

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 数字语音处理 |
| 姓 名： | 汪辉 |
| 学 院： | 计算机科学与技术学院 |
| 系： |  |
| 专 业： | 计算机科学与技术 |
| 学 号： | 3190105609 |
| 指导教师： | 杨莹春 |

2021年 12月 24日

**浙江大学实验报告**

课程名称： 数字语音处理 实验类型： 综合

实验项目名称： 低值语音检测算法

学生姓名： 汪辉 专业： 计算机科学与技术 学号： 3190105609

同组学生姓名： 个人实验 指导老师： 杨莹春

实验地点： 实验日期：2021 年 9 月17 日—2022 年 1 月 4 日

一、实验目的和要求

采集语音信息，构建说话人语音模型并验证。

理解梅尔倒谱系数和高斯混合模型的简要原理和实际应用。

分析测试文件的预测结果和差异原因。

二、实验环境和第三方库

实验使用python3.9.8开发环境,

使用到的库有：skgmm、sklearn、numpy、scipy、python\_speech\_features。

PRAAT语音分析软件进行音频分析。Audition进行语音采集。

三、实验内容和原理

内容：

(1)15次以上诗歌朗诵录音（每周1次以上）(2分)

(2)运用python及其它库完成MFCC，GMM 训练与测试，记录结果（2分）

(3)找出识别得分偏低的语句，用PRAAT分析其与模板语音之间的听感、特征等差异（2分）

(4)设计算法检测得分偏低语句，并计算检测准确率（2分）

原理：

1. MFCC

梅尔倒谱系数是一种在自动语音和说话人识别中广泛使用的特征，在1980年由Davis和Mermelstein提出的。

依据人的听觉实验结果来分析语音的频谱，MFCC分析依据的听觉机理有两个：梅尔刻度和临界带。

人耳对不同频率的声波有不同的听觉敏感度。从200Hz到5000Hz的语音信号对语音的清晰度影响较大。两个响度不等的声音作用于人耳时，则响度较高的频率成分的存在会影响到对响度较低的频率成分的感受，使其变得不易察觉，这种现象称为掩蔽效应。

由于频率较低的声音（低音）在内耳蜗基底膜上行波传递距离大于频率较高的声音（高音），因此低音容易掩蔽高音。低音掩蔽的临界带宽较高频要小。所以，人们从低频到高频这一段频带内按临界带宽的大小由密到疏安排一组带通滤波器，对输入信号进行滤波。将每个带通滤波器输出的信号能量作为信号的基本特征，对此特征经过进一步处理后就可以作为语音的输入特征。由于这种特征不依赖于信号的性质，对输入信号不做任何的假设和限制，又利用了听觉模型的研究成果。因此，这种参数比基于声道模型的LPCC相比具有更好的鲁棒性，更符合人耳的听觉特性，而且当信噪比降低时仍然具有较好的识别性能。

计算MFCC主要分为以下六个步骤：

1. 将信号帧化为短帧
2. 对于每一帧，计算功率谱的[周期图估计](http://en.wikipedia.org/wiki/Periodogram)
3. 将mel滤波器组应用于功率谱，求滤波器组的能量，将每个滤波器中的能量相加
4. 取所有滤波器组能量的对数
5. 取对数滤波器组能量的离散余弦变换（DCT）。
6. 保持DCT系数2-13，其余部分丢弃

