به نام خدا

گزارش پروژه پایانی درس سیگنال و سیستم ها – دکتر آرمین سلیمی‌بدر

اعضا گروه : امیرقائمی – آراس ولی‌زاده – سجاد نعیمی

بخش پردازش صوت:

در این بخش از ما خواسته شده بود تا یک صوت داده شده که نویزی هستش رو به صوت تمیز و یک سری اعمالات مثل تندکردن صوت و کندکردن صوت و ... روش انجام بشه.
همراه با توضیح اینکه روال کار ما چه شکلی بوده بخش های مختلف کدمون رو توضیح میدیم.

1.Read_voice:

```python
def read_voice(path):
    rate, data = wavfile.read(path)
    Amplitude = rfft(data)
    Frequency = rfftfreq(len(data), 1 / rate)
    return Amplitude, Frequency , data , rate

```

در ابتدا مسیر فایل (path) رو به تابع پاس میدیم تا بدونه از کجا باید ویس نویزی رو بخونه. تو خط اول میام دیتا و ریت صوت رو میخونم که دیتا ما ویسمون به صورت آرایه و ریت اون رو میگیریم (اگه ریت رو ۲ برابر کنیم سرعت ویس ۲ برابر میشه). تابع rfft میاد برامون یه تبدیل فوریه گسسته روی مقادیر حقیقی دیتامون میزنه که سهم هم فرکانس رو برامون درست میکنه، مقدار اون رو ذخیره می‌کنیم. تو خط بعدی میایم با استفاده از rfftfreq فرکانس هایی که داخل ساخت سیگنالمون نقش داشتن رو بدست میاریم، الان ضریب فرکانس ها داخل Amplitude و خود عدد فرکانس داخل Frequency موجود هستش.

2.Write_voice:

```
1
2  def write_voice(data, rate, path):
3      wavfile.write(path, rate, data.astype(np.int16))
4
```

تابع write_voice کاری که برامون انجام میده اینه دیتا و ریت یک ویس رو بهش پاس میدیم و درنهایت اونارو تو مسیری که بهش پاس دادیم برامون سیو میکنه.

3.ChangeVoiceSpeed:

```
def change_voice_speed(data, rate, speed_factor):
    new_rate = int(rate * speed_factor)
    indices = np.round(np.arange(0, len(data), speed_factor))
    indices = indices[indices < len(data)].astype(int)
    new_data = data[indices]
    return new_data, new_rate
```

تابع change voice speed کاری که انجام میده برامون اینه سرعت صدامون رو تغییر میده، اول میایم خود ویس رو تو غالب دیتا و سرعتش رو تو غالب ریت بهش پاس میدیم، یک آرگومان ضریب سرعت بهش پاس بدیم تا بفهمیم سرعت ویسمون میخواد چند برابر بشه، همین کارو دقیقا تو خط اول انجام میدیم. با استفاده از arrange میایم یه آرایه به طول دیتامون میسازیم و بعدش از ۰ شروع میکنیم به مقدار دهی داخل آرایه و اختلاف هر دو مقدار speed_factor هستش یعنی اگه ۲ باشه مقدار آرایمون به این شکل میشه : [0,2,4,6,8,....] و این ایندکس هایی هستش که باید از دیتامون بخونیم تا ویس جدید شکل بگیره که برای این مثال یعنی ویسمون سریعتر بشه.

3.LowPassFilter:

```
def low_pass_filter(Frequency, Amplitude, F, t):
    lower_bound = F - t / 2
    upper_bound = F + t / 2
    for i in range(len(Frequency)):
        if  Frequency[i] > upper_bound or Frequency[i]<
lower_bound  or np.abs(Amplitude[i]) > 5*10**8 :
            Amplitude[i] = 0
    return Amplitude
```

تابع low pass filter ما بدین شکل هستش که ابتدا میانه فرکانس هایی که میخوایم فیلتر کنیم بهش میدیم و یک عرض برای آن قرار میدیم. تا یک رنج پایین و بالا بود اون رو پاک کنیم و اینکه یک سری طیف این بین مقدار بیش از حد زیادی دارن که میایم حذف میکنیم.

