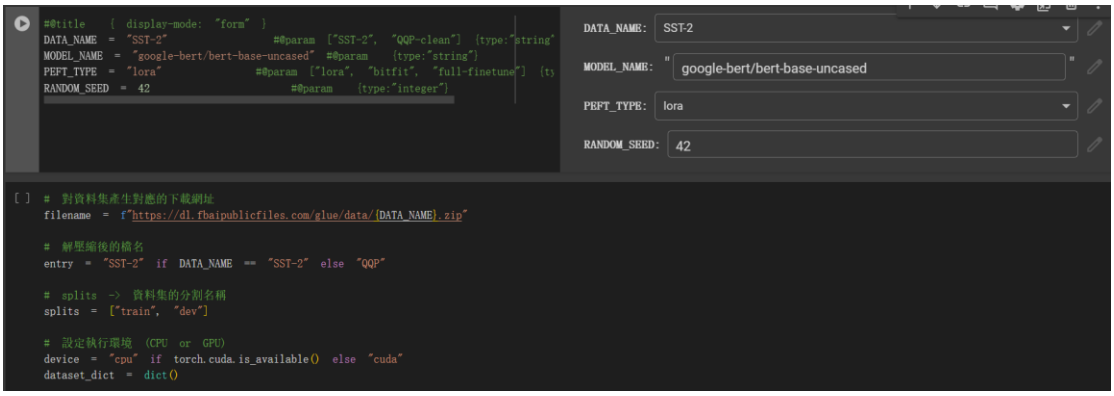
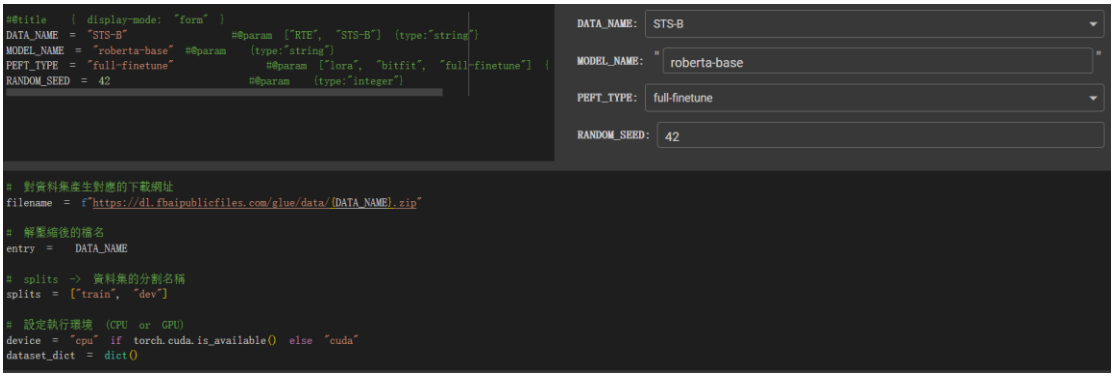
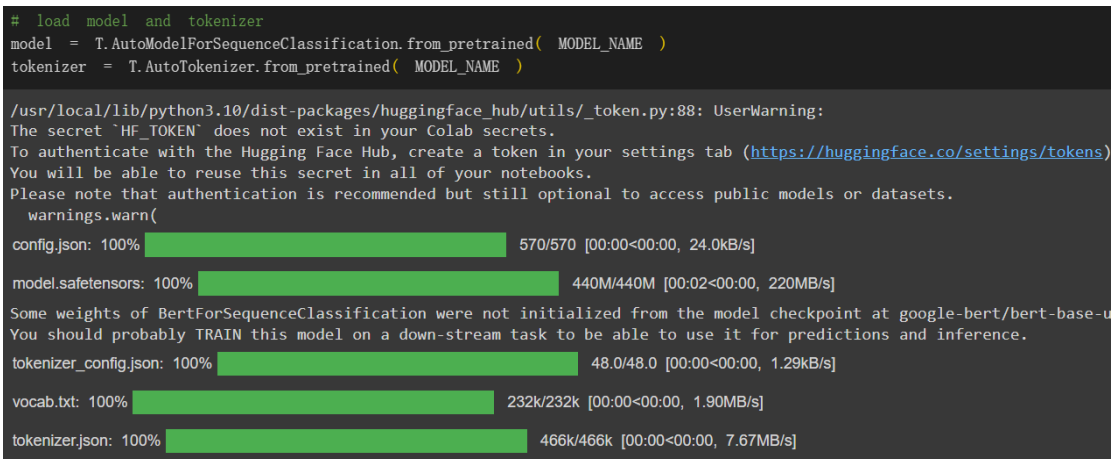
1. Data (Task1, 20 pts)
    1. You can choose any perferring dataset in GLUE benchmark to fine-tuning.Load the train and validation split of data (10 pts). You can use pandas, Dataset and Dataloader.
    如下兩張圖所示，我選用的 dataset 是 SST-2 以及 STSB，並載入他們各別的分割資料集。