实验中，采用python第三方库调用mfcc函数计算该特征值。

1. GMM

得到说话人的特征后，利用高斯混合模型进行训练。高斯模型是一种常用的变量分布模型，即常说的正态分布。一维高斯分布的概率密度函数如下：

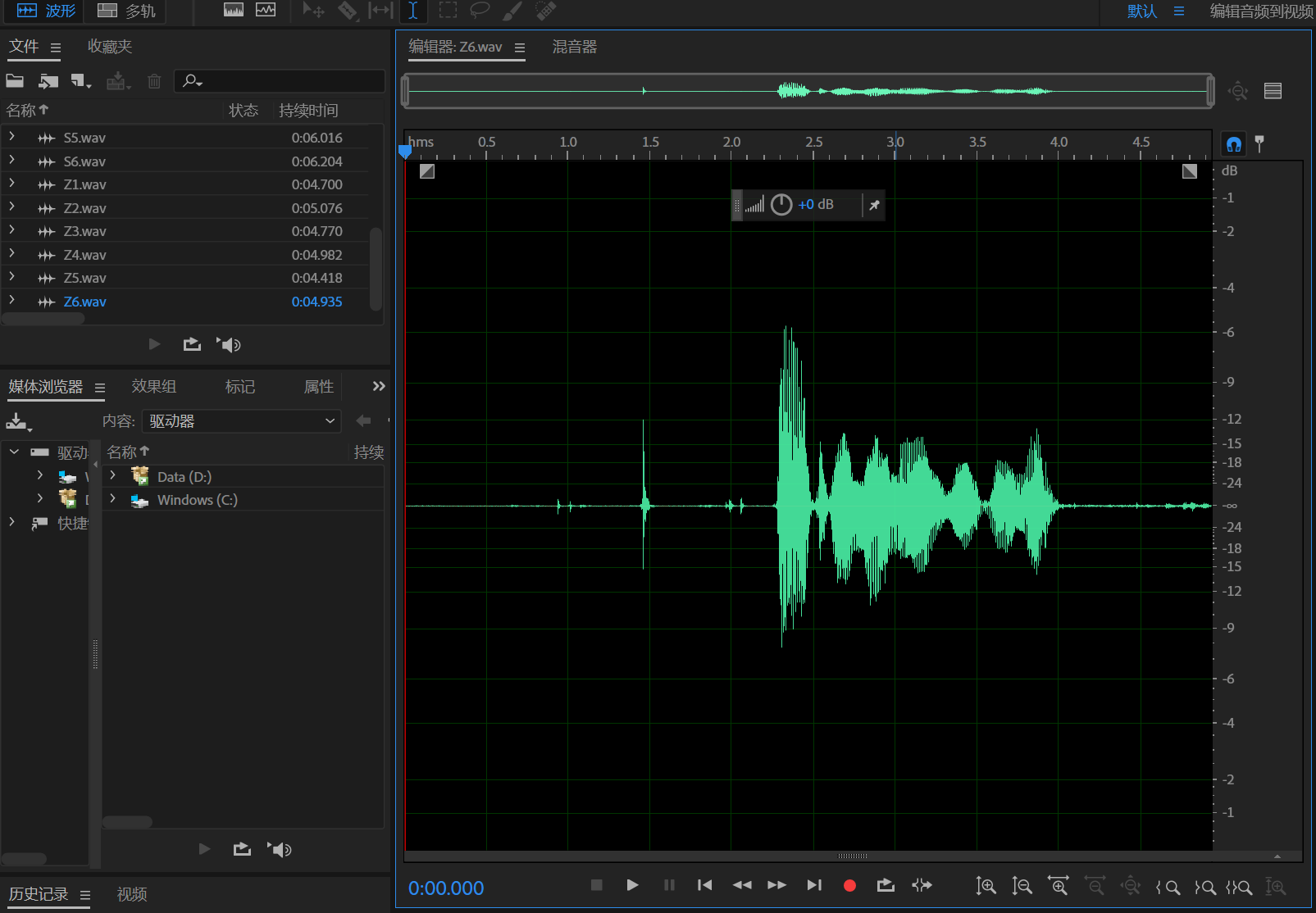
GMM通过求解两个或以上高斯模型，并通过一定的权重将这些高斯模型融合成一个模型，得到最终的混合高斯模型。假设混合高斯模型由K个高斯模型组成（即数据包含K个类），则GMM的概率密度函数如下：

高斯混合模型的本质是是融合几个单高斯模型，来使得模型更加复杂，从而产生更复杂的样本。理论上，如果某个混合高斯模型融合的高斯模型个数足够多，它们之间的权重设定得足够合理，这个混合模型可以拟合任意分布的样本。

四、实验过程



1. 使用Adobe Audition采集语音



1. 调用python第三方库进行特征提取和模型训练

python\_speech\_features是一个用于人语音特征计算的库，调用其中的mfcc函数能直接得到输入声音信号对应的mfcc参数。

1. from python\_speech\_features import mfcc
2. def get\_feature(fs, signal):
3. mfcc\_feature = mfcc(signal, fs)
4. if len(mfcc\_feature) == 0:
5. print >> sys.stderr, "ERROR.. failed to extract mfcc feature:", len(signal)
6. return mfcc\_feature

