

Bài tập lớn môn Xử lý tiếng nói

Đề tài: Audio to text

Thành viên: Mạc Đình Minh – 17021296

Trịnh Thanh Tùng – 17021356

A. Giới thiệu bài toán

1. Vấn đề Speech-to-Text (STT)

Vấn đề Speech-to-Text (STT) hay còn gọi là Speech Recognition – Nhận dạng giọng nói là 1 lĩnh vực con trong Computational Linguistics, trong đó STT phát triển lý thuyết và công nghệ nhận dạng và chuyển dữ liệu ngôn ngữ dạng âm thanh thành ký tự.

Speech-to-Text tập trung vào khả năng nhận dạng nội dung của người nói (speaker) và chuyển nội dung đó thành dữ liệu dạng ký tự, điều này hoàn toàn khác với Voice Recognition hoặc Speaker Identification - tập trung vào phân loại và nhận dạng người nói.

Các đặc điểm chính của vấn đề:

- Âm thanh được truyền dưới dạng sóng, để có thể lưu trữ âm thanh được ghi lại bằng độ cao của sóng âm sau các khoảng thời gian bằng nhau.

Những thách thức:

- Cùng 1 nội dung nhưng khác nhau về âm điệu, cao độ, âm lượng,...
- Cùng 1 từ nhưng có thể phát âm khác nhau tùy theo vùng, lãnh thổ cùng nói 1 ngôn ngữ.
- Dữ liệu âm thanh khá dễ bị nhiễu. Điều này ảnh hưởng đến quyết định chọn lựa tập dữ liệu cho việc train.

2. Những việc đã làm được

a. Cắt file audio:

Nhóm đã sử dụng 1 chương trình python để cắt các file audio gk của lớp thành các file wav ngắn theo âm tiết dựa vào file srt và xếp vào các folder riêng cho từng âm tiết. trong phần này nhóm sử dụng pysrt và pydub hỗ trợ cho việc cắt file wav theo srt.

- **pysrt** là 1 thư viện để chỉnh sửa hoặc tạo file SubRip, bên cạnh đó pysrt cũng cung cấp một số lệnh srt hữu ích để thay đổi, tách hoặc chia tỷ lệ một tệp .srt. (SRC:[<https://pypi.org/project/pysrt/>]) (<https://pypi.org/project/pysrt/>)
- **pydub** gồm các thư viện và chương trình để xử lý video, audio, multimedia files và streams. Trong chương trình bọn em sử dụng chính là ffmpeg, được thiết kế để xử lý các video và audio dựa trên command line và được sử dụng rộng rãi để chuyển đổi định dạng, chỉnh sửa cơ bản (cắt và ghép), chia tỷ lệ video, hiệu ứng hậu kỳ video (SCR: <https://github.com/jiaaro/pydub>)

```

7 AudioSegment.converter = "E:\Documents\Desktop\XLTN_FinalTerm\ffmpeg-4.3-win32-static\bin\ffmpeg.exe" #thu vien ff
8 AudioSegment.ffmpeg = "E:\Documents\Desktop\XLTN_FinalTerm\ffmpeg-4.3-win32-static\bin\ffmpeg.exe"
9 AudioSegment.ffprobe = "E:\Documents\Desktop\XLTN_FinalTerm\ffmpeg-4.3-win32-static\bin\ffprobe.exe"
10
11 # pip install pydub
12 # pip install pysrt
13
14 import pysrt
15
16 import os
17 from pathlib import Path
18 import pathlib
19 import glob
20
21 def convert_to_millisecond(miniSub):
22     start = miniSub.start.ordinal
23     end = miniSub.end.ordinal
24
25     return start, end
26
27 if __name__ == '__main__':
28     word_dict = dict()
29
30     dirName = "E:\Documents\Desktop\XLTN_FinalTerm\Data2" #todo : sua thanh folder muon chua data da xu ly
31
32     print("Start processing .....")
33
34
35
36
37

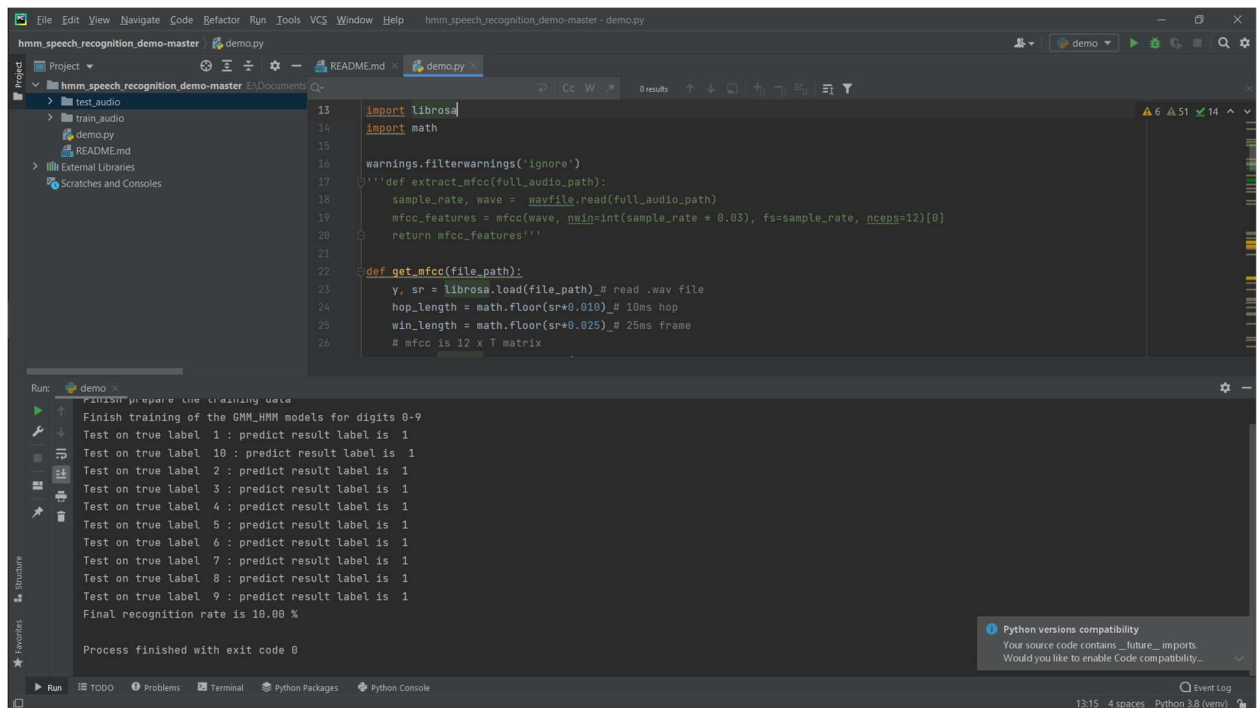
```


