	
2	实验内容书写规范、字迹工整认真	10	10	8	7	6	3	
3	实验过程叙述详细、概念正确，语言表达准确，结构严谨，条理清楚，逻辑性强，自己努力完成，没有抄袭。	30	30	26	23	20	10	
4	对实验过程中存在的问题分析详细透彻、深刻、全面、规范、，结合实验内容，有自己的个人见解和想法，并能结合该实验提出相关问题，给出解决方法。	30	30	26	23	20	10	
5	实验结果、分析和结论正确无误	20	20	17	15	13	6	
总得分								

签名（签章）：

日期： 年 月 日

河南理工大学教学上机实验报告

上机时间 2021 年 5 月 26 日

实验题目：

MFCC 特征提取

实验目的和要求：

目的：

- 1.掌握双门限端点检测方法
- 2.理解语音信号的分帧与加窗
- 3.了解语音信号的倒谱分析的意义
- 4.掌握 MFCC 参数的计算

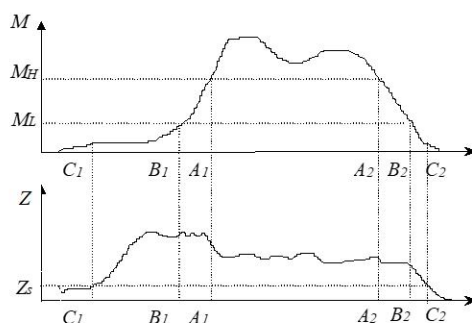
要求：

根据所学知识，能够独立提取一段语音信号的 MFCC 特征。每位同学独立录制数字 0-9 的语音，并提取每段语音的 MFCC 特征序列，观察特征序列的维数。

实验过程：

一、端点检测。

端点检测，也叫语音活动检测，Voice Activity Detection，VAD，它的目的是对语音和非语音的区域进行区分。通俗来理解，端点检测就是为了从带有噪声的语音中准确的定位出语音的开始点，和结束点，去掉静音的部分，去掉噪声的部分，找到一段语音真正有效的内容。



双门限法语音端点检测：

1、第一步是取一个较高的短时能量作为阈值 M_H ，利用这个阈值，我们就可以先分出语音中的浊音部分（如图， A_1 到 A_2 区间）。

2、第二步是取一个较低的能量阈值 M_L ，利用这个阈值，我们可以从 A_1 ， A_2 ，向两端进行搜索，将较低能量段的语音部分也加入到语音段，进一步扩大语音段范围（如图所示， B_1 - B_2 之间还是语音段）。

3、第三步是利用短时过零率，短时过零率的阈值为 Z_s 。由于语音的两端部分是辅音（也就是清音部分），也是语音中的一部分，但是辅音的能量与静音部分的能量一样低，但是过零率比静音部分高出很多。为了区分开二者，将利用短时能量区分完的语音段继续向两端进行搜索，短时过零率大于 3 倍 Z_s 的部分，则认为是语音的清音部分。将该部分加入语言段，就是求得的语音段（如图 C_1 - C_2 部分）。

```

1. %双门限法端点检测
2. clear all; clc; close all;
3.
4. [x,fs]=wavread('C4_1_y.wav'); % 读入数据文件
5. x=x/max(abs(x)); % 幅度归一化
6. N=length(x); % 取信号长度
7. time=(0:N-1)/fs; % 计算时间
8. subplot 311
9. plot(time,x,'k');
10. title('双门限法的端点检测');
11. ylabel('幅值'); axis([0 max(time) -1 1]);
12. xlabel('时间/s');
13. wlen=200; inc=80; % 分帧参数
14. IS=0.1; overlap=wlen-inc; % 设置 IS
15. NIS=fix((IS*fs-wlen)/inc +1); % 计算 NIS
16. fn=fix((N-wlen)/inc)+1; % 求帧数
17. frameTime=FrameTimeC(fn, wlen, inc, fs);% 计算每帧对应的时间
18. [voiceseg,vsl,SF,NF,amp,zcr]=vad_TwoThr(x,wlen,inc,NIS); % 端点检测
19. subplot 312
20. plot(frameTime,amp,'k');
21. ylim([min(amp) max(amp)])
22. title('短时能量');
23. ylabel('幅值');
24. xlabel('时间/s');
25. subplot 313
26. plot(frameTime,zcr,'k');
27. ylim([min(zcr) max(zcr)])
28. title('短时过零率');
29. ylabel('幅值');
30. xlabel('时间/s');
31. for k=1 : vsl % 画出起止点位置
32.     subplot 311
33.     nx1=voiceseg(k).begin; nx2=voiceseg(k).end;
34.     nx1=voiceseg(k).duration;
35.     line([frameTime(nx1) frameTime(nx1)],[-1.5 1.5],'color','r','LineStyle','-');
36.     line([frameTime(nx2) frameTime(nx2)],[-1.5 1.5],'color','b','LineStyle','--');
37.     subplot 312
38.     line([frameTime(nx1) frameTime(nx1)],[min(amp) max(amp)],'color','r','LineStyle','-');
39.     line([frameTime(nx2) frameTime(nx2)],[min(amp) max(amp)],'color','b','LineStyle','--');
40.     subplot 313
41.     line([frameTime(nx1) frameTime(nx1)],[min(zcr) max(zcr)],'color','r','LineStyle','-');
42.     line([frameTime(nx2) frameTime(nx2)],[min(zcr) max(zcr)],'color','b','LineStyle','--');

