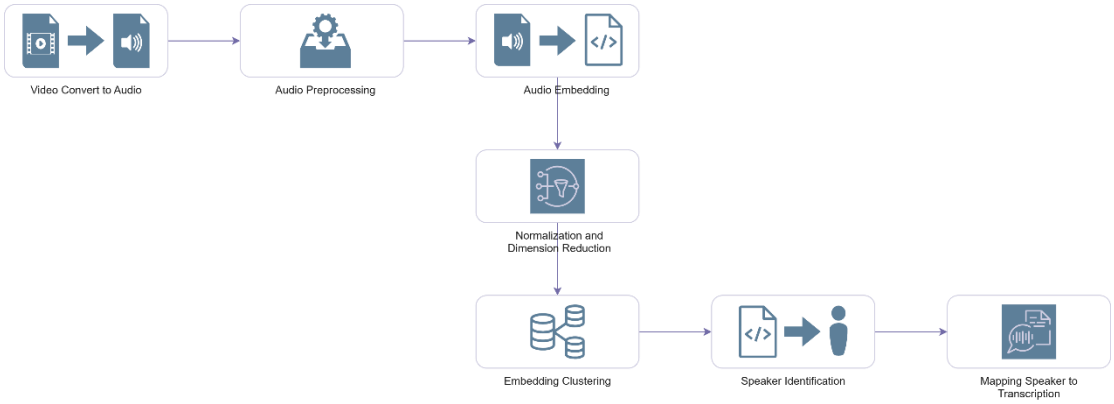
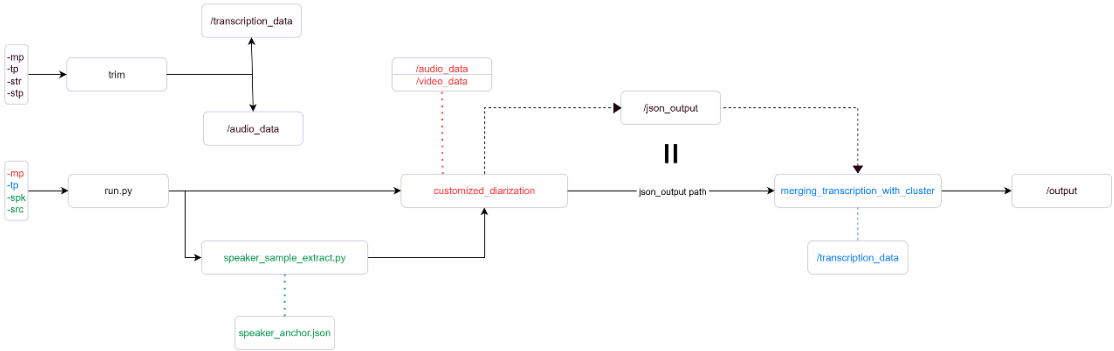
Speaker Diarization

Environment

- Operation System: Microsoft Windows 11 Pro
- CPU: 13th Gen Intel(R) Core(TM) i5-1335U
- GPU: NVIDIA GeForce RTX 2050
- Python: 3.11.9 (Using pyenv + venv)

