Pyannote Tutorial

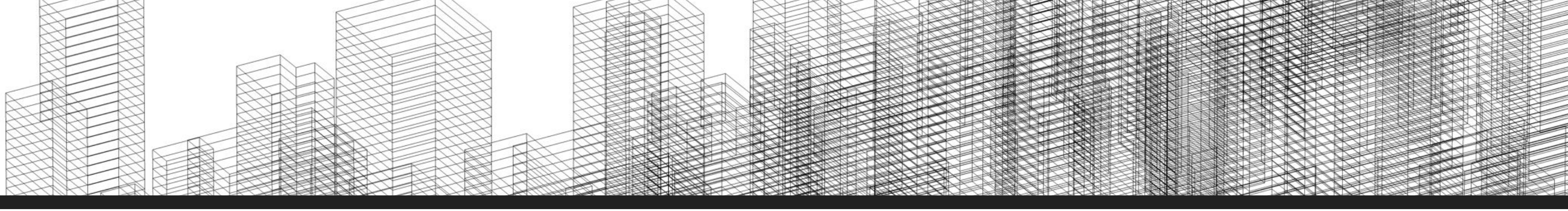
By Akshat Agrawal
(akshatagrawal1729@gmail.com)

Contents

1. What is Pyannote?
    a. Tasks Available
    b. Pipelines Available
2. Installation and setup
    a. Pyannote Installation
    b. Installing AMI-Diarisation setup
    c. Hugging Face Setup
3. Using Pyannote
    a. Pyannote Pipelines
    b. How to use pre trained pyannote models?
    c. Pyannote Data-Loader
    d. How to train a model on your dataset?
    e. Fine Tuning on your dataset
    f. Playing around with pyannote
    g. Custom Modifications in Pyannote
4. Tips
    a. For Training/ Fine Tuning/ Modifications
    b. For NSCC
5. Contribution

What is Pyannote?

1. Open Source toolkit for Speaker Diarization
2. Based on Pytorch

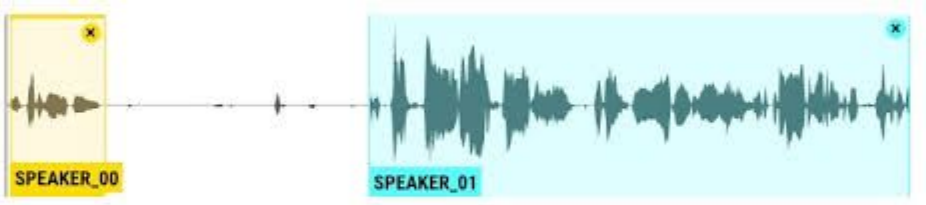


Tasks Available

1. Voice Activity Detection
2. Overlapped Speech Detection
3. Segmentation

Pyannote Pipelines

- Available Pipelines
    a. Voice Activity Detection
    b. Speaker Diarisation
    c. Overlapped Speech Detection
    d. Speaker Segmentation

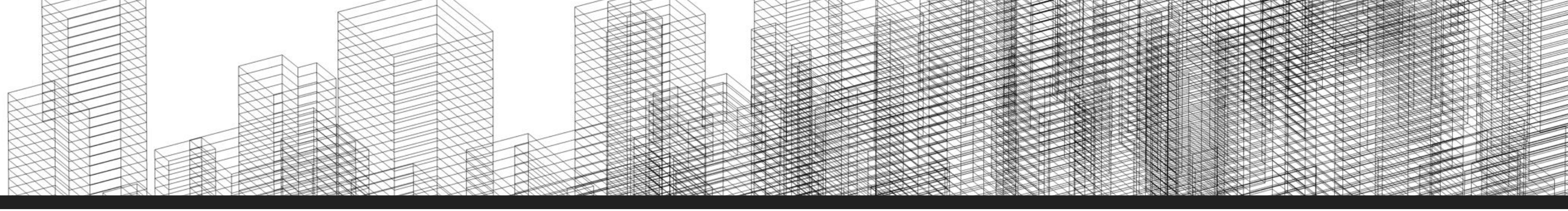


Installation

1. #Create a conda environment named pyannote
   a. conda create -n pyannote python=3.8
   b. conda activate pyannote
2. #Install other dependencies
   a. conda install pytorch==1.11.0 torchvision==0.12.0 torchaudio==0.11.0 -c pytorch
3. #Install pyannote.audio
   a. pip install pyannote.audio

Installing AMI-Diarisation setup

1. Clone AMI-Diarisation setup
   a. git clone https://github.com/pyannote/AMI-diarization-setup.git
2. Downloading dataset
   a. cd AMI-diarization-setup/pyannote
   b. sh download_ami.sh



Hugging Face

Setting Up Hugging Face Model Hub Access Token

1. Visit hf.co/pyannote/speaker-diarization and hf.co/pyannote/segmentation and accept user conditions
2. Visit hf.co/settings/tokens to create an access token