```

43. end

二、预加重。

预加重的目的是提升高频部分，使信号的频谱变得平坦，保持在低频到高频的整个频带中，能用同样的信噪比求频谱。同时，也是为了消除发生过程中声带和嘴唇的效应，来补偿语音信号受到发音系统所抑制的高频部分，也为了突出高频的共振峰。

```
1. % 预加重功能测试
2. clc
3. clear all
4. close all
5. [s,fs]=wavread('C2_5_y_3.wav');
6. e=s(2000:2225); %提取一段进行分析，容易看出变化
7. un=filter([1,-0.95],1,e); %预加重信号 b=[1,-0.95];
8.
9. %原始信号频谱
10. N=512;
11. pinlv=(0:1:N/2-1)*fs/N;
12. x=fft(e,N);
13. r1=abs(x);
14. t1=20*log10(r1);
15. signal=t1(1:N/2);
16.
17. %预加重信号频谱
18. [h1,w1]=freqz([1,-0.95],1,256,fs);
19. pha=angle(h1);
20. H1=abs(h1);
21. r2=r1(1:N/2);
22. u=r2.*h1;
23. u2=abs(u);
24. signalPre=20*log10(u2);
25.
26. figure(1);
27. subplot(211)
28. plot(e,'b*-')
29. ylim([-0.4,1])
30. hold on
31. plot(real(un),'ro-')
32. legend('原始语音信号','预加重后的语音信号')
33. title('原始语音信号和预加重后的语音信号');
34. xlabel('采样点');ylabel('幅度');
35. subplot(212);
36. plot(pinlv,signal,'g+-')
37. hold on
38. plot(pinlv,signalPre,'kx-')
```

```

39. legend('原始语音信号频谱','预加重后的语音信号频谱')
40. title('预加重前后的语音信号频谱');
41. xlabel('频率');ylabel('幅度/dB');

```

三、分帧加窗。

为了方便对语音分析，可以将语音分成一个个小段，称之为：帧。先将 N 个采样点集合成一个观测单位，称为帧。通常情况下 N 的值为 256 或 512，涵盖的时间约为 20~30ms 左右。为了避免相邻两帧的变化过大，因此会让两相邻帧之间有一段重叠区域，此重叠区域包含了 M 个取样点，通常 M 的值约为 N 的 1/2 或 1/3。通常语音识别所采用语音信号的采样频率为 8KHz 或 16KHz，以 8KHz 来说，若帧长度为 256 个采样点，则对应的时间长度是 $256/8000 \times 1000 = 32\text{ms}$ 。

语音在长范围内是不停变动的，没有固定的特性无法做处理，所以将每一帧代入窗函数，窗外的值设定为 0，其目的是消除各个帧两端可能会造成的信号不连续性。常用的窗函数有方窗、汉明窗和汉宁窗等，根据窗函数的频域特性，常采用汉明窗。

