DLHLP 2021 Fall
HW1 E2E ASR

2021.10.03

TA: 張恆瑞

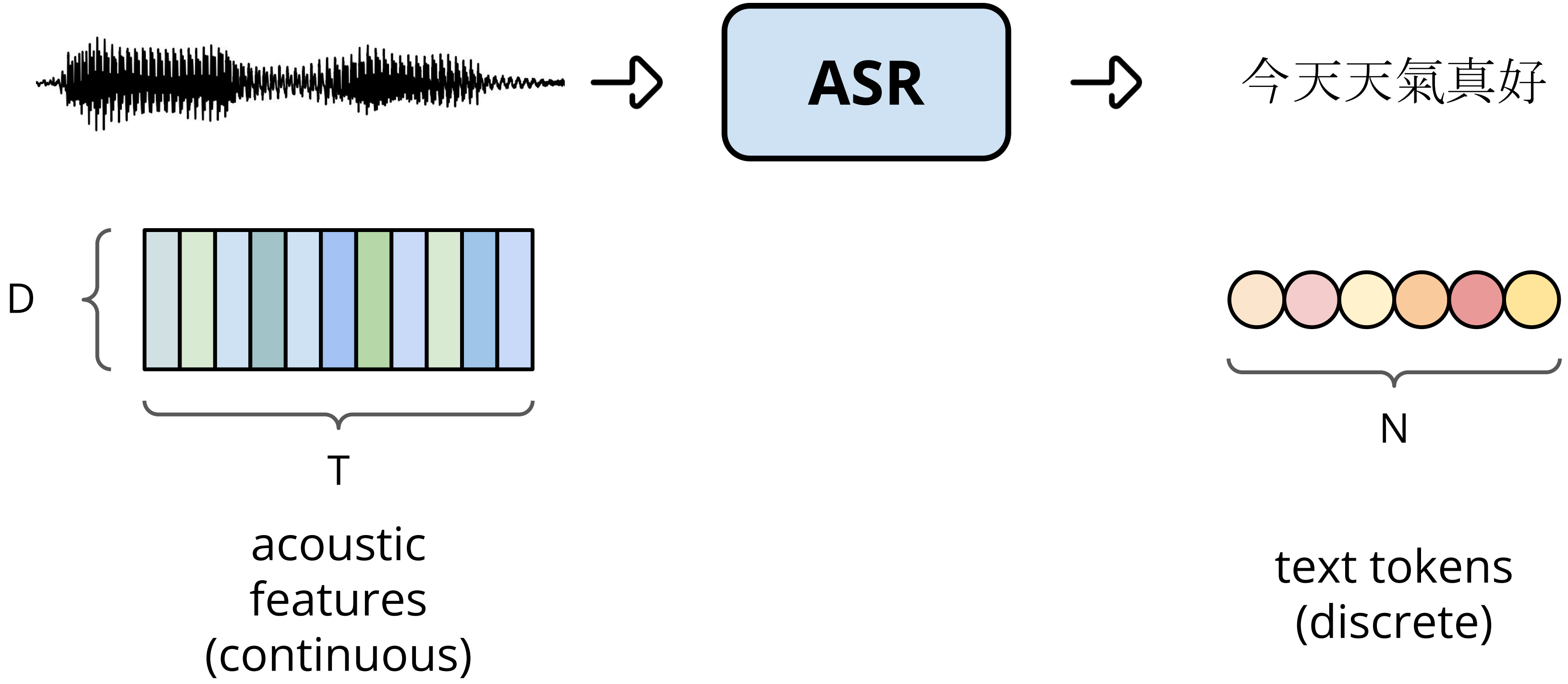
Outline

- Intro to E2E ASR (CTC)
- MiniASR Toolkit
  - Preprocess
  - Modify Config
  - Training
  - Testing (evaluation metric)
- Hints for improving performance
- Reference

