NYCU IEE Deep Learning
Lab 3 Report: Machine Translation

陳柏翔　313510156

1 Introduction

本次作業旨在完成中文對英語的文句翻譯任務，從頭實作完整的 Transformer [1] 模型架構，並在給定的 Testing set 上達到 BLEU (1-gram) > 0.25 以及 BLEU (2-gram) > 0.1 的成果。而在此作業中，我最終達到 BLEU (1-gram) 為 0.46 ，以及 BLEU (2-gram) 為 0.33 的成果。

2 Dataset

2.1 English-Chinese Translation Dataset

給定的資料集源自於 Tatoeba [2] 與 XDailyDialog [3]，範例資料如 Figure 1 所示。其中，Training/Validation/Testing Set(Public) 的資料量分別為 47,500/2,500/200 個中英文對照句組。

	English	Chinese
0	I'm Susan Greene.	我是蘇珊格林。
1	You don't have to take an examination.	你不需要考试。
2	I can't leave.	我走不了。
3	A cold beer would hit the spot!	来杯冰啤酒就太棒了!
4	Let's start!	讓我們開始吧。
...	...	...
49995	Just buy a cask of wine. Have you bought ice yet?	买一桶酒就行了。你买冰块了吗?
49996	OK. No problem.	好的,没问题。
49997	I'm not really in the mood for Italian, actual...	实际上,我不太喜欢意大利菜。我想吃点辣的。
49998	It's OK. It seems we have a lot in common.	还行吧。看来我们有很多共同点。
49999	What? You've got to be kidding me!	什么?开什么玩笑!

Figure 1: Examples from the English-Chinese translation dataset.

2.2 Tokenization

不同於先前作業在 Computer Vision (CV) 領域中的任務，在 Natural Language Processing (NLP) 領域中，我們需要先將文句中的每個字 (或詞) 轉換為 Tokens 才能讓機器有辦法學習，而 Token 就是每個有意義的詞被轉換後所對應的數值，轉換後結果如 Figure 2 範例所示。而其中也不乏一些具有特殊意義的 Token，例如 `101` 代表 `[Begin of Sentence]` (`[BOS]`)、`102` 代表 `[End of Sentence]` (`[EOS]`)。

```
Chinese Input:  她知道您的電話號碼嗎?
Token IDs:  [101, 1961, 4761, 6887, 2644, 4638, 7442, 6282, 5998, 4826, 1621, 136, 102]
```

Figure 2: Example of tokenization.

在此作業中我們所使用的 Tokenizer 為 BERT Tokenizer。訓練時我們會額外進行 Padding 前處理，將較短的句子補上 `0` 代表 `[Pad]` 的 Token 到相同的長度，這樣才有辦法組成一個 Batch 送入模型進行訓練。

