Using SSL models for Multilingual ASR

Lucas Ondel & Léa-Marie Lam-Yee-Mui

For the workshop

Working with 2 teams:

- Multilingual/Code-Switching ASR:
  - Building ASR systems coping with 2 languages at once
- Leveraging Pre-Training Models :
  - Adapting self-supervised models for speech processing

Research focus for the workshop:

- “Universal” Speech Recognition System

Universal ASR

An ASR system is universal if it usable **for everyone** and **by everyone**:

- It can recognize all languages (i.e. usable for everyone)
- Its construction and deployment is simple enough (i.e. usable by everyone)

Using SSL models

SSL models:

- Strong improvements on multilingual ASR 🙂
- Ease of use: easily adapted less target data 🙂
- Huge memory and computation requirements 🙁
- Decoding several languages is still a big issue 🙁

Towards Universal Speech Recognition…

- **Lightweight SSL** models
  - Using FNet architecture for pre-training on speech
- Using **semiring algebra** for adaptation and inference in SSL models for ASR
  - Efficient adaptation of SSL models with LF-MMI
  - Decoding speech

Simplification of models

- Transformers need lots of computation/memory

- Can we simplify the network architecture:
    - Use FNet[1] instead of Transformer

[1] “FNet: Mixing Tokens with Fourier Transforms" https://arxiv.org/pdf/2105.03824.pdf

Progress

Finished Pytorch implementation of TDNN-FNET and TDNN-Transformer architectures, integrated them in PyChain

	miniLibrispeech	**WSJ**	nb of params
5 TDNN (default)	24.99	4.42	1.85M
5 TDNN + 2 FNet + posEnc	26.31	5.18	3.04M
5 TDNN + 2 Transformer + posEnc	98.65	5.39	4.22M

Table 1: Preliminary results on miniLibrispeech and WSJ with different AM

Some future research directions

On WSJ and MLS, the default TDNN architecture is always better.

- Need of a strong baseline with the Transformer architectures
- Find the amount of data for the FNet architecture to work
- Use the data from the multilingual team, mostly code-switching speech (all in QCRI cluster, still working with QCRI support to install things and run the recipes correctly)

Is there any alternatives to the Transformer ?

- Time measurements needed

Multilingual/CS team

4 work packages

WP1: multilingual ASR

WP2: CS text data generation

WP3: evaluation of CS ASR

WP4: analytic, CS explaining

Use WP1 for the pretraining team

Using features from SSL models

Compare them to traditional features, for instance on the CS data

Started preparing features with HuBERT on WSJ (QCRI is not quite ready)

- too much memory requirements
- can take quite a lot of time

Ideally, use the pretrained models from the pretraining team as alternatives to the models currently released.

For the workshop…

- Alternatives to the Transformer models
- Efficient adaptation of SSL models with LF-MMI loss function for ASR
    - PyChain
    - Matrix-based (see Lucas)
- Matrix-based decoder (Multilingual team)