汉明窗：

$$w(n) = \begin{cases} 0.54 - 0.46 \cos[2\pi n / (N-1)], & 0 \leq n \leq (N-1) \\ 0, & n = \text{else} \end{cases}$$

```

1. %分帧
2. function frameout=enframe(x,win,inc)
3.
4. nx=length(x(:));          % 取数据长度
5. nwin=length(win);         % 取窗长
6. if (nwin == 1)             % 判断窗长是否为 1，若为 1，即表示没有设窗函数
7.     len = win;             % 是，帧长=win
8. else
9.     len = nwin;            % 否，帧长=窗长
10. end
11. if (nargin < 3)            % 如果只有两个参数，设帧 inc=帧长
12.     inc = len;
13. end
14. nf = fix((nx-len+inc)/inc); % 计算帧数
15. frameout=zeros(nf,len);    % 初始化
16. indf= inc*(0:(nf-1)).';    % 设置每帧在 x 中的位移量位置
17. inds = (1:len);           % 每帧数据对应 1:len
18. frameout(:) = x(indf(:,ones(1,len))+inds(ones(nf,1),:)); % 对数据分帧
19. if (nwin > 1)              % 若参数中包括窗函数，把每帧乘以窗函数
20.     w = win(:)';           % 把 win 转成行数据
21.     frameout = frameout .* w(ones(nf,1),:); % 乘窗函数
22. end

```

```

1. %加窗，不同窗函数的显示
2. clc
3. clear all

```

```

4. close all
5. N=32;nn=0:(N-1);
6. subplot(311);
7. w = ones(N,1); %矩形窗实现
8. stem(nn,w)
9. xlabel('点数');ylabel('幅度');title('(a)矩形窗')
10. subplot(312);
11. w = 0.54 - 0.46*cos(2*pi*(0:N-1)/(N-1)); %汉明窗实现
12. stem(nn,w)
13. xlabel('点数');ylabel('幅度');title('(b)汉明窗')
14. subplot(313)
15. w = 0.5*(1 - cos(2*pi*(0:N-1)/(N-1))); %汉宁窗实现
16. stem(nn,w)
17. xlabel('点数');ylabel('幅度');title('(c)汉宁窗')

```

四、快速傅里叶变换。

对第 n 帧语音信号 $x_n(m)$ 进行傅里叶变换(离散时域傅里叶变换, DTFT), 可得到短时傅里叶变换, 其定义如下:

$$X_n(e^{j\omega}) = \sum_{m=0}^{N-1} x_n(m) e^{-j\omega m}$$

由于信号在时域上的变换通常很难看出信号的特性, 所以通常将它转换为频域上的能量分布来观察, 不同的能量分布, 就能代表不同语音的特性。所以在乘上汉明窗后, 每帧还必须再经过快速傅里叶变换以得到在频谱上的能量分布。对分帧加窗后的各帧信号进行快速傅里叶变换得到各帧的频谱。并对语音信号的频谱取模平方得到语音信号的功率谱。

五、计算谱线能量。

对语音信号的频谱取模平方得到语音信号的谱线能量。

六、计算通过梅尔滤波器的能量。

将能量谱通过一组 Mel 尺度的三角形滤波器组, 定义一个有 M 个滤波器的滤波器组(滤波器的个数和临界带的个数相近), 采用的滤波器为三角滤波器, 中心频率为 $f(m)$ 。

七、计算 DCT 倒谱。

经离散余弦变换(DCT)得到 MFCC 系数:

$$C(n) = \sum_{m=0}^{N-1} s(m) \cos(\pi n(m+0.5)/N), n=1, 2, \dots, L$$

将上述的对数能量带入离散余弦变换, 求出 L 阶的 Mel 参数。 L 阶指 MFCC 系数阶数, 通常取 12-16。这里 M 是三角滤波器个数。

```

1. %计算 DTW
2. for i = 1:N
3.     fMatrix1 = fMatrixall1{i,1};

