Outline

- Task Description
- Dataset
- Data segmentation
- Kaggle
- Guidelines
Task Introduction

- Self-attention
  - Proposed in GOOGLE's work, *Attention is all you need*. It combines the strengths of RNN (consider whole sequence) and CNN (processing parallelly).
- Main goal: Learn how to use transformer.
HW2: Phoneme classification

Task: Multiclass Classification

Framewise phoneme prediction from speech.

What is a phoneme?

A unit of speech sound in a language that can serve to distinguish one word from the other.

- bat / pat, bad / bed
- Machine Learning → M AH SH IH N  L ER N IH NG
HW4: Speaker classification

Task: Multiclass Classification

Predict speaker class from given speech.
Dataset

- Training: 69438 processed audio features with labels.
- Testing: 6000 processed audio features without labels.
- Label: 600 classes in total, each class represents a speaker.
Data Preprocessing

Acoustic Feature

Waveform → DFT → mel-spectrogram

log

spectrogram

... → filter bank

ref. prof. Hung-Yi Lee
[2020Spring DLHLP] Speech Recognition
Data formats

- Data Directory
  - metadata.json
  - testdata.json
  - mapping.json
  - uttr-\{random string\}.pt

- The information in metadata
  - "n_mels": The dimension of mel-spectrogram.
  - "speakers": A dictionary.
    - Key: speaker ids.
    - value: "feature_path" and "mel_len"
Data segmentation during training

Different length
Data segmentation during training

Different length

Segment during training

Segment = 2
Sample Code

Colab Link: link

- Baselines:
  - Simple: Run sample code and know how to use transformer.
  - Medium: Know how to adjust parameters of transformer.
  - Hard: Construct conformer which is a variety of transformer.
Requirements - Simple

- Build a self-attention network to classify the speaker with the sample code.
- Simple public baseline: 0.82523
Requirements- Medium

- Modify the parameters of the transformer modules in the sample code.
- Medium public baseline: 0.90547

```python
class Classifier(nn.Module):
    def __init__(self, d_model=80, n_spks=600, dropout=0.1):
        super().__init__()
        # Project the dimension of features from that of input into d_model.
        self.prenet = nn.Linear(40, d_model)
        # TODO:
        # Change Transformer to Conformer.
        self.encoder_layer = nn.TransformerEncoderLayer(
            d_model=d_model, dim_feedforward=256, nhead=2
        )
        # self.encoder = nn.TransformerEncoderLayer(encoder_layer, num_layers=2)

        # Project the the dimension of features from d_model into speaker nums.
        self.prenet = nn.Sequential(
            nn.Linear(d_model, d_model),
            nn.ReLU(),
            nn.Linear(d_model, n_spks),
        )
```
Requirements - Hard

- Improve the performance by constructing the **conformer** layer.
- Hard public baseline: 0.95404

```python
class Classifier(nn.Module):
    def __init__(self, d_model=80, n_spks=600, dropout=0.1):
        super().__init__()
        # Project the dimension of features from that of input into d_model.
        self.prenet = nn.Linear(40, d_model)

        # TODO:
        # Change Transformer to Conformer.
        self.encoder_layer = nn.TransformerEncoderLayer(
            d_model=d_model, dim_feedforward=256, nhead=2
        )

        # Project the the dimension of features from d_model into speaker nums.
        self.pred_layer = nn.Sequential(
            nn.Linear(d_model, d_model),
            nn.ReLU(),
            nn.Linear(d_model, n_spks),
        )
```
Grading

- Evaluate Metrics = @1 Accuracy.
- Simple baseline (public) +1 pt (sample code)
- Simple baseline (private) +1 pt (sample code)
- Medium baseline (public) +1 pt
- Medium baseline (private) +1 pt
- Hard baseline (public) +1 pt
- Hard baseline (private) +1 pt
- Upload code to NTU COOL +4 pts

Total: 10 pts
# Submission Format

- "Id, Category" split by ',', in the first row.
- Followed by 6000 lines of "filename, speaker name" split by ',',.

<table>
<thead>
<tr>
<th>Id</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>uttr-7eadda33f5fe4c9fa884c30ca0c05381.pt</td>
<td>id11111</td>
</tr>
<tr>
<td>uttr-7e0673bd280e4d5e8f352c8b9b5872b3.pt</td>
<td>id22222</td>
</tr>
<tr>
<td>uttr-9681040a85a8490cb7486f968c26131a.pt</td>
<td>id33333</td>
</tr>
<tr>
<td>uttr-dc680bc998a84069835e4422e3b46324.pt</td>
<td>id44444</td>
</tr>
<tr>
<td>uttr-3184e679b6ab43d7a4b5016ac35b38cb.pt</td>
<td>id55555</td>
</tr>
</tbody>
</table>
Deadlines

- Kaggle: 2021/04/16 23:59 (UTC+8)
- NTU COOL: 2021/04/18 23:59 (UTC+8)
Grading -- Bonus

- **If you got 10 points**, we make your code **public** to the whole class.

- In this case, if you also submit a **PDF report briefly describing your methods** (<100 words in English), you get a bonus of **0.5 pt**. (your report will also be available to all students)

- **Report template**
Code Submission

● **NTU COOL** (4pts)
  ○ Compress your code and report into

    <student ID>_hw4.zip

* e.g. b06901020_hw4.zip

○ We can only see your last submission.
○ Do not submit your model or dataset.
○ If your code is not reasonable, your semester grade x 0.9.
Code Submission

- Your .zip file should include only
  - Code: either .py or .ipynb
  - Report: .pdf (only for those who got 10 points)

- Example:
Links

Kaggle: [link]
Colab: [link]
Data: [link]
Video (Chinese): [link]
Video (English): [link]
Self-Attentive Speaker Embeddings: [link](#)

**Hints**

Speaker classification system

Self-attention pooling
Hints

Conformer: [link]
Hints

Additive margin softmax: link
Regulation

- You should NOT plagiarize, if you use any other resource, you should cite it in the reference. ( * )
- You should NOT modify your prediction files manually.
- Do NOT share codes or prediction files with any living creatures.
- Do NOT use any approaches to submit your results more than 5 times a day.
- Do NOT search or use additional data or pre-trained models.
- Your final grade x 0.9 if you violate any of the above rules.
- Prof. Lee & TAs preserve the rights to change the rules & grades.

( * ) Academic Ethics Guidelines for Researchers by the Ministry of Science and Technology
If any questions, you can ask us via...

- **NTU COOL (recommended)**
  - [https://cool.ntu.edu.tw/courses/4793](https://cool.ntu.edu.tw/courses/4793)

- **Email**
  - [ntu-ml-2021spring-ta@googlegroups.com](mailto:ntu-ml-2021spring-ta@googlegroups.com)
  - The title should begin with “[hw4]”

- **TA hour**
  - Each Friday during class