L635-Fall2025-Assignment 02: Fine-tuning Speech Foundation Models using ESPnet EZ

Main references:

- ESPnet repository
- ESPnet documentation
- ESPnet-EZ repo

Important Notes

- Please submit PDF files of your completed notebooks to Canvas. You can print the notebook using `File -> Print` in the menu bar.

Acknowledgement

- This homework is adapted from the ESPnet online demos and tutorials.

Install ESPnet

- The temporary version we used for Assignment 1 just got merged to ESPNET. We are now installing espnet as should be. You may see some dependency errors. It should be safe for you to ignore them for now.

```
!git clone https://github.com/espnet/espnet.git
!cd espnet && pip install .
```

```
fatal: destination path 'espnet' already exists and is not an empty directory.
Processing /content/espnet
  Preparing metadata (setup.py) ... done
Requirement already satisfied: setuptools<74.0.0,>=38.5.1 in /usr/local/lib/pyth
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: configargparse>=1.2.1 in /usr/local/lib/python3.1
Requirement already satisfied: typeguard in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: humanfriendly in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: librosa>=0.10.2 in /usr/local/lib/python3.12/dist
Requirement already satisfied: jamo==0.4.1 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: PyYAML>=5.1.2 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: soundfile>=0.10.2 in /usr/local/lib/python3.12/di
Requirement already satisfied: h5py>=2.10.0 in /usr/local/lib/python3.12/dist-pa
```

```
Requirement already satisfied: kaldiio>=2.18.0 in /usr/local/lib/python3.12/dist
Requirement already satisfied: torch>=1.11.0 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: torch_complex in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: nltk>=3.4.5 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: numpy>=2.0.0 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: protobuf in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: hydra-core in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: lightning in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: sentencepiece==0.2.0 in /usr/local/lib/python3.12
Requirement already satisfied: pyworld>=0.3.4 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: pypinyin<=0.44.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: espnet_tts_frontend in /usr/local/lib/python3.12/
Requirement already satisfied: ci_sdr in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: fast-bss-eval==0.1.3 in /usr/local/lib/python3.12
Requirement already satisfied: asteroid_filterbanks==0.4.0 in /usr/local/lib/pyt
Requirement already satisfied: editdistance in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: importlib-metadata<5.0 in /usr/local/lib/python3.
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.12/di
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: audioread>=2.1.9 in /usr/local/lib/python3.12/dis
Requirement already satisfied: numba>=0.51.0 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: scikit-learn>=1.1.0 in /usr/local/lib/python3.12/
Requirement already satisfied: joblib>=1.0 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: pooch>=1.1 in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: soxr>=0.3.2 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: lazy_loader>=0.1 in /usr/local/lib/python3.12/dis
Requirement already satisfied: msgpack>=1.0 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (
Requirement already satisfied: cffi>=1.0 in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: fsspec in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/l
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/py
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/py
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/pyt
```

```
1 !pip install espnet-model-zoo # for downloading pre-trained models
2 !apt install ffmpeg # for audio file processing
3 !pip install ipywebrtc notebook # for real-time recording
4 !pip install datasets==3.6.0 # for downloading ASR datasets
```

```
Requirement already satisfied: espnet-model-zoo in /usr/local/lib/python3.12/dis
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (
Requirement already satisfied: numpy in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: espnet in /usr/local/lib/python3.12/dist-packages
```

```
Requirement already satisfied: espnet in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: huggingface-hub in /usr/local/lib/python3.12/dist
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: setuptools<74.0.0,>=38.5.1 in /usr/local/lib/pyth
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: configargparse>=1.2.1 in /usr/local/lib/python3.1
Requirement already satisfied: typeguard in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: humanfriendly in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: librosa>=0.10.2 in /usr/local/lib/python3.12/dist
Requirement already satisfied: jamo==0.4.1 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: PyYAML>=5.1.2 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: soundfile>=0.10.2 in /usr/local/lib/python3.12/di
Requirement already satisfied: h5py>=2.10.0 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: kaldiio>=2.18.0 in /usr/local/lib/python3.12/dist
Requirement already satisfied: torch>=1.11.0 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: torch-complex in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: nltk>=3.4.5 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: protobuf in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: hydra-core in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: lightning in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: sentencepiece==0.2.0 in /usr/local/lib/python3.12
Requirement already satisfied: pyworld>=0.3.4 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: pypinyin<=0.44.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: espnet-tts-frontend in /usr/local/lib/python3.12/
Requirement already satisfied: ci-sdr in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: fast-bss-eval==0.1.3 in /usr/local/lib/python3.12
Requirement already satisfied: asteroid-filterbanks==0.4.0 in /usr/local/lib/pyt
Requirement already satisfied: editdistance in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: importlib-metadata<5.0 in /usr/local/lib/python3.
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.12/di
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/d
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/d
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.12/dist-packa
Requirement already satisfied: audioread>=2.1.9 in /usr/local/lib/python3.12/dis
Requirement already satisfied: numba>=0.51.0 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: scikit-learn>=1.1.0 in /usr/local/lib/python3.12/
Requirement already satisfied: joblib>=1.0 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python3.12/dis
Requirement already satisfied: pooch>=1.1 in /usr/local/lib/python3.12/dist-pack
Requirement already satisfied: soxr>=0.3.2 in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: lazy_loader>=0.1 in /usr/local/lib/python3.12/dis
Requirement already satisfied: msgpack>=1.0 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.12/dist
```