```

```
4.      fMatrix1 = CMN(fMatrix1);
5.      Scores1(i) = myDTW(fMatrix1,rMatrix);
6.  end
7.
8.  for j = 1:N
9.      fMatrix2 = fMatrixall2{j,1};
10.     fMatrix2 = CMN(fMatrix2);
11.     Scores2(j) = myDTW(fMatrix2,rMatrix);
12. end
13.
14. for k= 1:N
15.     fMatrix3 = fMatrixall3{k,1};
16.     fMatrix3 = CMN(fMatrix3);
17.     Scores3(k) = myDTW(fMatrix3,rMatrix);
18. end
```

实验结果：

本次实验中，我们了解了语音识别中的一些基本原理和流程：预加重、加窗分帧、端点检测、之后是特征提取 MFCC 等，并进行了相应模块的实验实现。实验中通过调整相应参数以实现不同参数之间实现效果的对比。

实验分析：

特征参数提取的目标，顾名思义，就应该使相同的语音之间的差别尽可能的小，不同的语音之间的差异尽可能的大。而 MFCC 就是特征提取的有效方法。长久以来语音信号处理和模型训练是分开的，因为信号处理的输入信号是原始音频，而模型训练的输入特征由于要求对相位不敏感，一般是基于原始音频的能量谱得到的特征(mfcc,fbank)。

深度学习发展之后，语音识别业界也一致在尝试使用深度学习从原始音频当中提取特征去替代 mfcc 和 mel fbank。然而通过实验证明使用原始音频作为输入，相比传统特征 mel fbank 作为输入的性能并没有明显的优越性。由此我们可以看出 MFCC 在语音识别界有其重要地位。

实验成绩：

日期：____年____月____日

河南理工大学教学上机实验报告

上机时间 2021 年 6 月 24 日

实验题目：

基于 DTW 的孤立词语音识别

实验目的和要求：

目的：

- 1.掌握语音识别的模板匹配法的原理和过程；
- 2.掌握动态时间规整技术；
- 3.应用 matlab 实现基于 DTW 的 10 个阿拉伯数字的识别。

要求：

每位同学自己录制 0-9 这 10 位数字的模板，然后分别尝试使用自己的模板和其他同学的模板来进行孤立词识别，并观察识别结果。

实验过程：

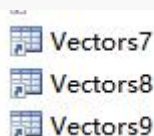
一、录制 10 个数字的语音信号作为模板，每个数字发音三次。

原本文件中存在有 6 个模板数据，所以首先录制自己的发音模板三遍，将第一遍录制的 0-9 十位数字的声音模板命名为 Vectors7.mat，后两次依次参照本次录制，并分别命名为：Vectors8.mat、Vectors9.mat。

修改 GetMyVectors.m 的 128 行为对应每次的名称：

```
1. save('Vectors___.mat')
```

每次录制会生成一个 Vectors__ 的文件，这就是我们后面需要用到的模板数据。



二、提取每段模板语音的 MFCC 特征向量序列，发出任意数字的声音，提取相应的 MFCC 特征向量序列。

MFCC 参数是基于人的听觉特性利用人听觉的屏蔽效应，在 Mel 标度频率域提取出来的倒谱特征参数。美尔频标倒谱系数(Mel Frequency Cepstrum Coefficient, MFCC)考虑了人耳的听觉特性，将频谱转化为基于 Mel 频标的非线性频谱，然后转换到倒谱域上。由于充分考虑了人耳的听觉特性，而且没有任何的前提假设，MFCC 参数具有良好的识别性能和抗噪声能力，但其计算量和计算精度要求较高。

三、使用自己的 30 个模板语音的 MFCC 特性向量，分别与步骤 3 提取的 MFCC 特征向量进行动态时间规整，获取对应的匹配度得分，然后根据匹配度得分的大小判断步骤 3 发出的声音所对应的数字。观察并记录识别结果。

在进行了前面的工作后我们进行动态时间规整（DTW），DTW 是一种衡量两个长度不

同的时间序列的相似度的方法。应用也比较广，主要是在模板匹配中，比如说用在孤立词语音识别（识别两段语音是否表示同一个单词），手势识别，数据挖掘和信息检索等中。

1、打开 DTWscores.m，首先初始化 DTW 判别矩阵，然后加载前面我们录制的 Vectors7.mat、Vectors8.mat、Vectors9.mat 三个模板数据：

```
1. %加载模板数据
2. %s1 = load('Vectors1.mat');
3. s1 = load('Vectors7.mat');
4. fMatrixall1 = struct2cell(s1);
5. %s2 = load('Vectors2.mat');
6. s2 = load('Vectors8.mat');
7. fMatrixall2 = struct2cell(s2);
8. %s3 = load('Vectors3.mat');
9. s3 = load('Vectors9.mat');
10. fMatrixall3 = struct2cell(s3);
```