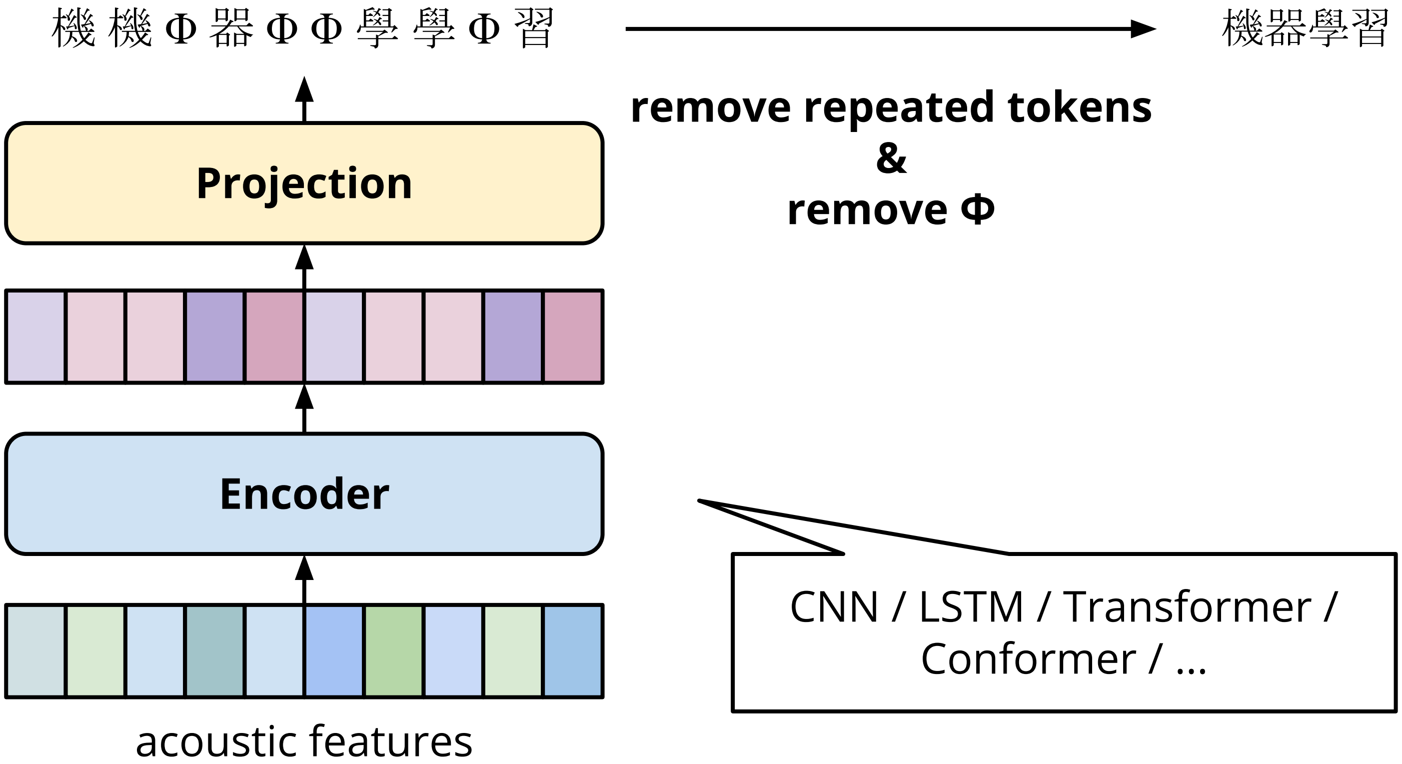
Preliminaries

- Strongly recommended:
  - Speech Recognition 1
  - Speech Recognition 2
  - Speech Recognition 3
  - Speech Recognition 4
  - Speech Recognition 5
  - Speech Recognition 6
- Optional:
  - Language Modeling

End-to-end Automatic Speech Recognition



Connectionist Temporal Classification (CTC)



Connectionist Temporal Classification (CTC)

Target: Maximize P(機器學習 l X)

P(機器學習 l X) = P(機機Φ器ΦΦ學學Φ習 l X)

+ P(Φ機Φ器ΦΦΦ學習Φ l X)

+ P(機ΦΦ器器器學Φ習習 l X)

