实验二 基于 GMM 的语音数字识别

学号：2019101404　姓名：马正一

1 任务描述

基于数字语音数据集，编写代码，使用 GMM 算法完成语音识别，对输入的一段音频进行分类，输出语音中的数字，如“2”、“10”。

2 实验环境

操作系统使用 MacOS，Python=3.6，python-speech-features=0.6，pyaudio，scikit-learn=0.18.1。

3 实验方案

3.1 MFCC 特征提取

我们使用课程提供的英文数据集，包括数字 0-9 共 150 个 wav 格式的音频文件。我们使用 Python 的 wav 包读取 wav 文件，使用 python-speech-features 获得每条音频数据的 13 维 MFCC 特征。我们在本实验中对加入一阶导与二阶导的 39 维特征同样进行了实验，但识别结果不如 13 维 MFCC 特征。我们分析原因很可能为训练数据过少导致数据的过拟合。具体来说，MFCC 特征提取算法首先进行预加重，然后对语音文件进行分帧，加窗，然后进行快速傅里叶变换，将它转换为频域上的能量分布来观察；将能量谱通过一组 Mel 尺度的三角形滤波器组，对频谱进行平滑化，并消除谐波的作用，突显原先语音的共振峰；计算每个滤波器输出的对数能量，经离散余弦变换（DCT）得到 MFCC 系数；然后计算对数能量；最后提取动态差分参数。

在实际编写代码时，我们在 features.py 中编写了特征提取函数，返回每个数据样本的 13 维 MFCC 特征。

3.2 GMM 分类

在获取了 MFCC 特征之后，我们将编写基于 GMM 的分类算法。我们使用了 scikit-learn 的 GaussianMixture 高斯混合分布模型来编写程序。在本实验中，我们面的是一个十分类问题。我们将训练十个单核 GMM。对于每个数字的所有样本，我们用这部分训练数据训练同一个单核 GMM。在测试阶段，我们将每一个待分类样本输入至每一个 GMM，得到在该 GMM 下的对应评分，即该 GMM 对应数字的评分。我们对十个评分进行降序排序，将评分最高的 GMM 对应数字作为预测标签进行输出。

Scikit-learn 中对 GaussianMixture 模型的定义如下

```
class sklearn.mixture.GaussianMixture(n_components=1, covariance_type='full', tol=0.001,
reg_covar=1e-06, max_iter=100, n_init=1, init_params='kmeans', weights_init=None,
means_init=None, precisions_init=None, random_state=None, warm_start=False, verbose=0,
verbose_interval=10)
```

1. n_components: 混合高斯模型个数，默认为 1
2. covariance_type: 协方差类型，包括｛‘full’，‘tied’，‘diag’，‘spherical’｝四种，full 指每个分量有各自不同的标准协方差矩阵，完全协方差矩阵（元素都不为零），tied 指所有分量有相同的标准协方差矩阵（HMM 会用到），diag 指每个分量有各自不同对角协方差矩阵（非对角为零，对角不为零），spherical 指每个分量有各自不同的简单协方差矩阵，球面协方差矩阵（非对角为零，对角完全相同，球面特性），默认‘full’ 完全协方差矩阵
3. tol：EM 迭代停止阈值，默认为 1e-3.
4. reg_covar: 协方差对角非负正则化，保证协方差矩阵均为正，默认为 0
5. max_iter: 最大迭代次数，默认 100
6. n_init: 初始化次数，用于产生最佳初始参数，默认为 1
7. init_params: ｛‘kmeans’，‘random’｝, defaults to ‘kmeans’. 初始化参数实现方式，默认用 kmeans 实现，也可以选择随机产生
8. weights_init: 各组成模型的先验权重，可以自己设，默认按照 7 产生
9. means_init: 初始化均值，同 8
10. precisions_init: 初始化精确度（模型个数，特征个数），默认按照 7 实现
11. random_state : 随机数发生器
12. warm_start : 若为 True，则 fit（）调用会以上一次 fit（）的结果作为初始化参数，适合相同问题多次 fit 的情况，能加速收敛，默认为 False。
13. verbose : 使能迭代信息显示，默认为 0，可以为 1 或者大于 1（显示的信息不同）
14. verbose_interval : 与 13 挂钩，若使能迭代信息显示，设置多少次迭代后显示信息，默认 10 次。