Import the dependencies and check the state of installation

```
import torch
import datasets
import espnetez as ez # ESPnet wrapper that simplifies integration. If you
import numpy as np
import librosa
from espnet2.bin.s2t_inference import Speech2Text # Core ESPnet module for

print("Installation success!")
```

```
Installation success!
```

Data Processing

For this tutorial, we will use the FLEURS dataset from HuggingFace: https://huggingface.co/datasets/google/fleurs .

FLEURS is a 102-language multilingual speech dataset, supporting tasks such as Automatic Speech Recognition (ASR), Speech Translation (ST), and Language Identification (LID).

While the total size of FLEURS is relatively large at ~1000 hours of training data, each individual language only has 7-10 hours of audio.

For this tutorial, we will focus on monolingual ASR for one of the 102 languages.

Data Downloading

We will first download the data for one language of FLEURS. FLEURS organizes the languages by its ISO2 language code and locale. For example, American English is `en_us`.

We will use English for the first fine-tuning experiment. You will have the opportunity to try a different language later on in the assignment.

If you want to download the data for another language, you can map the language name to the ISO2 code using Table 9 in the FLEURS paper: https://arxiv.org/pdf/2205.12446. Then, you can use that to identify the language+region combination using the HuggingFace data previewer: https://huggingface.co/datasets/google/fleurs .

(Please select y for the prompt of running custom code to download the data)

```
!mkdir downloads
%cd downloads
```

```
%cd downloads
!pip install --upgrade --no-cache-dir gdown
!gdown 1pzBIeUL1H0z-lLaBFyyKQFBbJhDJXRnM
!tar -xzf TIDIGITS_children_boy.tar.gz && ls TIDIGITS_children_boy
```

```
mkdir: cannot create directory ‘downloads’: File exists
/content/downloads
Requirement already satisfied: gdown in /usr/local/lib/python3.12/dist-packages
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.12/dist-
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packag
Requirement already satisfied: requests[socks] in /usr/local/lib/python3.12/dist
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-packages (
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: typing-extensions>=4.0.0 in /usr/local/lib/python
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/d
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/d
Requirement already satisfied: PySocks!=1.5.7,>=1.5.6 in /usr/local/lib/python3.
Downloading...
From (original): https://drive.google.com/uc?id=1pzBIeUL1H0z-lLaBFyyKQFBbJhDJXRn
From (redirected): https://drive.google.com/uc?id=1pzBIeUL1H0z-lLaBFyyKQFBbJhDJX
To: /content/downloads/TIDIGITS_children_boy.tar.gz
100% 127M/127M [00:00<00:00, 142MB/s]
data  readme.1st
```