Workflow





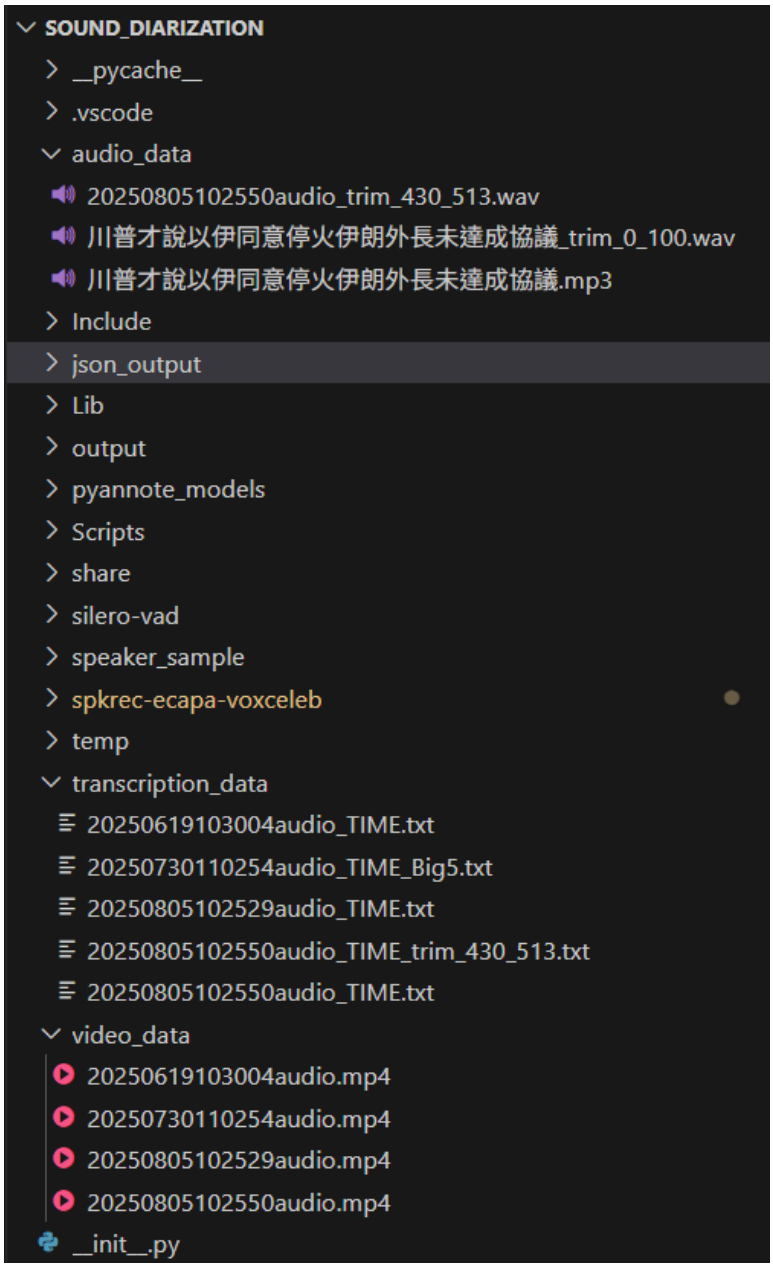
File Hierarchy

```
|—/sound_diarization
    └—— |
        —pyannote_diarization_config.json
        —customized_diarization_config.json
        —speaker_anchor.json
        —__init__.py
        —run.py
        —pyannote_diarization_offline.py
        —customized_diarization.py
        —speaker_sample_extract.py
        —merging_transcription_with_cluster.py
        —utils.py
        —requirements.txt
        —/pyannote_models
            └—— |
                —...
                —pyannote_diarization_config.yaml
        —/silero-vad
            └—— |
                —...
                —hubconf.py
        —/spkrec-ecapa-voxceleb
            └—— |
                —...
                —hyperparams.yaml
        —/audio_data
            └—— |
                —...
                —test.mp3
        —/video_data
            └—— |
                —...
                —test.mp4
        —/transcription_data
            └—— |
                —...
                —test.txt
        —/speaker_sample
            └—— |
                —...
                —speaker1.npy
                —speaker1.wav
        —/temp
        —/json_output
        —/output
```

Data

This project has 3 types of input data:

- Audio Data: The audio file can be any audio format. All audio data will be stored in the folder `audio_data`.
- Video Data: The video file can be any video format. All video data will be stored in the folder ` video_data`.
- Transcription Data: The transcription file should be a .txt file with a strict row-by-row format; [(start:.2f)s -> (stop:.2f)s] sentences. All transcription data will be stored in the folder ` transcription_data`.

