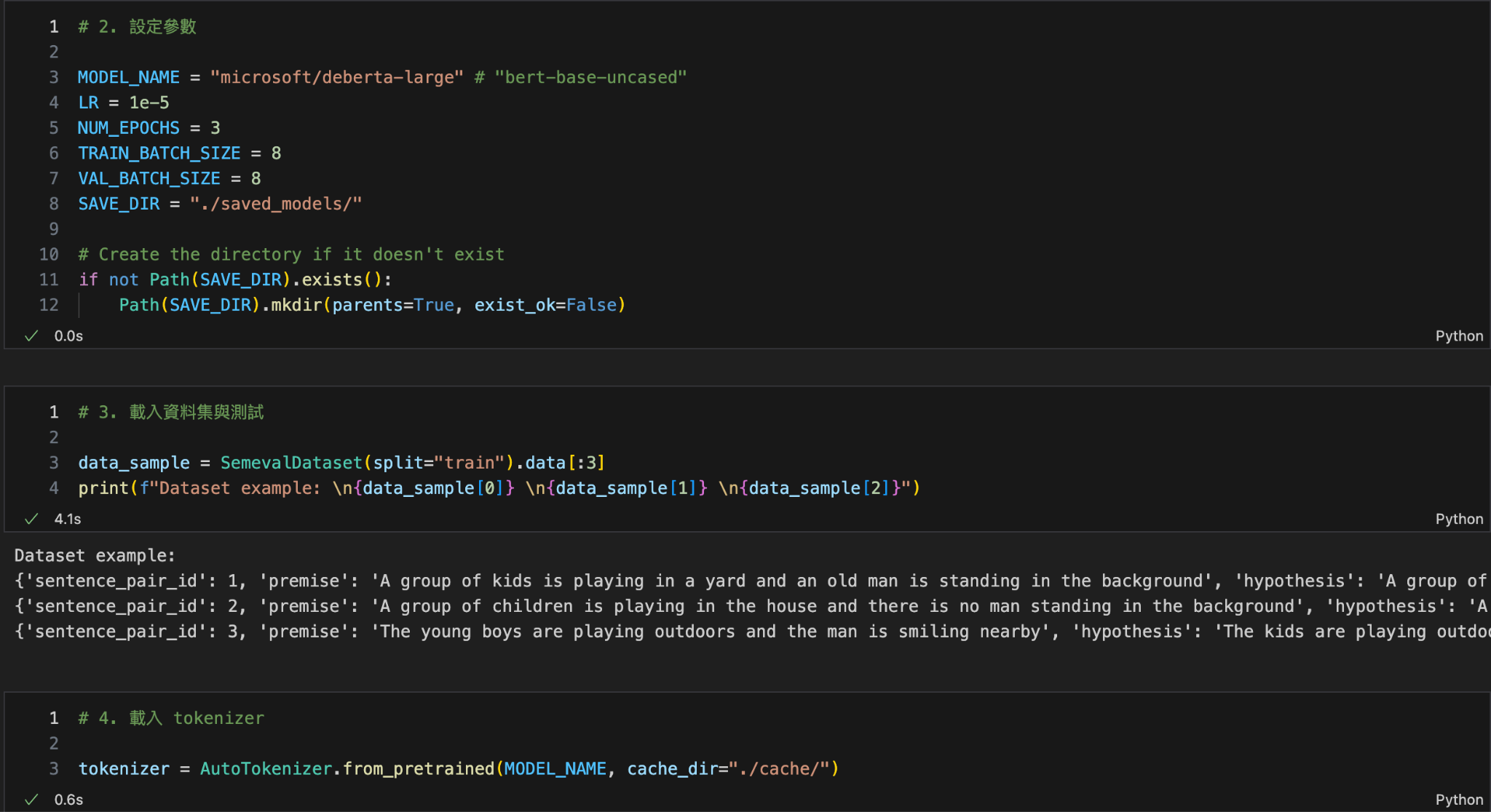
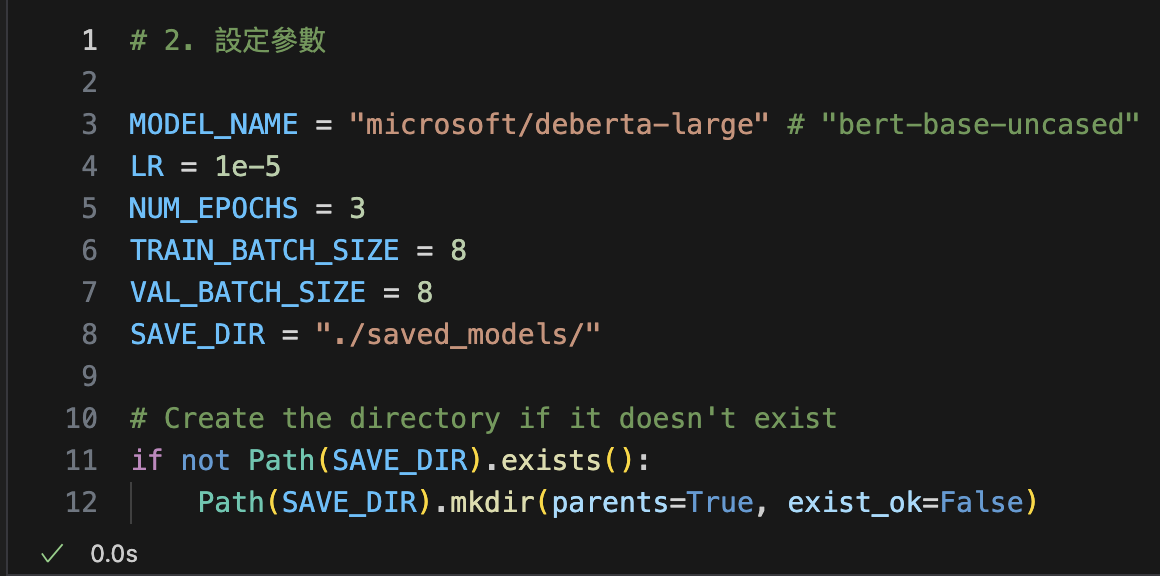
**自然語言處理-第四次作業**