3 Architecture

3.1 Network Overview

我基於標準的 Transformer [1] 做了一些修改，得到的最終模型如 Figure 3 所示。

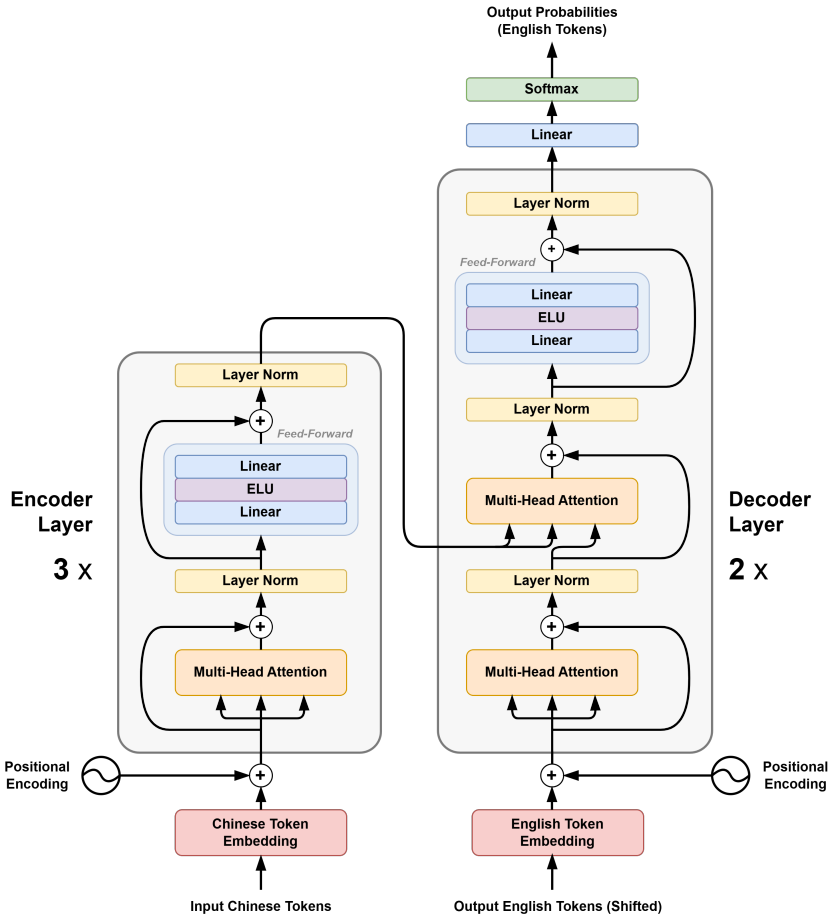

Figure 3: Seq2Seq network architecture used in this work.

而模型的主要超參數設置如下：

- Embedding size (`EMB_SIZE`, d_{model}): 256
- Number of heads in multi-head attention (`N_HEAD`, h): 4
- Feed-forward dimensions (`FFN_HID_DIM`, d_{ff}): 2048
- Number of encoder layers (`NUM_ENCODER_LAYERS`): 3
- Number of decoder layers (`NUM_DECODER_LAYERS`): 2

3.2 Layer Details

在 Encoder 與 Decoder 中，基本架構仍與標準 Transformer 一致，唯獨 Feed-Forward Network 的 Activation Function 都替換成 Exponential Linear Unit (ELU)，以得到更好的反向傳播能力。

而相較於標準 Transformer [1] 的 d_{model}、h、d_{ff} 等超參數，我都設置得明顯較小，這是因為訓練的資料量比起 [1] 少了許多，如果設置過大的超參數很容易導致模型 Overfitting。經過數次測試後，我發現此次作業中的 Encoder 與 Decoder 層數大約設置為 2 或 3 就已非常充足。

在模型訓練時，非常重要的一點是在 Attention 的計算中必須使用 Padding Masking 與 Causal Masking，前者可以避免模型關注到 `[Pad]` 的無意義 Tokens，而後者則是避免模型看到未來資訊。然而我注意到在課程講義中，Causal Masking 連同 Attention 矩陣 (即 QK^T) 對角線數值也一併設為負無窮大，這其實是錯誤的而且會使訓練無效，因為對角線為負無窮大會使當前要預測的 Token 權重在經過 Softmax 後為數值 0，相當於無法進行任何預測。

3.3 Positional Encoding

在標準 Transformer [1] 中，Positional Encoding 的資訊是透過以下算式計算：

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

然而，這樣的計算式其實會受到 d_{model} 的大小而影響其表現力，根據我的模型超參數設置 $d_{model} = 256$，得到的結果如 Figure 4 左圖所示，可以看到後半部分的 Embedding Dimensions 並沒有被有效利用到，使得鄰近的位置所加的 Positional Encoding 向量差異並不大，難以分辨不同位置的 Tokens。