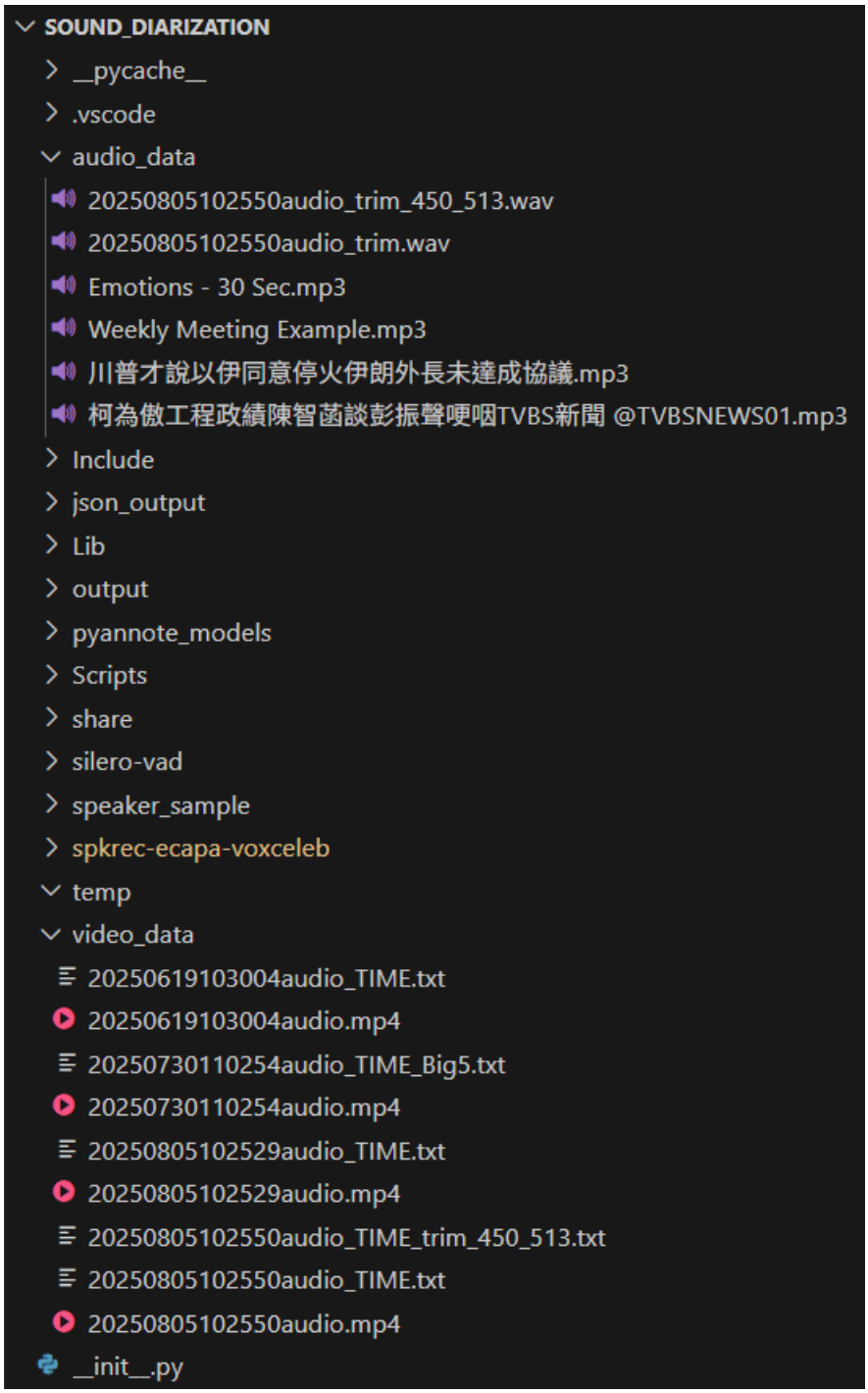


Introduction

This package includes 2 diarization programs:

- pyannote_diarization_offline:
  This program uses the `pyannote/speaker-diarization-3.1` model to perform speaker diarization, this program serves as the benchmark for our `customized_diarization` program.
- customized_diarization:
  This program performs speaker diarization by implementing the whole workflow from scratch. It supports detailed model tuning and customized parameters adjustment at each stage of the workflow.