3.3 GMM 实现

具体在编写代码时，我们编写核心模型类 GMMSet，用于维护 10 个 GMM 高斯混合分布。对于每个数字的训练样本，我们使用 GMMSet 定义的 fit_new 函数调用每个 GMM 的 fit 函数进行模型训练。在测试阶段，我们调用 GMMSet 的 predict_one 函数，使用十个 GMM 分别进行评分，并对评分进行降序排序取最高分对应 label，作为预测输出。

```
from sklearn.mixture import GaussianMixture

import operator

import numpy as np

import math



class GMMSet:



    def __init__(self, gmm_order = 1):

        self.gmms = []

        self.gmm_order = gmm_order

        self.y = []



    def fit_new(self, x, label):

        self.y.append(label)

        gmm = GaussianMixture(self.gmm_order)

        gmm.fit(x)

        self.gmms.append(gmm)
```

```
def gmm_score(self, gmm, x):
        # print(gmm.score(x))
        return np.sum(gmm.score(x))


@staticmethod
def softmax(scores):
        scores_sum = sum([math.exp(i) for i in scores])
        score_max  = math.exp(max(scores))
        return round(score_max / scores_sum, 3)


def predict_one(self, x):
        # print(x.shape)
        scores = [self.gmm_score(gmm, x) / len(x) for gmm in self.gmms]
        p = sorted(enumerate(scores), key=operator.itemgetter(1), reverse=True)
        # print("----------------------")
        # print(p)
        # print(scores)
        p = [(str(self.y[i]), y, p[0][1] - y) for i, y in p]
        # print(p)
        result = [(self.y[index], value) for (index, value) in enumerate(scores)]
        # print(result)
```

```
        p = max(result, key=operator.itemgetter(1))

        # print(p)

        softmax_score = self.softmax(scores)

        return p[0], softmax_score



    def before_pickle(self):

        pass



    def after_pickle(self):

        pass
```

4 运行手册

本项目已上传至 github 并编写 Readme 说明手册，链接为：[Link]

在实验开始前，用户需使用 Anaconda 或 pip 安装实验所需的 python 环境。

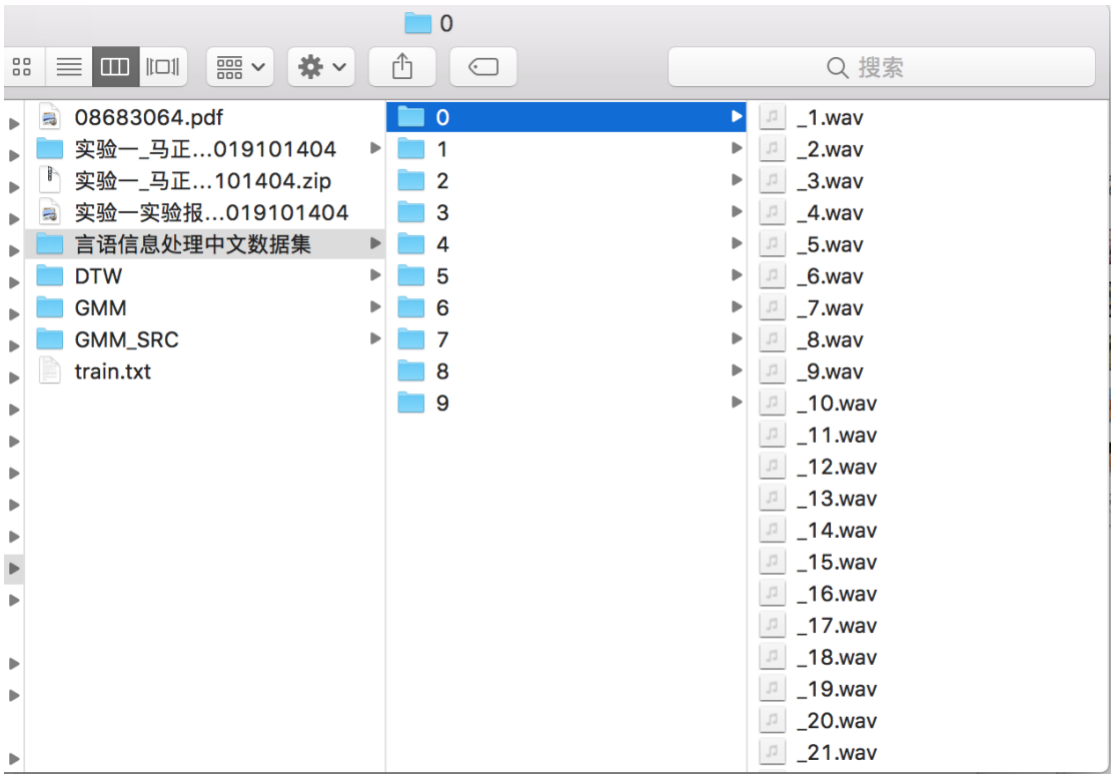
```
conda create -n GMM -c anaconda python=3.6 numpy pyaudio scipy
conda activate GMM
pip install -r requirements.txt
```

