HW6 - BERT

- **執行環境**
  - ✓ **kaggle kernal**

- **使用套件**
  - ✓ **numpy 處理矩陣以及數學邏輯運算**
  - ✓ **pandas 處理資料格式**
  - ✓ **torch 的 nn 網路和 AdamW 優化器**
  - ✓ **tansformers 執行 bert**

```
from time import time
from datetime import timedelta
from copy import deepcopy

import random
import numpy as np
import pandas as pd
from ml_metrics import mapk

import torch
from torch.optim import AdamW
from torch.nn.utils.rnn import pad_sequence
from torch.utils.data import Dataset, DataLoader
from transformers import BertTokenizerFast, BertForMultipleChoice
```

- **資料前處理**
  1. **一開始會將 document 和 query 拆成 token，並將其轉成 BERT 看得懂的 id。**
  2. **最後輸入至模型的 input 格式為 [CLS] query_id [SEP] document_id [SEP]**
  3. **限制 input 的長度為 512，太長則截斷，太短則補齊。**
  4. **將輸入資料以 8:2 的比例分成訓練集和驗證集。**

- **Training**
  1. **利用 dataloader 將輸入資料批次處理，設定 batch_size = 2。**
  2. **使用 BertForMultipleChoice 作為 model 進行分類任務。**
  3. **將 input_id, attention_mask, token_type_id, label 四個參數放入 model 訓練。**
  4. **利用 back-propagation 計算 loss function 的權重，並利用 AdamW 進行優化，設定 learning rate = 3e-5。**

- **Validation**
  1. **利用 validation 來實作 early-stopping，防止過擬和。**

- **Testing**
  1. **將測試資料如資料前處理所述，將原始資料轉成 BERT 相容的輸入格式。**
  2. **放入訓練好的模型中預測輸出分數**
  3. **結合 bm25 的分數，利用驗證集進行 grid search，來找到最好的 bert_weight。**

- **實作遇到的困難、心得**

這次作業除了講求 NLP 經驗，電腦的硬體設備更是重要，原本使用自己 4G 顯卡的電腦，但是將 batch_size 條成 1 之後仍然出現 cuda out of memory，無奈之下想起有 kaggle kernel 這方便的平台可使用，才順利訓練模型。此外也很感謝助教提供的填空版本，對於沒有 NLP 經驗的我，看了助教提供的 code 才稍微理解 BERT 整個訓練架構，參考 pytorch 和 BERT 的官方文件後，很快便能將程式補齊。