hỗ trợ trên Python 3, do đó nhóm đã sử dụng mfcc từ librosa đồng thời thay thế hàm get_mfcc từ model HMM của thầy để trích xuất ra các đặc trưng âm thanh.

Dùng GMMHMM trong hmmlearn để xây dựng mô hình cho từ đơn lẻ, quá trình nhận dạng sẽ so sánh xác suất nào là lớn nhất làm kết quả nhận dạng, các trạng thái nối tiếp nhau theo trình tự thời gian nên một trạng thái sẽ đi kèm với các xác suất chuyển trạng thái.

Tuy nhiên kết quả thu được không khả quan do khó khăn chính là không biết dc ma trận chuyển đổi trạng thái, và xác suất khởi tạo, ngay cả khi đã thay đổi giá trị của ma trận và xác suất khởi tạo kết quả vẫn ra thấp với dataset mà tác giả đã cung cấp từ trước (Bản tiếng Trung:

https://github.com/wblgers/hmm_speech_recognition_demo)



```
13 import librosa
14 import math
15
16 warnings.filterwarnings('ignore')
17
18 def extract_mfcc(full_audio_path):
19     sample_rate, wave = wavfile.read(full_audio_path)
20     mfcc_features = mfcc(wave, nmin=int(sample_rate * 0.03), fs=sample_rate, ncoeffs=12)[0]
21     return mfcc_features'''
22
23 def get_mfcc(file_path):
24     y, sr = librosa.load(file_path) # read .wav file
25     hop_length = math.floor(sr*0.010) # 10ms hop
26     win_length = math.floor(sr*0.025) # 25ms frame
27     # mfcc is 12 x T matrix
```

Run demo x

```
Finish preparing the training data
Finish training of the GMM_HMM models for digits 0-9
Test on true label 1 : predict result label is 1
Test on true label 10 : predict result label is 1
Test on true label 2 : predict result label is 1
Test on true label 3 : predict result label is 1
Test on true label 4 : predict result label is 1
Test on true label 5 : predict result label is 1
Test on true label 6 : predict result label is 1
Test on true label 7 : predict result label is 1
Test on true label 8 : predict result label is 1
Test on true label 9 : predict result label is 1
Final recognition rate is 10.00 %
Process finished with exit code 0
```

Python versions compatibility
Your source code contains _future_ imports. Would you like to enable Code compatibility...