修改完成后运行该文件。




2、打开 matchTemplates.m 进行模板匹配检测。每次运行该 matlab 代码后，我们任意按键即可实时录制我们发出的 2 秒声音，发出的声音应该为 0-9 之间的任意数字，此时可以得到匹配得到的声音数字是多少。

<pre>Nbr = 7 2 7 1 1 7 You have just said Six .</pre> <p>发出声音数字 0 识别结果</p>	<pre>Nbr = 2 8 2 8 2 8 You have just said One .</pre> <p>发出声音数字 1 识别结果</p>
<pre>Nbr = 3 9 3 9 3 4 You have just said Two .</pre> <p>发出声音数字 2 识别结果</p>	<pre>Nbr = 4 9 4 5 4 9 You have just said Three.</pre> <p>发出声音数字 3 识别结果</p>
<pre>Nbr = 2 5 5 4 5 4 You have just said Four .</pre> <p>发出声音数字 4 识别结果</p>	<pre>Nbr = 6 10 6 10 6 2 You have just said Five .</pre> <p>发出声音数字 5 识别结果</p>
<pre>Nbr = 7 10 7 5 7 1 You have just said Six .</pre> <p>发出声音数字 6 识别结果</p>	<pre>Nbr = 8 2 8 5 8 2 You have just said Seven.</pre> <p>发出声音数字 7 识别结果</p>

<pre> Nbr = 9 3 9 4 9 3 You have just said Eight. </pre> <p>发出声音数字 8 识别结果</p>	<pre> Nbr = 10 7 10 7 10 6 You have just said Nine . </pre> <p>发出声音数字 9 识别结果</p>
--	--

四、使用其他同学的 30 个模板语音的 MFCC 特性向量序列，分别与步骤 6 提取的 MFCC 特征向量进行动态时间规整，获取对应的匹配度得分，然后根据匹配度得分的大小判断步骤 6 发出的声音所对应的数字。观察并记录识别结果。

1、重复以上一、二、三步骤，录制同学（杨毛强）的模板语音，并分别命名为：Vectors10.mat、Vectors11.mat、Vectors12.mat，在进行 MFCC 特征提取和 DTW 之后重新进行匹配识别。

 Vectors10
 Vectors11
 Vectors12

2、以同学（杨毛强）的数字声音作为模板，以我发出的实时语音作为匹配进行实验。

<pre> Nbr = 7 9 9 10 3 9 You have just said Eight. </pre> <p>以同学声音为模板，匹配数字 0 结果</p>	<pre> Nbr = 2 8 2 6 2 6 You have just said One . </pre> <p>以同学声音为模板，匹配数字 1 结果</p>
<pre> Nbr = 9 3 9 3 9 3 You have just said Two . </pre> <p>以同学声音为模板，匹配数字 2 结果</p>	<pre> Nbr = 9 3 9 2 3 9 You have just said Eight. </pre> <p>以同学声音为模板，匹配数字 3 结果</p>
<pre> Nbr = 6 10 5 10 7 2 You have just said Nine . </pre> <p>以同学声音为模板，匹配数字 4 结果</p>	<pre> Nbr = 6 10 10 7 7 10 You have just said Nine . </pre> <p>以同学声音为模板，匹配数字 5 结果</p>
<pre> Nbr = 7 6 7 6 7 10 You have just said Six . </pre> <p>以同学声音为模板，匹配数字 6 结果</p>	<pre> Nbr = 8 2 10 1 8 7 You have just said Seven. </pre> <p>以同学声音为模板，匹配数字 7 结果</p>

<pre>Nbr = 9 3 9 3 3 9 You have just said Two .</pre> <p>以同学声音为模板，匹配数字 8 结果</p>	<pre>Nbr = 3 10 3 2 9 7 You have just said Two .</pre> <p>以同学声音为模板，匹配数字 9 结果</p>
--	---