4.ReverseVoice:

```
1  def reverse_voice(data):
2      new_data = data[::-1]
3      return new_data
```

این تابع هم کارش اینه ویس رو به عنوان ورودی پاس بدیم بهش که یه آرایس و در نهایت اونو برعکس میکنه که نتیجتا ویس ما رو یه ریت ثابت برعکسش میکنیم.

5.MixVoices:

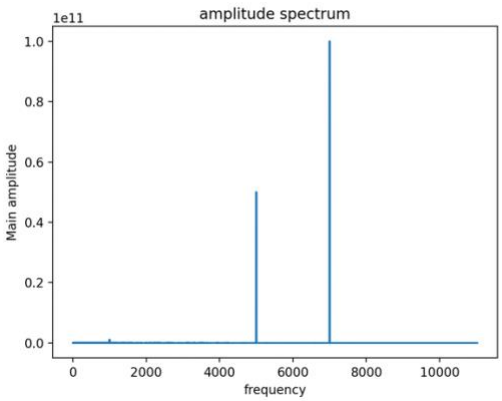
```python
def mix_voices(Datas, Rates):
    Rate = Rates[0]
    min_length = min([len(data) for data in Datas])
    Data = np.zeros(min_length)
    for data in Datas:
        Data += data[:min_length]
    Data = Data / len(Datas)
    return Data, Rate
```

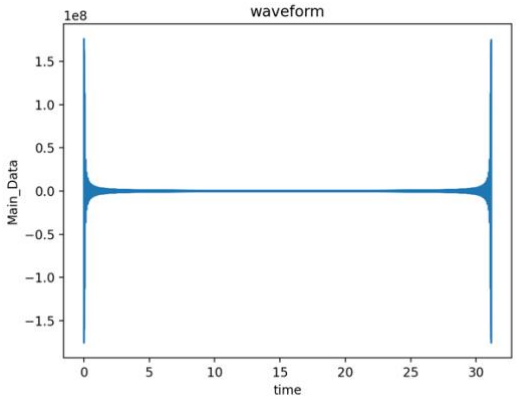
تابع میکس کردن هم به این شکله که اولین ریت از ویس ها رو به عنوان ریت ویس خروجی انتخاب میکنه و در نهایت طول ویس رو با کوچک ترین ویس انتخاب میکنیم و در نهایت مقادیر دیتا های ویس های مختلف رو با هم جمع میکنیم تا ویس میکس شده شامل تمامی ویس ها باشد.

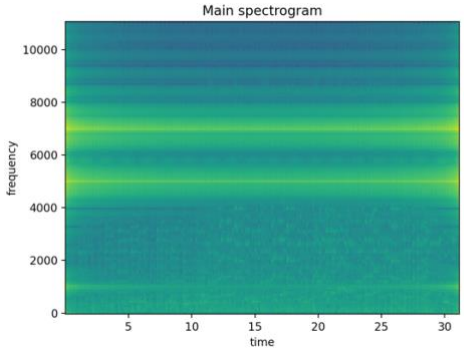
```python
path = '/Users/arasvalizadeh/Downloads/audio/potc.wav'
amplitude, frequency , data , rate = read_voice(path)
amplitudecopy= amplitude.copy()
good = low_pass_filter(frequency,amplitude,5000,10000)
filteredVoice = irfft(good)
write_voice(filteredVoice,rate,
'/Users/arasvalizadeh/Downloads/audio/newaudio3/cleanpotc.wav')
slow_speed_factor = 0.8
slowed_data, slowed_rate = change_voice_speed
(filteredVoice, rate, slow_speed_factor)
fast_speed_factor = 1.5
fast_data, fast_rate = change_voice_speed
(filteredVoice, rate, fast_speed_factor)
write_voice(slowed_data, slowed_rate,
'/Users/arasvalizadeh/Downloads/audio/newaudio3/slowpotc.wav')
write_voice(fast_data, fast_rate,
'/Users/arasvalizadeh/Downloads/audio/newaudio3/fastpotc.wav')
path2=
"/Users/arasvalizadeh/Downloads/audio/newaudio3/cleanpotc.wav"
rateclean , dataclean = wavfile.read(path2)
reversed_data = reverse_voice(dataclean)
write_voice(reversed_data, rateclean,
'/Users/arasvalizadeh/Downloads/audio/newaudio3/reversepotc.wa
v'
)

rateslow , dataslow = wavfile.read(
"/Users/arasvalizadeh/Downloads/audio/newaudio3/slowpotc.wav")
ratefast , datafast = wavfile.read(
"/Users/arasvalizadeh/Downloads/audio/newaudio3/fastpotc.wav")
ratereverse , datareverse = wavfile.read(
"/Users/arasvalizadeh/Downloads/audio/newaudio3/slowpotc.wav")

Datas = [dataclean,datafast,datareverse,dataslow]
Rates = [rateclean,ratefast,ratereverse,rateslow]
mixed_data, mixed_rate = mix_voices(Datas, Rates)
write_voice(mixed_data, mixed_rate,
'/Users/arasvalizadeh/Downloads/audio/newaudio3/mixpotc.wav')
```