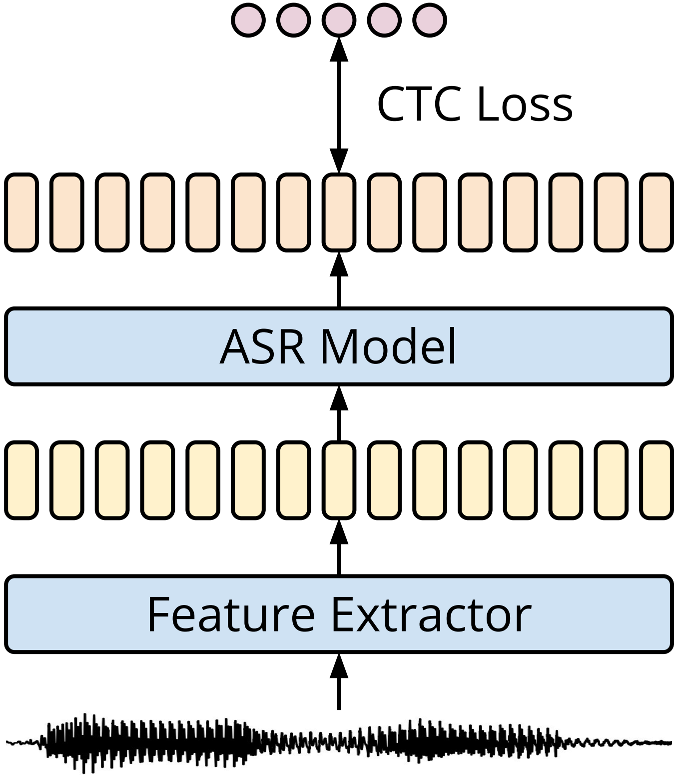
+ P(ΦΦ機Φ器學Φ習ΦΦ l X)

+ …

Too many possibilities!

Dynamic Programming

End-to-end Automatic Speech Recognition



text token sequence: L x V ($L \leq T'$)

probability distributions over all possible vocabularies: T' x V ($T' \leq T$)

acoustic features: T x D (e.g. $T = T_{raw}$ / 160)

raw waveform: T_{raw} x 1 (sample rate = 16kHz)

Toolkit

- https://github.com/vectominist/MiniASR
- A mini, simple, and fast E2E ASR toolkit. (still in development)
- Core: PyTorch, PyTorch Lightning, S3PRL
- https://github.com/vectominist/MiniASR/blob/main/example/example_librispeech_training.ipynb

Installation

```
git clone https://github.com/vectominist/MiniASR.git

cd MiniASR

pip3 install -e ./
```

Available commands:

```
miniasr-asr
miniasr-preprocess
```

Brief Intro to MiniASR

```
.
├── LICENSE
├── README.md
├── egs
├── example
├── hubconf.py
├── logo.png
├── miniasr
├── run_asr.py
├── run_preprocess.py
├── script
├── setup.py
└── tools
```

Most functions are in here

ASR training / testing

Data preprocessing

```
── egs
   └── librispeech
       ├── README.md
       ├── config
       │   ├── ctc_test_100h.yaml
       │   ├── ctc_test_960.yaml
       │   ├── ctc_test_example.yaml
       │   ├── ctc_test_fbank.yaml
       │   ├── ctc_train_100h.yaml
       │   ├── ctc_train_960h.yaml
       │   ├── ctc_train_example.yaml
       │   └── ctc_train_fbank.yaml
       ├── path.sh
       ├── preprocess.sh
       ├── test.sh
       └── train.sh
```

Config files.

Scripts.

```
── miniasr
   ├── bin
   │   └── asr_trainer.py
   ├── data
   │   ├── README.md
   │   ├── audio.py
   │   ├── dataloader.py
   │   ├── dataset.py
   │   └── text.py
   ├── model
   │   ├── base_asr.py
   │   └── ctc_asr.py
   ├── module
   │   ├── __init__.py
   │   ├── encoder_rnn.py
   │   ├── feat_selection.py
   │   ├── masking.py
   │   └── scheduler.py
   ├── preprocess
   │   ├── README.md
   │   ├── generate_vocab.py
   │   └── librispeech.py
   └── utils
       ├── __init__.py
       ├── basic_setups.py
       ├── eval_metrics.py
       └── model_utils.py
```

Setup PyTorch Lightning Trainer.