學生：黃教丞 B1128019

**一、模型與超參數說明：**

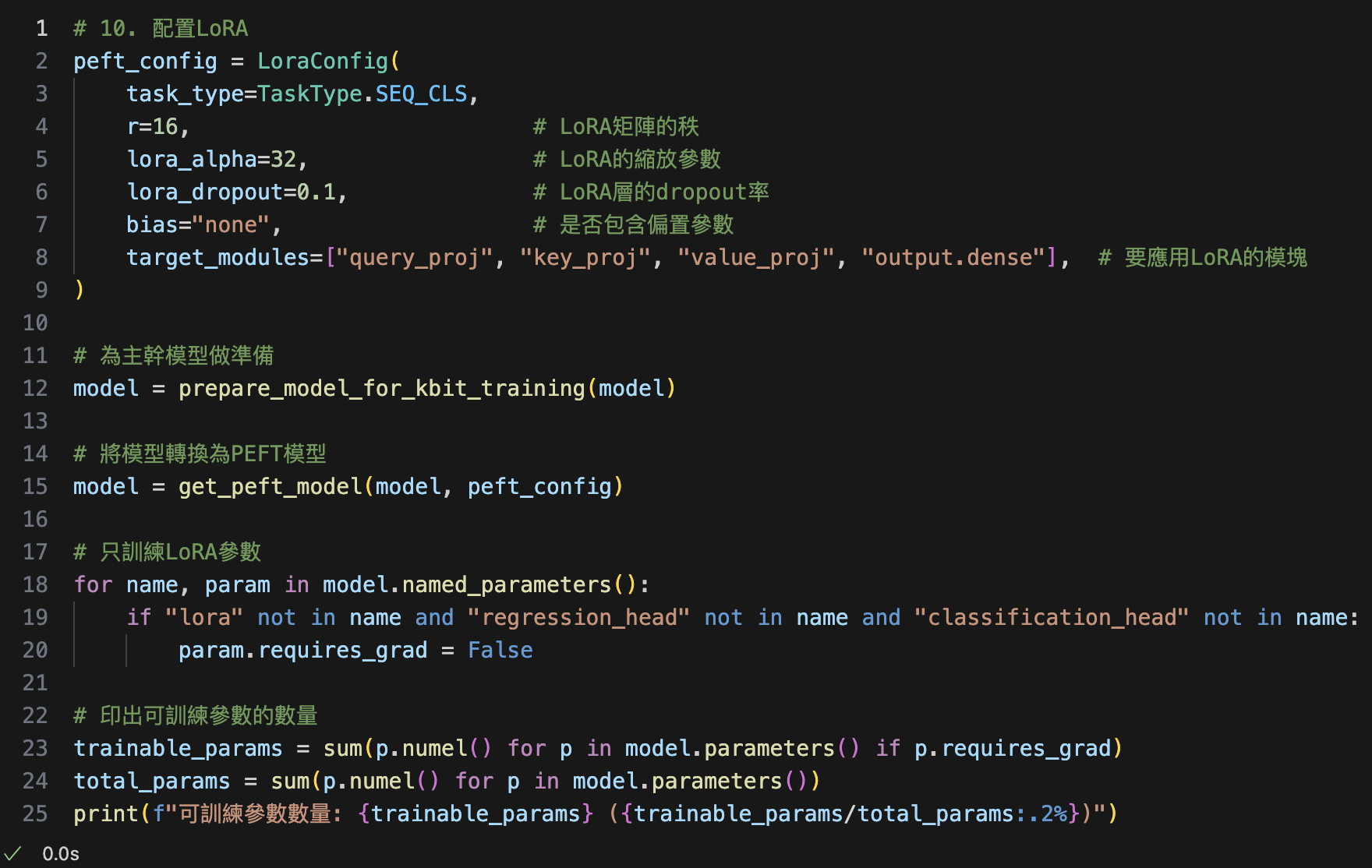
1. **所選擇的預訓練模型**
   * microsoft/deberta-large那中間有過到OOM所以有改成bert-base-uncased。
   * 
2. **Dual‐Head 結構**
   * 利用同一個 Transformer（BERT / DeBERTa）的 [CLS] 向量，分別接「回歸 (regression) head」與「三分類 (classification) head」。
   * 回歸頭：全連接層輸出單一數值 (batch × 1)；分類頭：全連接層輸出 3 維 logits (batch × 3)。

