Final Project – Written judgment summarization

第 28 組 書面報告

組長：趙昌昱

組員：吳瀚惟、陳諺樟、鄧新泰

首先，install 與 import 所有會使用到的套件，如下圖所示：

```
import transformers
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM, AutoModelForCausalLM
import torch
from torch.utils.data import Dataset, DataLoader
from torch.optim import AdamW
from tqdm.auto import tqdm
from ignite.metrics import Rouge
import re
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
import jieba
```

1.Data preprocessing (30 pts)

Truncation

我們觀察到原文判決中的資料非常長，有的多達 8000 字以上，因此要讓資料有效率地被放入模型訓練，就必須做 truncation。而我們對 truncation max_length 做了很多的嘗試，最終試到當 max_length=1892 時，模型會有最佳的表現。

```
def get_tensor(sample):
    # 將模型的輸入和ground truth打包成Tensor
    model_inputs = tokenizer([each["origin_context"] for each in sample], max_length=1892 ,padding=True, truncation=True, return_tensors="pt")
    model_outputs = tokenizer([each["summary"] for each in sample], max_length=1892 ,padding=True, truncation=True, return_tensors="pt")
    return model_inputs["input_ids"].to(device), model_outputs["input_ids"].to(device)
```

Sliding window

將 input 原文依據 sliding window 大小分割成多段重疊的文字，並透過文字間的關聯性做摘要，最後再將所有結果串接而成，來處理比較長的文本。

```
def __init__(self, window_size=512, overlap=128) -> None:
        super().__init__()
        data_df = pd.read_excel(dataset_path)
        self.data = []
        self.window_size = window_size
        self.overlap = overlap
        for index, row in data_df.iterrows():
                origin_context = str(row['裁判原文'])
                summary = row['摘要']
                windows = self.sliding_window(origin_context)
                for window in windows:
                        self.data.append({"origin_context": "summary: " + window, "summary": summary})

def sliding_window(self, text):
        start = 0
        text_length = len(text)
        windows = []
        while start < text_length:
                end = min(start + self.window_size, text_length)
                windows.append(text[start:end])
                start += self.window_size - self.overlap
        return windows
```

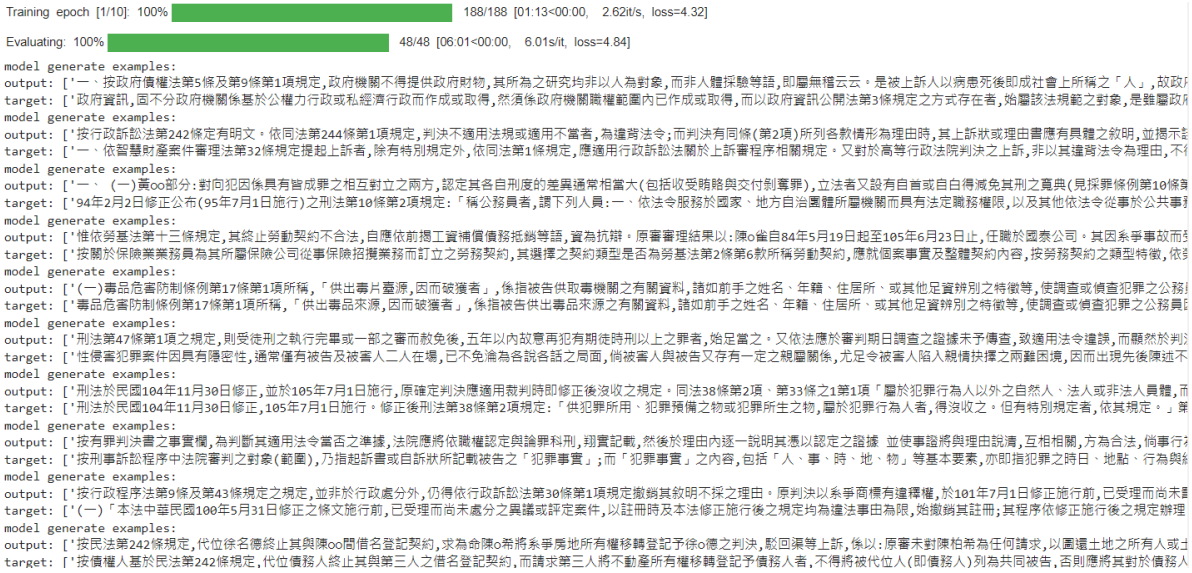
2.Model & Training method (20 pts)