为了进行声音模型的训练和后续测试，实现一个名为GMMSet的对象，存储所有的训练特征数据，提供对外的接口用于训练和预测。

1. class GMMSet:
2. def \_\_init\_\_(self, gmm\_order = 32):
3. self.gmms = []
4. self.gmm\_order = gmm\_order
5. self.y = []

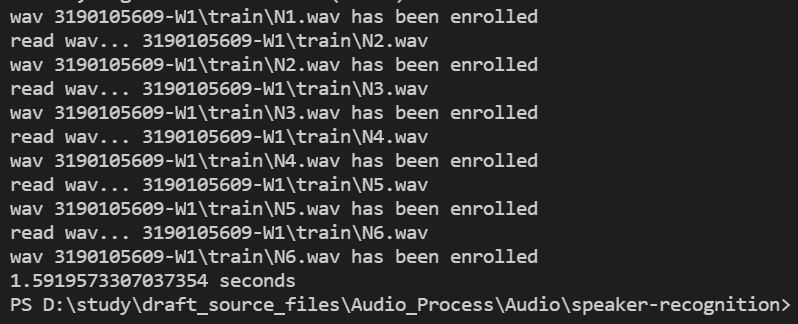
列表gmms存储训练的所有GMM集合，GaussianMixture是库sklearn.mixture中的对象：

1. from sklearn.mixture import GaussianMixture

GMMSet对外的接口fit\_new和predict\_one分别用于训练和测试：

1. def fit\_new(self, x, label):
2. self.y.append(label)
3. gmm = GaussianMixture(self.gmm\_order)
4. gmm.fit(x)
5. self.gmms.append(gmm)
6. def gmm\_score(self, gmm, x):
7. return np.sum(gmm.score(x))
9. def predict\_one(self, x):
10. scores = [self.gmm\_score(gmm, x) / len(x) for gmm in self.gmms] *# get average scores of all gaussian mixture models*
11. return math.exp( np.mean(scores) ) *# exp the average of scores*

通过15周录音的正常语速的语音训练模型：



1. 采集不同对象的语音数据测试模型

首先用每周录音的快速、慢速和含糊语音分别测试模型，测试分数基于GaussianMixture的score函数取指数得到，当分数接近于1时表明测试文件接近说话人的声音。

下列是用第十周的语音数据进行预测的结果：

1. read wav... 3190105609-W10\F1.wav
2. 3190105609-W10\F1.wav  score-> 0.9020088730844658
3. read wav... 3190105609-W10\F2.wav
4. 3190105609-W10\F2.wav  score-> 0.9010066975175153
5. read wav... 3190105609-W10\F3.wav
6. 3190105609-W10\F3.wav  score-> 0.9020462238997244
7. read wav... 3190105609-W10\F4.wav
8. 3190105609-W10\F4.wav  score-> 0.886098601327435
9. read wav... 3190105609-W10\F5.wav
10. 3190105609-W10\F5.wav  score-> 0.903109844371423
11. read wav... 3190105609-W10\F6.wav
12. 3190105609-W10\F6.wav  score-> 0.9104934998468861
13. read wav... 3190105609-W10\S1.wav
14. 3190105609-W10\S1.wav  score-> 0.9358878956421507
15. read wav... 3190105609-W10\S2.wav
16. 3190105609-W10\S2.wav  score-> 0.9219119893611302
17. read wav... 3190105609-W10\S3.wav
18. 3190105609-W10\S3.wav  score-> 0.9233115538127816
19. read wav... 3190105609-W10\S4.wav
20. 3190105609-W10\S4.wav  score-> 0.9223110330061005
21. read wav... 3190105609-W10\S5.wav
22. 3190105609-W10\S5.wav  score-> 0.9181485842899326
23. read wav... 3190105609-W10\S6.wav
24. 3190105609-W10\S6.wav  score-> 0.9230866630937307
25. read wav... 3190105609-W10\Z1.wav
26. 3190105609-W10\Z1.wav  score-> 0.9245290441269188
27. read wav... 3190105609-W10\Z2.wav
28. 3190105609-W10\Z2.wav  score-> 0.9046219377289034
29. read wav... 3190105609-W10\Z3.wav
30. 3190105609-W10\Z3.wav  score-> 0.8989199958713836
31. read wav... 3190105609-W10\Z4.wav
32. 3190105609-W10\Z4.wav  score-> 0.9079195425789713
33. read wav... 3190105609-W10\Z5.wav
34. 3190105609-W10\Z5.wav  score-> 0.9157962315478624
35. read wav... 3190105609-W10\Z6.wav
36. 3190105609-W10\Z6.wav  score-> 0.9168674471876312