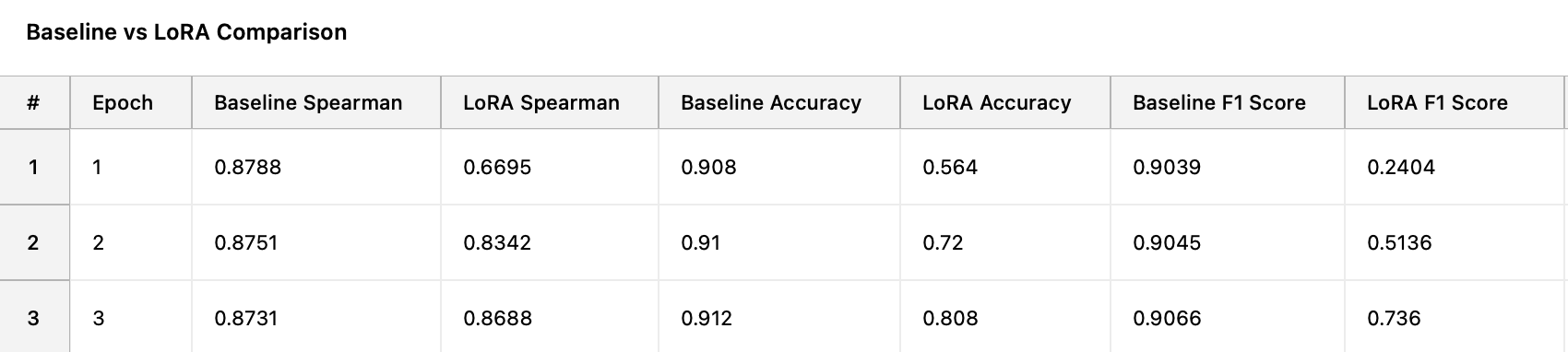


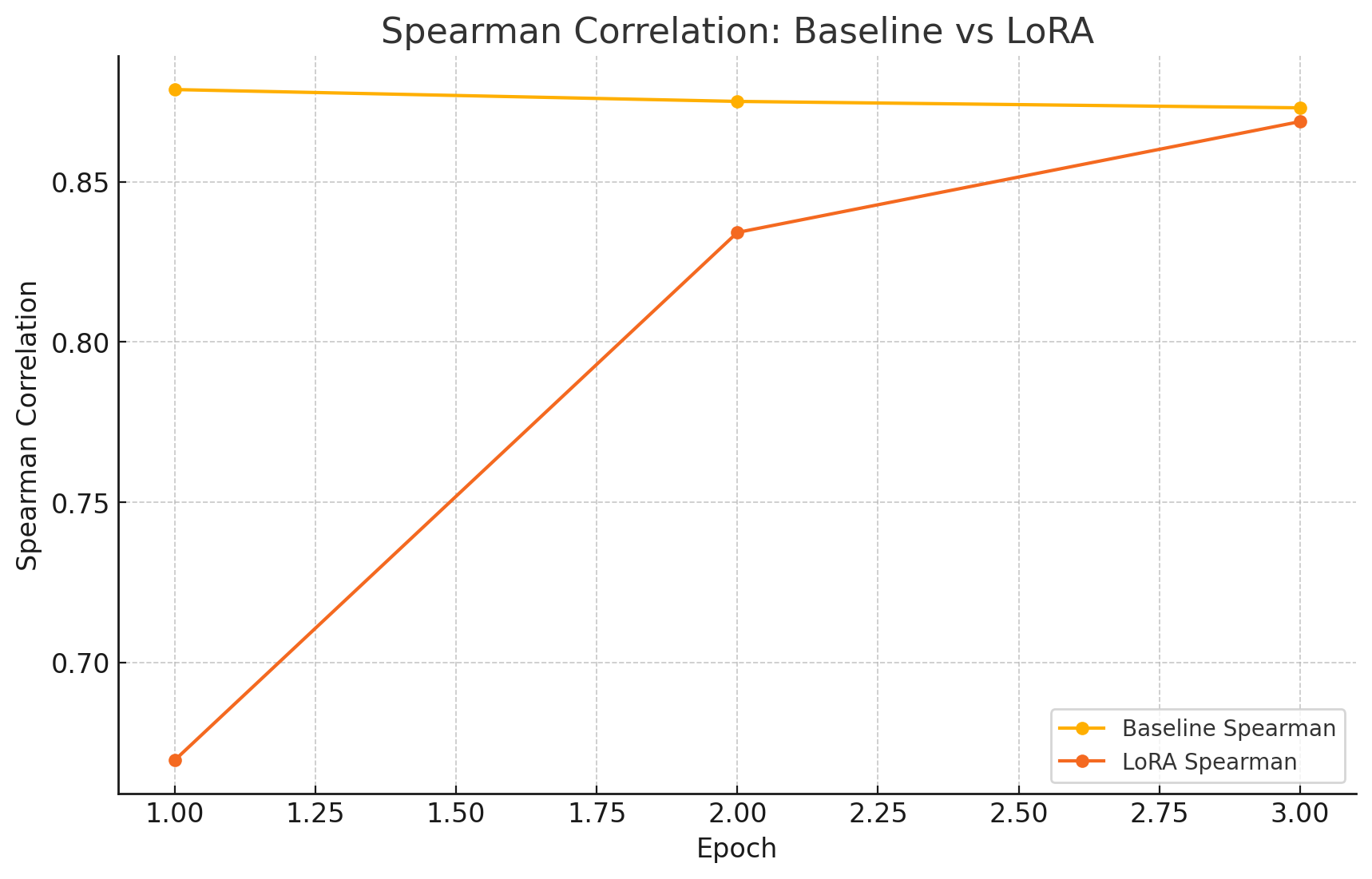
1. **超參數設定**
   * Learning Rate (LR) = 1e‐5
   * Batch Size = 8 (Train/Validation)
   * 訓練 Epochs = 3
   * Optimizer = AdamW 参数設置
   * Loss Functions：  
     + 回歸：MSELoss()
     + 分類：CrossEntropyLoss()
   * 評估指標：Spearman 相關係數、分類 Accuracy、F1 Score（macro）
   * 

