Python 音频处理入门

——数字媒体基础 课程 王晗

这一次实验我们练习如何使用python解析一段格式为wav的音频。

实验内容：
1、 读取音频数据
2、 绘制单通道及双通道音频波形
3、 计算语音信号短时能量与短时过零率
4、 绘制语谱图并观察语谱图中音频的基音周期、频率与共振峰

准备工作：
首先，我们需要 import 几个工具包，一个是 python 标准库中的 wave 模块，用于音频处理操作，另外两个是 numpy 和 matplot，提供数据处理函数，这两个工具包的安装请参考 Python图像处理入门中的相关介绍。

```
import wave
# 导入 wave 模块
import matplotlib.pyplot as plt
# 用于绘制波形图
import numpy as np
# 用于计算波形数据
import os
#  用于系统处理，如读取本地音频文件
```

一：读取本地音频数据
处理音频第一步是需要从让计算机“听到”声音，这里我们使用 python 标准库中自带的 wave 模块进行音频参数的获取。

（1） 导入 wave 模块
（2） 使用 wave 中的函数 open 打开音频文件，wave.open(file,mode)函数带有两个参数，第一个 file 是所需要打开的文件名及路径，使用字符串表示；第二个 mode 是打开的模式，也是用字符串表示 （'rb'或'wb'）
（3） 打开音频后使用 getparams() 获取音频基本的相关参数，其中 nchannels:声道数；sampwidth:量化位数或量化深度（byte）；framerate:采样频率；nframes:采样点数

完整代码：

```
f = wave.open(r"./data/cn.wav",'rb' )
params = f.getparams ()
nchannels,sampwidth, framerate, nframes = params [:4]
print(framerate)
```

结果：

```
44100
```

二：读取**单通道**音频，并绘制波形图（常见音频为左右，2个声道）

（1） **通过第一步，可以继续读取音频数据本身，保存为字符串格式**

strData = f.readframes(nframes)

（2） **如果需要绘制波形图，则需要将字符串格式的音频数据转化为 int 类型**

waveData = np.fromstring(strData,dtype=np.int16)

此处需要使用到 numpy 进行数据格式的转化

（3） **将幅值归一化**

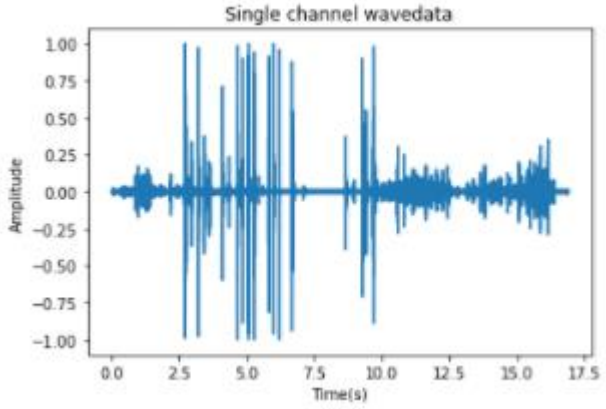
waveData = waveData*1.0/(max(abs(waveData)))

这一步去掉也可画出波形图，大家可以尝试不用此步，找出波形图的不同

（4） **绘制图像**

```
time = np.arange(0,nframes)*(1.0 / framerate)#计算音频的时间
plt.plot(time,waveData)
plt.xlabel("Time(s)")
plt.ylabel("Amplitude")
plt.title("Single channel wavedata")
plt.show()
```