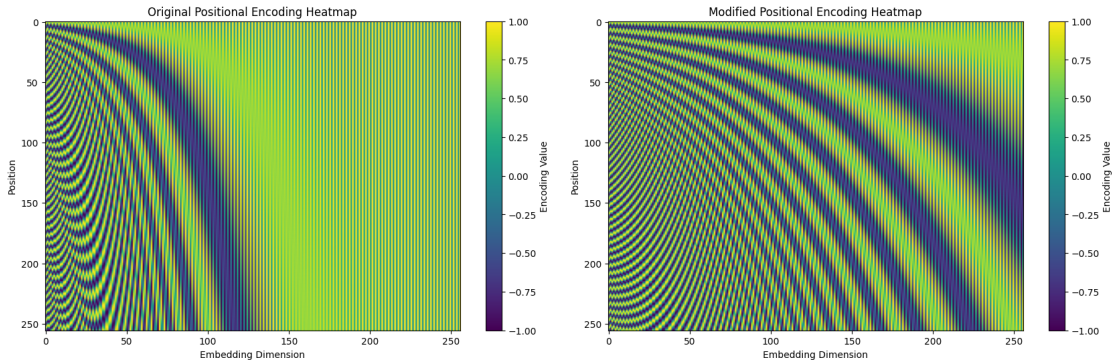

Figure 4: Positional encoding heatmaps.

因此，我將計算式的週期 (分母) 部分改為更小的數值，不再固定設為 10000，而是會根據可能的最長 Tokens 數 (即 `max_len`) 來決定週期範圍：

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{(max\_len/2\pi)^{2i/d_{model}}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{(max\_len/2\pi)^{2i/d_{model}}}\right)$$

得到的結果如 Figure 4 右圖所示，可以發現後半的 Embedding Dimensions 也隨著位置不同而有明顯的變化。經過我多次實測，這項改動不只比原本的 Positional Encoding 來得好，甚至比起使用 Learnable Positional Encoding 有更好的效果。

3.4 Beam Search

在本作業中，我採用了 Beam Search 技巧作為解碼策略，以在翻譯過程中取得較佳的輸出品質。相較於傳統的 Greedy Search 僅選取每一步中機率最高的下一個 Token，Beam Search 會同時保留多個潛在候選序列（稱為「beam」），並根據其累積的對數機率分數進行排序與裁剪。具體而言，我在每個解碼步驟中保留前 k 個分數最高的候選句（k 為 `num_beams` 參數），並對這些候選句分別生成下一個 Token 的分佈，最後再從所有擴展後的序列中選取新的前 k 個，形成下一輪的搜尋空間。

此過程會持續進行，直到所有候選句皆生成結束符號 (`[EOS]`) 或達到最長長度上限。如此設計能在合理的計算成本下減少模型陷入局部最優解的可能性。根據我在實驗中的觀察，使用 `num_beams = 3` 的設定即可在生成品質與推論速度之間取得良好平衡，比起 Greedy 算法提高了約 0.02 的 BLEU (1-gram) 準確率。

3.5 Model Details

以下 Table 1 為使用 `torchinfo` 套件所測試出的模型參數量、運算量與記憶體使用量。在此作業中，我的模型參數量為題目要求的參數量限制 200M 的 12.4%，符合題目要求。

Total Params	24,759,876
MACs	198.08 M
(Encoder and Decoder) Input Size	(1, 128)
Forward/backward Pass Size	47.26 MB
Estimated Total Size	146.30 MB

Table 1: Model summary.

上表的後三項為假設輸入序列在 Encoder 端與 Decoder 端皆為 (1, 128) 形狀的序列。

4 Training Strategy

為了提高模型的準確率，首先透過前面章節 3 中所討論的內容，在模型設計上調整超參數到適合的大小，以及更改 Positional Encoding 的設計，就能夠大幅度的提高模型準確率。但是除此之外，模型訓練方式也很重要，以下是針對我在訓練時的細節做說明。

4.1 Loss Function and Optimizer

在此作業中，我所使用的 Loss Function 為 Cross Entropy Loss，搭配 SGD Optimizer 來進行模型優化，並使用 Learning Rate Scheduler (Reduce LR On Plateau) 調整更新幅度，參數設置如下：