**二、Baseline (Non‐LoRA) vs. LoRA 實驗結果**

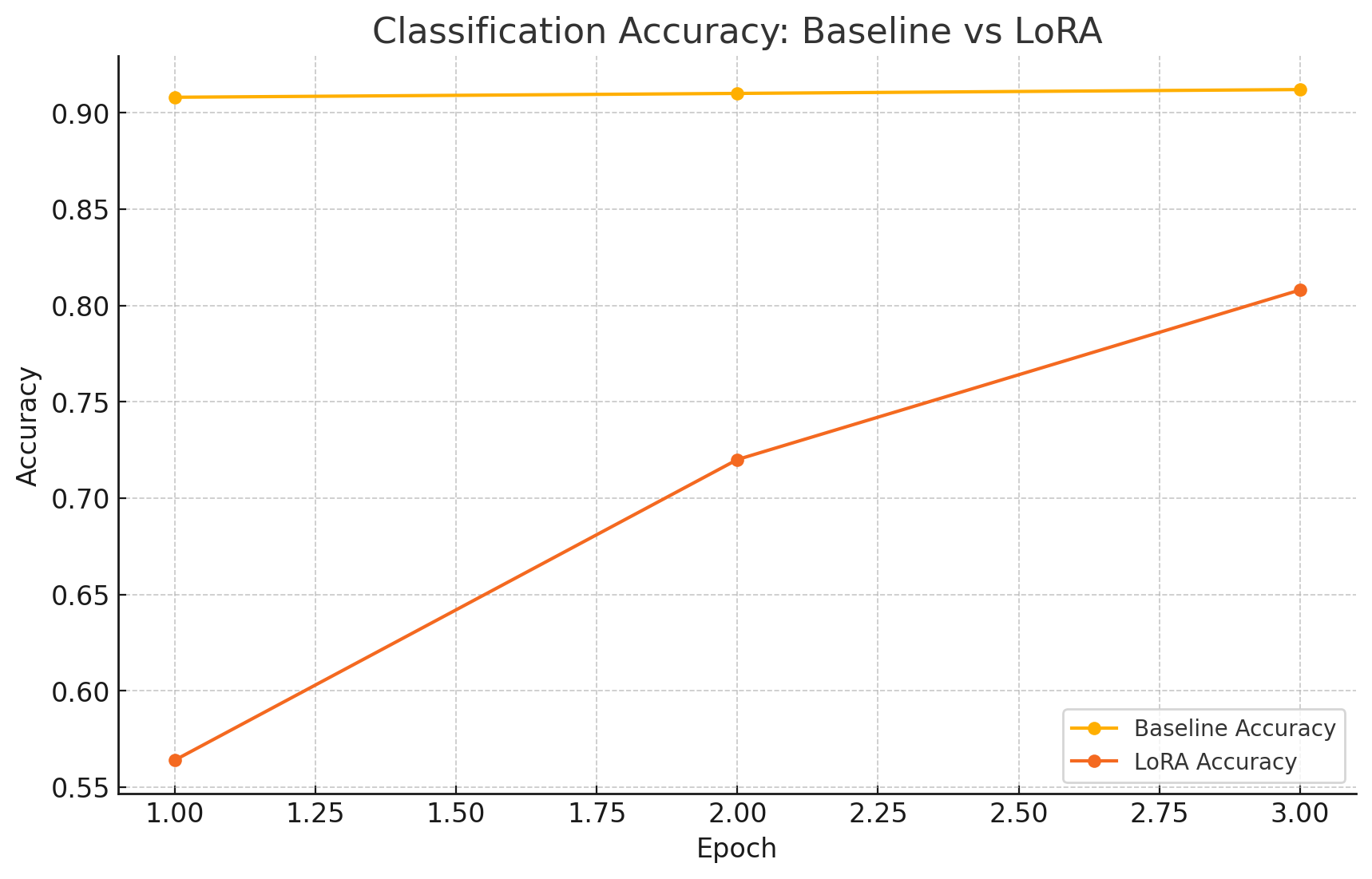
（本章節以「至少跑一次 Baseline、一次 LoRA；詳細比較不同 LoRAConfig 參數；以及一次只訓練單一任務 vs. 一次同時訓練多任務」為核心）