Pyannote Pipelines

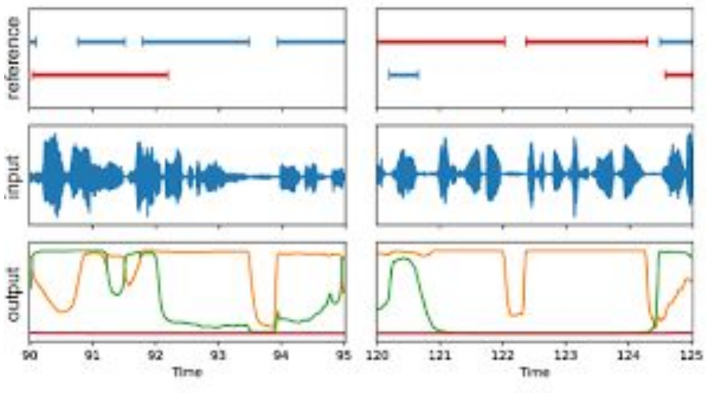
Data
Operation 1
Operation 2
Operation 3
Data
Pipeline

1. For applying a pre-trained pipeline, refer:
   a. https://github.com/pyannote/pyannote-audio/blob/develop/tutorials/applying_a_pipeline.ipynb

2. For playing around with pyannote pipelines, refer:
   a. https://github.com/AKSHAT2429/Pyannote-Tutorial/blob/main/Pyannote_Pipeline.ipynb

How to use pre trained pyannote models?

1. Please refer:
   a. https://github.com/pyannote/pyannote-audio/blob/develop/tutorials/applying_a_model.ipynb

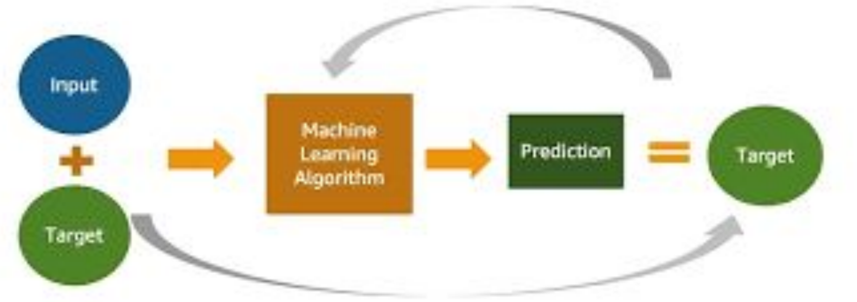


Pyannote Data-Loader

1. The details of the dataset should be included in a database.yml file, which contains:
    a. Name of all the audio files.
    b. RTTM files of the audio.
    c. UEM files of the audio: It is basically the duration of the audio for which you need to train/test your model.
    d. Keep in mind the protocol to be used (Important) (Check: pyannote/database/protocol)
2. You can find a sample database.yml file at:
    a. https://github.com/AKSHAT2429/Pyannote-Tutorial/blob/main/Creating_Database/database.yml
3. Files useful while creating a database.yml can be found at:
    a. https://github.com/AKSHAT2429/Pyannote-Tutorial/tree/main/Creating_Database

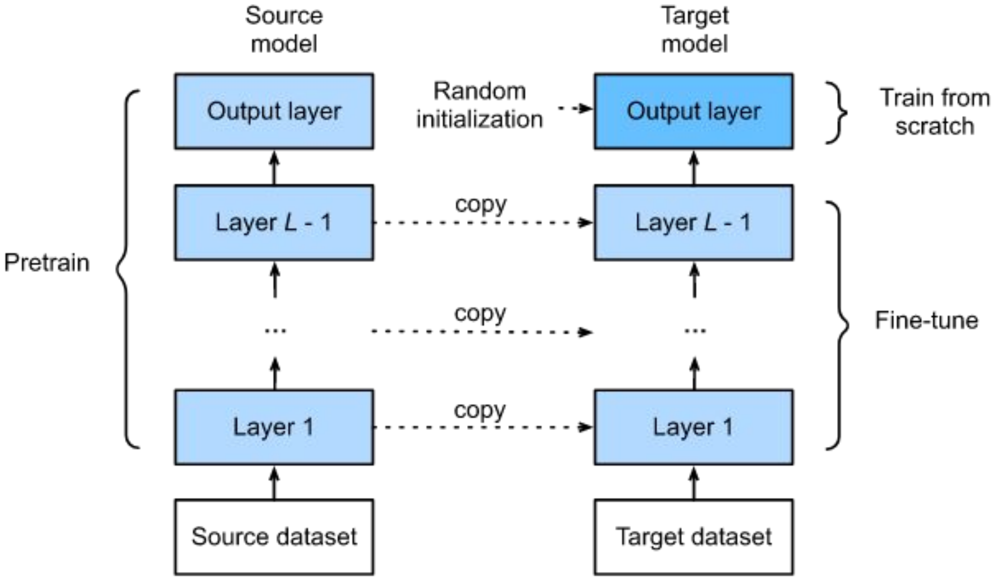
How to train a model on your dataset?