Model

我們嘗試過很多 Transformer-based 的模型，盡量找尋有被中文 pretrain 過的模型，最後**選用”heack/HeackMT5-ZhSum100k”作為最終模型**，原因是它不僅被中文文本 pretrain 過，在 pretrain 的過程中也有學習過中文 summarization 的任務，且實驗的結果，用此模型訓練出的 evaluation score Rouge-2 最高，達到 0.20 左右，此外，訓練時間也不會太久，使用 colab 上的 A100 GPU 進行訓練，一個 epoch 大約花 7 分鐘左右。

Training 過程

我們訓練 10 個 epochs，每個 epoch 的訓練時長約為七分鐘，以下為訓練過程圖。



超參數設定

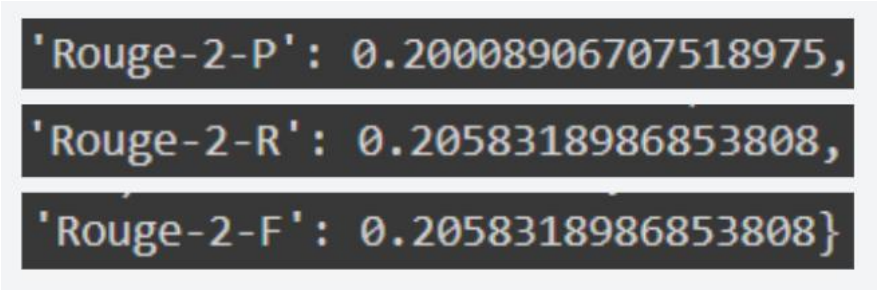
以下是實驗出最佳的一組超參數，其中，truncation_max_length 如上文所述，用於將資料前處理的 truncation 步驟，而 model_output_max_length 代表 model generate 出的文字最多可達到 2048 個字元。

- ***lr = 1e-4***
- ***epochs = 10***
- ***train_batch_size = 1***
- ***validation_batch_size = 1***
- ***test_batch_size = 1***

- ***random_seed = 42***
- ***truncation_max_length = 1892***
- ***model_output_max_length = 2048***

Evaluation 結果:

以下是透過上述 model 與超參數訓練出來的結果，達到 Rouge-2 分數 0.20 的成績。



3.Analysis (30 pts)

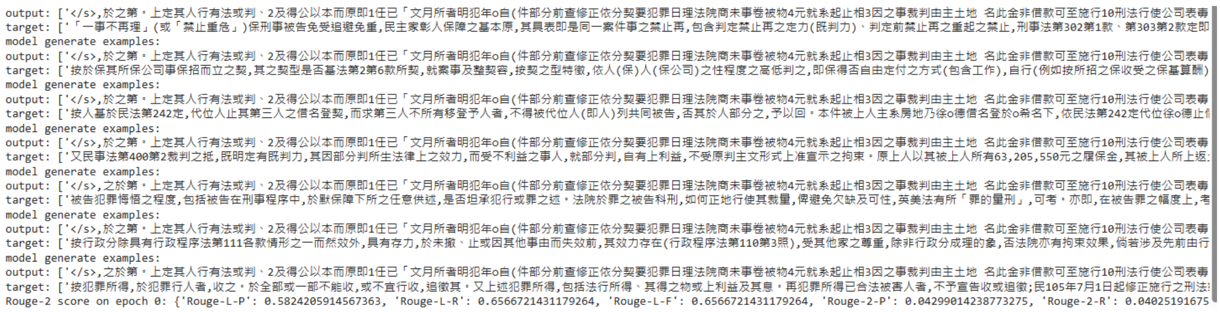
Difficulties

1. 算力不足: 在訓練大模型或 sliding window 時需要花大量時間，使用我們的資源 train 不太動。因為沒有能高速運算的本地端 GPU，因此我們購買 colab pro 的雲端 GPU 來補足我們的算力。

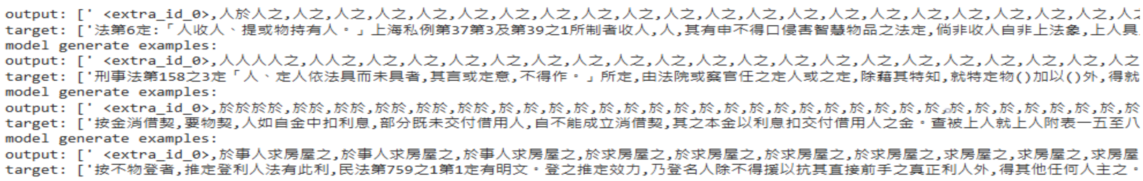
2. 嘗試過很多 Transformer-based 的模型，效果大多不佳，最後採用的模型如上文第二點所述，以下列出我們嘗試過的模型及失敗原因。