- Learning Rate: 10^{-2}
- Number of Epochs: 80
- SGD Momentum: 0.9
- SGD Weight Decay: 10^{-5}
- Scheduler Factor: 0.7
- Scheduler Patience: 5
- Minimum Learning Rate: 10^{-5}

這樣的設置讓我的模型可以快速的收斂，如果持續數個 Epochs 都沒有收斂的現象再下調 Learning Rate。此外，我的模型儲存條件允許準確率較最高準確率低 0.5% 也會進行儲存。

4.2 Data Augmentation: Random Token Dropout

為了提高模型的泛化能力，我希望增加模型的上下文推理能力。因此我透過隨機覆蓋 Token 的方式來使輸入句中的某個詞被 `[Pad]` 覆蓋掉，具體作法是在每次取樣一筆訓練資料時，模型以機率 $p = \text{augment\_prob}$ 決定是否進行增強。如果 (中文) 輸入樣本的長度大於 7，則隨機選取輸入序列中的一個位置，並將該位置的 Token 替換為 `[Pad]`。

這樣的操作可模擬真實應用中缺字或斷句不完整所造成的資訊缺失情境。而我之所以設下長度需要大於 7 的限制，是因為當句子太短的時候 (e.g. 句子長度 ≤ 5) 每個詞都有非常重要的意義，隨機缺失就很容易覆蓋到句中很重要的詞。另外，我還嘗試過以新的 `[Mask]` Token 來隨機覆蓋，但是得到的訓練結果不如預期，因此改用 `[Pad]` 來做覆蓋。

4.3 Data Augmentation: Back-translation

Back-translation 是一種常見的資料增強方法，會透過一個反向翻譯模型（即將目標語言翻譯回來源語言的模型）來生成新的平行句對，以擴充訓練資料。

然而，由於本次作業的限制，不能使用任何 Pre-trained 模型或外部 Datasets，因此無法利用外部語料訓練出具備語意遷移能力的反向翻譯模型。如果只使用相同的 Training Set 進行反向翻譯訓練，實際上等同於讓模型學習原始資料的雙向對應關係，並未產生任何新的語料或語意變化。因此這樣的 Back-translation 並無實質增強效果。

基於上述原因，且經過我實測後發現模型準確率只會下降 (BLEU score (1-gram) 約為 0.41) 而不會上升，因此最終沒有採用 Back-translation 作為資料增強策略。

5 Experiments

5.1 Model Training

模型的訓練過程如 Figure 5 所示，左圖為訓練過程中的 Loss 收斂變化曲線，右圖則是在 Validation Set 上的準確率變化曲線。

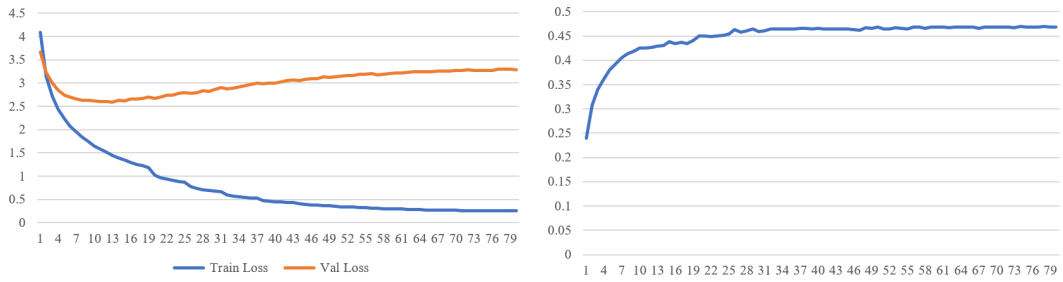

Figure 5: Loss (left) and validation accuracy (right) during training.

