1. Model analysis

本次作業三我所選擇的模型是 RoBERTa，他是一種基於 BERT 的自然語言處理模型。它通過對 BERT 模型進行一系列優化和調整，以達到更好的性能。

RoBERTa 移除了 BERT 中的下一句預測任務（Next Sentence Prediction），因為研究發現這個任務對模型性能的提升幫助有限。相反，RoBERTa 只使用了 Masked Language Model 任務進行訓練。此外，RoBERTa 訓練了更長的時間和更多的步數，使模型能夠更好地捕捉語言特徵。

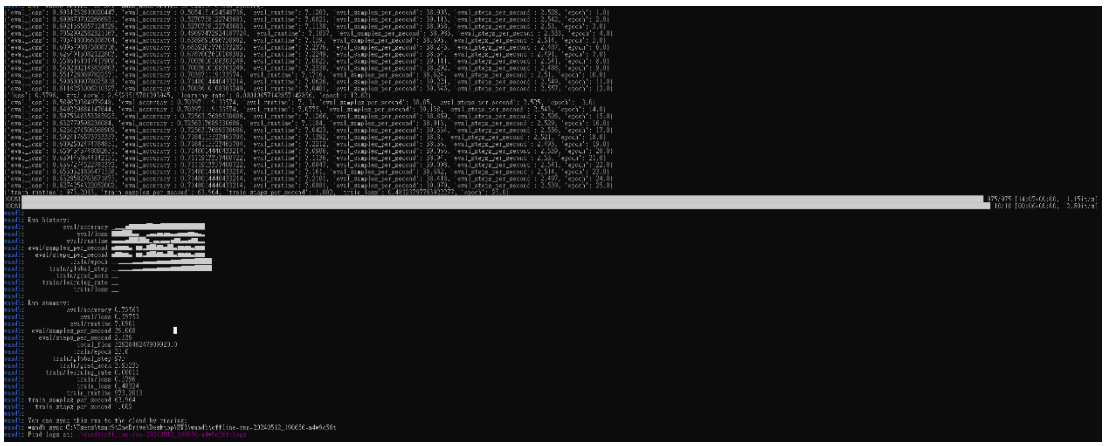
在程式中我使用 transformers.AutoModelForSequenceClassification.from_pretrained() 函式並將 num_labels 設置為 2 分類之後 load 模型進來，其後便使用 trainer 函式將模型與資料集放進去訓練，這個函式透過反向傳播和梯度下降算法來優化模型參數，並在每個訓練周期結束時計算驗證集上的性能指標，如準確率、損失等，最後保存訓練好的模型。其中 optimizer 我使用 default 的 AdamW。

Rte dataset_bitfit:

Acc: 0.74007

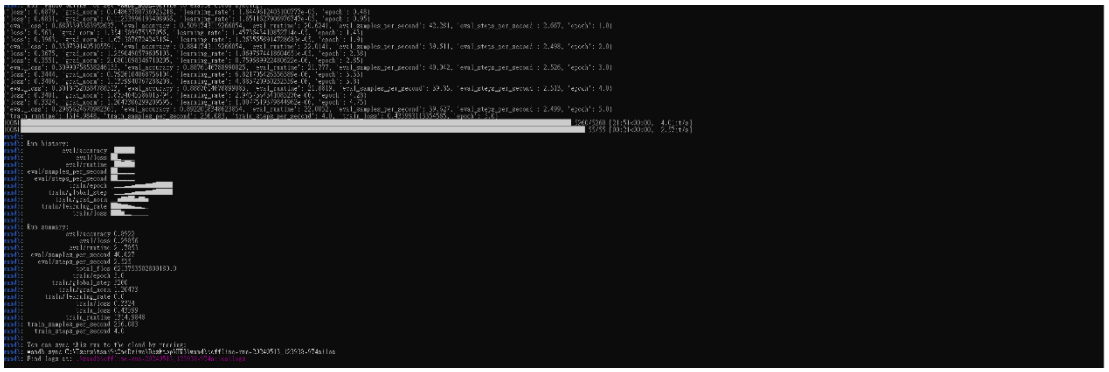
Rte dataset_lora:

Acc:0.72563



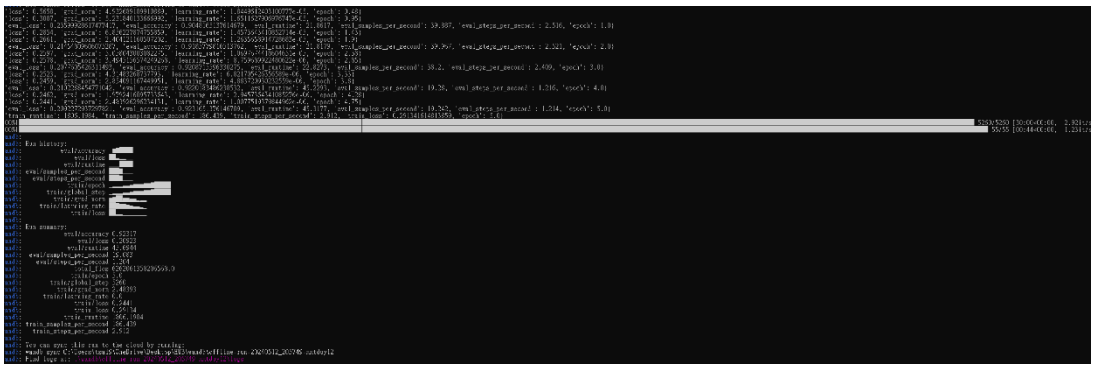
Sst2 dataset_bitfit:

Acc:0.8922



Sst2 dataset_lora:

Acc:0.92317



2. PEFT Discussion

對於 transformer 模型而言，Bitfit 凍結大部分模型參數，只訓練更新 bias 參數跟特定任務的分類層參數。

Bias 的參數舉例來說有:

1). MLP 層中的 bias

2). attention 模塊中計算 query、key、value 跟合併多個 attention 結果的涉及到的 bias

3).Layer normalization 層的 bias 參數

這些 bias 參數佔部分模型，像是 bert 模型全部參數量的 0.08%左右。

他透過只更新部分的 bias 參數與特定任務的分類層參數，來加強模型對於特定下游任務的識別，因此能夠在不訓練大量參數的條件下，增加模型的準確率。

我所找到的最好的超參數是:

Rte dataset:

```
args = transformers.TrainingArguments(
    output_dir= "/save",
    num_train_epochs= 25,
    learning_rate= 5e-4,
    per_device_train_batch_size= 16,
    per_device_eval_batch_size= 16,
    gradient_accumulation_steps= 4,
    warmup_steps= 100,
    weight_decay= 0.01,
    evaluation_strategy= "epoch",
    save_strategy= "epoch",
    save_steps= 100,
    eval_steps= 100,
```

```
        load_best_model_at_end=True,
        metric_for_best_model="accuracy",
)
lora_config = LoraConfig(
        task_type=TaskType.SEQ_CLS, r=2, lora_alpha=8, lora_dropout=0.01
)
```

由於 Rte dataset 較小，因此我使用更多的 epoch 來增加其訓練量

Sst2 dataset:

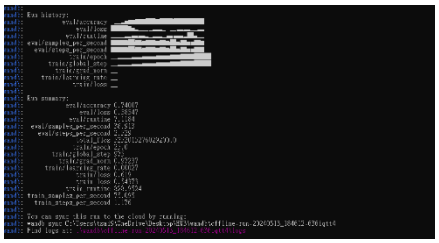
```
args = transformers.TrainingArguments(
        output_dir= "/save",
        num_train_epochs= 5,
        learning_rate= 2e-5,
        per_device_train_batch_size= 16,
        per_device_eval_batch_size= 16,
        gradient_accumulation_steps= 4,
        warmup_steps= 100,
        weight_decay= 0.01,
        evaluation_strategy= "epoch",
        save_strategy= "epoch",
        save_steps= 100,
        eval_steps= 100,
        load_best_model_at_end=True,
        metric_for_best_model="accuracy",
)
```

```
lora_config = LoraConfig(
    task_type=TaskType.SEQ_CLS, r=2, lora_alpha=8, lora_dropout=0.01
)
```

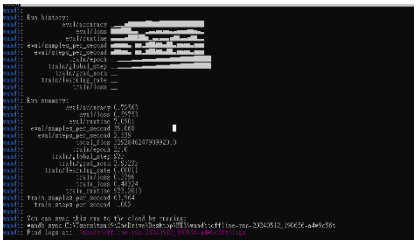
在 full-finetuning 中，我們更新整個模型的所有參數，因此通常需要較小的學習率，以免破壞先前學習到的知識。但是，在 peft，我們只更新模型的一部分參數，通常是最後幾層或某些層的參數。這些層可能具有不同的特徵表示和學習速度，因此可能需要調整學習率以更好地適應它們的更新。如果某些層的參數需要更大的變化，那麼可能需要較大的學習率，而對於其他層，可能需要較小的學習率以避免過度調整。因此，根據部分微調所選擇的層和任務的特定要求，需要對學習率進行調整以獲得最佳的性能。

3. PEFT Comparison

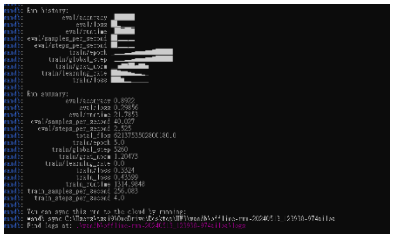
Rte dataset_bitfit:



Rte dataset_lora:



Sst2 dataset_bitfit:



sst2 dataset_lora:

我們可以發現，不論在哪個 dataset，所測出來的 Accuracy 都是差不多的，我認為這有可能是因為 SST-2 與 Rte 數據集可能相對簡單，對於這些不同的模型來說，它們的能力可能足以應對這個任務，因此沒有看到明顯的性能差異。

Lora rank 大小對於模型訓練的影響有:

較低的 rank 可能會導致模型對數據的擬合不足，雖然可能會提高模型的泛化能力，但模型可能無法捕捉數據中的重要特徵，從而降低了模型的性能，parameter count 會減少，而較高的 rank 可能會導致過擬合，還會增加模型的計算複雜度，因為它涉及更多的參數和更大的矩陣。這可能會導致訓練時間增加，特別是對於大規模數據集和複雜模型，模型可能會過度擬合訓練數據，導致在未見過的數據上性能下降，parameter count 增加。