Inspect the data

```
%cd /content/
!mkdir TIDIGITS_subset2
!mkdir TIDIGITS_subset2/train
!mkdir TIDIGITS_subset2/validation
!mkdir TIDIGITS_subset2/test
```

```
/content
mkdir: cannot create directory ‘TIDIGITS_subset2’: File exists
mkdir: cannot create directory ‘TIDIGITS_subset2/train’: File exists
mkdir: cannot create directory ‘TIDIGITS_subset2/validation’: File exists
mkdir: cannot create directory ‘TIDIGITS_subset2/test’: File exists
```

```
!gdown 17TlHt41TFZPYNQJ3TuMdw8TyW5gufvtC
!python data_prep.py /content/downloads/TIDIGITS_children_boy
!sh move_files.sh
```

```
Streaming output truncated to the last 5000 lines.
    utterance_production: a
    recording_date  : 20-AUG-1982
    sample_checksum : 2767
  Duration: 00:00:01.08, bitrate: 112 kb/s
  Stream #0:0: Audio: shorten, 20000 Hz, mono, s16p
File 'TIDIGITS_subset2/train/ds-5a.wav' already exists. Overwrite? [y/N]
```

```
Not overwriting - exiting
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg develop
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.22.04.1 --toolchain=hard
  libavutil      56. 70.100 / 56. 70.100
  libavcodec     58.134.100 / 58.134.100
  libavformat    58. 76.100 / 58. 76.100
  libavdevice    58. 13.100 / 58. 13.100
  libavfilter     7.110.100 /  7.110.100
  libswscale      5.  9.100 /  5.  9.100
  libswresample   3.  9.100 /  3.  9.100
  libpostproc    55.  9.100 / 55.  9.100
Guessed Channel Layout for Input Stream #0.0 : mono
Input #0, nistsphere, from '/content/downloads/TIDIGITS_children_boy/data/childr
  Metadata:
    database_id     : TIDIGITS
    database_version: 1.0
    utterance_id    : ds_z7567_a
    sample_min      : -2566
    sample_max      : 1932
    speaker_id      : ds
    prompt_code     : z7567
    utterance_production: a
    recording_date  : 20-AUG-1982
    sample_checksum : 46270
  Duration: 00:00:02.36, bitrate: 144 kb/s
  Stream #0:0: Audio: shorten, 20000 Hz, mono, s16p
File 'TIDIGITS_subset2/train/ds-z7567a.wav' already exists. Overwrite? [y/N]
Not overwriting - exiting
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg develop
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.22.04.1 --toolchain=hard
  libavutil      56. 70.100 / 56. 70.100
  libavcodec     58.134.100 / 58.134.100
  libavformat    58. 76.100 / 58. 76.100
  libavdevice    58. 13.100 / 58. 13.100
  libavfilter     7.110.100 /  7.110.100
  libswscale      5.  9.100 /  5.  9.100
  libswresample   3.  9.100 /  3.  9.100
  libpostproc    55.  9.100 / 55.  9.100
Guessed Channel Layout for Input Stream #0.0 : mono
Input #0, nistsphere, from '/content/downloads/TIDIGITS_children_boy/data/childr
  Metadata:
    database_id     : TIDIGITS
    database_version: 1.0
    utterance_id    : ds_242_a
    sample_min      : -2464
    sample_max      : 2444
    speaker_id      : ds
    prompt_code     : 242
    utterance_production: a
```

```
1 # Datasets library
2 from datasets import load_dataset, Audio
3 train_dataset = load_dataset("audiofolder", data_dir=f"/content/TIDIGITS_su
```

```
4 valid_dataset = load_dataset("audiofolder", data_dir=f"/content/TIDIGITS_su
5 test_dataset = load_dataset("audiofolder", data_dir=f"/content/TIDIGITS_sub
```

```
Resolving data files: 100% 1326/1326 [00:00<00:00, 14422.54it/s]
Resolving data files: 100% 101/101 [00:00<00:00, 8639.41it/s]
Resolving data files: 100% 201/201 [00:00<00:00, 8967.05it/s]
Downloading data: 100% 1326/1326 [00:00<00:00, 17898.07files/s]
Downloading data: 100% 101/101 [00:00<00:00, 9013.48files/s]
Downloading data: 100% 201/201 [00:00<00:00, 7699.63files/s]
Generating train split: 1325/0 [00:00<00:00, 17633.25 examples/s]
Generating validation split: 100/0 [00:00<00:00, 4095.04 examples/s]
Generating test split: 200/0 [00:00<00:00, 6827.61 examples/s]
Resolving data files: 100% 1326/1326 [00:00<00:00, 13905.75it/
```

```
1
2 train_dataset = train_dataset.cast_column("audio", Audio(sampling_rate=1600
3 valid_dataset = valid_dataset.cast_column("audio", Audio(sampling_rate=1600
4 test_dataset = test_dataset.cast_column("audio", Audio(sampling_rate=16000)
```