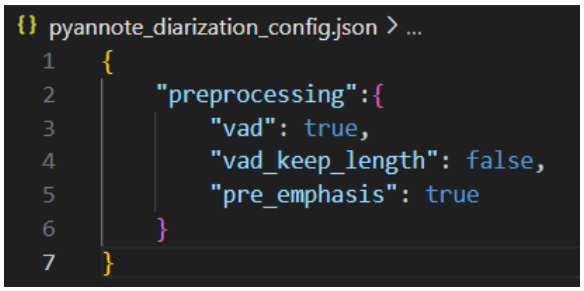
Each program supports audio or video as the input file. The audio files should be placed in folder `audio_data`, and the video files should be placed in folder `video_data`.



pyannote_diarization_offline

pyannote_diarization_config.json

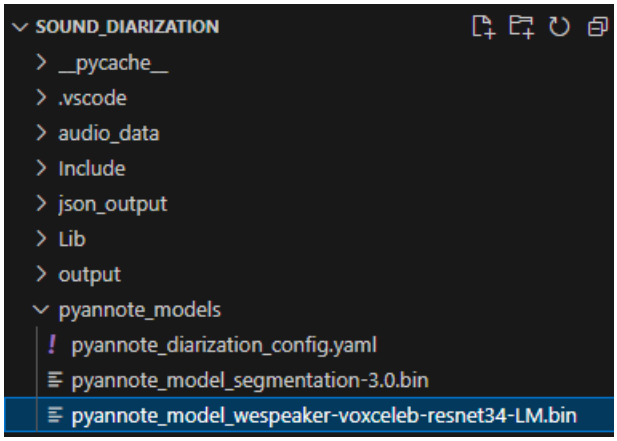
The JSON file ` pyannote_diarization_config.json` controls what kind of preprocessing techniques will apply to the input file.



Parameter	Data Type	Description
vad	Bool	If True, voice activity detection will apply. Default is True.
vad_keep_length	Bool	If True, vad will remove the non-speech portions and concatenate the speech portions. Otherwise, the non-speech portions will be replaced with silence. Default is False.
pre_emphasis	Bool	If True, Pre-emphasis will apply. Default is True.

/ pyannote_models

The folder ` pyannote_models` contains the necessary files for deploying the speaker diarazation model` pyannote/speaker-diarization-3.1` on-premises. To know more details about the implementation, see https://github.com/pyannote/pyannote-audio/blob/main/tutorials/community/offline_usage_speaker_diarization.ipynb



customized_diarization

run

The program `run.py` is the main program of the customized_diarization workflow.

This program accepts 4 parameters:

```
parser.add_argument("-mp", "--media_path", help="input media path", dest="media_path", type=str, required=True)
parser.add_argument("-tp", "--transcription_path", help="transcription path", dest="transcription_path", type=str, required=False)
parser.add_argument("-spk", "--speaker", help="The string assigned speakers join with `|`, e.g. `Jeremy|Barry`", dest="speaker", type=str, required=False)
parser.add_argument("-src", "--source", help="The string assigned source join with `|`, e.g. `./file1.mp4|./file2.mp4`", dest="source", type=str, required=False)
```

customized_diarization_config.json

The JSON file ` customized_diarization_config.json` controls the detailed parameters at each stage applying to the input file.

Parent	Parameter	DataType	Description
init	embed	Str	The embedding method. Default is “ecapa”.
init	reduction	Str	The dimension reduction method. It can be “pca” or “kpca”, with a default value of “pca”.
init	normalizartion	Bool	If True, the L2 normalization will apply to the audio embeddings. Default is True.
init	cluster	Str	The clustering method. It can

			be “HDBSCAN”, “AP” (affinity propagation) or “SC” (spectral clustering), with a default value of “HDBSCAN”.
init	speakers	Int	The number of speakers present in the input file, only needs to be specified if “SC” is chosen as the clustering method.
init	threshold	Float	The threshold to consider that a segment embedding belongs to a certain speaker, the value range is between 0 and 1, with a default value of 0.65.
segmentation	window	Float	The window size of a segment in seconds. Default is 1.6.
segmentation	stride	Float	The stride size of a segment in seconds. Default is 0.08.
segmentation	batch	Int	The batch size of a segment. Default is 1.
preprocessing	vad	Bool	If True, voice activity detection will apply. Default is True.
preprocessing	vad_keep_length	Bool	If True, vad will remove the non-speech portions and concatenate the speech portions. Otherwise, the non-speech portions will be replaced with silence. Default is False.
preprocessing	pre_emphasis	Bool	If True, Pre-emphasis will apply. Default is True.
dimension_reduction	ratio	Float	The expected explained variance ratio used to determine how many principal components to retain. Default is 0.85.