可以看到，预测分数都在0.9上下，接近1，基本预测准确。部分语句分数较低，低于0.9，将在后续进行音频分析。

为了验证预测的准确性，我也采集了我以外的同学的音频样本进行预测，由于预测模型是用我的语音数据进行训练，预测别人的语音信号应该得到较低的分数：

1. xtl\week1\3190101016-W1-F1.wav  score-> 0.5060988602940119
2. xtl\week1\3190101016-W1-F2.wav  score-> 0.6780581154968419
3. xtl\week1\3190101016-W1-F3.wav  score-> 0.6427569118684018
4. xtl\week1\3190101016-W1-F4.wav  score-> 0.6725179357878625
5. xtl\week1\3190101016-W1-F5.wav  score-> 0.6617905651450378
6. xtl\week1\3190101016-W1-F6.wav  score-> 0.7102533233393596
7. xtl\week1\3190101016-W1-N1.wav  score-> 0.5585911997237109
8. xtl\week1\3190101016-W1-N2.wav  score-> 0.7296366219306235
9. xtl\week1\3190101016-W1-N3.wav  score-> 0.7719828949393276
10. xtl\week1\3190101016-W1-N4.wav  score-> 0.7747077439359047
11. xtl\week1\3190101016-W1-N5.wav  score-> 0.7987624637959636
12. xtl\week1\3190101016-W1-N6.wav  score-> 0.7942679533740612
13. xtl\week1\3190101016-W1-S1.wav  score-> 0.7295650299204757
14. xtl\week1\3190101016-W1-S2.wav  score-> 0.8419910906522693
15. xtl\week1\3190101016-W1-S3.wav  score-> 0.8732818706323338
16. xtl\week1\3190101016-W1-S4.wav  score-> 0.8790900594167392
17. xtl\week1\3190101016-W1-S5.wav  score-> 0.8887333235842959
18. xtl\week1\3190101016-W1-S6.wav  score-> 0.8935879677269339
19. xtl\week1\3190101016-W1-Z1.wav  score-> 0.6404212356449691
20. xtl\week1\3190101016-W1-Z2.wav  score-> 0.7424791850759329
21. xtl\week1\3190101016-W1-Z3.wav  score-> 0.7679193519764103
22. xtl\week1\3190101016-W1-Z4.wav  score-> 0.7503996990746721
23. xtl\week1\3190101016-W1-Z5.wav  score-> 0.7076380022259203
24. xtl\week1\3190101016-W1-Z6.wav  score-> 0.7968140647752181

从结果中可以看到，预测分数都不超过0.9，相较于自己的语音的预测分数明显低，基本符合预测模型的预想。然而，仍然有部分语音数据得到了接近0.9的分数，这与gmm模型预测的目的有出入，在后续音频分析中分析这些数据。