Pretrained Model

In low-resource settings, training a model from scratch is unlikely to lead to good results. So instead, we will fine-tune a pre-trained foundation model.

We will use the base version of OWSM 3.1, an open-source speech foundation model trained on 180K hours of multilingual ASR and ST.

Downloading

Since it needs to support many language varieties, OWSM uses ISO3 for the language

IDs. The ISO3 code for your language of choice can also be found in Table 9 in the FLEURS paper: https://arxiv.org/pdf/2205.12446

```
FINETUNE_MODEL="espnet/owsm_v3.1_ebf_base"
owsm_language="eng" # language code in ISO3
```

```
pretrained_model = Speech2Text.from_pretrained(
    FINETUNE_MODEL,
    lang_sym=f"<{owsm_language}>",
    beam_size=1,
    device='cuda'
)
torch.save(pretrained_model.s2t_model.state_dict(), 'original.pth')
pretrain_config = vars(pretrained_model.s2t_train_args)
tokenizer = pretrained_model.tokenizer
converter = pretrained_model.converter
```

```
Fetching 29 files: 100%  29/29 [00:00<00:00, 1763.62it/
```

Setup Training

We first need to convert the HuggingFace data into a format that ESPnet can read. This can be easily done by defining a `data_info` dictionary that maps each field required for OWSM fine-tuning to a column in our dataset.

```
'''
pretrained_model -> the pre-trained model we downloaded earlier
tokenizer -> Tokenizes raw text into subwords
converter -> Converts subwords into integer IDs for model input
'''

def tokenize(text):
    return np.array(converter.tokens2ids(tokenizer.text2tokens(text)))
data_info = {
    "speech": lambda d: d['audio']['array'].astype(np.float32), # 1-D raw w
    "text": lambda d: tokenize(f"<{owsm_language}><asr><notimestamps> {d['t
    "text_prev": lambda d: tokenize("<na>"), # tokenized text of previous u
    "text_ctc": lambda d: tokenize(d['transcription']), # tokenized text ma
}
test_data_info = {
    "speech": lambda d: d['audio']['array'].astype(np.float32),
    "text": lambda d: tokenize(f"<{owsm_language}><asr><notimestamps> {d['t
    "text_prev": lambda d: tokenize("<na>"),
    "text_ctc": lambda d: tokenize(d['transcription']),
    "text_raw": lambda d: d['transcription'], # raw untokenized text as the
```

```
}
train_dataset = ez.dataset.ESPnetEZDataset(train_dataset, data_info=data_ir
valid_dataset = ez.dataset.ESPnetEZDataset(valid_dataset, data_info=data_ir
test_dataset = ez.dataset.ESPnetEZDataset(test_dataset, data_info=test_data
```