در ادامه برنامه صرفا به فراخوانی صوت ها از فایل ساخت موارد خواسته شده می پردازیم که در عکس فوق مشخص شده (تمامی فایل ها بر روی سیستم شخصی آراس تست و ران شده برای همین مسیر فایل ها روی سیستم آراس هستش)

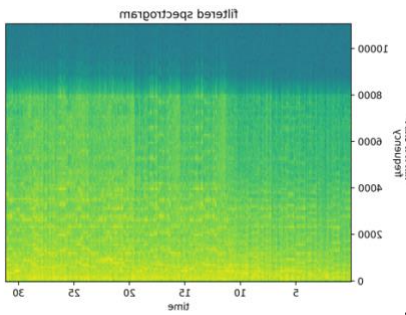
خروجی های سیگنال صوت اولیه:

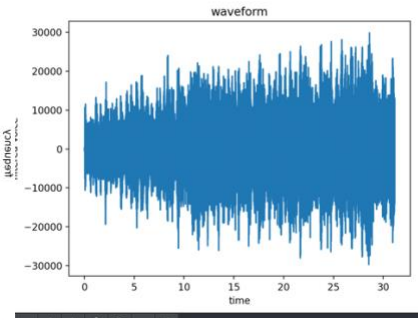


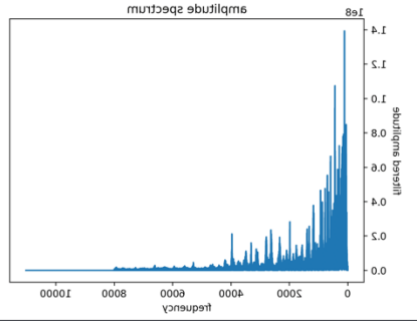




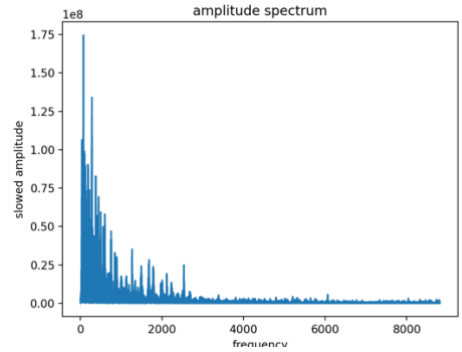
خروجی های صدای فیلتر شده :