    2. Select two datasets on GLUE benchmarks (10 pts). Tokenize the text. You can design your own tokenizer or use any API (if available).
    SST-2 部分所使用的 tokenizer:



    STSB 部分所使用的 tokenizer:

```
num_labels = 1 # 定義不同資料集的label數量(1 -> regression -> MSELoss)

# 讀入 huggingface 的 model 與 tokenizer
model = T.AutoModelForSequenceClassification.from_pretrained( MODEL_NAME, num_labels=num_labels )
tokenizer = T.AutoTokenizer.from_pretrained( MODEL_NAME )
```

2. PEFT Validation (Task3, 10 pts):
    1. Evaluate your model when you are training. (10 pts)

SST-2(Lora)(只節錄最後的部分，紅色方框為 Accuracy):

24000	0.200000	0.277719	0.913991	0.916944
24500	0.193600	0.279096	0.915138	0.917960
25000	0.196800	0.276865	0.913991	0.916574

SST-2(Bitfit)(只節錄最後的部分，紅色方框為 Accuracy):

24000	0.283700	0.285326	0.888761	0.890643
24500	0.281300	0.286404	0.891055	0.893855
25000	0.294700	0.285603	0.888761	0.891134

SST-2(full-finetune) (只節錄最後的部分，紅色方框為 Accuracy):

24150	0.080200	0.357183	0.930046	0.930603
24200	0.082200	0.314532	0.935780	0.937500
24250	0.071000	0.316744	0.934633	0.936313

STSB(Lora) (只節錄最後的部分，紅色方框為 Pearson-Spearman Corr):

3150	0.713400	0.659183	0.862514
3300	0.672900	0.668659	0.860924
3450	0.704600	0.661946	0.862584

STSB(Bitfit) (只節錄最後的部分，紅色方框為 Pearson-Spearman Corr):

3150	2.091900	2.325861	0.493635
3300	2.006500	2.330186	0.496860
3450	2.057200	2.322372	0.498652

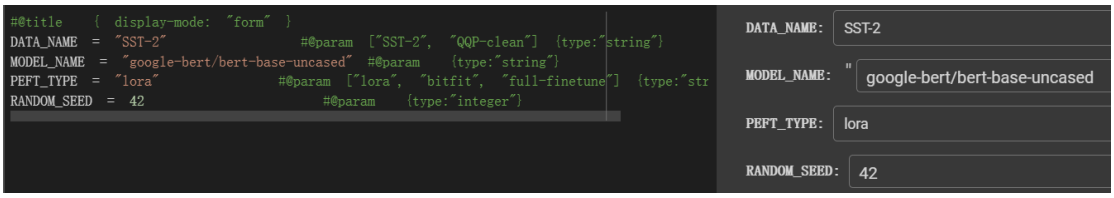
(這個跑出來的結果有點低，推測是因為資料較少所導致(為了加速有縮減訓練資料)，但整體結果是有持續上升的，Loss 也有持續下降。)

STSB(full-finetune) (只節錄最後的部分，紅色方框為 Pearson-Spearman Corr):

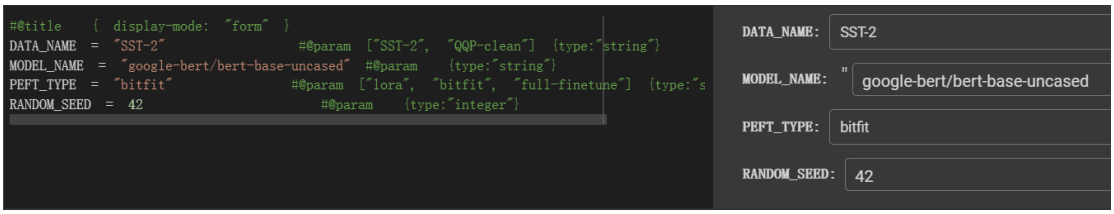
3150	0.079300	0.453216	0.905665
3300	0.076300	0.432045	0.906484
3450	0.071500	0.418188	0.906895