实验结果：

本次的基于动态时间归整（DTW）的孤立字语音识别实验中，通过录制不同人的数字声音模板，通过 MFCC 提取特征后进行 DTW。在实验中，可以发现，对于我本人的数字声音模板，在本人测试中，效果也是十分明显，可以达到 90% 的正确率，如果在进行修正的情况下，达到 100% 是可以的。在利用同学（杨毛强）的声音作为模板，以我的声音去匹配的时候，正确率仅仅为 40%，在多次测试情况下仍不能达到 100% 的正确结果。

实验分析：

DTW 的基本思想就是把未知量均匀地伸长或缩短，直到它与参考模式的长度一致为止。在时间归整过程中，未知单词的时间轴要不均匀地扭曲或弯折，以便使其特征与模型特征对正。动态时间归整是较早的一种模式匹配和模型训练技术，它应用动态规划方法成功地解决了语音信号特征参数序列在进行比较时时长不等的难题。

由以上我们可以知道，DTW 主要解决了录入时声音不规整的因素，将录制的声音规整以后，通过计算录制与模板之间的距离作为一种相似度的衡量。基于这样的思想，在进行实验的测试中，由于同学（杨毛强）发音与我的发音存在一些区别，因此始终不能保持将我的声音与他的声音模板全部完美匹配，但是相比较之下，DTW 用了一种简单的思路来实现了一些孤立词的优秀识别。

实验成绩：

日期：____年____月____日

河南理工大学教学上机实验报告

上机时间 年 月 日

实验题目：

基于 HMM 的孤立词语音识别

实验目的和要求：

目的：

- 1.掌握语音识别的 HMM 的原理和过程
- 2.掌握维特比解码算法
- 3.应用 matlab 实现基于 HMM 的 10 个阿拉伯数字的识别

要求：

每位同学自己或与同学共同录制 0-9 这 10 位数字的语音信号，并训练 10 个 HMM 模型，最后进行孤立词识别，并观察识别结果。

实验过程：

一、自己录制 10 个数字的语音信号，每个数字发音 8 次。

编写录制声音代码，设置参数：MFCC 参数阶数为 12、录制数字个数为 10 个、采样频率 fs=16000、录音时长为 2s、训练样本的人数为 1。

```
1. ncoeff = 12;           %MFCC 参数阶数
2. N = 10;                %10 个数字
3. fs=16000;              % 采样频率
4. duration2 = 2;         %录音时长
5. k = 1;                  %训练样本的人数
6. a = cell(1,10);
```

声音录制代码 record_cell_file.m:

```
1. clear all;
2. close all;
3. ncoeff = 12;           %MFCC 参数阶数
4. N = 10;                %10 个数字
5. fs=16000;              % 采样频率
6. duration2 = 2;         %录音时长
7. k = 1;                  %训练样本的人数
8. a = cell(1,10);
9.
10. for i = 1:10
11.     fprintf('\n 计算数字%d\n',i);
12.     b = cell(1,8);
```

```

13.     for j= 1:8
14.         speech = audiorecorder(fs,16,1);
15.         disp('Press any key to start 2 seconds of speech 1 recording...');
16.         pause
17.         disp('Recording speech...');
18.         recordblocking(speech,duration2)           % duration*fs 为采样点数
19.         speechIn=getaudiodata(speech);
20.         disp('Finished recording. ');
21.         disp('System is trying to recognize what you have spoken...');
22.         speechIn = my_vad(speechIn);               %端点检测
23.         b{1,j} =speechIn;
24.     end
25.     a{1,i} = b;
26. end
27. save('Cell.mat')

```

运行之后会生成对应的模板的 cell。

Cell 2021/6/24 16:12 Microsoft Acces... 6,918 KB

二、提取每段语音的 MFCC 特征向量序列，使用对应的样本分别训练每个 HMM 模型。

1、提取 MFCC 向量序列：

```

1. fprintf('\n 计算数字%d 的 mfcc 特征参数\n',i);
2. for k = 1:length(a{i}) % 样本数的循环
3.     obs(k).sph = a{i}{k}; % 数字 i 的第 k 个语音
4.     obs(k).fea = mfcc(obs(k).sph); % 对语音提取 mfcc 特征参数
5. end