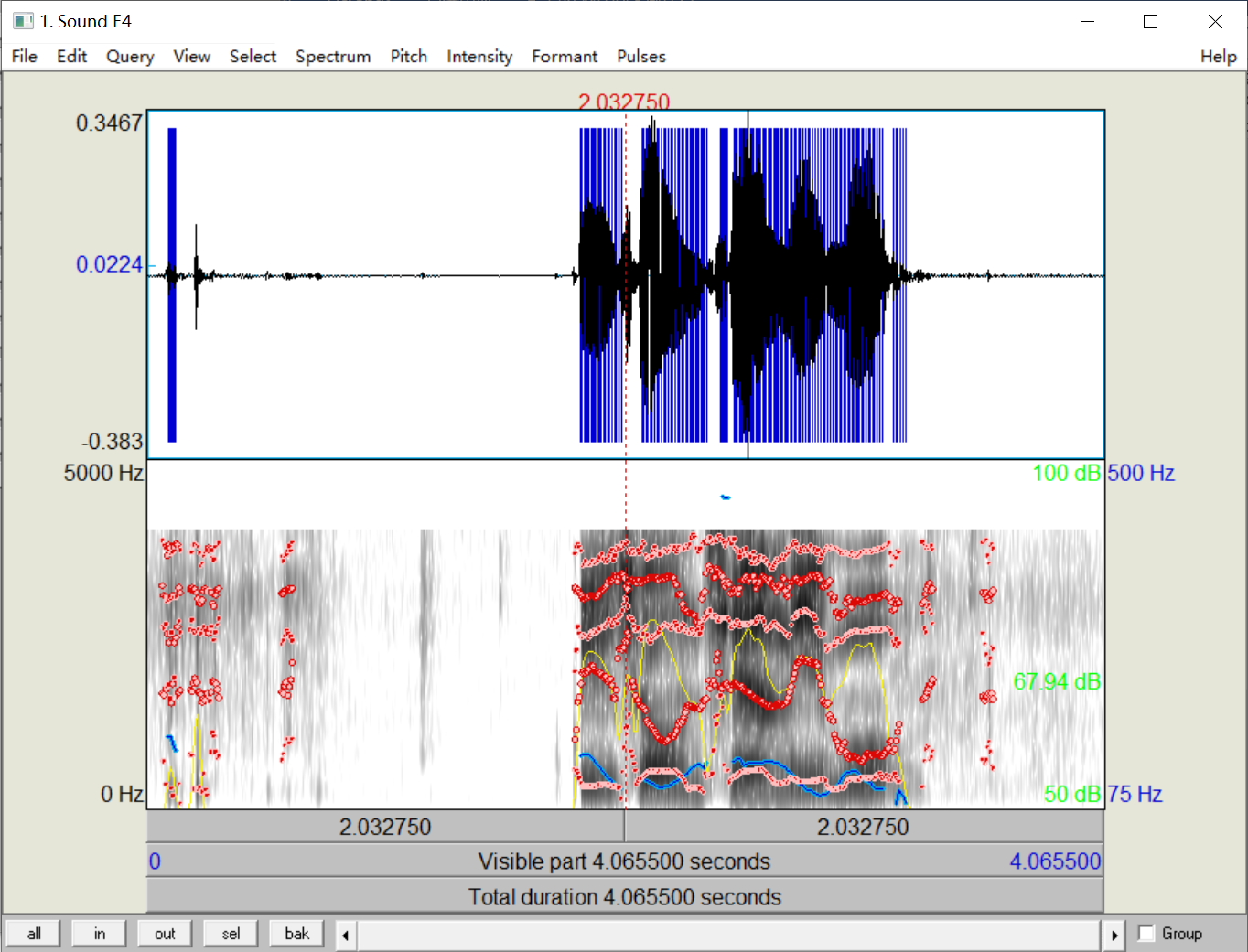
1. 对比分值低的语音音频

尽管我们仍然可以通过已经有的音频数据确定一个分数的阈值来将自己的语音和其他的语音区分开来，从而实现识别说话人的目的，但是部分数据的过高或过低的分数仍然需要引起注意。以下通过praat简要地分析了这些音频文件。