3. PEFT Models (Task2, 15 pts)
    1. Model design (5 pts). You can use any Pretrained-Models for your base model

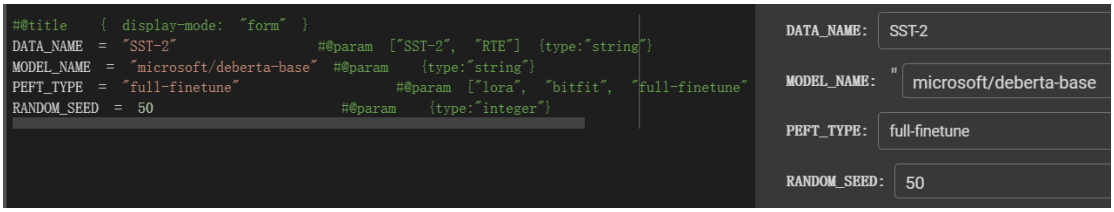
SST-2(Lora):



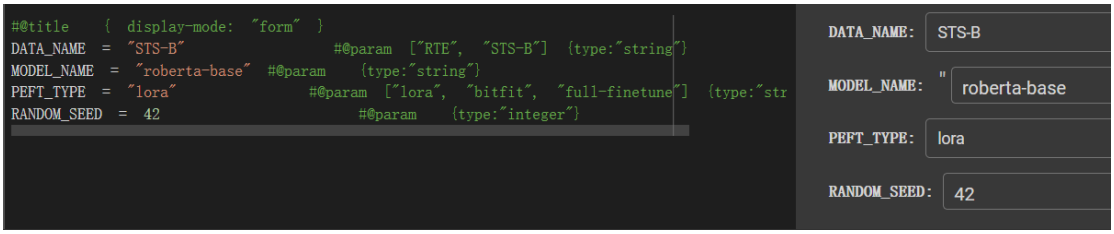
SST-2(Bitfit):



SST-2(full-finetune):



STSB(Lora):



STSB(Bitfit):

STSB(full-finetune):

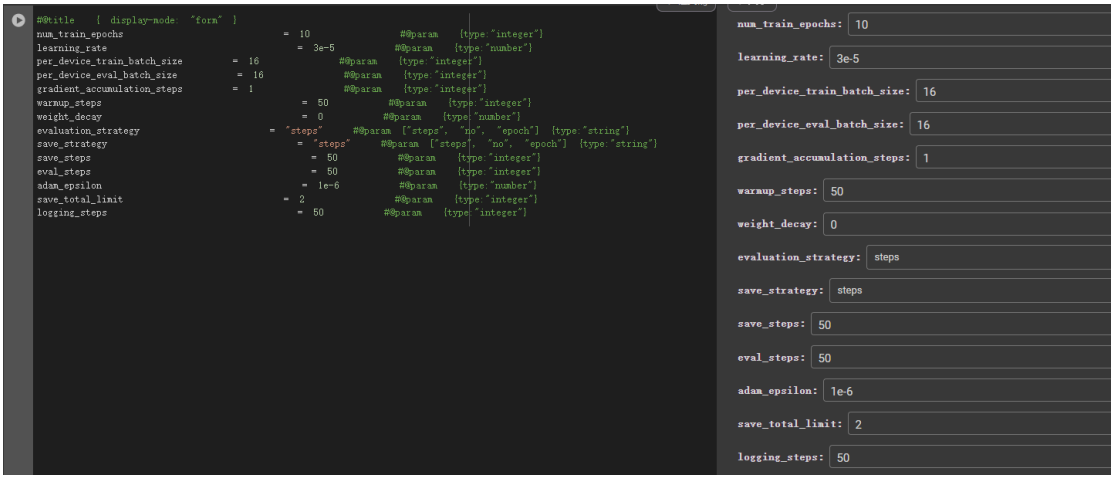
2. Train(finetune) the model (separately by PEFT methods) (10 pts). You should mark some of the parameter as trainable while others are untrainable.