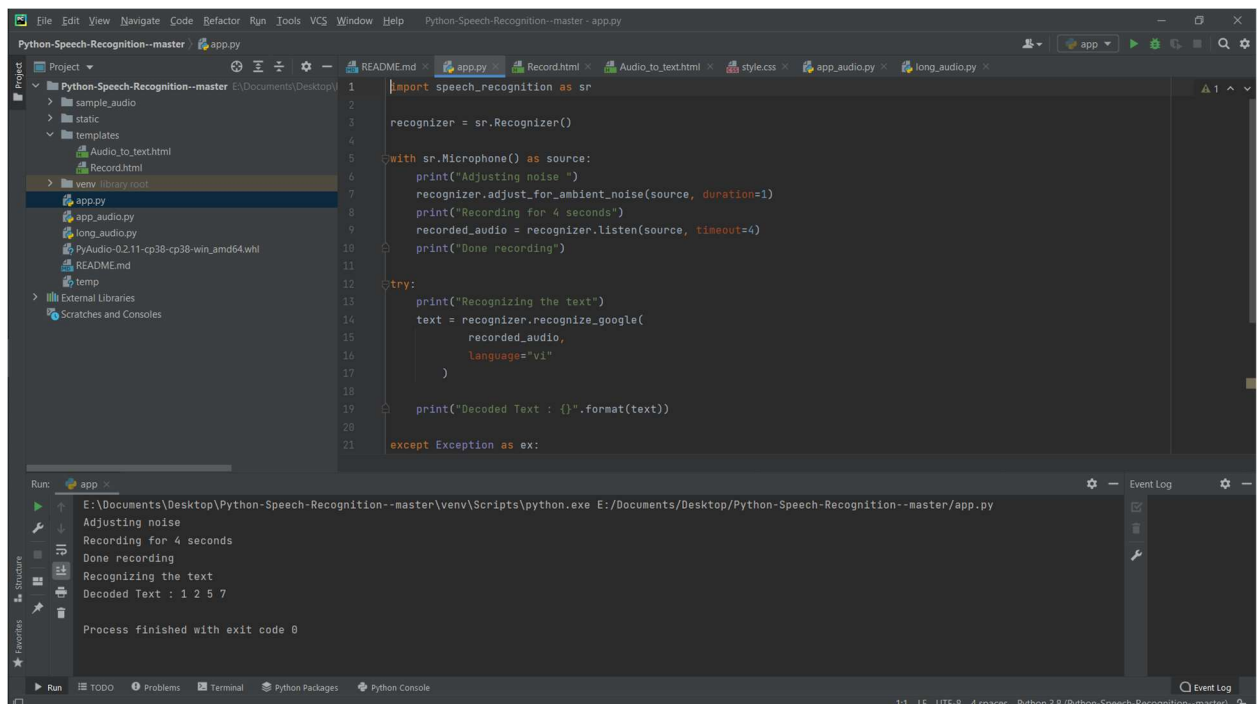
c. Application sử dụng Api

Cuối cùng, để đạt được kết quả tốt cho web application nhóm sử dụng api speech_recognition. 2 phần mà nhóm em đã thực hiện xong:

- **Phần 1:** Web Application sử dụng API upload file wav và chuyển file wav sang dạng text. API sử dụng là Google Cloud Speech Library for Python Speech-to-Text API recognition : <https://cloud.google.com/speech-to-text>. Tham khảo thêm: https://github.com/Uberi/speech_recognition. Flask là một web frameworks, thuộc loại micro-framework được xây dựng bằng ngôn ngữ lập trình Python. Flask cho phép xây dựng các ứng dụng web từ đơn giản tới phức tạp.

Link Youtube: <https://youtu.be/Bvv81f1cq-0>

- **Phần 2:** Sử dụng API để ghi âm và chuyển sang dạng text: Sử dụng Speech Recognition API để chuyển speech to text. PyAudio là một thư viện liên kết của Python và PortAudio , là một thư viện I / O âm thanh đa nền tảng. Với PyAudio có thể sử dụng Python để phát và ghi âm thanh trên nhiều nền tảng khác nhau, chẳng hạn như GNU / Linux, Microsoft Windows và Apple Mac OS X / macOS.



```
1 import speech_recognition as sr
2
3 recognizer = sr.Recognizer()
4
5 with sr.Microphone() as source:
6     print("Adjusting noise ")
7     recognizer.adjust_for_ambient_noise(source, duration=1)
8     print("Recording for 4 seconds")
9     recorded_audio = recognizer.listen(source, timeout=4)
10    print("Done recording")
11
12 try:
13     print("Recognizing the text")
14     text = recognizer.recognize_google(
15         recorded_audio,
16         language="vi"
17     )
18
19     print("Decoded Text : {}".format(text))
20
21 except Exception as ex:
```

Run: app

E:\Documents\Desktop\Python-Speech-Recognition-master\venv\Scripts\python.exe E:/Documents/Desktop/Python-Speech-Recognition-master/app.py

Adjusting noise
Recording for 4 seconds
Done recording
Recognizing the text
Decoded Text : 1 2 5 7

Process finished with exit code 0

Link Github toàn bộ project: https://github.com/MinhMac3011/XLTN_Final