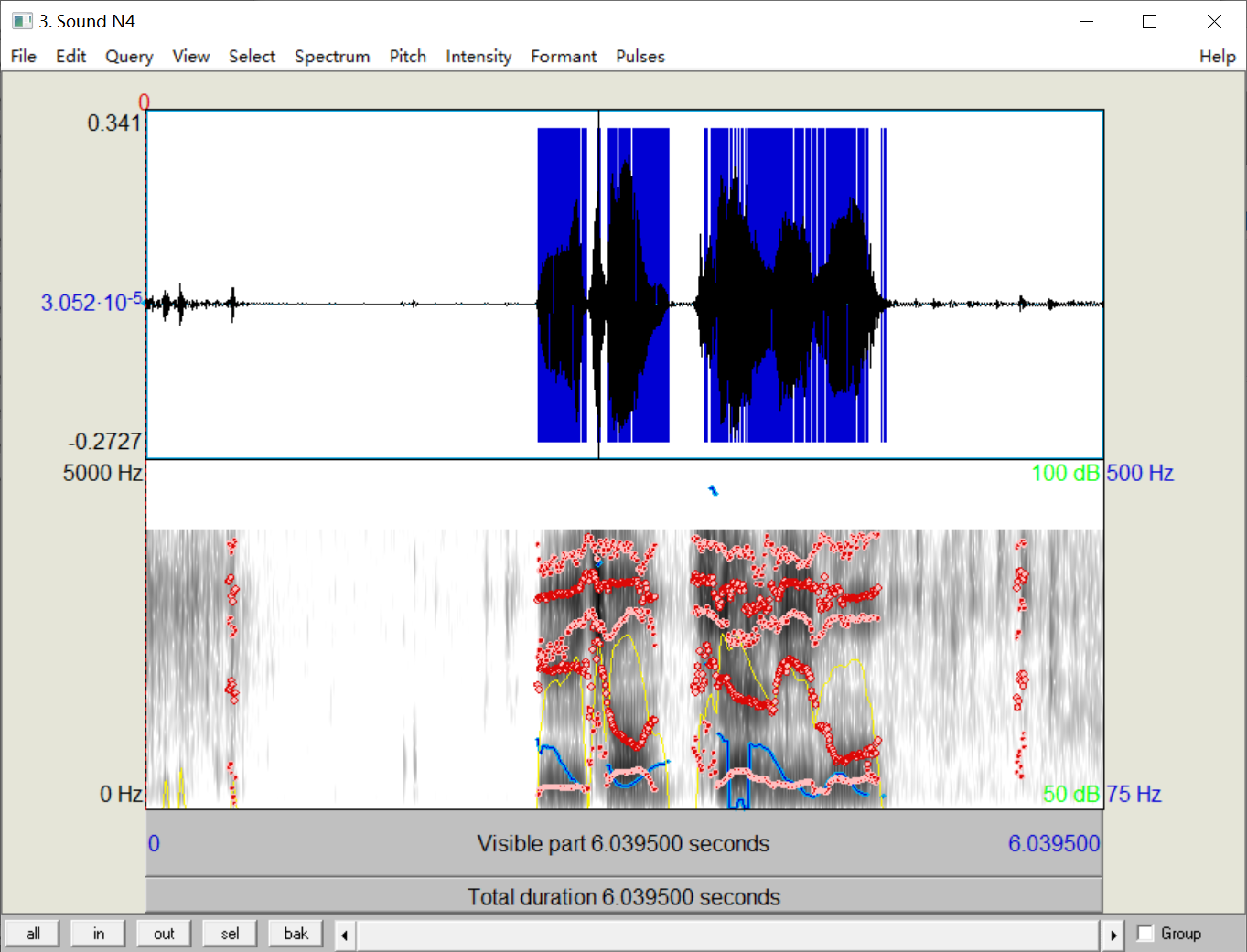
通过对比15周的每句得分，选择F4即快速的“欲穷千里目”进行音频分析，从下面的截图看到，部分F4的得分甚至低至0.8左右。