各方法所使用的參數設定如圖所示:

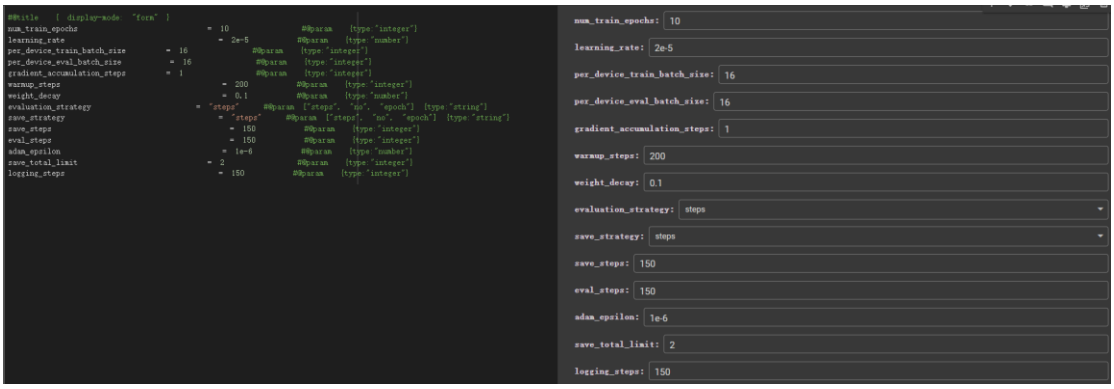
SST-2(Lora):

SST-2(Bitfit):

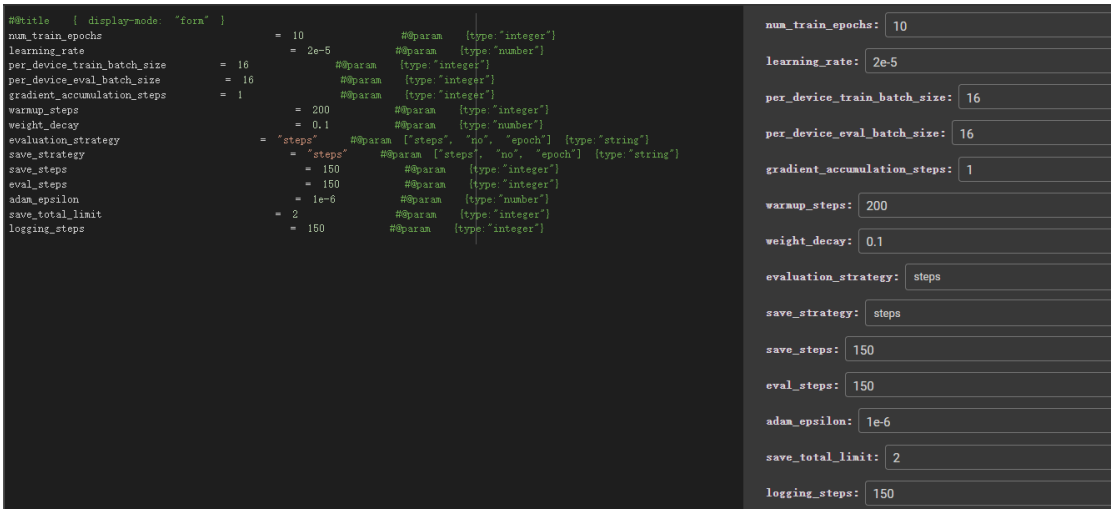
SST-2(full-finetune):