```

2、使用对应样本训练 HMM 模型：

```

1. fprintf('\n 训练数字%d 的 hmm\n',i);
2. hmm_temp=inithmm(obs,N,M); %初始化 hmm 模型
3. hmm{i}=baum_welch(hmm_temp,obs); %迭代更新 hmm 的各参数

```

这里要注意，读入训练集时保证是正确的要训练的样本，例如第一次我使用的是我自己的声音作为训练，数据名称为：Cell，第二次训练采用同学（杨毛强）作为训练的数据：Cell2。

三、自己发出不同数字的待识别语音信号，提取 MFCC 特征向量序列，并进行识别，直到所有数字的发音都被识别一次。

```

>> hmm_recog
开始识别
该语音的真实值为1
该语音识别结果为1

```

发出声音数字 1 识别结果

```

>> hmm_recog
开始识别
该语音的真实值为2
该语音识别结果为2

```

发出声音数字 2 识别结果

<pre>>> hmm_recog 开始识别 该语音的真实值为3 该语音识别结果为3</pre> <p>发出声音数字 3 识别结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为4 该语音识别结果为4</pre> <p>发出声音数字 4 识别结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为5 该语音识别结果为5</pre> <p>发出声音数字 5 识别结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为6 该语音识别结果为6</pre> <p>发出声音数字 6 识别结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为7 该语音识别结果为7</pre> <p>发出声音数字 7 识别结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为8 该语音识别结果为8</pre> <p>发出声音数字 8 识别结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为9 该语音识别结果为9</pre> <p>发出声音数字 9 识别结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为10 该语音识别结果为10</pre> <p>发出声音数字 10 识别结果</p>

四、与同学合作录制 10 个数字的语音信号，每个数字发音 8 次。

重复以上步骤一、二、三，使用同学（杨毛强）的声音去训练 HMM，并把自己的声音作为测试，观察识别结果。

<pre>>> hmm_recog 开始识别 该语音的真实值为1 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 1 的结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为2 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 2 的结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为3 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 3 的结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为4 该语音识别结果为2</pre> <p>以同学声音训练，匹配数字 4 的结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为5 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 5 的结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为6 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 6 的结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为7 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 7 的结果</p>	<pre>开始识别 该语音的真实值为8 该语音识别结果为6</pre> <p>以同学声音训练，匹配数字 8 的结果</p>
<pre>>> hmm_recog 开始识别 该语音的真实值为9 该语音识别结果为5</pre> <p>以同学声音训练，匹配数字 9 的结果</p>	<pre>>> hmm_recog 开始识别 该语音的真实值为10 该语音识别结果为1</pre> <p>以同学声音训练，匹配数字 10 的结果</p>

实验结果：

本次的基于 HMM 的孤立词语音识别实验中，通过录制不同人的数字声音模板，然后训练，建立模板参考。在实验中，可以发现，对于我本人的数字声音模板，在本人测试中，效果十分明显，可以达到 100% 的正确率。在利用同学（杨毛强）的声音作为模板训练模板参考后，再去以我的声音去匹配的时候，正确率仅仅为 10%，在多次测试情况下仍不能大幅度提高识别效果。

实验分析：

对比 DTW 的实验结果，可以看到 DTW 的效果比 HMM 的效果要好，通过查询相关资料，结合上课内容，可以知道 HMM 采用了 GMM 模型，可识别范围比 DTW 更好，同时，HMM 把孤立语音识别分为几个不同的状态统一训练，这就与 DTW 分不同模板匹配差不多，但是 HMM 更加方便。相比较 DTW，HMM 更像是一个学习过程，一个监督性分类器，分类出多个状态而非一个供识别的模板。DTW 比起 HMM 显得较为生硬。

通过上述可以知道，实验的结果应该在 HMM 的模型精确度上会好一些，但是比较而言，我的 DTW 会好，这可能由于我录制 HMM 训练的声音时产生了噪音所致。对于后续，HMM 仍有很大的改进空间。

实验成绩：

日期：____年____月____日