Next we need to define a function that will pass our pre-trained model to ESPnet. This function here doesn't do much since our setup is simple, but its required for more complex settings.

```
# define model loading function
def count_parameters(model):
    return sum(p.numel() for p in model.parameters() if p.requires_grad)

def build_model_fn(args):
  model = pretrained_model.s2t_model
  model.train()
  print(f'Trainable parameters: {count_parameters(model)}')
  return model
```

Training

Training requires tuning many hyper-parameters. Here is an initial config to help start you off.

```
!gdown 1RuOXmN9nyhLSbRZVGHcxlQjOQ-dygtUu
!mkdir config
!mv finetune.yaml config/finetune.yaml
```

```
Downloading...
From: https://drive.google.com/uc?id=1RuOXmN9nyhLSbRZVGHcxlQjOQ-dygtUu
To: /content/finetune.yaml
100% 987/987 [00:00<00:00, 5.07MB/s]
mkdir: cannot create directory ‘config’: File exists
```

Before we begin training, we need to define where our model files and logs will be saved. We also need to override some of the settings used to pre-train the foundation model with our own settings.

```
EXP_DIR = f"./exp/finetune"
STATS_DIR = f"./exp/stats_finetune"
finetune_config = ez.config.update_finetune_config(
    's2t',
    pretrain_config,
    f"./config/finetune.yaml"
```

```
)

# You can edit your config by changing the finetune.yaml file directly (but
# You can also change it programatically like this
finetune_config['max_epoch'] = 1
finetune_config['num_iters_per_epoch'] = 500
```

Finally, we just need to pass our model, data, and configs to a trainer.

```
trainer = ez.Trainer(
    task='s2t',
    train_config=finetune_config,
    train_dataset=train_dataset,
    valid_dataset=valid_dataset,
    build_model_fn=build_model_fn, # provide the pre-trained model
    data_info=data_info,
    output_dir=EXP_DIR,
    stats_dir=STATS_DIR,
    ngpu=1
)
```

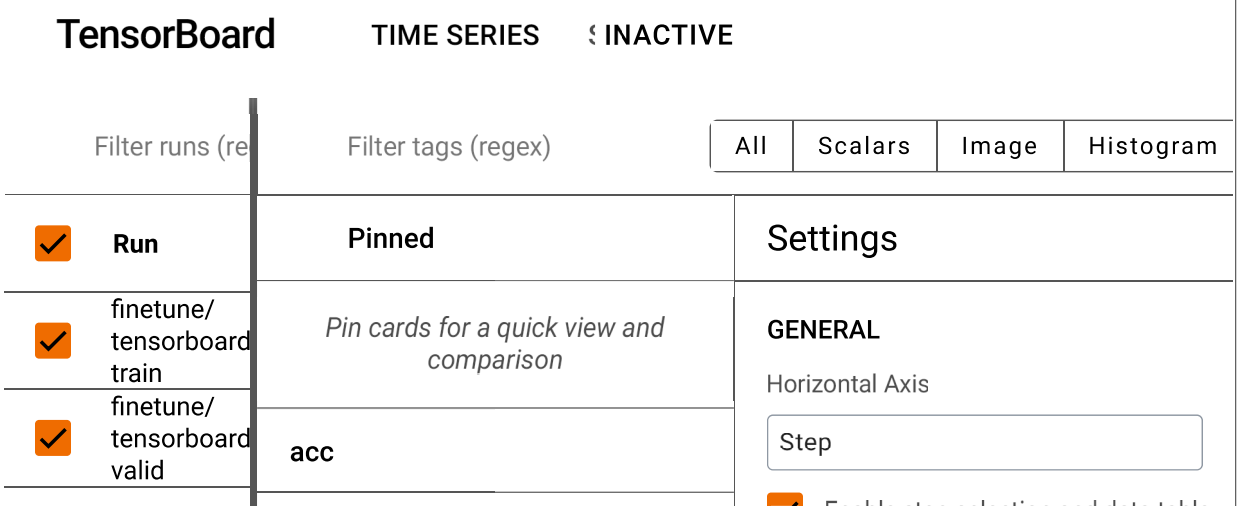
```
trainer.collect_stats() # collect audio/text length information to construc
```

