NEMO

This speech - to - text module is built using NVIDIA NeMo. To train an ASR model from NeMo you need two sets of data (manifests), one for training/validation and one for testing and a configuration .yaml file. Listed below are details about each component.

Manifest

https://docs.nvidia.com/deeplearning/nemo/user-guide/docs/en/stable/asr/datasets.html

Configuration

https://docs.nvidia.com/deeplearning/nemo/user-guide/docs/en/stable/asr/configs.html

HUGGING FACE

This speech - to - text model is built using Wav2Vec2 from HuggingFace and is used for transcription of .wav files. To retrain this model, a base model and labeled training data file should be provided.

The two options for selecting a base model are as follows:

Providing a HuggingFace model:

HuggingFace hosts a variety of models that can be used as a base for training. This can be useful for training in a new language or improving upon other people's works.

Selecting a model from the website:

If you want to continue working on a previous model and improve it with new/more accurate data, this is a good option for the base model. To do so, choose from the drop down the model you would like to build upon.

The datafile must be of .json format. For every audio input that needs to be retrained, there should be the processed wav file (the output from librosa.load()) under the tag “audio” and the labeled transcript under the label “text”. The transcript should be lowercase with all punctuation except apostrophes. For example (note: this example file does not contain valid data):

{“text”:{“0”: "According to all known laws of aviation, there is no way a bee should be able to fly."},
“audio”:{“0”: [0.0,0.0,0.0,0.0,-0.0000610352,0.0,-0.0000610352,0.0000305176,-0.0001525879,0.0001220703,-0.000213623,0.0002441406,-0.0003967285,0.0005187988,-0.0010070801,0.0036621094,0.0098266602,0.0093078613,0.009765625,0.008605957,0.0099182129,0.0062561035,0.0065307617,0.0067443848,0.00390625,0.0025939941,0.003692627,-0.0018615723,0.005279541,]}}

Diarization

This speaker diarization model is built using Pyannote and SpeechBrain and is used to differentiate between speakers on an audio file. To retrain this model, a base model and training directory must be provided

To select a base model, choose from the drop-down menu the model you would like to build upon.

The data directory should have three subfolders in it by the name of "RTTM_set", "UEM_set" and "WAV_set". When those three directories are provided, it will automatically generate the configuration file, which is of '.yml' format, that allows the speaker diarization model to be retrained (the database format from PyannoteDatabase). For example:

```
{SampleData:
  {SpeakerDiarization:
    { only_words:
      { development:
          {annotated: ./data_preparation/TrainingData/SampleData/UEM_set/{uri}.uem,
           annotation: ./data_preparation/TrainingData/SampleData/RTTM_set/{uri}.rttm,
           uri: ./data_preparation/TrainingData/SampleData/LIST_set/dev.txt},
      { test:
          {annotated: ./data_preparation/TrainingData/SampleData/UEM_set/{uri}.uem,
           annotation: ./data_preparation/TrainingData/SampleData/RTTM_set/{uri}.rttm,
           uri: ./data_preparation/TrainingData/SampleData/LIST_set/test.txt,}
      { train:
          {annotated: ./data_preparation/TrainingData/SampleData/UEM_set/{uri}.uem
           annotation: ./data_preparation/TrainingData/SampleData/RTTM_set/{uri}.rttm
           uri: ./data_preparation/TrainingData/SampleData/LIST_set/train.txt}}}}}}
```