结果：



三： 多通道语音数据读取

唯一不一样的地方就是 waveData 是一个多维的矩阵了

plt.plot(time,waveData[:,n])#绘制waveData数据的第n列，即音频第n个通道的数据

关键代码

```
waveData = np.reshape(waveData,[nframes,nchannels])
f.close()
```

```
time = np.arange(0,nframes)*(1.0 / framerate)
plt.figure()
plt.subplot(5,1,1)
plt.plot(time,waveData[:,0])
plt.xlabel("Time(s)")
plt.ylabel("Amplitude")
plt.title("Ch-1 wavedata")
plt.subplot(5,1,3)
plt.plot(time,waveData[:,1])
plt.xlabel("Time(s)")
plt.ylabel("Amplitude")
plt.title("Ch-2 wavedata")
plt.subplot(5,1,5)
plt.plot(time,waveData[:,2])
plt.xlabel("Time(s)")
plt.ylabel("Amplitude")
plt.title("Ch-3 wavedata")
plt.show()
```

四：语音信号短时能量

计算较短时间内的语音能量。这里短时指的是一帧（256个采样点）。

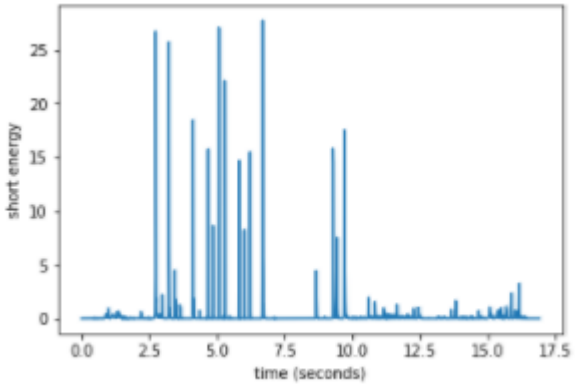
关键代码：

```
def calEnergy(wave_data) :
    energy = []
    sum = 0
    frameSize = 256
    for i in range(len(wave_data)) :计算每一帧的数据和
        sum = sum + (wave_data[i] * wave_data[i])#采样点数据平方
        if (i + 1) % frameSize  == 0 :
            energy.append(sum)
            sum = 0
        elif i == len(wave_data) - 1 :
            energy.append(sum)
    return energy
```

绘制

```
energy = calEnergy(waveData[:,0])
time2 = np.arange(0, len(energy)) * (len(waveData[:,0])/len(energy) / framerate)

plt.plot(time2, energy)
plt.ylabel("short energy")
plt.xlabel("time (seconds)")
plt.show()
```

输出：



五：语音信号短时过零率

在每帧中（256个采样点），语音信号通过零点（从正变为负或从负变为正）的次数。

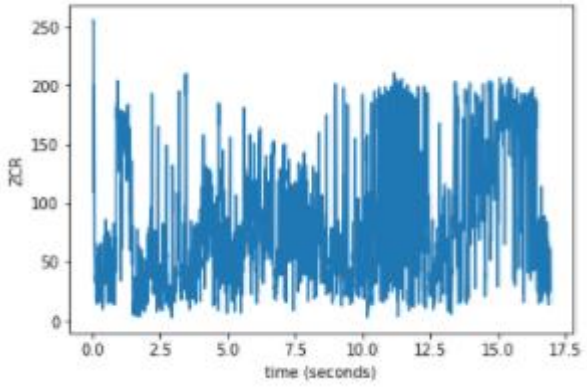
关键代码：

```
def ZeroCR(wave_data,frameSize,overLap):
    wlen = len(wave_data)
    step = frameSize - overLap
    frameNum = math.ceil(wlen/step)#帧的数量
    zcr = np.zeros((frameNum,1))
    for i in range(frameNum):#每帧过零的次数计算
        curFrame = wave_data[np.arange(i*step,min(i*step+frameSize,wlen))]
        curFrame = curFrame - np.mean(curFrame)
        zcr[i] = sum(curFrame[0:-1]*curFrame[1::]<=0)
    return zcr
```

绘制

```
frameSize = 256
overLap = 0
zcr = ZeroCR(waveData[:,0],frameSize,overLap)
Time3 = np.arange(0, len(zcr)) * (len(waveData[:,0])/len(zcr) / framerate)

plt.plot(time3, zcr)
plt.ylabel("ZCR")
plt.xlabel("time (seconds)")
plt.show()
```