dimension_reduction	n_components	Int	The number of compoemts that will yield after dimension reduction. Default is 25.
dimension_reduction	criteria	Str	Specifies whether to apply `n_components` or `ratio` during dimensionality reduction. Default is `n_components`.
dimension_reduction	random_state	Int	The random state for the dimension reduction methods.
dimension_reduction	kpca	Dict	The specific parameters for `KPCA` method including `kernel` and `gamma`. See https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.KernelPCA.html to get more info.
clustering	hdbscan	Dict	The specific parameters for `HDBSCAN`. See https://scikit-learn.org/stable/modules/generated/sklearn.cluster.HDBSCAN.html to get more info.
clustering	affinitypropagation	Dict	The specific parameters for `affinitypropagation`. The parameter `distance` isn't from the original scikit-learn docs, this parameter determines the kind of distance applied to calculating the similiarity of datapoints For other parameters, see https://scikit-

			learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html to get more info.
clustering	spectralclustering	Dict	The specific parameters for ` spectralclustering ` . See https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html to get more info.

speaker_anchor.json

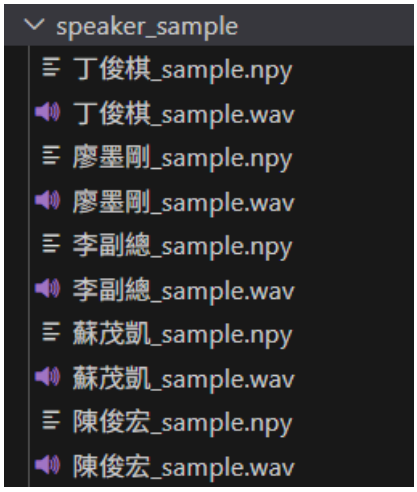
The JSON file ` speaker_anchor.json` stores a list speaker info, where each item defines a specific speaker's audio clip in a video/audio file.

Parameter	Data Type	Description
name	Str	The name of the speaker.
source	Str	The file path of the media file (audio or video) from which the segment is taken.
start	Int	The start time of the speaker's segment in seconds.
stop	Int	The stop time of the speaker's segment in seconds.

/speaker_sample

The folder ` speaker_sample` stores speaker sample embeddings and clips generating by `speaker_sample_extract.py`.

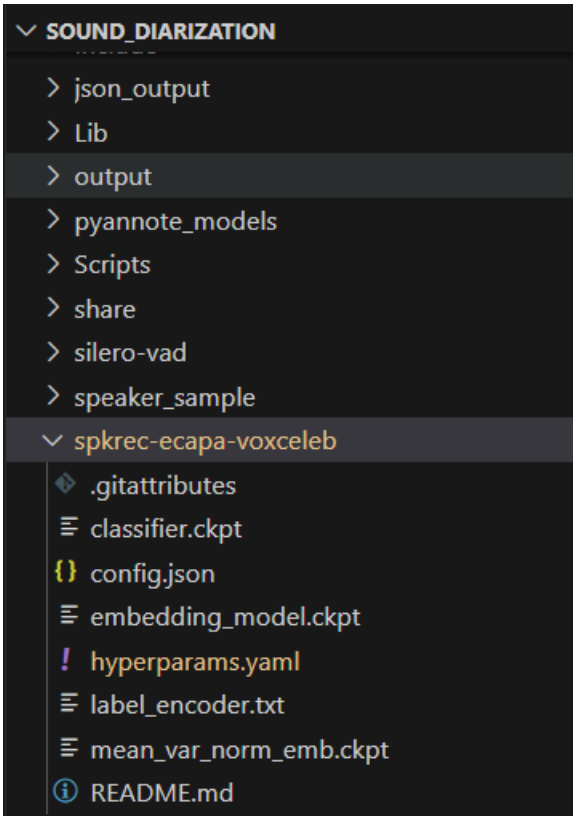
- Speaker sample embeddings are saved as .npy file.
- Speaker sample clips are saved as .wav file.



/ spkrec-ecapa-voxceleb

The folder `spkrec-ecapa-voxceleb` contains the ECAPA-TDNN speaker embedding model repository. We download the entire repository and place it under the root directory of the environment to enable the on-premises speaker diarization embedding workflow.