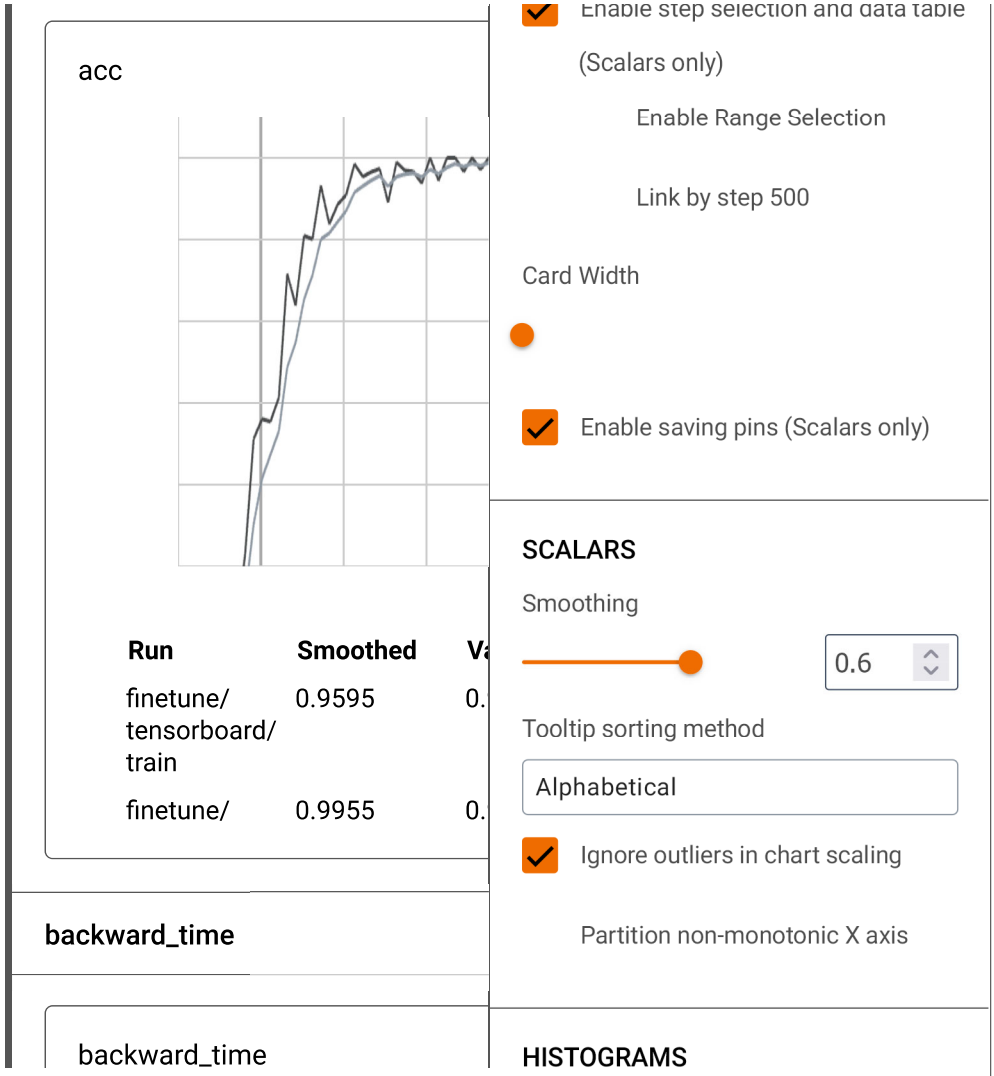
```
/usr/bin/python3 /usr/local/lib/python3.12/dist-packages/colab_kernel_launcher.p
Trainable parameters: 101182628
```

```
# Load the TensorBoard notebook extension
%load_ext tensorboard

# Launch tensorboard before training
%tensorboard --logdir /content/exp
```

```
The tensorboard extension is already loaded. To reload it, use:
  %reload_ext tensorboard
```






```
1 trainer.train() # every 100 steps takes ~1 min
```

```
/usr/bin/python3 /usr/local/lib/python3.12/dist-packages/colab_kernel_launcher.p
Trainable parameters: 101182628
WARNING:root:The training has already reached at max_epoch: 2
```