Data processing:
audio loading, data augmentation, text tokenizer, Dataset / Dataloader

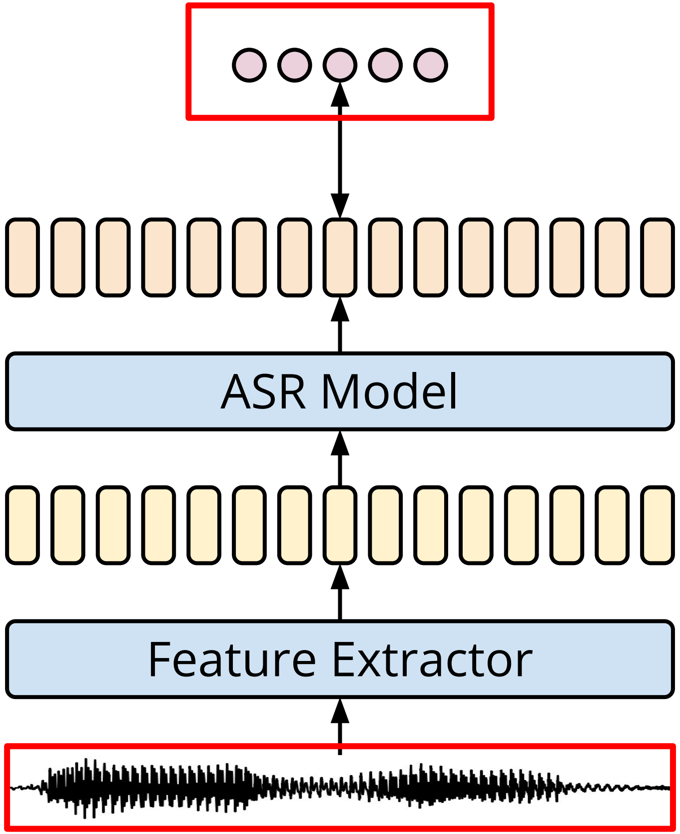
ASR model:
- base_asr.py: general ASR class
- ctc_asr.py: CTC ASR class

Modules:
RNN encoder, feature selection, ...

Preprocess:
Process LibriSpeech dataset & vocabulary

Utilities:
Setup, evaluation metrics, loading, ...

Step 1: Data Preprocessing



- **Training set: Libri-light fine-tuning set**
  - https://github.com/facebookresearch/libri-light
  - 10 hours
  - English audio books
- **Development set: LibriSpeech dev-clean**
  - https://www.openslr.org/12
- **Testing set: LibriSpeech test-clean**
  - https://www.openslr.org/12

Step 1: Data Preprocessing

```
miniasr-preprocess
  --corpus Corpus name.
  --path Path to dataset.
  --set Which subsets to be processed.
  --out Output directory.
  --gen-vocab Specify whether to generate
              vocabulary files.
  --char-vocab-size Character vocabulary size.
```

Step 1: Data Preprocessing

data/libri_train_1h/data_list_sorted.json

```
├── data_dict.json
├── data_list_sorted.json
├── text.txt
└── vocab_char.txt
```


```
[
    {
        "text": "BRING ME ROUND TO THE BOWER SAID SILAS WHEN THE BARGAIN WAS CLOSED NEXT SATURDAY EVENING AND IF A SOCIABLE GLASS OF OLD JAMAIKEY WARM SHOULD MEET YOUR VIEWS I AM NOT THE MAN TO BEGRUDGE IT YOU ARE AWARE OF MY BEING POOR COMPANY SIR REPLIED MISTER VENUS BUT BE IT SO",
        "file": "/work/harry87122/dataset/librispeech_finetuning/1h/4/other/978/125137/978-125137-0011.flac"
    },
    {
        "text": "AN INDISPENSABLE AND AGREEABLE GUARANTEE OF WOMANLY POWER BUT TO BECOME A WIFE AND WEAR ALL THE DOMESTIC FETTERS OF THAT CONDITION WAS ON THE WHOLE A VEXATIOUS NECESSITY HER OBSERVATION OF MATRIMONY HAD INCLINED HER TO THINK IT RATHER A DREARY STATE",
        "file": "/work/harry87122/dataset/librispeech_finetuning/1h/4/clean/248/130644/248-130644-0008.flac"
    },
    {
        "text": "SHE COULD NOT LOOK FORWARD TO A SINGLE LIFE BUT PROMOTIONS HAVE SOMETIMES TO BE TAKEN WITH BITTER HERBS A PEERAGE WILL NOT QUITE DO INSTEAD OF LEADERSHIP TO THE MAN WHO MEANT TO LEAD AND THIS DELICATE LIMBED SYLPH OF TWENTY MEANT TO LEAD",
        "file": "/work/harry87122/dataset/librispeech_finetuning/1h/4/clean/248/130644/248-130644-0010.flac"
    },
```