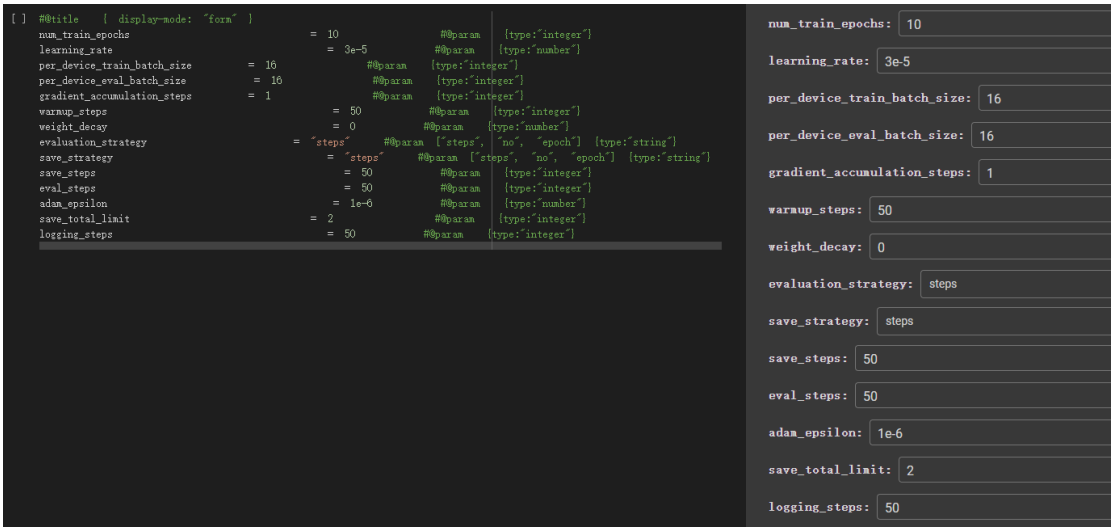
STSB(Lora):



STSB(Bitfit):



STSB(full-finetune):

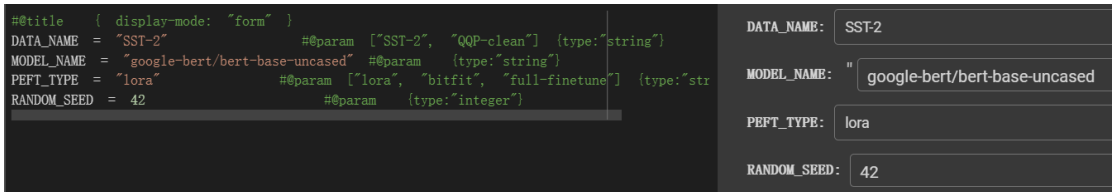


Analysis (45 pts)

1. Model analysis (5 pts)

   Include your model design and the process of loss reduction and the results of validation and testing (if available).

   由於其他檔案主要只是選用的 dataset 不同或是使用了不同的 PEFT method，以下我會針對 main.ipynb 來做介紹(SST-2, Lora):



   如上圖，我具體選用的模型是 google-bert/bert-base-uncased，dataset 為 SST-2，PEFT method 則是選用 Lora。

Step	Training Loss	Validation Loss	Accuracy	F1
23000	0.199300	0.278516	0.910550	0.913140
23500	0.210300	0.278457	0.912844	0.915556
24000	0.200000	0.277719	0.913991	0.916944
24500	0.193600	0.279096	0.915138	0.917960
25000	0.196800	0.276865	0.913991	0.916574

   在訓練過程中，模型的損失函數（Loss）會隨著訓練步數的增加而逐漸減少。以下是模型訓練的具體過程和結果：

   A. Loss Reduction：在訓練過程中，每個步驟都會計算訓練損失，

並根據設置的保存和評估步數進行中間結果的記錄。隨著訓練的進行，模型的損失值會逐漸下降，表明模型在不斷學習和優化。

B. Validation Results：在設定的評估步驟或 Epoch 結束後，模型會在驗證集上進行評估，計算準確率（Accuracy）和 F1 分數（F1 Score）。這些評估指標能幫助我們了解模型在未見數據上的表現，並防止過擬合。

dataset	metrics	baseline
CoLA	Matthew's Corr	0.6
SST2	Accuracy	0.88
MRPC	Accuracy	0.8
STSB	Pearson-Spearman Corr	0.8
QQP	F1 / Accuracy	0.8/0.8
MNLI_Matched	Accuracy	0.8
MNLI_Mismatched	Accuracy	0.8
QNLI	Accuracy	0.85
RTE	Accuracy	0.7
WNLI	Accuracy	0.8

由以上的這些結果表明，我採用的模型在驗證集上達到了不錯的準確率(0.91 左右，超過助教訂定的標準 0.88)和 F1 分數，展示了模型在此任務上的良好性能。

2. PEFT Discussion (25 pts)

What is the main feature of Bitfit? Why does it work? (10 pts) What are the best hyper-parameters(e.g. learning rate, batch size, warmup ...) you find? Comparing with full-finetuning, why do we need larger/smaller learning rate? (10 pts)

