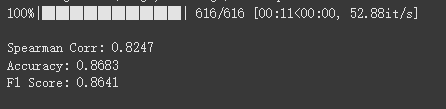
Assignment 3: Multi-output learning

NYCU\_313581016

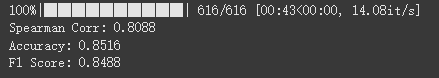
1. Which (pre-trained) model do you use? Why to choose the model?

(5%)

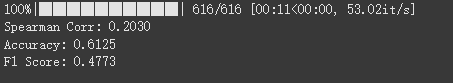
bert-base-uncased



bert-large-uncased



distilbert-base-uncased

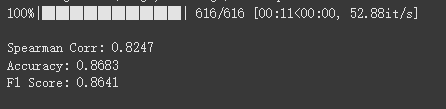


說明：我共嘗試了三種模型，首先我判斷uncased 與 cased 的選擇應該不會讓這次作業的結果更好，因此我選擇了 uncased，並測試三種不同的模型，根據測試結果發現 bert-large-uncased 需要跑最久，但針對 test dataset 的效果還不錯，而bert-base-uncased、distilbert-base-uncased兩者時間大約只需要 1/3 ，且bert-base-uncased 效果跟 bert-large-uncased 差不多，有時候甚至更好，因此選擇 bert-base-uncased 作為這次作業使用的模型。

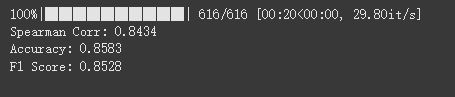
1. Compared with models trained separately on each of the sub-task,

does multi-output learning improve the performance? (8%)

multi-output learning



models trained separately



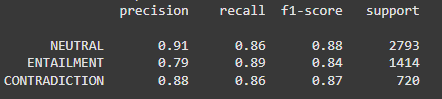
說明：以上是我使用 multi-output learning 與 models trained separately

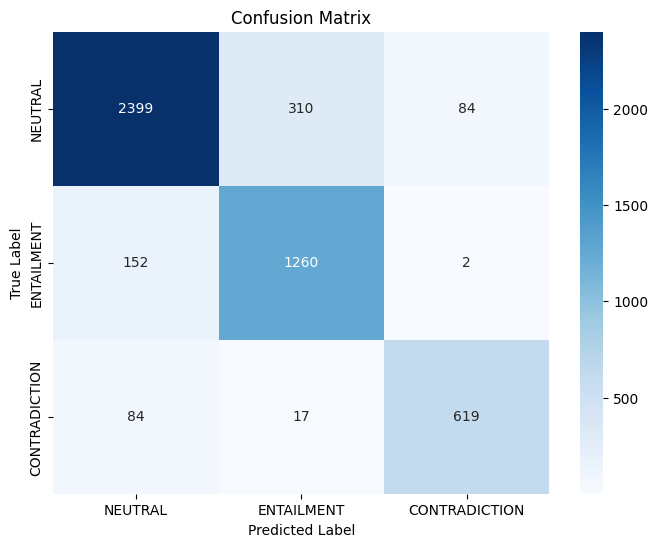
實作的結果，在這邊可以看到因為第二種方法的 model 要分開預測，所以所需的時間是第一種方式的兩倍。從 test dataset 的結果來看， 在 regression 的部分是從0.8247增加到了 0.8434，大約增加了 0.2 左右，而就 classification 來看，Accuracy 和 F1 Score 都有所下降，大約下降了 0.1 左右， 就這次的結果來說，無法判定哪一種方式比較好，不過我想在任務之間有相關性的情況下，或許multi-output learning 可以通過共享資訊來提升整體效能，此外，同時訓練多個任務可以減少計算資源的浪費，也可能提高模型的推論效率。

1. Why does your model fail to correctly predict some data points?

Please provide an error analysis. (8%)

Classification :