Inference

Here is a demo to perform inference using the original and fine-tuned model.

```
1 id, sample_test_utterance = test_dataset.__getitem__(0)
```

```
1 pretrained_model.s2t_model.cuda()
2 pretrained_model.device = 'cuda'
3
4 d = torch.load("original.pth")
5 pretrained_model.s2t_model.load_state_dict(d)
6 pred = pretrained_model(sample_test_utterance['speech'])
```

```
print('PREDICTED: ' + pred[0][0])
print('REFERENCE: ' + sample_test_utterance['text_raw'])
```

```
PREDICTED: <eng><asr><notimestamps> sx
REFERENCE: 6
/usr/local/lib/python3.12/dist-packages/espnet2/s2t/espnet_model.py:279: FutureW
  with autocast(False):
```

Inference with fine-tuned model

```
d = torch.load("./exp/finetune/1epoch.pth")
pretrained_model.s2t_model.load_state_dict(d)
pred = pretrained_model(sample_test_utterance['speech'])
print('PREDICTED: ' + pred[0][0])
print('REFERENCE: ' + sample_test_utterance['text_raw'])
```

```
PREDICTED: <eng><asr><notimestamps> 6
REFERENCE: 6
```

✅ Task 1

Now that you have performed inference with both the pre-trained model and your fine-tuned model, provide some qualitative analyses of the results. How does the output between the two models differ? Do you observe any stylistic differences in the transcriptions?

The difference between the pre-trained and fine-tuned models is pretty stark, demonstrating the importance of domain-specific adaptation. The pre-trained model, designed for general speech, transcribed the audio as "sx." This output is not a real word but rather a phonetic artifact, suggesting the model heard the acoustic signals of the digit "six" but failed to map them to its correct semantic meaning, instead producing a nonsensical character sequence. In contrast, the fine-tuned model achieved a perfect transcription of "6." This dramatic improvement shows that the fine-tuning process successfully specialized the model for the TIDIGITS domain, teaching it to correctly interpret the acoustic patterns of children's digit pronunciation. Stylistically, both models used the same output format, but the fine-tuned model replaced phonetic guessing with accurate digit recognition.

```
Start coding or generate with AI.
```

✅ Task 2

Following the inference sample, perform inference on the whole test dataset and report both WER and CER using jiwer.

WER Calculation

```
!pip install jiwer
```

```
Collecting jiwer
  Downloading jiwer-4.0.0-py3-none-any.whl.metadata (3.3 kB)
Requirement already satisfied: click>=8.1.8 in /usr/local/lib/python3.12/dist-pa
Collecting rapidfuzz>=3.9.7 (from jiwer)
  Downloading rapidfuzz-3.14.1-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_
Downloading jiwer-4.0.0-py3-none-any.whl (23 kB)
Downloading rapidfuzz-3.14.1-cp312-cp312-manylinux_2_27_x86_64.manylinux_2_28_x8
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.2/3.2 MB 50.4 MB/s eta 0:00:00
Installing collected packages: rapidfuzz, jiwer
Successfully installed jiwer-4.0.0 rapidfuzz-3.14.1
```