根據我查閱到的資料結果，Bitfit（BERT Itself Fine-Tuning）是一種針對預訓練語言模型（如 BERT）的微調方法，其主要特點在於只調整模型中的偏置參數，而保持其餘權重參數不變。這種方法的主要優點在於以下：

A. 參數高效：Bitfit 僅需要調整偏置參數，參數數量遠少於全模型微調，這大大減少了計算資源和存儲需求。

B. 簡化模型更新：由於只有偏置參數發生變化，模型更新過程更簡單且計算成本更低。

C. 減少過擬合風險：由於調整的參數較少，模型更不容易過擬合於特定的訓練數據集。

而 Bitfit 之所以有效，是因為偏置參數在模型的各層中對輸出有著顯著影響，通過調整這些參數，模型能夠適應新的任務要求，而無需大幅度改變所有權重參數。

另外，經過我多次實驗最終所使用的超參數為 learning rate = 1e-4, batch size = 8, warmup step = 50...(圖片可參考上方第三大題的第二小題)，

以上是我實測下來能跑出最佳結果的參數。

最後，和 full-finetuning 相比，Bitfit 通常需要較大的學習率來促進快速收斂，我找到的主要原因有以下兩點:

A. 參數更新範圍小：Bitfit 只更新偏置參數，這些參數量較少，因此可以承受較大的更新步長，從而加速訓練過程。
B. 收斂速度需求：由於參數更新範圍有限，需要較大的學習率來確保模型能夠在合理的訓練時間內達到收斂。

3. PEFT Comparison (25 pts)
Compare the performance of Bitfit with Lora on the two datasets you select. (10 pts) How does the rank(r) affect LoRA's parameter count and performance? (10 pts)

(1).在這個實驗中，我比較了 Bitfit 和 LoRA 兩種微調方法在 SST-2 及 STSB 數據集上的性能表現。為了進行這個比較，我採用了相同的訓練設置（如訓練週期數、學習率、批次大小等），並在驗證集上評估它們的準確率和 F1 分數。

結果如下(紅色方框前者為 Accuracy，後者為 F1 score):

- SST-2(Lora): Accuracy = 0.91 , F1 score = 0.92

- SST-2(Bitfit): Accuracy = 0.88 , F1 score = 0.89

從結果可以看出，LoRA 在驗證準確率和 F1 分數上都略微優於 Bitfit。這表明 LoRA 能夠更好地捕捉到數據中的模式，並且在情感分類任務中具有更高的性能表現。

- STSB(Lora):

- STSB(Bitfit):

同樣地，STSB 也有差不多類似的效果，只是由於我使用較少的訓練資料，所以造成結果差異較大，推測若使用完完整資料訓練，則會更趨近於 SST-2 dataset 所呈現的狀況。

(2). LoRA（Low-Rank Adaptation）是一種將權重矩陣分解為兩個低秩矩陣的技術，其目的是減少模型參數數量，同時保持較高的性能；而 Rank

(r) 係指這兩個低秩矩陣的秩（或列數），它會直接影響 LoRA 的參數數量和性能，以下我將針對兩點來個別作解釋:

A. 影響參數數量

LoRA 通過將全尺寸的權重矩陣分解為兩個低秩矩陣，顯著減少了參數數量。具體地說，假設原始權重矩陣的尺寸為 d×d，則其參數數量為 d^2。通過 LoRA 分解後，兩個低秩矩陣的尺寸分別為 d×r 和 r×d，其參數總數為 2dr。當 Rank (r) 增加時，參數數量也會增加，具體表現為：

- ✧ r 小：參數數量減少，但可能限制了模型的表達能力。
- ✧ r 大：參數數量增加，模型的表達能力增強，但計算和存儲成本也會相應增加。

B. 影響性能

Rank (r) 對模型性能的影響如下：

- ✧ 低 Rank (r)：
  - ✓ 優點：顯著減少參數數量，降低計算和存儲成本。
  - ✓ 缺點：過低的 Rank 可能導致模型表達能力不足，從而降低性能。
- ✧ 高 Rank (r)：
  - ✓ 優點：提高模型的表達能力，可能提升性能。
  - ✓ 缺點：參數數量增加，導致更高的計算和存儲成本。