嘗試過的模型	失敗原因
yihsuan/mt5_chinese_small	效果不佳，重複生成一樣的文字 如: 不斷生成判決文本中的名字「葉文勇」
csebuetnlp/mT5_multilingual_XLSum	memory爆掉
IDEA-CCNL/Randeng-Pegasus-238M-Summary-Chinese	tokenizer有bug: “AttributeError: 'PegasusTokenizer' object has no attribute 'vocab' ”

3. 調整 learning rate 做訓練時，比想像中來得困難，不論調大或調小都會導致效果不佳，**仍然是**以 sample code 最初提供的 **1e-4 為最佳**。以下是將 learning rate 調大為 1e-3 產生的問題: 模型對於不同判決會產生相同的輸出，如下圖所示。



而將 learning rate 調小為 1e-5 也會產生另一種問題: 模型會在一筆資料中產生大量重複的文字，如下圖所示。



Testing analysis

將訓練好的模型預測給定的 test 資料集，觀察到以下現象：

testing example1:

以 test 資料集中的第 45 筆資料為例，**將原文摘要後的結果大多文句通順且文整。**以下為模型生成第 45 筆資料的摘要:

G	H	I	J	K
裁判字號	**爭點**	**裁判日期**	**裁判案由**	**裁判原文**
最高行政法院106年度判字第305號判決	原住民身分取得要件。	106年6月15日	原住民身分法	最高行政法院判決106年

摘要:

一、 原住民身分法第 8 條第 1 項之「應具」一謂「取得」之不同用語即知，故「係指「本該具有」，此參原人間身分法採血緻主義精神，應指自然血統而言，其可從二者一稱「發」所謂『取得』之方式即為「兼顧」者，並兼問其子女相關權益，爰增訂第 2 項如當事人已死亡，及子女準用本法第 4 條及第 7 條之規定取得該住民時間，以保障當人直系血親卑親屬之義務。」而該立法理由雖未提及「具有原簽人在 80 年 10 月 4 及過世未能於生前親自申請復當，而未取得新住民變身者。是故，若具本住民血運，為落實原滿住民文書法施行前，因.....或其他原因緝失或未得取得第住民性質者為限。而上訴人之祖母既具山地與泰雅族，依血〇〇之精神等語，益足證之。 二、 (二)上開卷(三)第 3 款、第 4 項)所列情形，包括第 6 項「原種與非原□結婚所生子女，從具原個人身分之父或母之姓或前住民傳統名字者』，於完成從父 or 母姓及傳運名字後，即能取得。則「並具(本於具有)之母性，自有「即有」或「已」的「未具』者」為不合。

testing example2:

以 test 資料集中的第 24 筆資料為例，裁判原文未出現"行政罰法第 5 條「從新從輕原則」"等文字，然而摘要中卻能產生出此法條，因此**我們推測模型能從之前學習過的資料中預測相關的知識。**以下為模型生成第 24 筆資料的摘要:

G	H	I	J	K
裁判字號	**爭點**	**裁判日期**	**裁判案由**	**裁判原文**
最高行政法院106年度判字第417號	按出賣人究應於何時辦竣戶籍登記	106年8月3日	特種貨物及勞務稅條例	最高行政法院判決106年

摘要:

一、 行政法院就行政法上義務之行為，應於出售日次日起 3 個月內將戶籍地由原房地直接遷移至新房用地」之限制，自無以行政機關內部自行規定之作業標準，作為對外裁罰人民、限制人民權利之依據。又依行政罰法第 5 條「從新從輕原則」，被上訴人於 102 年間購置系爭房子地，並於 1992 年 4 月 10 日函釋未發布為人民所周知。