從 Loss 變化可以發現只有 Training Loss 在收斂，而 Validation Loss 則越來越差，但是這是一個正常現象，因為計算 Loss 的方式是根據 Ground Truth 與 Predicted Tokens 之間計算 Cross Entropy，然而隨著模型對語句的理解，模型的輸出可能是合理的，但是卻不是跟 Ground Truth 一模一樣的 Tokens 順序或位置，這樣就會使得 Loss 特別大。

而從 Validation Set 上的準確率變化可以看到，模型的準確率是有在正常提升的，最終爬升到 0.47 的位置，因此能推斷訓練過程是正常的。

5.2 Testing Result

以下 Figure 6 為最終在 Testing Set 上的測試結果，`num_beams` 數量設置為 3，執行時間約 8 秒。

```
The parameter size of model is 24759.876 k
====================  PASS parameter size requirement  ====================
BLEU score (1-gram) =  0.4644069808535278
BLEU score (2-gram) =  0.3297111455723643
BLEU score (3-gram) =  0.24184932246804236
BLEU score (4-gram) =  0.17688478745520114
====================  PASS BLEU score requirement      ====================
execution time = 8.469s
====================  PASS execution time requirement  ====================
```

Figure 6: The results on testing set.

5.3 Ablation Study

	BLEU score (1-gram / 2-gram)	**Execution time (seconds)**
Using PE in [1] + Greedy Search	0.419 / 0.278	3.3
Using PE in [1] + Beam Search	0.441 / 0.298	8.9
Using modified PE + Greedy Search	0.446 / 0.306	3.3
Using modified PE + Beam Search	0.464 / 0.330	8.5

Table 2: Test results in different scheme.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] Tatoeba. Tatoeba: The sentence collection project. https://tatoeba.org/zh-cn/. Accessed: 2025-10-15.

[3] Zeming Liu. Xdailydialog: An extended dailydialog dataset. https://github.com/liuzeming01/XDailyDialog, 2022. Accessed: 2025-10-15.

A Translation Testing

此章節展示模型訓練後的實際翻譯成效，在 Figure 7 中以簡體中文句子做為輸入語句，而 Figure 8 則以繁體中文做為輸入語句，結果上並沒有明顯差異。

```
Input:          : 你好，欢迎来到中国。
Prediction      : You are a good reputation for China.
Ground truth    : Hello, welcome to China.
Bleu Score (1-gram):  0.1428571492433548
Bleu Score (2-gram):  0.0
Bleu Score (3-gram):  0.0
Bleu Score (4-gram):  0.0
```

```
Input:          : 她知道您的電話號碼嗎?
Prediction      : Does she know your telephone number?
Ground truth    : Does she know your telephone number?
Bleu Score (1-gram):  1.0
Bleu Score (2-gram):  1.0
Bleu Score (3-gram):  1.0
Bleu Score (4-gram):  1.0
```

```
Input:          : 你现在在哪里工作?
Prediction      : Where do you work right now?
Ground truth    : Where do you work now?
Bleu Score (1-gram):  0.8333333134651184
Bleu Score (2-gram):  0.7071067690849304
Bleu Score (3-gram):  0.6299605369567871
Bleu Score (4-gram):  0.5372849702835083
```

Figure 7: Translation Examples (Simplified Chinese)

```
Input           : 歡迎來到台灣。
Prediction      : Welcome to Taiwan.
Ground truth    : Welcome to Taiwan.
Bleu Score (1-gram):  1.0
Bleu Score (2-gram):  1.0
Bleu Score (3-gram):  1.0
Bleu Score (4-gram):  0.0
```

```
Input           : 你好，歡迎來到台灣。
Prediction      : You are competent studying in Taiwan.
Ground truth    : Hello, welcome to Taiwan.
Bleu Score (1-gram):  0.1666666567325592
Bleu Score (2-gram):  0.0
Bleu Score (3-gram):  0.0
Bleu Score (4-gram):  0.0
```

Figure 8: Translation Examples (Traditional Chinese)

此外，從 Figure 7 的最上圖以及 Figure 8 觀察可以發現，句子開頭如果加上「你好」就會導致翻譯結果與預期結果相差甚遠。