输出：

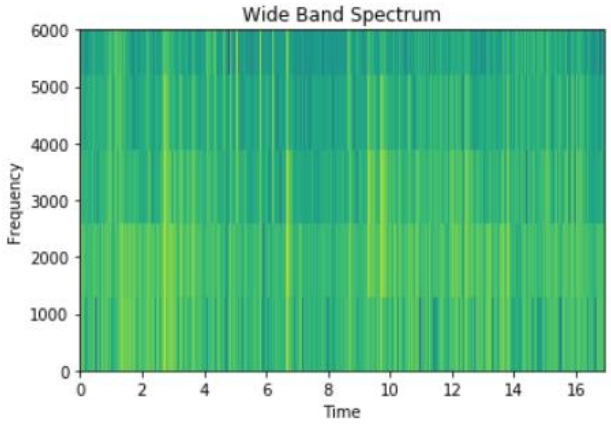


六：绘制语谱图

（1） 绘制宽带语谱图

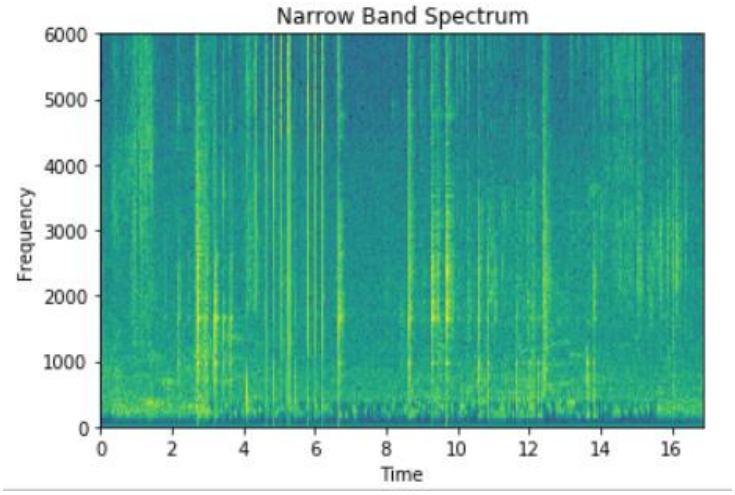
帧长，包含采样点数，必须为2^n，n取小（如32，画出的图为宽带频谱图），n取大（如2048，画出的图为窄带频谱图）

```
framesize = 32
计算离散傅里叶变换的点数，NFFT必须与时域的点数framsize相等，即不补零的FFT
NFFT = framesize
设置帧与帧重叠部分采样点数，overlapSize约为每帧点数的1/3~1/2
overlapSize = 1.0 / 3 * framesize
overlapSize = int(round(overlapSize))  # 取整
绘制频谱图
plt.specgram(waveData[:,0], NFFT=NFFT,
Fs=framerate,window=np.hanning(M=framesize), noverlap=overlapSize)

绘制
plt.ylabel('Frequency')
plt.xlabel('Time')
plt.ylim(0, 6000)
plt.title("Wide Band Spectrum")
plt.show()
```



（2）绘制窄带语谱图

```
framesize = 2048
framelength = framesize / framerate
NFFT = framesize
overlapSize = 1.0 / 3 * framesize
overlapSize = int(round(overlapSize))
plt.specgram(waveData[:,0],
NFFT=NFFT,Fs=framerate,window=np.hanning(M=framesize),
                              noverlap=overlapSize)

plt.ylabel('Frequency')
plt.xlabel('Time')
plt.ylim(0, 6000)
plt.title("Narrow Band Spectrum")
plt.show()
```



此版本为课堂交流版本，如发现文中有纰漏请随时联系。

实验二：数字音频技术

1、绘制一段语音（十个字以内）的语谱图和语音波形。对比不同窗长度，和帧覆盖率来分析语谱图反应出的语音特征，以及宽带和窄带语谱图的变化情况。