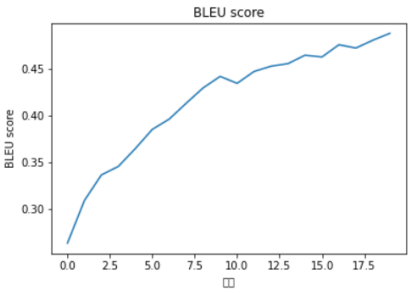
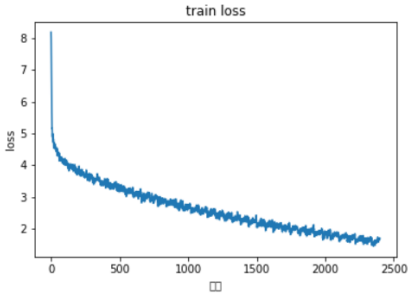
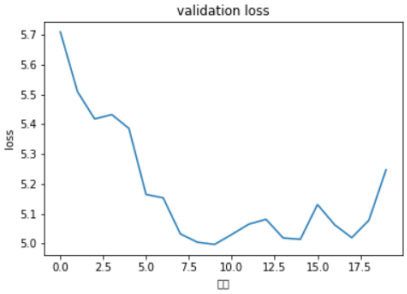
學號：b07901112　　系級：電機二　　姓名：劉聿珉

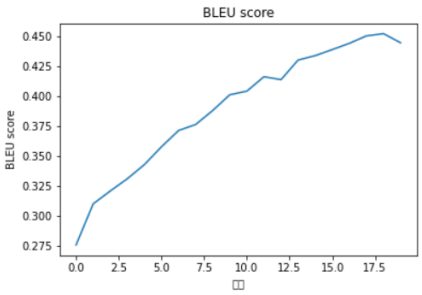
1. (20%) Teacher Forcing:
   a. 請嘗試移除 Teacher Forcing，並分析結果。
   兩組數據皆沒有做 attention、beam search。以這樣的條件下進行比較
   有 Teacher Forcing 的圖還有 result：
   val [12000] loss: 5.268, Perplexity: 206.518, bleu score: 1.011
   test loss: 5.5152, bleu_score: 0.4521

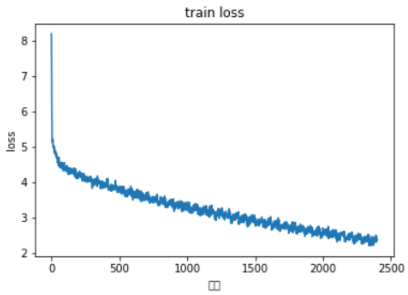


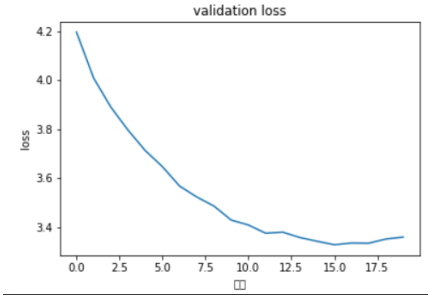




   移除 Teacher Forcing:
   val [12000] loss: 3.394, Perplexity: 29.509, bleu score: 0.449
   test loss: 3.6148, bleu_score: 0.3752







我們從上面的模型可以看到有 teacher forcing 的 Perplexity 比沒有 Teacher forcing 的 Perplexity 還要高上許多，但是沒有 teacher forcing 的 bleu score 比起有 teacher forcing 的 bleu score 還要低得多。我想這些結果是符合理想的，因為沒有 teacher forcing 的 model 會在模型樹狀圖上面亂走，沒有人告訴他正確的 output，所以 bleu score 自然就會非常的不好。而有 teacher forcing 的則是不斷的在沿著正確的 output 的那個分之在 train 使得他對於其他分支並沒有良好的理解，所以只要一開始走到其他分支，就會使模型走到一個對他來說未知的領域，Perplexity 自然就會高上許多。

而有 teacher forcing 的 loss 也比沒有 teacher forcing 的 loss 大上許多，由此可以知道，Teacher forcing 對於 training model 不會是一個很好的策略。

2. (30%) Attention Mechanism:

. 請詳細說明實做 attention mechanism 的計算方式，並分析結果。

沒有做：Attention Mechanism

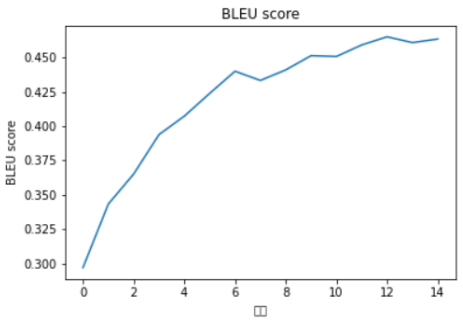
val [12000] loss: 3.421, Perplexity: 31.349, bleu score: 0.457

