**自然語言處理**

**Natural Language Processing**

**Assignment 3: Multi-output learning**

|  |  |
| --- | --- |
| 開設學校： | 國立清華大學 |
| 就讀校名： | 國立北科技大學 |
| 就讀班級： | 資工碩一 |
| 修課學號： | 113598043 |
| 修課學生： | 張育丞 |
| 授課老師： | 高宏宇 教授 |

Dec 15, 2024

**目錄**

[**1.** **前言** 1](#_Toc182524106)

[**2.** **程式實作** 1](#_Toc182524107)

[**3.** **環境建置資訊** 5](#_Toc182524108)

[**4.** **實驗心得** 5](#_Toc182524109)

**圖目錄**

[圖1 基本環境建置 1](#_Toc182524001)

[圖2 標點符號編碼 1](#_Toc182524002)

[圖3 下載Bert-Base-Uncased模型的Tokenizer 1](#_Toc182524003)

[圖4 自訂資料集之資料處理 2](#_Toc182524004)

[圖5 參數設定 2](#_Toc182524005)

[圖6 資料批次打包函數與 DataLoader 設定 3](#_Toc182524006)

[圖7 基於 BER 模型構建與前向傳播方法 3](#_Toc182524007)

[圖8 多標籤模型初始化與設備分配 3](#_Toc182524008)

[圖9 評估指標的設定 3](#_Toc182524009)

[圖10 模型訓練與驗證流程 4](#_Toc182524010)

[圖11 模型測試與相關性評分流程 5](#_Toc182524011)

**表目錄**

[表1 開發平台與工具 5](#_Toc182524092)

[表2 套件名稱與版本 5](#_Toc182524093)

# **前言**

在這次的第三次作業中，我們將實作使用深度學習框架 PyTorch，並結合 Hugging Face 的 API 來操作 Google BERT-Base 模型。我們將透過此模型來執行自然語言處理任務，包括文本的預處理與訓練，並以各種評估指標（如 Spearman 相關係數和 F1 分數）來測量模型的性能。這個作業的目標是加深對深度學習模型的理解，並學習如何在實務中應用現代 NLP 工具來解決語言處理問題。

# **程式實作**

在圖1中，這段程式碼負責初始化深度學習模型訓練的基本環境，包含匯入必要的工具庫與模組，例如： transformers （用於處理 NLP 模型）、torch（核心深度學習框架），並設置數據集加載、優化器、評估指標等功能。此外，程式會檢查是否有 GPU 可用，並自動選擇適合的運算設備，最後建立模型儲存的資料夾，為後續的模型訓練和評估奠定基礎。

在圖2中，由於中文的標點符號可能會影響模型的辨識，部分符號會轉換為 [UNK]（未知標記），其他符號則替換為對應的英文標點，以便模型能夠更好地處理文本並提高準確性。

實作中將採用 Google 提出的 Bert 模型。因此，在圖3我們下載並儲存 Bert-Base-uncased 模型的 tokenizer 到 cache 資料夾中，以完成基本的預處理工作。

|  |  |
| --- | --- |
| 一張含有 文字, 螢幕擷取畫面, 字型 的圖片  自動產生的描述 | 一張含有 文字, 螢幕擷取畫面, 數字 的圖片  自動產生的描述 |
| 圖 基本環境建置 | 圖 標點符號編碼 |
|  | |
| 圖 下載Bert-Base-Uncased模型的Tokenizer | |

圖4的這段程式碼定義了一個名為 SemevealDataset 的自訂資料集之類別 (Class) ，主要用於載入與處理 sem\_eval\_2014\_task\_1 資料集。程式碼包含初始化方法，負責選擇訓練或驗證集並載入數據，並透過索引方法實現資料提取與特定 token 替換，便於模型處理文字數據，接著在圖5的部分是針對學習率 (Learning Rate) 、週期 (Epochs) 、訓練批次大小 (Train Batch Size) 、驗證批次大小 (Validation Batch Size) 等進行調整，接著在以圖6 collate\_fn 函數進行 premise 與 hypothesis 的處理以及 relatedness\_scores 和 entailment\_judgment 的轉換，而後又以圖7定義針對深度學習模型的類別，主要進行多標籤的分類，建置以 Bert-base-uncased 模型去進行，並且定義出兩個線性層： Regressor 相關性分數預測層及 Classifier 的涵義判斷層，然後又以前向傳播 (Forward) 輸入 input\_ids 、 attention\_mask 和 token\_type\_ids ，又再透過 regressor 和 classifier 輸出相關性 relatedness\_scores 和涵義判斷 entailment\_judgment ，此部分我也嘗試過GitHub Copliot的提示但效果不彰，但架構部分還行，因此參考並嘗試用過去的經驗試錯出來，又以圖8設定於GPU並且使用。