Step 2-1: Modify Config Files

- egs/librispeech/config/ctc_train_example.yaml

```yaml
mode: train
data:
  train_paths: ['data/libri_train_1h/data_list_sorted.json',
                'data/libri_train_9h/data_list_sorted.json']
  dev_paths: ['data/libri_dev/data_list_sorted.json']
  text:
    mode: character
    vocab: data/libri_train_1h/vocab_char.txt
```

Do not modify this part unless you want to use other training data.

SpecAugment: https://arxiv.org/abs/1904.08779

Step 2-1: Modify Config Files

- egs/librispeech/config/ctc_train_example.yaml

```yaml
model:
  name: ctc_asr
  extractor:
    name: fbank
    train: false
    feature: hidden_states
  encoder:
    hid_dim: 256
    n_layers: 3
    module: GRU
    bidirectional: true
    dropout: 0.2
```

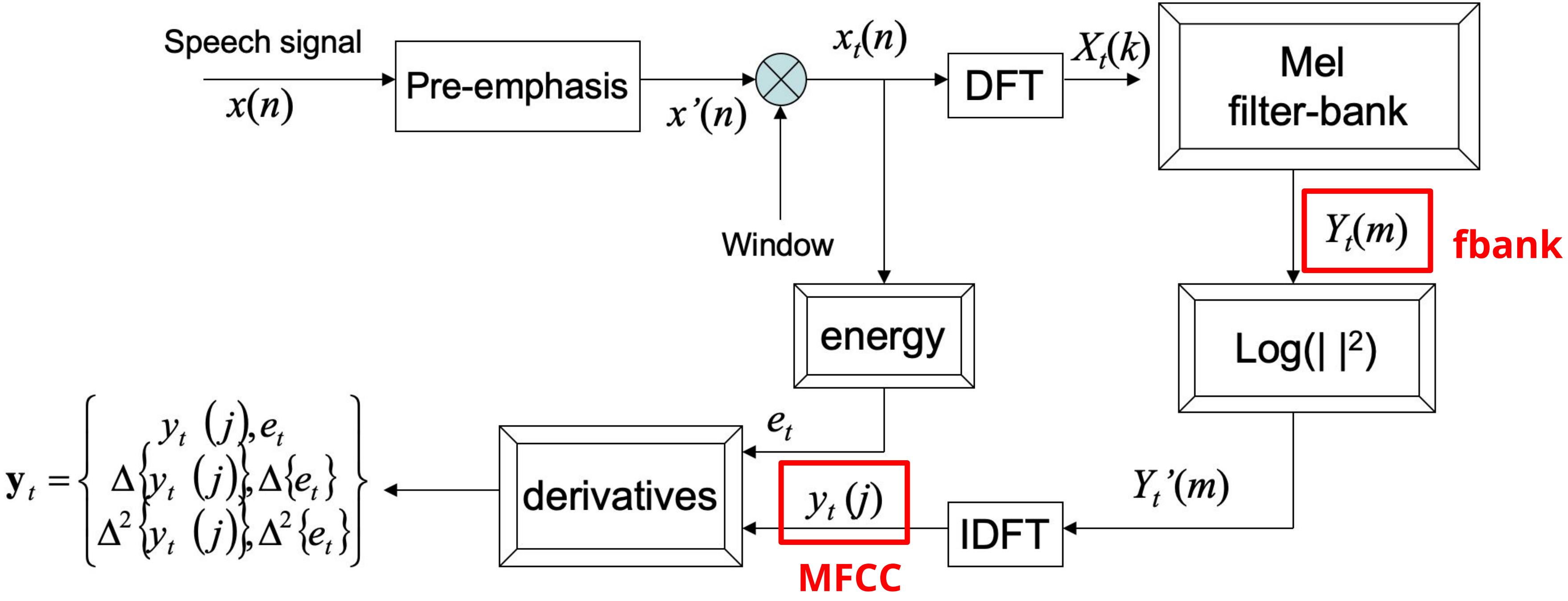
feature extractor

model architecture

```yaml
optim:
  algo: Adam
  kwargs:
    lr: 0.0001
specaugment:
  freq_mask_range: [0, 20]
  freq_mask_num: 2
  time_mask_range: [0, 40]
  time_mask_num: 2
  time_mask_max: 1.0
  time_warp_w: 80
```

comment this part to disable SpecAugment

Feature Extractors



source: Prof. Lin-shan Lee https://speech.ee.ntu.edu.tw/DSP2021Autumn/Pdfs/7.0.pdf

Feature Extractors

- Source: https://github.com/s3prl/s3prl

Feature	Name	Default Dim	Stride	Window	Backend