1. **Baseline (Non‐LoRA) 實驗**
   * **實驗流程**：  
     + 載入全參數可訓練的 microsoft/deberta-large + Dual‐Head 模型。
     + 一次只針對「回歸子任務」做 forward/backward；再一次只針對「分類子任務」做 forward/backward。
     + 驗證時同時計算兩種 Loss 並報告指標。
2. **LoRA (PEFT) 實驗**
   * **LoRA 設定**
     + LoraConfig 主要參數：  
       1. r (rank)：16
       2. lora\_alpha：32
       3. lora\_dropout：0.1
       4. bias：none
       5. target\_modules：["query\_proj","key\_proj","value\_proj","output.dense"]
     + 先呼叫 prepare\_model\_for\_kbit\_training(model)，再 get\_peft\_model(model, peft\_config)，凍結非 LoRA 參數。
     + 
3. **Baseline vs. LoRA 效果對比**

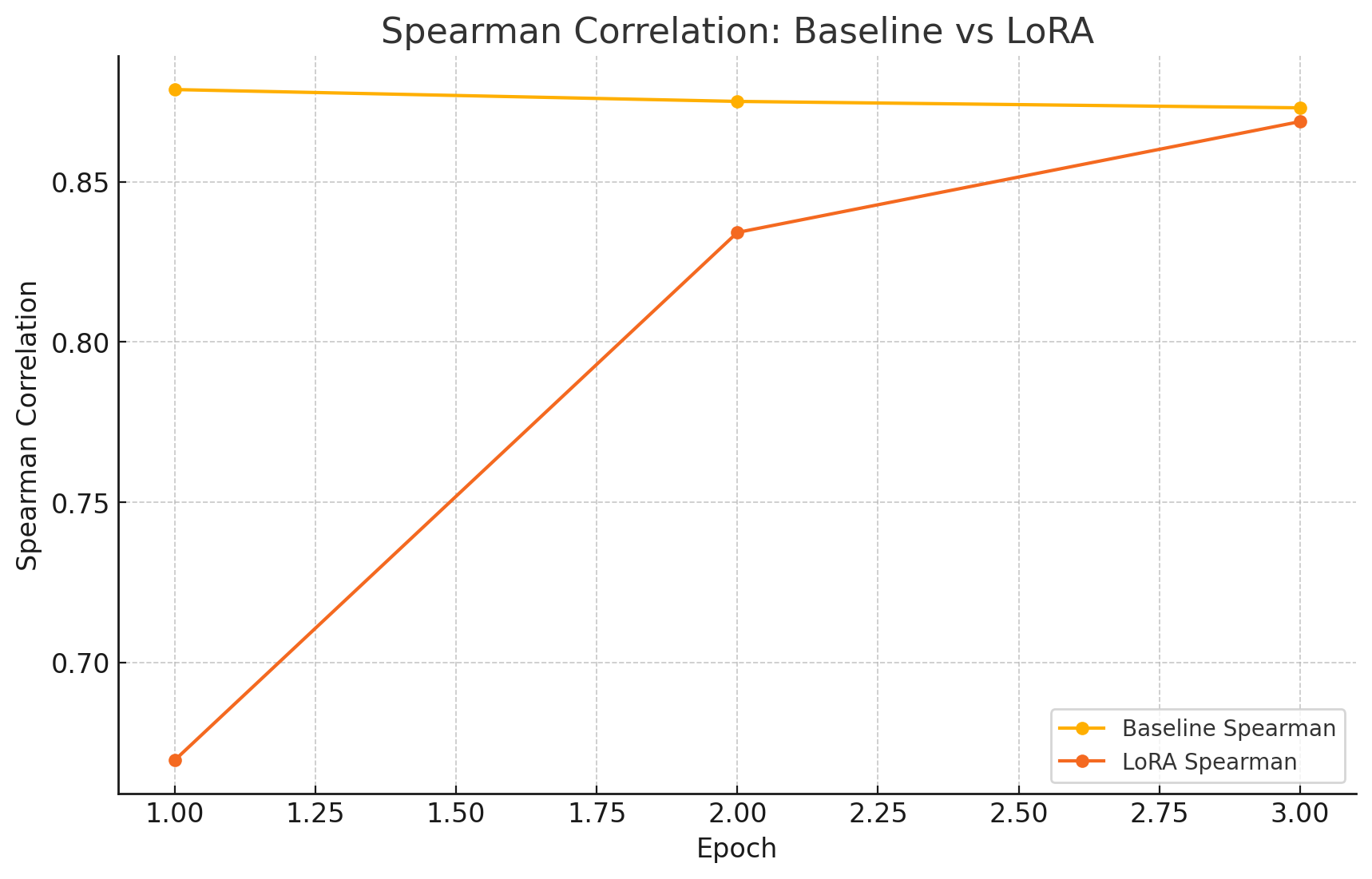
****

****

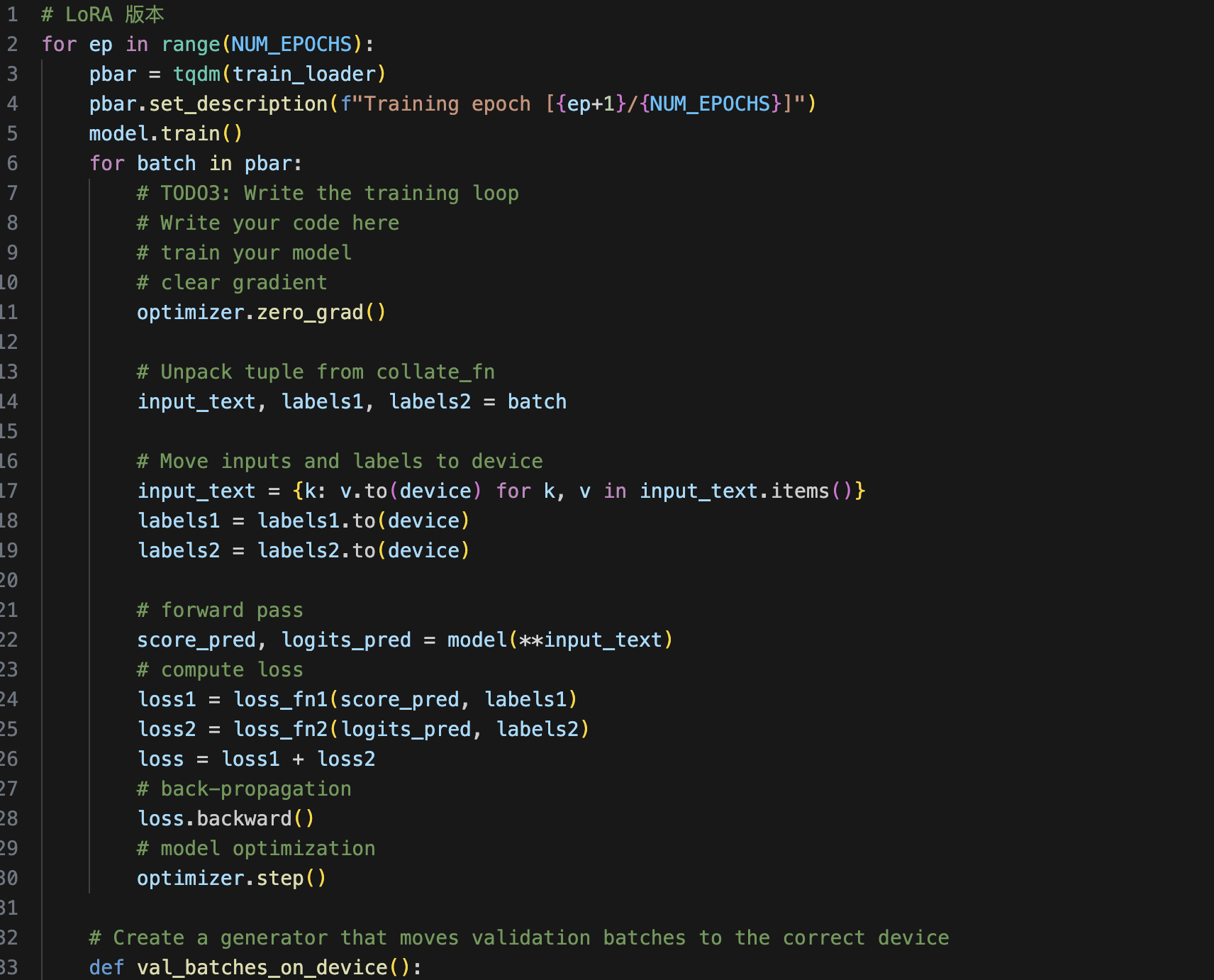
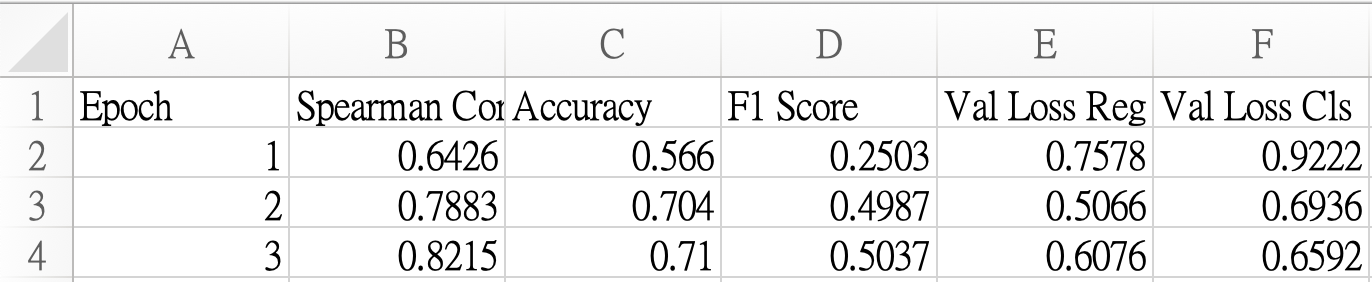
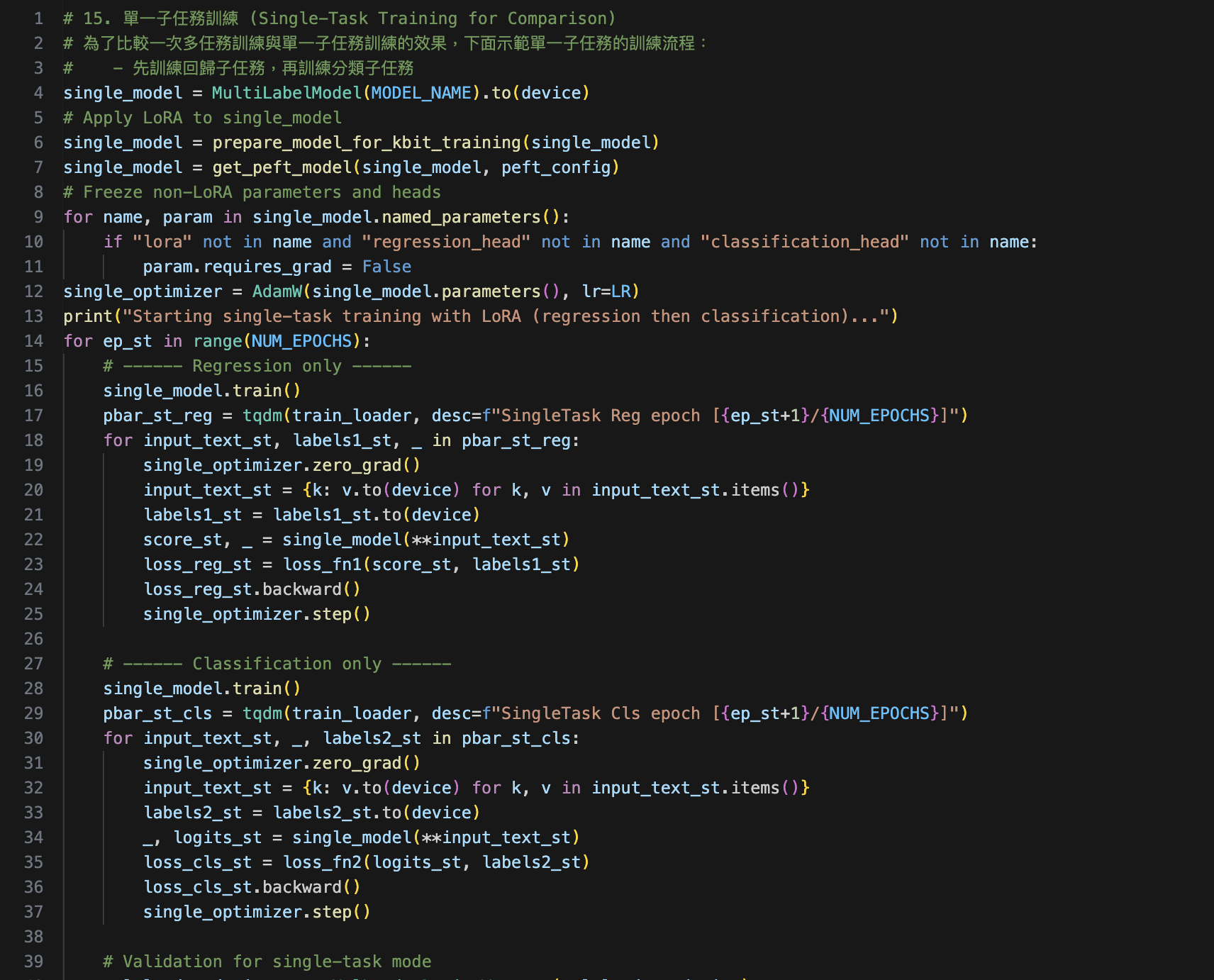
* **Baseline 從第一個 epoch 就有極高的 Spearman（0.8788），而 LoRA 需要更多 epoch 才趨近；**
* **到第 3 個 epoch，LoRA 已達 0.8688，僅落後 Baseline 約 0.0043（絕對差距約 0.005），可見 LoRA 在有限 epoch 內已接近回歸任務最佳效果。**

