

## Generative AI class 1:

### ■ 報告繳交時間:

8/19(一)上課前

### ■ Demo 時段:

8/13(三) 13:00 – 16:00

8/19(一) 17:00 – 18:00

去 Excel 裡面填時段，完成 Part1 就可以先來找我 Demo

<https://docs.google.com/spreadsheets/d/1GvIqz68xQSUCftC8fY19MHwbZohDc9quR6aPpO-Kjtw/edit?usp=sharing>

如果你有想要其他時間可以跟我約

### ■ 統一 QA 時段:

8/14 (三) 15:00 – 16:00 沒有一定要參加有問題再來就好  
要參加一樣去 excel 填寫參加

### ■ TODO

#### Part1: Model

1. PositionalEncoding
2. Attention
3. MultiHeadAttention
4. Residual
5. Feed forward

#### Part2: Trainer

6. Loss
7. Trainer

### ■ TODO Detail

1. PositionalEncoding : <https://medium.com/@hunter-j-phillips/positional-encoding-7a93db4109e6>
2. Attention
 
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$
3. MultiHeadAttention
 

In the multi-head attention mechanism, after first initializing Q, K, and V, we then transform their initial shapes from `(batch\_size, seq\_len, d\_model)` to `(batch\_size, num\_heads, seq\_len, d\_k)`, where `d\_k = d\_model / num\_heads`.
4. Residual
 
$$\text{Output} = \text{Input} + \text{dropout}(\text{layer}(\text{normalization}(x)))$$
5. Feed forward
 
$$\text{Dim} : (\text{batch}, \text{seq\_len}, \text{d\_model}) \rightarrow (\text{batch}, \text{seq\_len}, \text{ff\_dim}) \rightarrow (\text{batch}, \text{seq\_len}, \text{d\_model})$$
6. Loss
 
$$\text{CrossEntropyLoss}$$
7. Trainer
  - a. Input -> Encoder -> Encoder output
  - b. Encoder output, Decoder input -> Decoder -> Decoder output
  - c. Decoder output -> Projection -> output
  - d. Loss(Output, Label)

■ 環境需求:

1. Python>3.09
2. Pytorch > 1.10.0 **hint: sever cuda version=1.12 (11.3)**
3. datasets= 2.20.0
4. tokenizers=0.19.1
5. torchmetrics= 1.4.0.post0
6. tensorboard

## ■ 報告內容

### ■ Experimental Results

1. 執行 model.py 輸出(2, 5, 10)
2. Tensorboard 訓練 loss 收斂的截圖

`tensorboard --logdir=./your_logs_path`

### ■ Transformer 問題(拜託不要寫太多字最多最多一張 A4)

1. Transformer 模型採用的損失函數是什麼？它是如何運作的，使模型能夠學習到上下文關係和文法規則？
2. Attention 機制是如何改善過去 RNN 和 CNN 的缺點的？它具體解決了哪些問題，又是如何實現這些改進的？
3. Attention 機制是透過哪個部分進行視覺化的？如果可以的話，能否使用你們的模型來展示這個視覺化的過程？(套件: altair or seaborn heatmap)

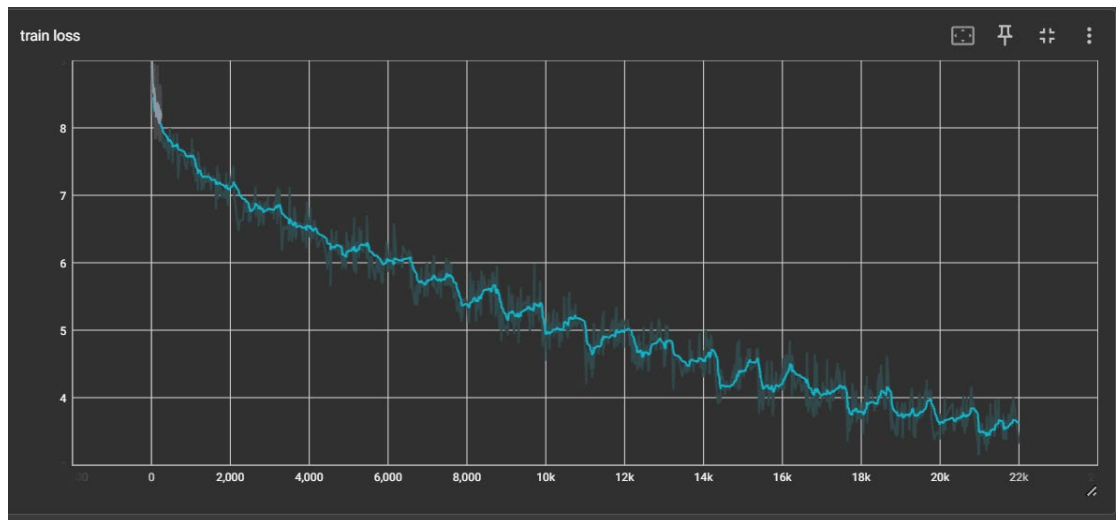
### ■ Example Report

1.

```
(base) tzeshinchen@cluster-ai3:/data/tzeshinchen/AI_example/transformer/src$ python model.py
Initialized model with random weights.

Final output size: torch.Size([2, 5, 10])
```

2.



1. Transformer 採用的 loss function 是甚麼是怎麼運作使模型可以學習到上下文關係?  
...
2. Attention 機制改善了過去 RNN 以及 CNN 分別的問題是甚麼，是如何改善的?  
...
3. Attention 是透過哪一個部分視覺化。可以的話裡用你們的模型畫出來