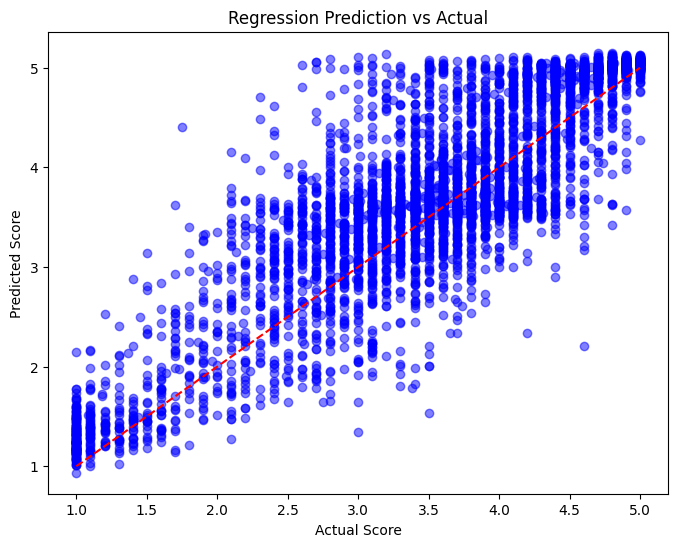
Number of Misclassified Samples: 649

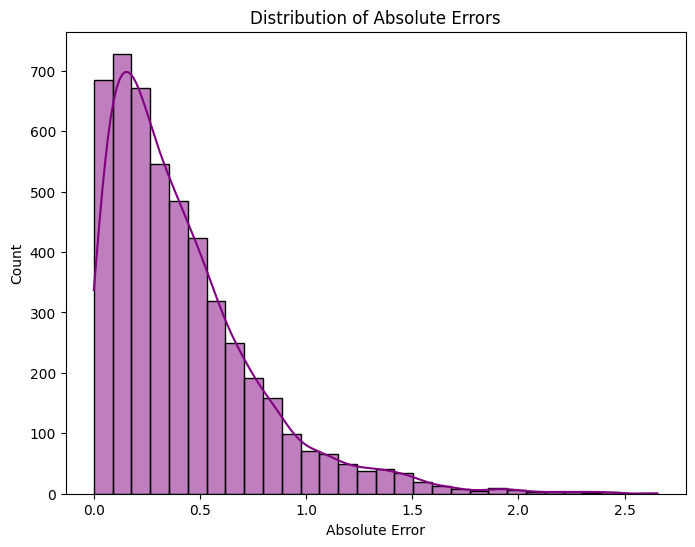
說明：首先可以看到 ENTAILMENT 的 precision 是較低的 ，但 recall 最高的 ，我們可以進一步的看到 confusion matrix ，有這樣的結果是因為 NEUTRAL 和 ENTAILMENT 較容易預測錯誤，尤其是實際為NEUTRAL 但被預測成 ENTAILMENT 的情況有 310 個 samples 是預測錯誤情況中最多的，模型將一些 NEUTRAL 或 CONTRADICTION 的樣本誤分類為 ENTAILMENT，也就造成了此數值，進而導致 ENTAILMENT 的F1-score 也是最低。

例外，還可以討論的是 CONTRADICTION 雖然 data的數量最少，但其訓練的成果算是不錯的，並沒有因為 imbalance 的問題導致 precision 和 recall 很低，

因此我沒有再針對 imbalance 做處理，這也考量到本身的data 數不多，做 undersampling 的會可能會導致模型效果很差。

Regression：





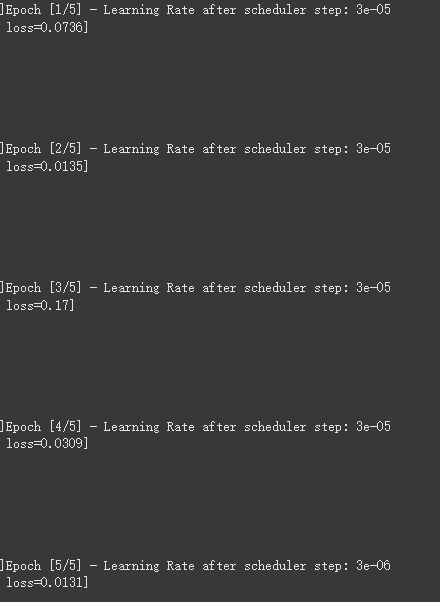
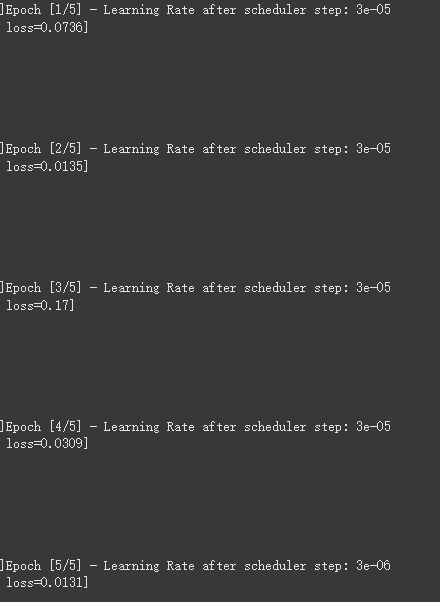
說明：我計算了MAE: 0.4172，Max Error: 2.6550 ，如果 MAE 很低，表示大多數樣本的預測結果接近實際值，而Max Error表示單個樣本的最大預測誤差，雖然沒有參考值，但我個人評估這樣的數值是可以接受的。

以上看到兩張分布圖，第一張散點圖展示每個樣本的實際值和預測值的對應關係，理想情況下，所有點應該接近紅色虛線，可以發現散點圖顯示大部分點接近紅線，但有一些偏離較遠的點。

第二張圖則是誤差分佈圖，理想情況下，誤差應集中在較小區間，可以看到其效果還不錯。

1. How do you improve your model performance? (9%)

針對模型優化部分，我修改了 Hyperparameter，epochs 增加到 5 ，並加入scheduler = ReduceLROnPlateau(optimizer, mode='min', factor=0.1, patience=1)調整 lr，因為 epoch 的數量較少，所以我的 patience=1 ，希望可以即時的更改 lr ，讓收斂更快，可以看到下圖 lr 從原本的 3e-5 變成 3e-6。



1. Anything that can strengthen your report. (10%)

* Loss function

除了 MSE、CrossEntropy 外，我也有嘗試 Huber Loss、L1 Loss等 loss function，但根據我測試的結果，通常 Huber Loss 和 MSE 的效果比較好，尤其是Huber Loss，雖然本身的 regression 部分數值差異不大，但 classification

的數值有明顯提升。



而L1 Loss效果明顯比較差，因此選用 MSE，而 CrossEntropy 因為是三個類別 所以就不另外測試其他的 loss function。

* Loss 計算



我有嘗試使用以上方法來計算 total loss 但最後呈現的效果跟原本的差不多，因此沒有使用。

附錄

|  |  |
| --- | --- |
| Environment | |
| Running environment | Colab |
| Python version | Colab |
| GPU(s) you used | T4 |

* 使用 GPT 修改 SpearmanCorrCoef, Accuracy, F1Score的相關程式碼，並添加註解。