Spectrogram	spectrogram	257	10ms	25ms	torchaudio-kaldi
FBANK	fbank	80 + delta1 + delta2	10ms	25ms	torchaudio-kaldi
MFCC	mfcc	13 + delta1 + delta2	10ms	25ms	torchaudio-kaldi
Mel	mel	80	10ms	25ms	torchaudio
Linear	linear	201	10ms	25ms	torchaudio

Step 2-1: Modify Config Files

- egs/librispeech/config/ctc_train_example.yaml

PyTorch Lightning Trainer: https://pytorch-lightning.readthedocs.io/en/latest/common/trainer.html#trainer-class-api

```yaml
hparam:
  train_batch_size: 32
  val_batch_size: 32
  accum_grad: 1
  grad_clip: 5
  njobs: 4
  pin_memory: true
```

```yaml
trainer:
  max_epochs: 500
  max_steps: 100000
  check_val_every_n_epoch: 5
  gpus: 1
  precision: 16
  logger: true
  log_every_n_steps: 5
  flush_logs_every_n_steps: 5
  default_root_dir: model/ctc_libri-10h_char
  deterministic: true
```

model name

Step 2-2: Modify Config ~~Files~~ via Command Line

```
--override “ \
        args.model.encoder.n_layers=2,, \
        args.model.extractor.name=’mfcc’,, \
        args.trainer.default_root_dir=’model/my_ctc_model’ \
    ”
```

Step 3: Training

```
minasr-asr --config egs/librispeech/config/ctc_train_example.yaml
```

or

```
python3 run_asr.py \
    --config egs/librispeech/config/ctc_train_example.yaml
```

- The results will be saved in `model/ctc_libri-10h_char/`

Step 3: Training (Resuming from a Checkpoint)