之后，运行以下命令完成实验并查看实验结果

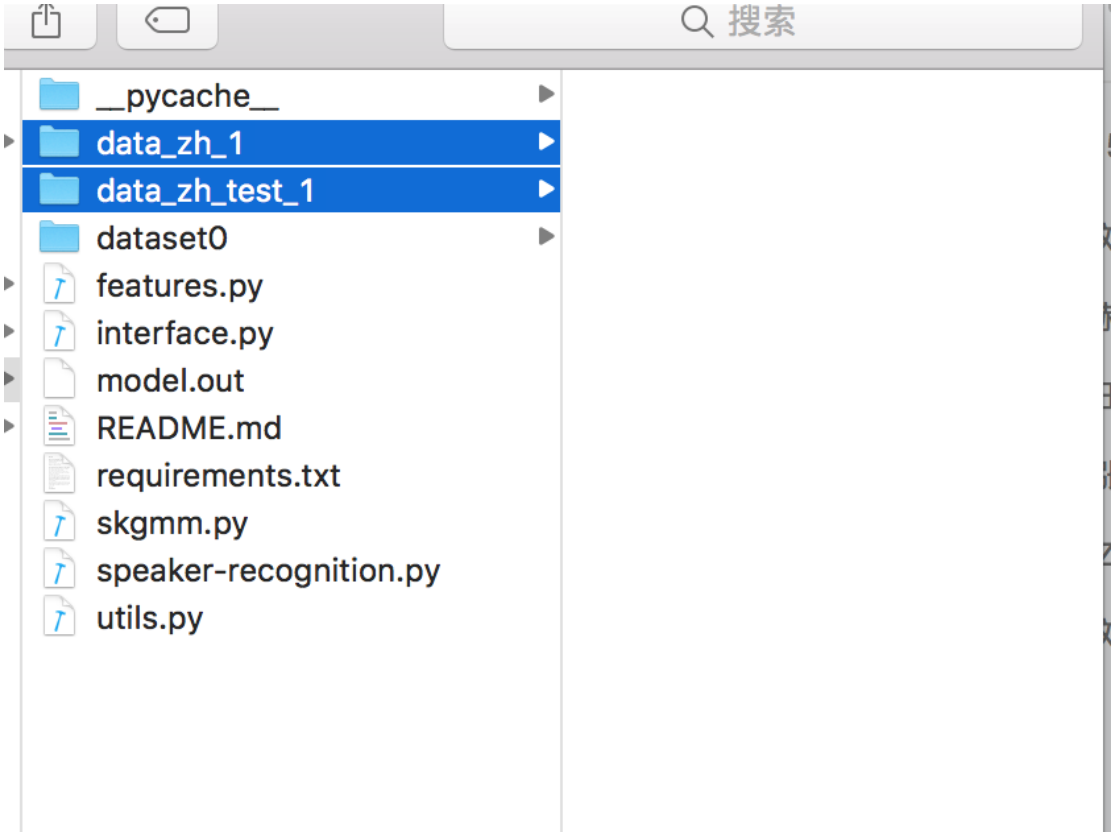
```
  git clone https://github.com/zhengyima/GMM_Digital_Voice_Recognition.git
GMM_DVR && cd GMM_DVR
  unzip /tmp/dataset.zip -d ./  # dataset.zip 是从百度网盘下载的数据
  python speaker-recognition.py -t enroll -i  "./data_zh_1/*/"  -m model.out
  python speaker-recognition.py -t predict  -i  "./data_zh_test_1/*/"  -m
model.out
```