圖9方面優化器選擇效率較好的Adamw而非Adam，並且套用在圖5進行設置的 lr ，至於在損失函數方面 regression\_loss\_fn 使用均方誤差損失 (ASELoss) 的策略計算相關性分數，classification\_loss\_fn 採用強項於分類的交叉熵損失 (CrossEntropyLoss) 進行含意判斷的損失，對於評估的指標有四個分別為斯皮爾曼相關係數 (SpearmanCorrCoef)、準確率 (Accuracy) 和多類別的 F1 分數 (F1Score)。

|  |  |
| --- | --- |
| 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  自動產生的描述 | 一張含有 文字, 字型, 螢幕擷取畫面, 行 的圖片  自動產生的描述 |
| 圖 自訂資料集之資料處理 | 圖 參數設定 |
| 一張含有 文字, 螢幕擷取畫面, 字型 的圖片  自動產生的描述 | 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  自動產生的描述 |
| 圖 資料批次打包函數與 DataLoader 設定 | 圖 基於 BER 模型構建與前向傳播方法 |
| 一張含有 文字, 字型, 螢幕擷取畫面, 白色 的圖片  自動產生的描述 | 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  自動產生的描述 |
| 圖 多標籤模型初始化與設備分配 | 圖 評估指標的設定 |

實驗到圖10，將進行兩項循環分別為訓練循環與驗證循環。

* 訓練循環

透過 dl\_train 將進行批次載入的動作，接著進行優化器的梯度處理，再來就是預測相關性分數和含意判斷，並計算回歸損失與分類所施，進行加總，而後再進行反傳播與更新模型參數。

* 驗證循環

使用 torch.no\_grad() 將梯度進用，減低運算成本，接著以 dl\_validation 載入驗證集，對此進行批次處理，進而計算模型的預測結果，最後收集所有預測與標籤，完成上述指標的顯示。

最後，圖11這是我自己追加的測試部分，為了確認所有的模型狀況，我將其檢查各項 ep{i}.ckpt 的檢查，其中以ep2.ckpt作為範例演示，並設置model.eval()進行評估推理，大多部分與訓練時相同，只是讀取結果並以spc\_score判斷最終Bonus的部份的成果，依據計算為10%。

|  |  |
| --- | --- |
| 一張含有 文字, 螢幕擷取畫面, 文件, 字型 的圖片  自動產生的描述 | 一張含有 文字, 螢幕擷取畫面, 字型, 文件 的圖片  自動產生的描述 |
| (a) | (b) |
| 一張含有 文字, 螢幕擷取畫面, 字型 的圖片  自動產生的描述 | |
| (c) | |
| 圖 模型訓練與驗證流程 | |
| 一張含有 文字, 螢幕擷取畫面, 字型 的圖片  自動產生的描述 | 一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片  自動產生的描述 |
| (a) | (b) |
| 一張含有 文字, 螢幕擷取畫面, 字型, 行 的圖片  自動產生的描述 | |
| (c) | |
| 圖 模型測試與相關性評分流程 | |

# **環境建置資訊**

本報告使用到良好的設備(表1)與幾個套件工具進行實驗(表2) ，這些工具可以使得整個實驗更加地有效率開發讓整體更加完善。

表1 開發平台與工具

|  |  |
| --- | --- |
| **Component** | **Specification** |
| Running environment | System: Windows 11 Pro,  CPU: Intel Core i9-12900K,  RAM: 16GB \* 2 (32GB),  GPU: NVIDIA RTX 3090 \* 1 |
| Visual Studio Code | Sep 2024 (version 1.94) |
| Environment management | Conda 24.9.0 |
| Python version | Python 3.11.9 |

表2 套件名稱與版本

|  |  |
| --- | --- |
| **Package Name** | **Version** |
| Transformers | 4.46.2 |
| Datasets | 2.21.0 |
| PyTorch | 2.4.0 |
| Tqdm | 4.66.5 |

# **實驗心得**

這次作業相比前兩次流暢了許多，或許是經驗的累積，也或許是我終於妥協並加入 Copilot 來進行錯誤說明和修正，讓整體流程更加順利。在 Spearman Corr 的測試環節中，透過使用該指標來驗證模型性能，使我更深入理解了模型在不同輸入條件下的表現差異，也更清楚數據批次處理的邏輯。這次作業的過程讓我深刻感受到工具對於程式開發的幫助，同時也意識到自己技術上的成長，未來希望能持續優化程式，提升整體效能和準確度。