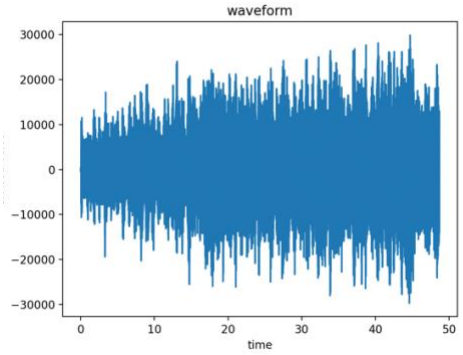


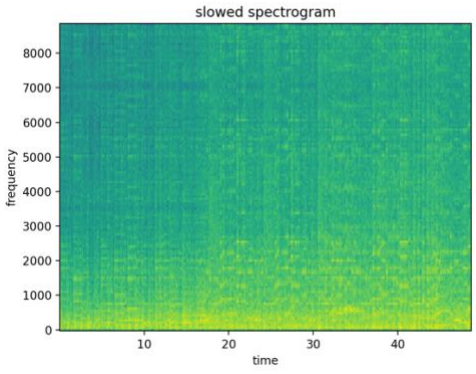




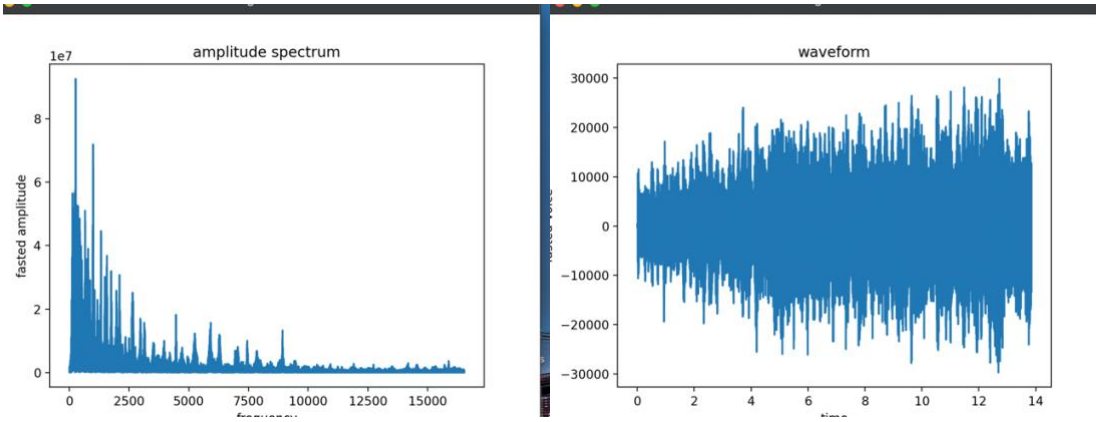
خروجی صدای آروم شده

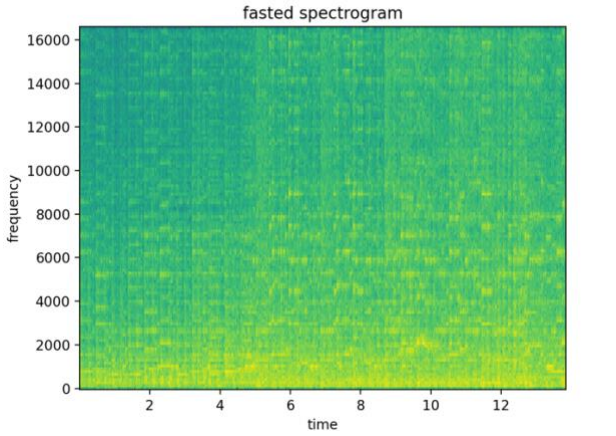




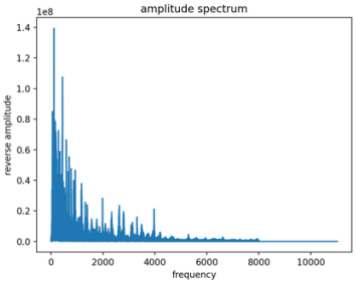


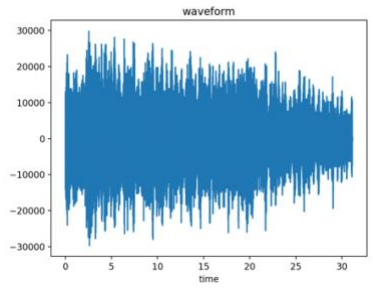
خروجی صدای سریع شده :

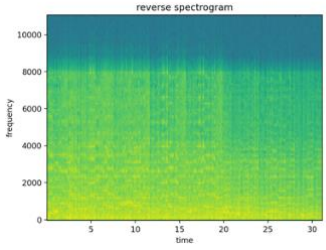




خروجی ریورس شده :







خروجی صدای میکس شده :