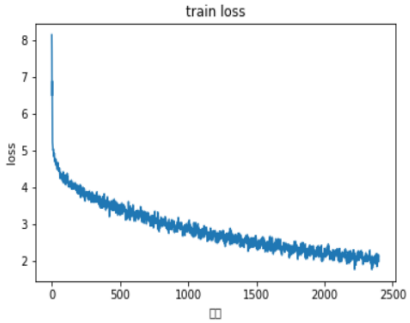
topk=3: test loss: 3.5807, bleu_score: 0.4322

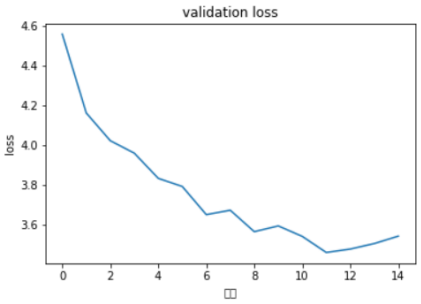
做 Attention Mechanism:

val [12000] loss: 3.561, Perplexity: 35.024, bleu score: 0.459

test loss: 3.5991, bleu_score: 0.4532







結果討論：

可以看到不管是 loss 或是 bleu score 有做 attention 的結果都比沒有 attention 的結果來的好。由此可以知道利用 attention 模型可以使模型注意力更集中在某些字詞上，不僅可以降低 loss 還可以使產生的語句更接近人所說的話(bleu score up!)

Attention 的使用時機：

在 decoder 中 input 經過 RNN 之後馬上跟 attention 進行對接，之後再將所產生的東西送入全連接層（Fully Connected Layer）。

Attention 的產生過程：

首先我會先將 encoder output 和 decoder 的 hidden vector reshape 之後 cat 在一起，然後餵到一個 RNN 裡面，RNN 模型結構如下：

```
self.dnn = nn.Sequential(
        nn.Linear(hid_dim*2*(sequence_len+num_layers), 1024),
        nn.Dropout(p=0.2),
        nn.ReLU(),
        nn.Linear(1024, 512),
        nn.Dropout(p=0.2),
        nn.ReLU(),
        nn.Linear(512, hid_dim*2),
        nn.Softmax()
    ).to(device)
```

然後 output 就是我們所需的 attention 了。

3. (30%) Beam Search:

. 請詳細說明實做 beam search 的方法及參數設定，並分析結果。

每次更新的方法

藉由 decoder 得到每個單詞的機率

計算累加的機率且存起來

求出新的分支

根據新 topk 分支，更新參數

找出 topk 的最新單字，當作下一輪的輸入

最後輸出紀錄的 top1。

參數設定：

(1) marked 紀錄所走過的字

(2) accuprob 累積機率

(3) probilitylist 紀錄 topk 分支進行下一次輸出後的 topk 機率

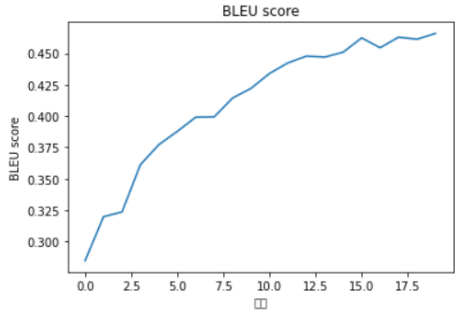
(4) canlist 紀錄 topk 追蹤的單字編號

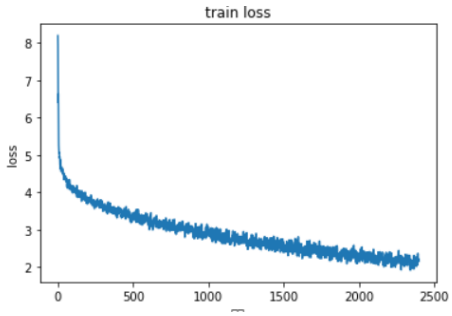
(5) hlist 紀錄 topk 分支的 decoder hidden state

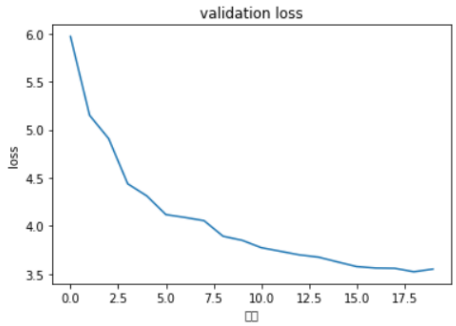
(6) hidden_buffer 的 buffer

因為設定了不少變數，在這邊只舉出比較重要的幾個

實驗作圖：







Test: Without Beam Search:

test loss: 3.6322, bleu_score: 0.4566

Test: With Beam Search:

test loss: 3.6105, bleu_score: 0.4393

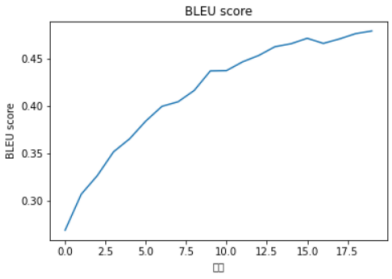
這樣的結果跟預期的有些出入，沒有 beam search 得到的 bleu score 高於有 beam search 的 bleu score。