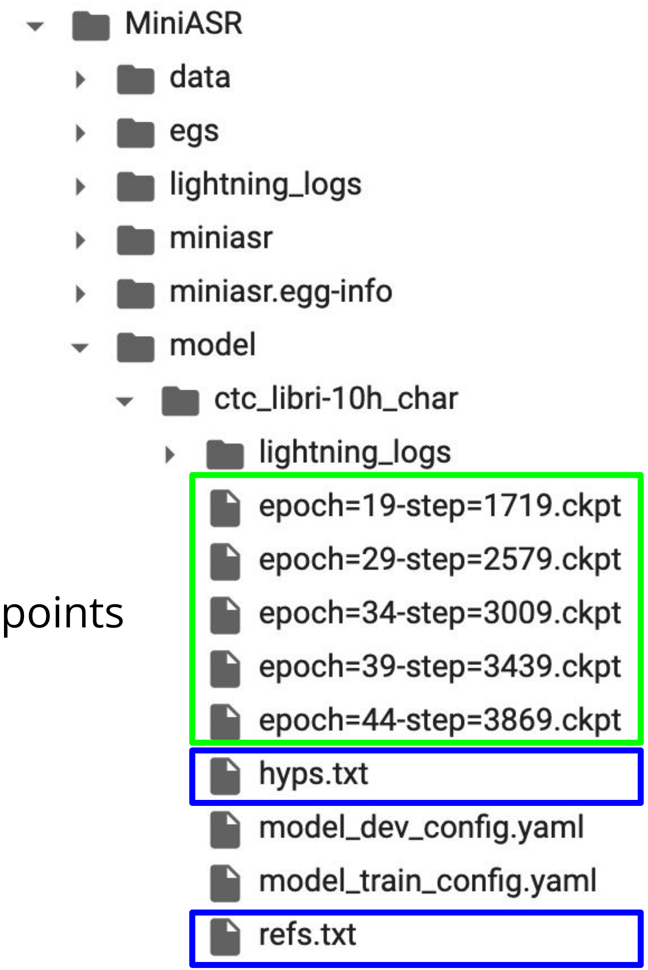
```
minasr-asr --ckpt model/ctc_libri-10h_char/???.ckpt
```

Step 4: Testing

```
minasr-asr \
    --config egs/librispeech/config/ctc_test_example.yaml \
    --test \
    --ckpt model/ctc_libri-10h_char/???.ckpt
```

- The results will be saved in `model/ctc_libri-10h_char/`

Step 4: Testing



Model Checkpoints

Hypothesis

Reference

Character Error Rate (CER)

- Reference

```
I like math until they  added the alphabet in it.
```

- Hypothesis

```
I like mat  until they addded the alphabat in it.
```

deletion insertion substitution

Word Error Rate (WER)

- Reference

  `I like math until they   added the alphabet in it.`

- Hypothesis

  `I like      until they a added the alphapad in it.`

  deletion insertion substitution

Step 4: Testing

substitution
deletion
insertion
character error rate (report this)

Character errors

#Snt	#Tok	Sub	Del	Ins	Err	SErr
2620	281530	14.9	11.8	2.7	29.5	100.0

Word errors

#Snt	#Tok	Sub	Del	Ins	Err	SErr
2620	52576	62.3	6.3	4.1	72.7	100.0

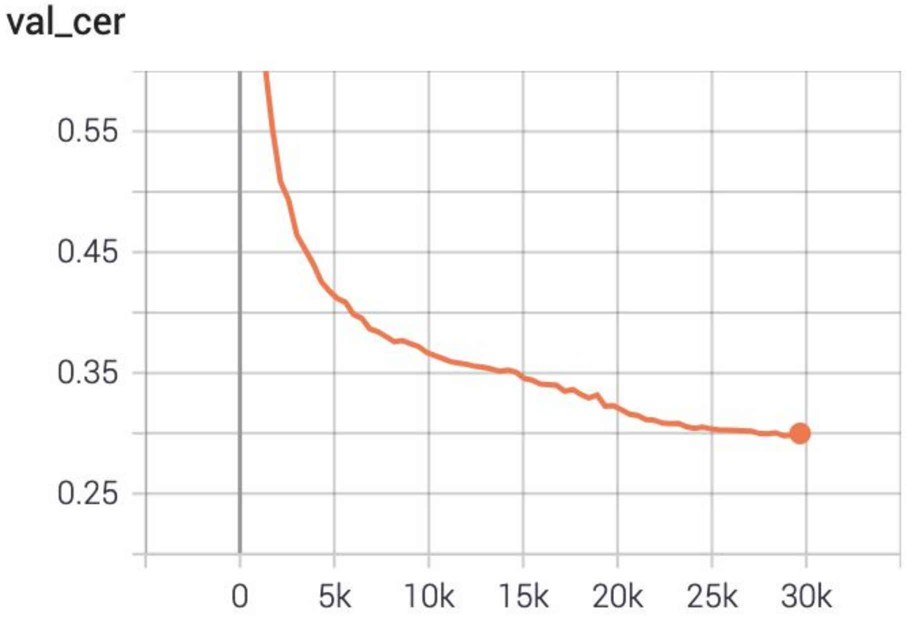
word error rate

Baseline -- FBANK (CER%)