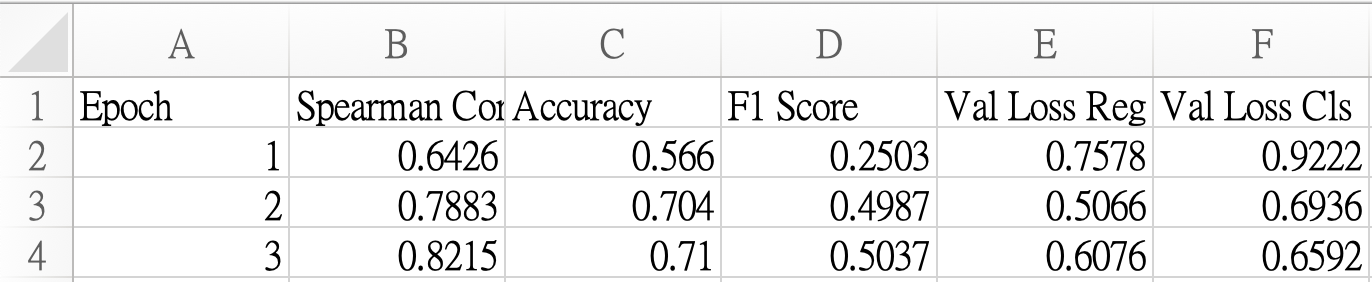
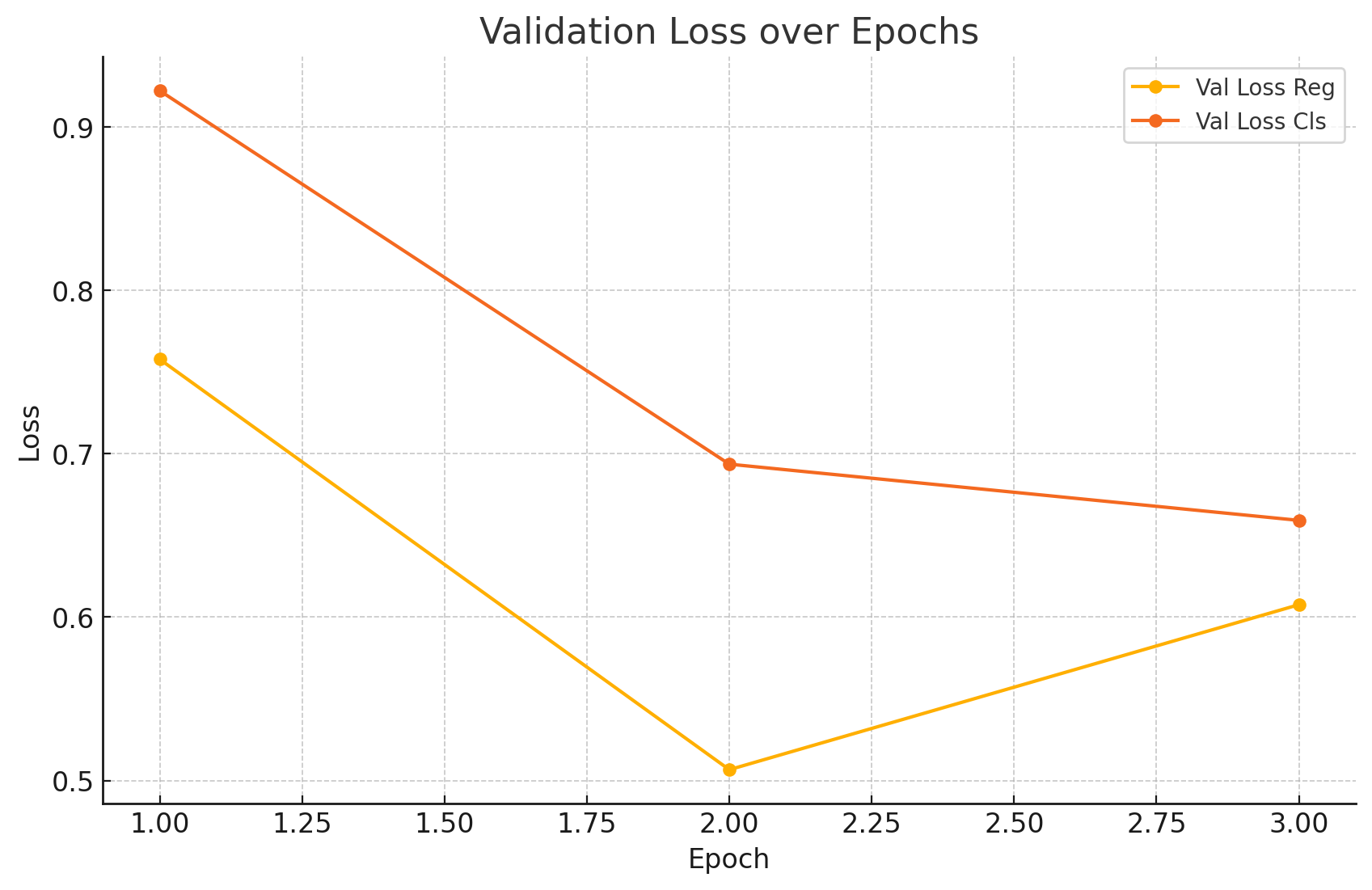
****