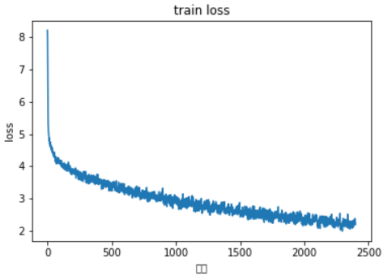
我想有可能是因為 model 沒有 train 好。或是在 training 的過程中，它將正確答案剪枝掉了，以至於得不到理想得答案。這部分就是 beam search 不太好控制的地方了，或許可以再用一些演算法改進。

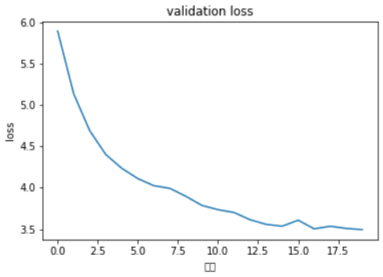
4. (20%) Schedule Sampling:

. 請至少實做 3 種 schedule sampling 的函數，並分析結果。

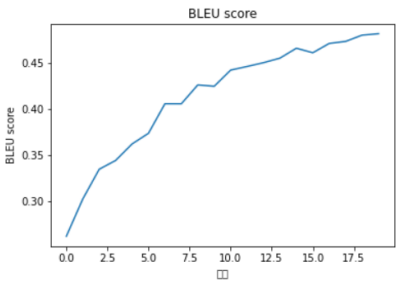
a. 第一個 schedule sampling

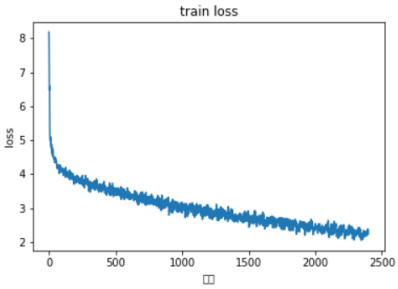


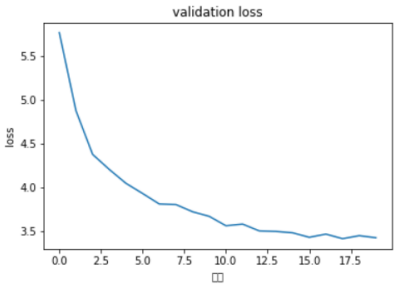




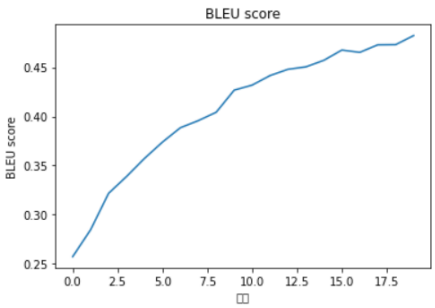
b. 第二個 schedule sampling

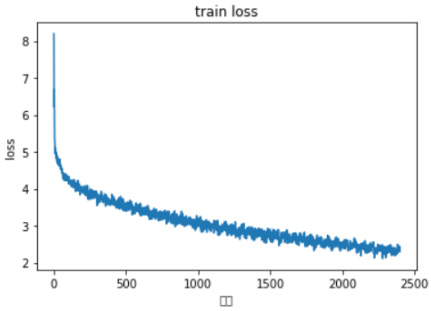


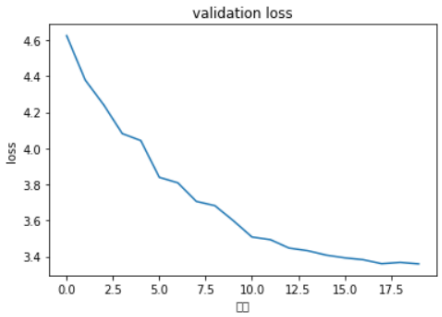




c. 第三個 schedule sampling







(a)linear

公式:teacher_forcing_ratio = max(bottom,init-slope*step)

bottom=0.3，init=1，slope=0.03

結果:

val [12000] loss: 3.628, Perplexity: 37.756, bleu score: 0.458

test loss: 3.8174, bleu_score: 0.4468

(2)exponential

公式: teacher_forcing_ratio= ratio**step

ratio = 0.949

結果:

val [12000] loss: 3.436, Perplexity: 31.076, bleu score: 0.413

test loss: 3.621, bleu_score: 0.4434

(3)inverse sigmoid

公式: teacher_forcing_ratio = tou/tou+exp(step/tou)

tou=6.218

結果:

val [12000] loss: 3.435, Perplexity: 29.241, bleu score: 0.428

test loss: 3.611, bleu_score: 0.4455

在 test bleu score 方面，linear>inverse sigmoid>exponential，在 test loss 方面，inverse sigmoid<exponental<linear。

但是三者的 bleu score 還是比 teacher forcing 的分數還要糟，我想可能是因為 schedule 參數需要再調整。