1. Please refer:
   a. https://github.com/AKSHAT2429/Pyannote-Tutorial/blob/main/Training_a_model_from_scratch.ipynb



Fine Tuning on your dataset

1. Please refer:
   a. https://github.com/AKSHAT2429/Pyannote-Tutorial/blob/main/Finetuning_a_model.ipynb



Playing around with pyannote

- To make custom changes:
  - Navigate to “anaconda3/envs/pyannote/lib/python3.8/site-packages/pyannote/audio” and make corresponding changes

    OR

  - Write custom function in the notebook itself

Custom Modification in Pyannote

1. How to define your own model?
    a. https://github.com/pyannote/pyannote-audio/blob/develop/tutorials/add_your_own_model.ipynb

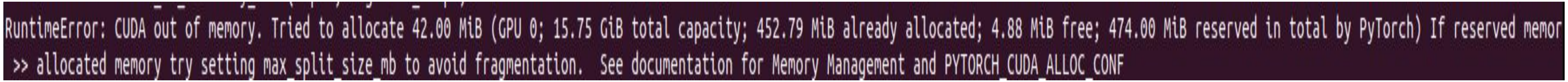
2. How to define your own task?
    a. https://github.com/pyannote/pyannote-audio/blob/develop/tutorials/add_your_own_task.ipynb

Tips for Training/ Fine Tuning/ Modifications

- Fine Tuning on new dataset
  - Always implement early stopping.
  - Try and compare different optimizers
  - Try learning rate scheduling
  - Try gradual unfreezing of layers
  - Try weight decay
  - Try data augmentation
  - Check if any kind of normalization (to any parameters or gradients is done) and take care of it while fine tuning.
  - Incorporate multiple evaluation methods (helps sometimes)
- When adding new feature/ modification:
  - Create a subset of full dataset (maybe 10 percent) to test if your approach is working well. If it works well then train whole model on full dataset to save time.
  - To test any new modifications, train on one epoch and explore the modification.

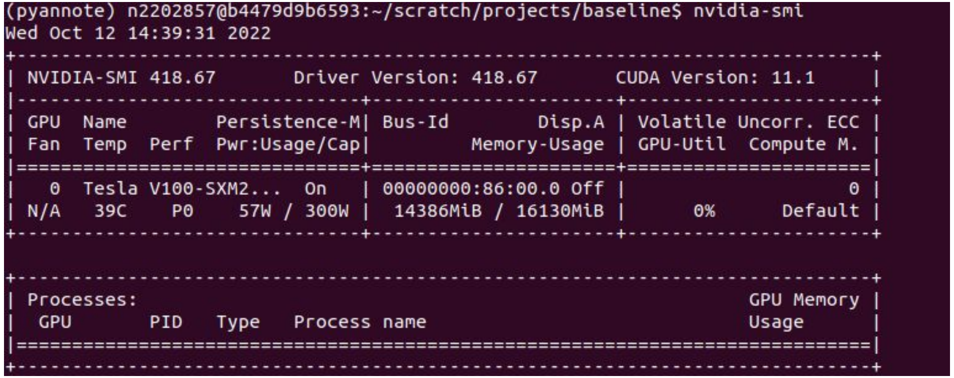
Tips for NSCC

- Sometimes you may need to decrease number of workers to fit it in NSCC GPU (To be done while initialising pyannote task).



```
RuntimeError: CUDA out of memory. Tried to allocate 42.00 MiB (GPU 0; 15.75 GiB total capacity; 452.79 MiB already allocated; 4.88 MiB free; 474.00 MiB reserved in total by PyTorch) If reserved memor
 >> allocated memory try setting max_split_size_mb to avoid fragmentation.  See documentation for Memory Management and PYTORCH_CUDA_ALLOC_CONF
```

- If you encounter the error above, check GPU memory (nvidia -smi)
    - If it is full - - -> Open a new terminal window and request qsub again (AND raise a ticket to NSCC).



```
(pyannote) n2202857@b4479d9b6593:~/scratch/projects/baseline$ nvidia-smi
Wed Oct 12 14:39:31 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 418.67       Driver Version: 418.67       CUDA Version: 11.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:86:00.0 Off |                    0 |
| N/A   39C    P0    57W / 300W |  14386MiB / 16130MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
+-----------------------------------------------------------------------------+
```

Contribution

- ➔ I would like to acknowledge with much appreciation
  - ◆ Prof. Chng Eng Siong
  - ◆ Lim Zhi Hao
  - ◆ Wong Chee Hoong Melvin
  - ◆ Liu Chenyu

Thank You!