5 实验结果及运行截图

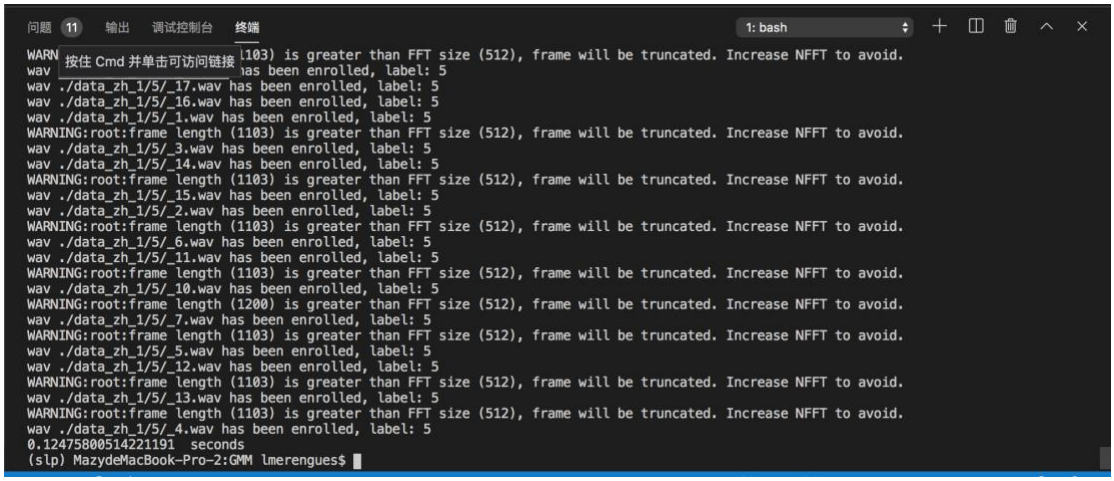
本实验在课程提供的中文数据集下，使用 285 条样本进行训练，30 条样本进行测试，对测试的 30 个样本可正确识别其中 19 个样本，准确率达到 63.33%。



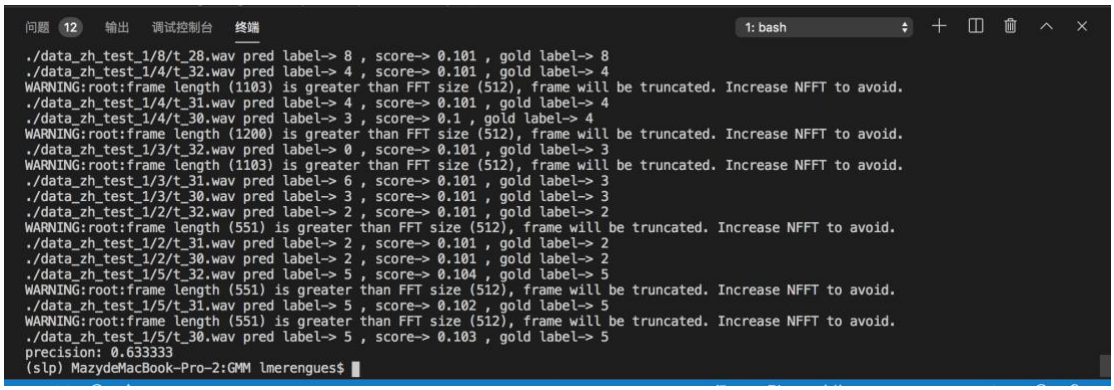
原始数据文件



划分训练集，测试集



模型训练阶段



模型测试，准确率 63.33%