The link of the repository: [speechbrain/spkrec-ecapa-voxceleb · Hugging Face]



/temp

The folder `temp` stores intermediate files, which will be removed once the

workflow complete.

speaker_sample_extract

The program `speaker_sample_extract.py` generates the speaker sample embeddings and clips, these outputs are saved in the folder `speaker_sample`.

merging_transcription_with_cluster

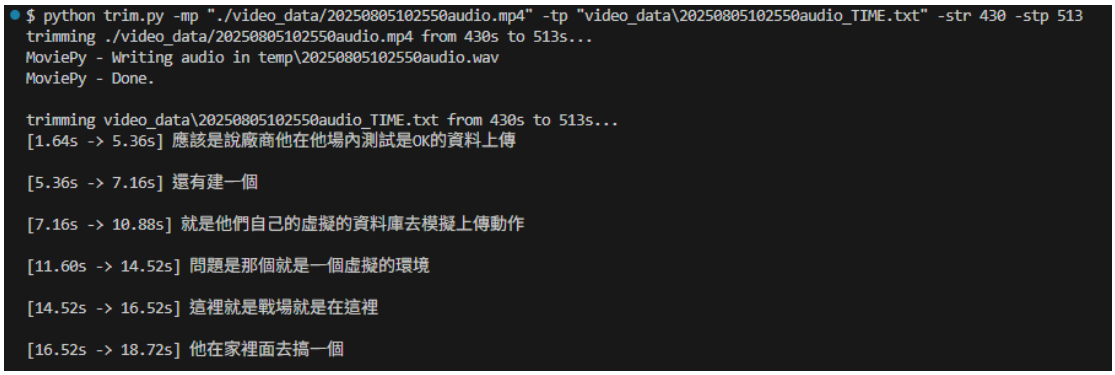
The program `merging_transcription_with_cluster.py` merges the transcription with the corresponding speakers/clusters line by line.

utils

The Python file `utils.py` contains all the general-purpose functions used throughout this project.

trim

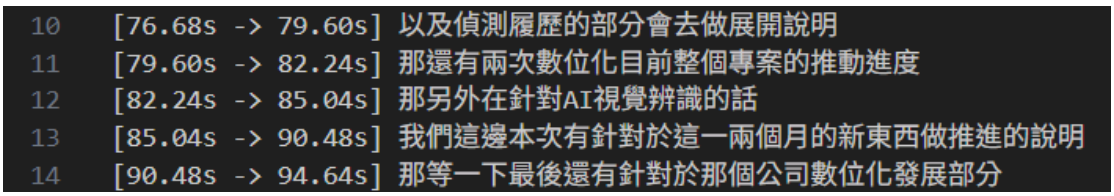
In some cases, the input media file may be too large to process, potentially causing the computer to shut down due to memory exhaustion. To address this, a simple function is provided to trim or extract a small clip of the media file along with the corresponding transcription.



The `trim` program accepts 4 parameters:

1. media_path: The path to the media file, which can be either a video or an audio file.
   transcription_path: The path to the transcription file, which must be a .txt file with a strict row-by-row format; [(start:.2f)s -> (stop:.2f)s] sentences .

regular expression: r"\[(\d+\.\d{2})s -> (\d+\.\d{2})s\] \w+\n"

2. 

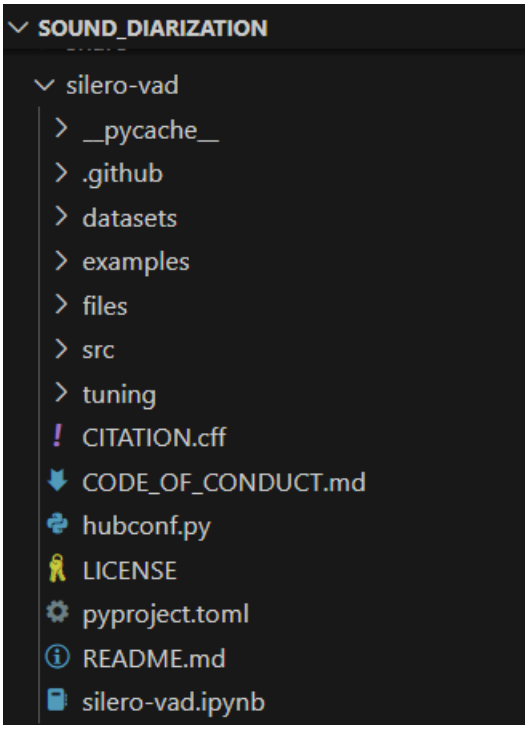
```
[76.68s -> 79.60s] 以及偵測履歷的部分會去做展開說明
[79.60s -> 82.24s] 那還有兩次數位化目前整個專案的推動進度
[82.24s -> 85.04s] 那另外在針對AI視覺辨識的話
[85.04s -> 90.48s] 我們這邊本次有針對於這一兩個月的新東西做推進的說明
[90.48s -> 94.64s] 那等一下最後還有針對於那個公司數位化發展部分
```

