



# Taiwanese Speech Recognition using Transformer Report

學生: 陳澤昕

學號: 311356003

# Code

- 在這次的作業中，完成了將音檔利用 sox 轉換、ESPnet 環境的架設，以及模型建立的所有步驟，這些步驟都包含在 run.ipynb 檔案中。
- 實作是由助教在上課時提供的程式碼，搭配上由卡內基梅隆大學提供的教學影片 (<https://www.youtube.com/watch?v=YDN8cVjxSik> 所產生的)。在run.ipynb中我先利用 AH4 資料集來進行 Stage 0 到 Stage 13 的實作，然後再進行這次的Kaggle 競賽題目。

```
# It takes a few seconds
!git clone https://github.com/espnet/espnet

%cd <espnet-root>/tools
!CONDA_TOOLS_DIR=$(dirname ${CONDA_EXE})/..

# We use a specific commit just for reproducibility.
%cd /data/tzeshinchen/deep_learning/kaggle/espnet
!git checkout 3a22d1584317ae59974aad62feab8719c003ae05
```

```
# It takes 30 seconds
!./setup_anaconda.sh ${CONDA_TOOLS_DIR} espnet 3.9
%cd /data/tzeshinchen/deep_learning/kaggle/espnet/tools
```

# Dataset

Evaluation :

單人女聲聲音（高雄腔）

輸入：台語語音音檔（格式：wav檔, 22 kHz, mono, 32 bits）

輸出：台羅拼音（依教育部標準）

- Training set :

kaggle資料夾下的training set利用sox轉音檔格式將22kHz的音檔轉換成16kHz，

- Testing seting :

kaggle資料夾下的testing set同樣利用sox轉音檔格式將22kHz因檔轉換成16kHz，

- Validation set :

從training set中隨機挑選10個資料當validation

# Model

- 這次作業嘗試了兩種變形的tranformer分別是branchformer 以及conformer這兩種。

經過幾次的嘗試最終採用branchformer

```
encoder_conf:
  output_size: 256
  use_attn: true
  attention_heads: 6
  attention_layer_type: rel_selfattn
  pos_enc_layer_type: rel_pos
  rel_pos_type: latest
  use_cgmlp: true
  cgmlp_linear_units: 2048
  cgmlp_conv_kernel: 31
  use_linear_after_conv: false
  gate_activation: identity
  merge_method: concat
  cgmlp_weight: 0.5 # used only if m
  attn_branch_drop_rate: 0.0 # used only if m
  num_blocks: 24
  dropout_rate: 0.1
  positional_dropout_rate: 0.1
  attention_dropout_rate: 0.1
  input_layer: conv2d
  stochastic_depth_rate: 0.0
```

Encoder

- Encoder  
主要input layer 是採用conv2d  
attention head : 6  
Drop rate : 0.1  
Output\_size : 256

- Decoder  
主要用原本的設定沒有特別更改

```
# decoder related
decoder: transformer
decoder_conf:
  attention_heads: 4
  linear_units: 2048
  num_blocks: 6
  dropout_rate: 0.1
  positional_dropout_rate: 0.1
  self_attention_dropout_rate: 0.
  src_attention_dropout_rate: 0.
```

Decoder

# Training

- 利用助教教的环境架设後去training
- GPU 使用4張1080 ti (跑了大概9小時)
- 有遇到RuntimeError: CUDA out of memory

```
length_normalized_loss: false  
  
# minibatch related  
batch_type: numel  
batch_bins: 2500000  
  
# optimization related
```





```
# minibatch related  
batch_type: numel  
batch_bins: 750000  
  
# optimization related  
accum_grad: 1
```


- 因為覺得batch size調低會導致訓練的量不夠多所以將epoch調整從原本60調整成150並且learning rate 調整成0.000125

# Result

## Public :

31	311356003		3.87378	10	21d
<div>Your Best Entry! Your submission scored 7.65048, which is not an improvement of your previous score. Keep trying!</div>					

## Private :

29	—	311552011		2.41563	8	22d
30	▲ 1	311356003		2.61316	10	21d
31	▼ 1	310581050		2.64609	6	5d

# Discussion

- 對於Testing set 的準確度只有97.3%多一點我認為有幾點可能的問題
  1. 為了防止out of memory 我將batch size設定較小導致模型沒辦法收斂到最小，下次作也可能要在memory 允許的範圍下多測試幾種batch\_size。
  2. 模型可能可以在繼續收斂，但是在那之前我設定的epoch就已經到了透過加大epoch應該可以解決這個問題。
  3. 測試的模型參數不夠多，沒有找出最佳的模型設定。
- 如過硬體資源足夠:
  1. 可以針對attention head，input size,以及block的數量進行測試
  2. 另外這次的調整主要針對encoder，下次可以連decoder的部分都更換合適的參數以及模型調整看看，可能可以得到更好的結果