Method	dev-clean	test-clean
FBANK	29.8	29.5

Learning Curve

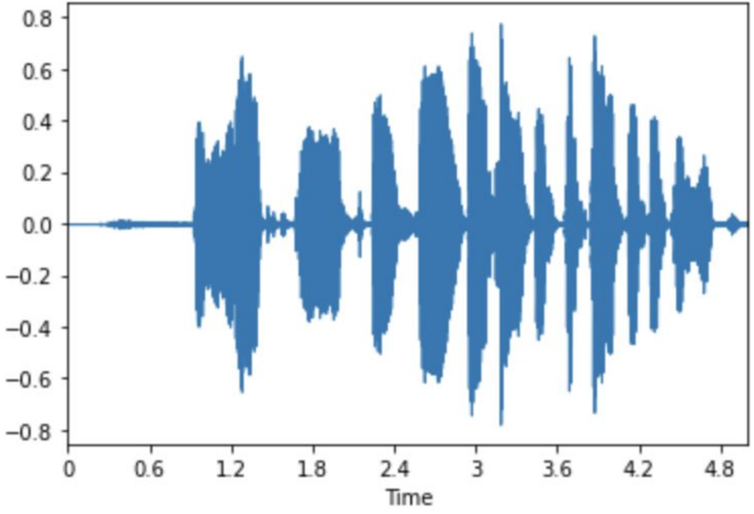
```
% tensorboard --logdir model
```



Play with Your Model

- https://github.com/vectominist/MiniASR/blob/main/example/recognition.ipynb

```
Speak to your microphone 5 sec...
Done!
```




```
[ ]  waves = [load_waveform('audio_ds.wav').to('cuda')]
     hyps = model.recognize(waves)
     print(hyps[0])
```

```
I LIKED MATH UNTIL THEY ADDED THE ALPHABET IN IT
```

What you can do to improve performance?

- Modify **configs**
  - egs/librispeech/config/ctc_train_example.yaml
- **Data augmentation**
  - w/ or w/o SpecAugment?
  - Speed perturbation
- Implement other model **architectures**
  - Encoder’s module (CNN, GRU, LSTM, Transformer, Conformer)
  - Framework (LAS / RNN-T)
- Input **features**
  - MFCC / fbank / spectrogram?

Reference

- CTC
    - https://www.cs.toronto.edu/~graves/icml_2006.pdf
    - http://proceedings.mlr.press/v32/graves14.pdf
    - https://distill.pub/2017/ctc/
- Data Augmentation
    - SpecAugment: https://arxiv.org/abs/1904.08779
    - Adaptive SpecAugment: https://arxiv.org/abs/1912.05533
    - Speed Perturbation: https://www.danielpovey.com/files/2015_interspeech_augmentation.pdf
- ASR Frameworks
    - RNN-T: https://arxiv.org/abs/1211.3711
    - LAS: https://arxiv.org/abs/1508.01211

Reference

- ASR Architectures
  - Transformer: https://arxiv.org/abs/1706.03762
  - Conformer: https://arxiv.org/abs/2005.08100
- Conventional ASR & Other Speech Processing Topics
  - http://ocw.aca.ntu.edu.tw/ntu-ocw/ocw/cou/104S204
- ASR Toolkits
  - Kaldi: https://github.com/kaldi-asr/kaldi (conventional ASR)
  - ESPnet: https://github.com/espnet/espnet
  - SpeechBrain: https://github.com/speechbrain/speechbrain
  - fairseq: https://github.com/pytorch/fairseq
  - flashlight: https://github.com/flashlight/flashlight

Reference

- Libraries
  - PyTorch: https://pytorch.org/docs/stable/index.html
  - PyTorch Ligtning: https://pytorch-lightning.readthedocs.io/en/latest/
  - torchaudio: https://pytorch.org/audio/stable/index.html
- Colab
  - Saving files to Google Drive: https://www.wongwonggoods.com/python/python-colab-mount-google-drive/