3. start: The start time (in seconds) of the trimmed segment.
4. stop: The stop time (in seconds) of the trimmed segment.

Regardless of whether the input is an audio or video file, the clips will be exported as a .wav file in the `audio_data` folder, and the corresponding transcription will be saved as a .txt file in the `video_data` folder.

/ silero-vad

The folder `silero-vad` contains the Voice Activity Detection (VAD) repository. We download the entire repository and place it under the root directory of the environment to enable on-premises speaker diarization VAD, which is utilized by both pyannote_diarization_offline and customized_diarization.
The link to the repository and implementation:
https://github.com/snakers4/silero-vad/blob/master/silero-vad.ipynb

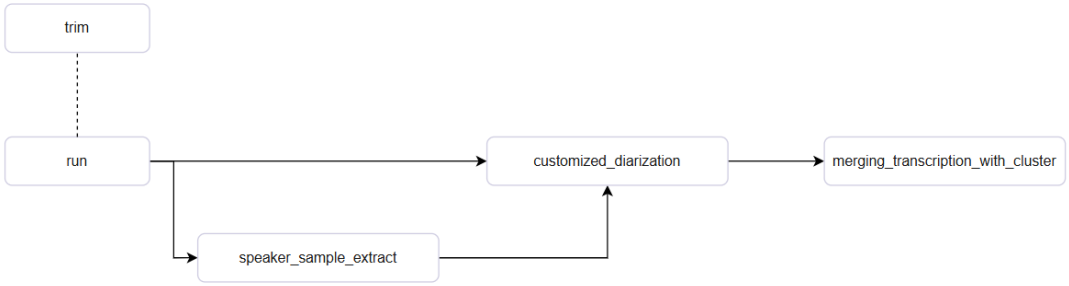


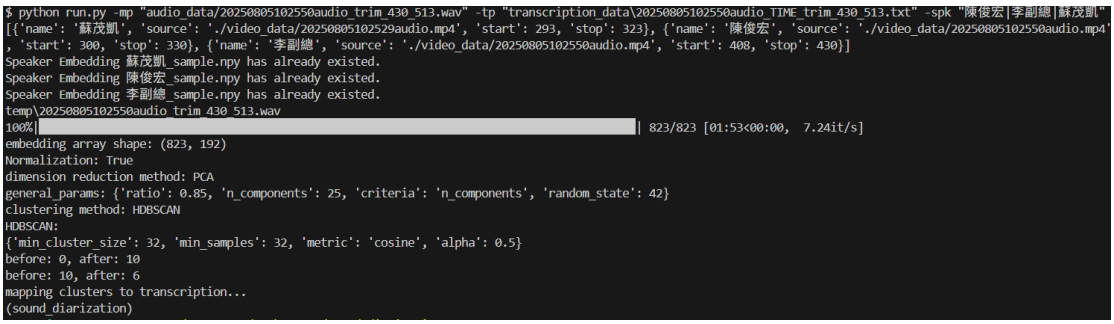
Usage

pyannote_diarization_offline

```
$ python pyannote_diarization_offline.py -p "audio_data\\川普才說以伊同意停火伊朗外長未達成協議.mp3"
Loading pyannote pipeline from pyannote_models\pyannote_diarization_config.yaml...
Changing working directory to C:\Users\680451\Documents\Work Space\sound_diarization
Changing working directory back to C:\Users\680451\Documents\Work Space\sound_diarization
start=0.03s stop=30.96s speaker_SPEAKER_01
start=31.05s stop=62.76s speaker_SPEAKER_02
start=62.76s stop=71.92s speaker_SPEAKER_03
start=72.14s stop=95.00s speaker_SPEAKER_02
start=95.07s stop=105.90s speaker_SPEAKER_00
start=106.04s stop=124.38s speaker_SPEAKER_02
start=124.62s stop=132.96s speaker_SPEAKER_04
start=133.14s stop=143.92s speaker_SPEAKER_02
```