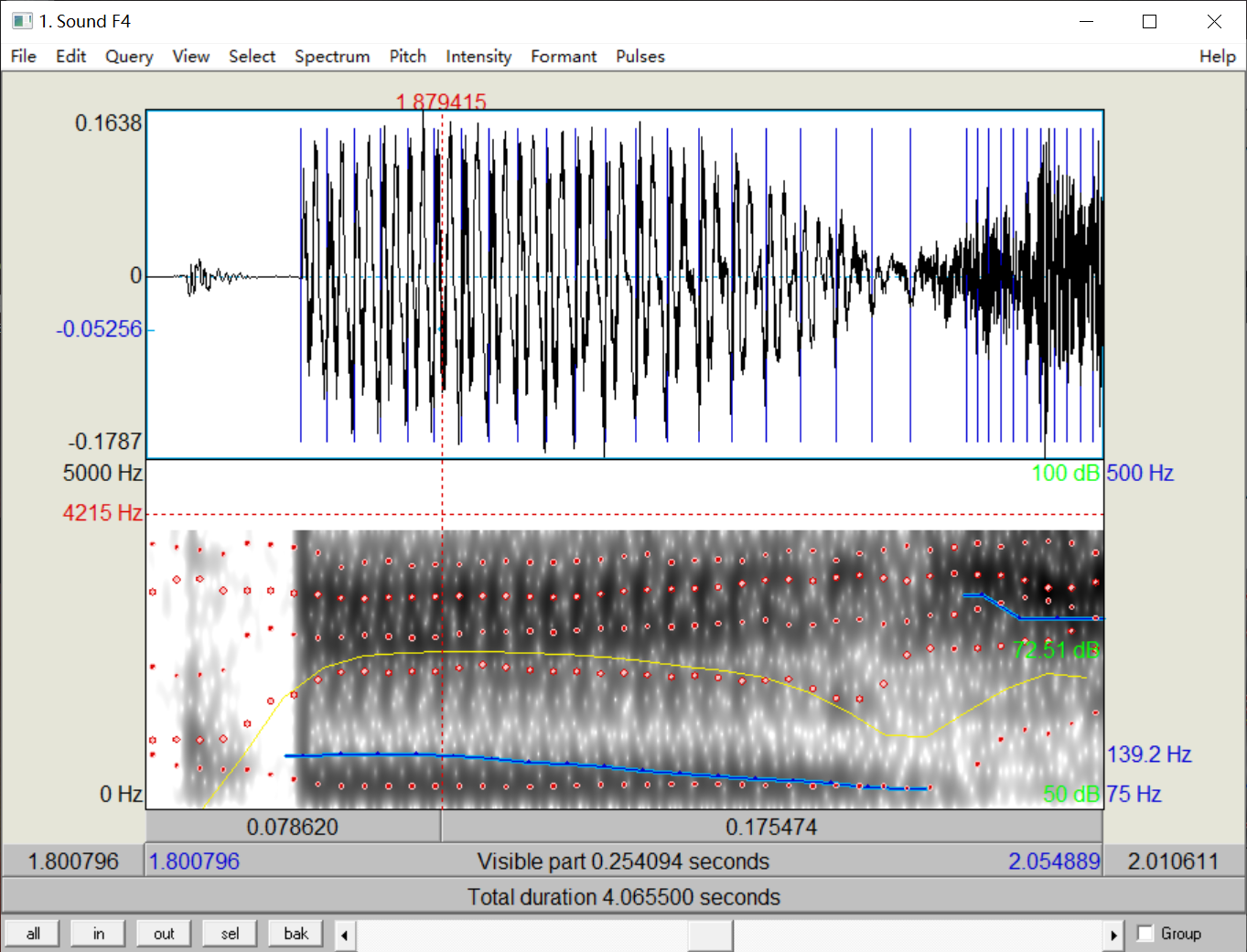
观察F4的音频频谱：



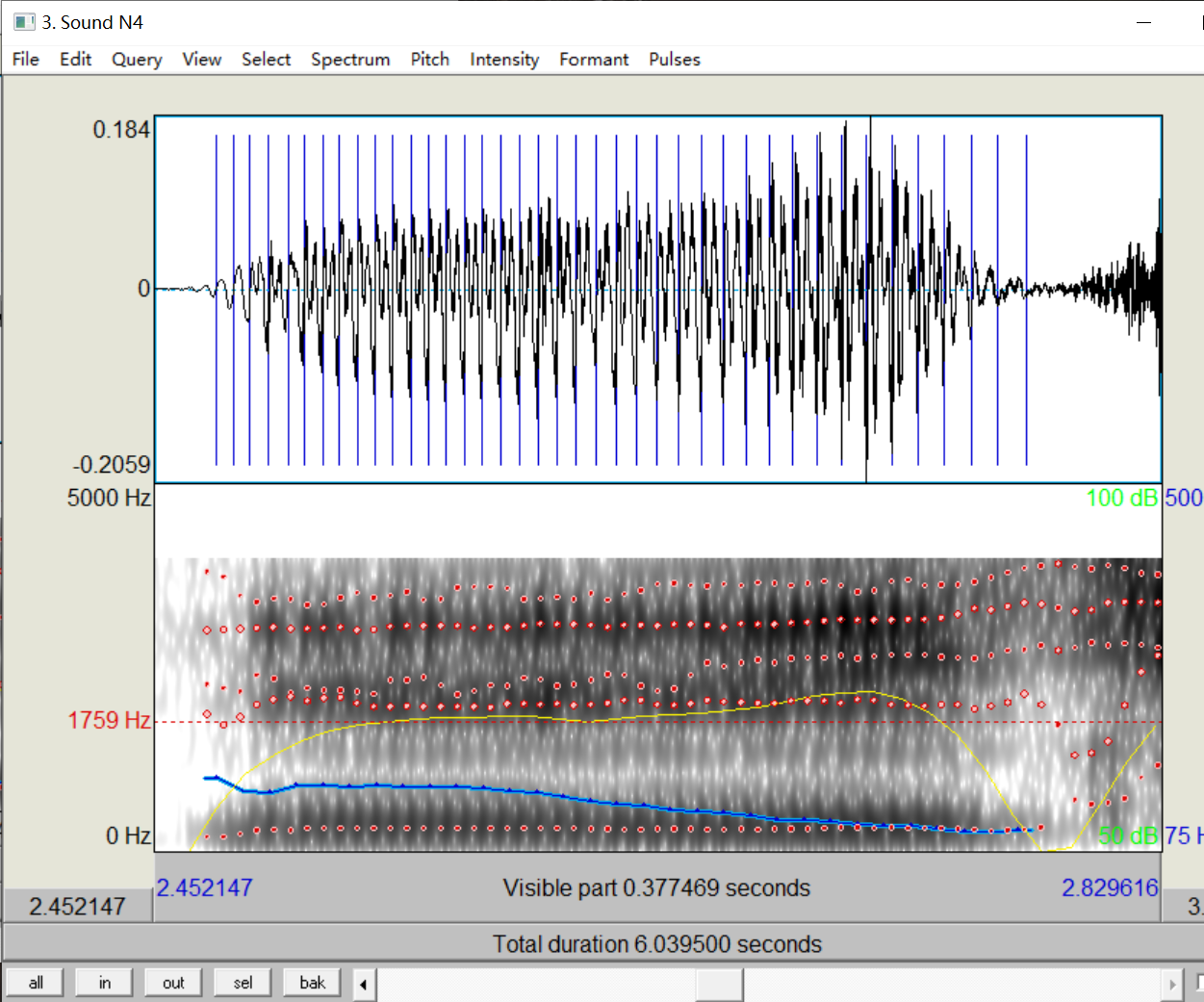
对应的模型训练语句N4音频的频谱：



观察对比可以发现，两者的音强基本吻合，分别在5个字出现波峰。F4具有更连贯的音高，更紧促的脉冲波形，这与F4是快速的有关，两者的共振峰具有明显的差别。然后以第一个字，“欲穷千里目”的欲为例，放大频谱观察。F4的频谱和波形：



N4中“欲”的波形和频谱：



五、实验结果与讨论

1、通过15周的语音训练建立了自己的说话识别模型，在一定的精度上实现了对说话人的识别。并且对预测的语音都能做出基于训练模型的测试分数，一定程度上表征了说话人模型和被测试语音的相似性。

2、通过分析低分语句的特点，简要总结了不同音频特征在mfcc中的表现，猜想梅尔倒谱系数对于音频共振这些信息存在表征不完备或丢失的情况。

3、结合高斯混合模型的应用大大促进了梅尔倒谱系数从理论到语音识别实践的过程，除了高斯模型外，还有很多概率分布模型在包括语音分析在内的领域有应用，高斯混合模型因为其极佳的适配性和在混合模型中良好的计算性得以广泛应用。

4、目前语音识别领域最受瞩目的应该是深度学习算法的应用，在近几年大量的实验中大有赶超原先的mfcc\_gmm体系的趋势。越来越多的更好更完备的神经网络正用来检测“说话人识别”、“语音识别”等领域且表现不俗。