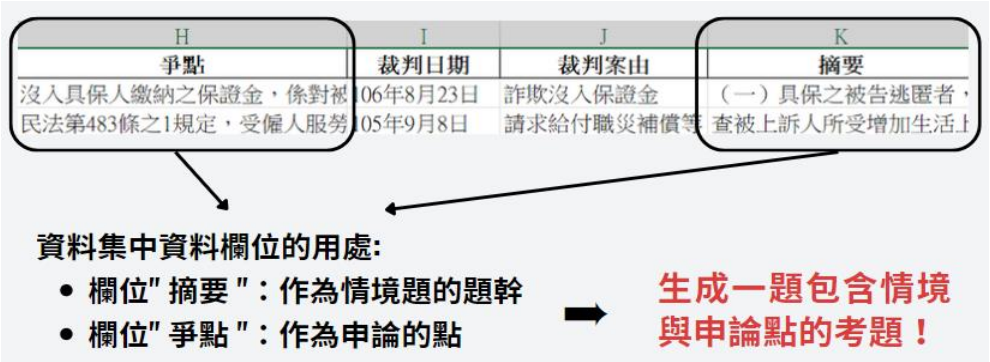
4.Proposal (20 pts)

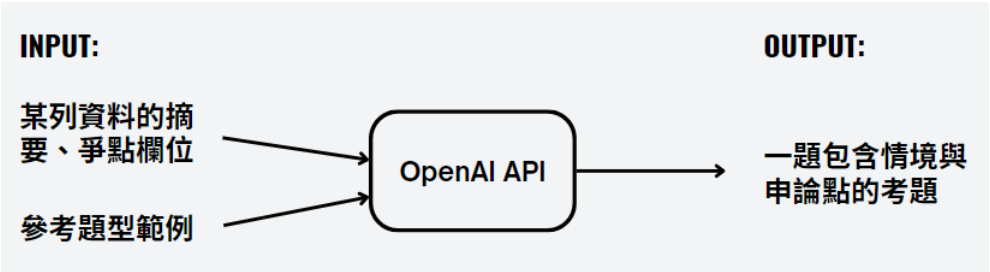
資料集應用１：作為法律課程情境與申論題的出題題庫

透過請教身邊的法律系朋友，我們得知法律系學生在課程中的考題很多時候會是以申論題的形式，且題目的來源多為一個判決。因此，我們認為這個資料集可以**作為法律課程情境與申論題的出題題庫，來幫助出題教授省去主動摘要判決與想題目的時間。**

實作方法：

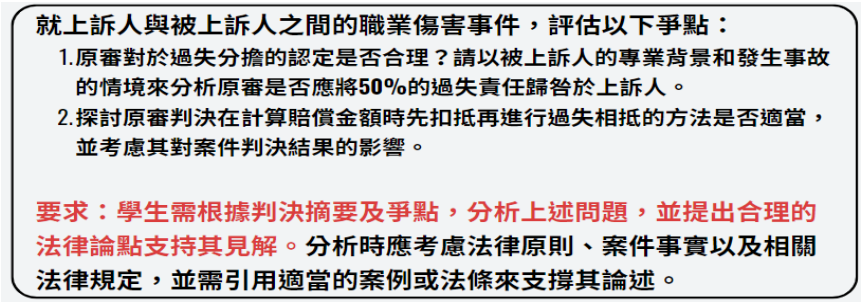
將資料集的資料欄位"摘要"作為情境題的題幹，欄位"爭點"則作為申論的考點。此外，也提供已存在的範例題型，透過 OpenAI API 的實作，便可以生成一題完整的題目。透過上述做法，一個法律課程的情境與申論題出題題庫就完成了。以下為實作步驟示意圖：





預期結果：

以 train 資料集的第二筆資料為例，是一個關於職業傷害案件的判決。下圖為透過上述步驟生成出的題目，可以看出此題目完整、符合判決內容且也要求學生將爭點作為申論點進行論述與作答。



資料集應用２：作為種子資料集，生產更大量且有品質的資料

經過對資料集的觀察後發現這樣的法律資料筆數不多，是相當珍貴的資料。因此，**若能將這些真實的判例資料作為種子資料集，生成更多高品質的仿真資料集，想必對法律應用面的發展會有幫助。**下圖為本次 train 資料集的資料筆數：

實作方式：

STEP1: 透過 LLM 將種子資料集生成仿真的合成資料集
STEP2: 兩個 LLM 對於合成出的資料集互相評分
STEP3: 根據評分、回饋做資料清洗
STEP4: 得到高品質資料集

預期結果：

經過上述的實作方式，便可以從少量珍貴的真實資料集產生大量高品質的資料集。

以上是我們的報告，謝謝。