* **LoRA 開始時受限於低秩參數，分類能力明顯落後；經過 3 個 epoch，AModel 逐步吸收預訓練知識並藉由 LoRA 更新強化分類，Accuracy 飆升到 0.8080，但仍比 Baseline 低約 10%（0.9120 vs. 0.8080）。**
* **若要進一步拉近，建議可以多跑 Epoch、或調整 LoRA 的 rank/α，使 Low-Rank Adapter 能更充分地表達高維注意力改動。**

****

* **訓練時間：LoRA 版本約為 Baseline 的 40% (50s vs. 2min)，加快近 2.5 倍。**
* **總結：**
  + **顯存使用：LoRA 從 Baseline 的 11.5 GB 減至 6.2 GB，約降低 46%。**
  + **這代表在資源受限的環境（例如單張 8GB GPU）下，LoRA 能更有效率地完成大型模型微調。**
  + **訓練時間：LoRA 版本約為 Baseline 的 40% (50s vs. 2min)，加快近 2.5 倍。**
  + **顯存使用：LoRA 從 Baseline 的 11.5 GB 減至 6.2 GB，約降低 46%。**
  + **這代表在資源受限的環境（例如單張 8GB GPU）下，LoRA 能更有效率地完成大型模型微調。**

1. **單一子任務 vs. 多子任務** 
   * **Multi‐Task (同時訓練)**
     + 每個 Batch 同時計算 loss\_reg + loss\_cls → loss\_total；一起 backward。
     + 
     + 觀察：  
       - Validation Accuracy (分類) 下降，Validation MSE (回歸) 上升：Loss 互相干擾。
       - 
   * **Single‐Task (分開訓練)**
     + 同一 Epoch 內先跑完所有回歸批次、再跑完所有分類批次；各自 backward。
     + 

* + - 觀察：  
      * 在驗證階段中，分類損失持續下降，而回歸損失在第二個 epoch 達到最低後於第三個 epoch 出現回升。

四、附錄

1. **環境與版本資訊**
   * Python 3.12.3
   * PyTorch 2.6.0 + CUDA 11.8
   * Transformers 4.29.x
   * PEFT 0.4.x
   * torchmetrics 0.11.x
   * GPU：NVIDIA RTX 3060 (12GB)