customized_diarization





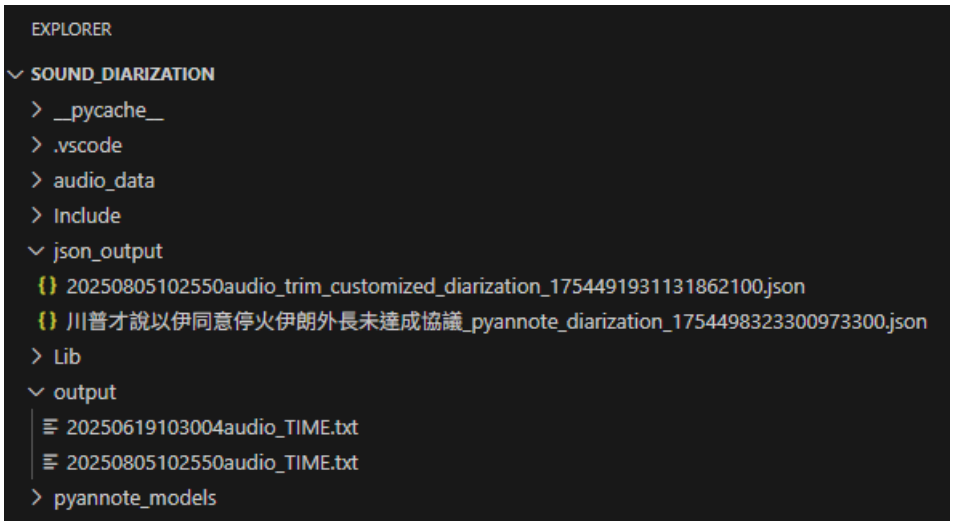
Output

In the clustering stage, the model will assign a cluster to each segment. And the subsequent down-stream processing flow will combine adjacent segments having the same cluster and apply other processes to smooth the output result.

There are 3 types of cluster labels:

1. Positive Number:
   The original cluster index output directly from the clustering model.
2. Name:
   When a transcription file is provided and the cosine similarity score between the speaker embedding and the segment embedding exceeds the threshold specified in `customized_diarization_config.json`, the original cluster index will be replaced by speaker label with the highest similarity score.
3. -1:
   Noise, or unrecognized voice.

There are 2 different output formats in this package.



- JSON file:

  The JSON format includes clusters along with their timelines, indicating start and end times in seconds. These files are stored in the folder `json_output `.

- Txt file

  The txt file resembles the original transcription input with the annotation for corresponding clusters or speakers in each row. These files are stored in the folder `output `.

```
[0.32s -> 1.48s] [李副總] 一切都很正常
[1.48s -> 3.76s] [李副總] 當然不會長出病毒來
[3.76s -> 6.20s] [李副總] 這裡就是什麼都有對不對
[6.20s -> 9.00s] [李副總] 當然病毒就會在這裡跑出來
[9.00s -> 12.68s] [李副總] 這樣子他去在家裡搞一個模擬的環境
[12.68s -> 16.88s] [-1] 跟實際現場打仗的狀況是一樣的嗎
[16.88s -> 20.28s] [-1] 好我會要求廠商就是看這禮拜最晚下禮拜
[20.28s -> 22.56s] [1] 看能不能入場拿去做一個實際的
[22.56s -> 25.96s] [-1] 因為就是他直接在場內做一個實際測試
[25.96s -> 27.68s] [李副總] 剛剛前面要講的問題是說
[27.68s -> 30.00s] [-1] 對啦現在就是我們想一個廠商對應一個
[30.00s -> 31.48s] [李副總] 想兩個廠商對應兩個
[31.48s -> 33.96s] [李副總] 廠商自己這樣好像也沒什麼招數可以拿出來
```

Since the `pyannote_diarization_offline` program just serves as a performance benchmark, it only yields the JSON format file and does not support the speaker identification process. That is, that it doesn't include speaker name labels.