Inference with the original model

```
hyps = []
refs = []
d = torch.load("original.pth") # Default to ckpt before fine-tuning.
pretrained_model.s2t_model.load_state_dict(d)

# make sure we use GPU
pretrained_model.s2t_model.cuda()
pretrained_model.device = 'cuda'

## My implementation
# Iterate through entire test dataset
for i in range(len(test_dataset)):
    # Get test sample
    id, sample = test_dataset.__getitem__(i)

    # Run inference
    pred = pretrained_model(sample['speech'])

    # Get the prediction text (remove special tokens for scoring)
    prediction_text = pred[0][0].replace('<eng><asr><notimestamps>', '').str
    reference_text = sample['text_raw']

    # Store for scoring
```

```
    hyps.append(prediction_text)
    refs.append(reference_text)
```

```
import jiwer


## Start your implementation here
# Compute WER and CER for test datasets
wer_score = jiwer.wer(refs, hyps)
cer_score = jiwer.cer(refs, hyps)

print(f"Pre-trained Model - Average WER: {wer_score:.2%}")
print(f"Pre-trained Model - Average CER: {cer_score:.2%}")

# Print a few examples to see what's happening
print("\nFirst 5 examples:")
for i in range(min(5, len(hyps))):
    print(f"Ref: '{refs[i]}' | Hyp: '{hyps[i]}'")
```

```
Pre-trained Model - Average WER: 95.92%
Pre-trained Model - Average CER: 198.14%

First 5 examples:
Ref: '6' | Hyp: 'six'
Ref: '5' | Hyp: 'fari'
Ref: '4 2' | Hyp: 'four'
Ref: '3 4 6 8 1' | Hyp: 'three four six eight one'
Ref: '6 3 6 6 7' | Hyp: 'six three six six seven'
```

Inference with the fine-tuned model

```
hyps = []
refs = []
d = torch.load("./exp/finetune/1epoch.pth") # Default to ckpt before fine-t
pretrained_model.s2t_model.load_state_dict(d)

# make sure we use GPU
pretrained_model.s2t_model.cuda()
pretrained_model.device = 'cuda'

## My implementation
# Iterate through entire test dataset
for i in range(len(test_dataset)):
    # Get the test sample
    id, sample = test_dataset.__getitem__(i)

    # Run inference
    pred = pretrained_model(sample['speech'])

```

```
    # Get the prediction text (remove the special tokens for scoring)
    prediction_text = pred[0][0].replace('<eng><asr><notimestamps>', '').st
    reference_text = sample['text_raw']

    # Store for scoring
    hyps.append(prediction_text)
    refs.append(reference_text)
```

```
import jiwer

## My implementation here
# Compute WER and CER for the test datasets
wer_score = jiwer.wer(refs, hyps)
cer_score = jiwer.cer(refs, hyps)

print(f"Fine-tuned Model - Average WER: {wer_score:.2%}")
print(f"Fine-tuned Model - Average CER: {cer_score:.2%}")

# Optional: Print a few examples to see what's happening
print("\nFirst 5 examples:")
for i in range(min(5, len(hyps))):
    print(f"Ref: '{refs[i]}' | Hyp: '{hyps[i]}'")
```

```
Fine-tuned Model - Average WER: 8.78%
Fine-tuned Model - Average CER: 9.48%

First 5 examples:
Ref: '6' | Hyp: '3'
Ref: '5' | Hyp: '4'
Ref: '4 2' | Hyp: '4'
Ref: '3 4 6 8 1' | Hyp: '3 4 6 8 1'
Ref: '6 3 6 6 7' | Hyp: '6 3 6 6 7'
```

```
Start coding or generate with AI.
```