

HW3 RAG

一、方法

1. 資料集前處理

在資料集前處理的方面，先將 full_text 依照大段落切開，即利用“\n\n\n”先做大概的切割，在每個段落中，再利用["\n\n", "\n", ".", " "]做切割，即 RecursiveCharacterTextSplitter 會在盡量滿足 chunk_size 長度的狀況下，仍大致依照語意做切割，切割優先序依序為兩次換行、一次換行、句點與空格。

由於有些被切割下來的 chunk 仍可能過於零碎，無法判斷原句的意思，因此每個 chunk 中文字實際對應到哪個完整句子我也有記下對應的 metadata，句子的切割則用 nltk 裡的函式幫忙切割。

2. 檢索方法

在檢索方法上則利用 dense retrieve 與 sparse retrieve 同時搜索，dense retrieve 較能根據語意找到適合的 chunk，而 sparse retrieve 則較能直接找到字面上相似的 chunk，最後把兩者找到的 chunk 所對應的完整句子全部抓回來，做我的 initial_evidences。

3. LLM 對答設計與 Prompt 技巧

在 LLM 的對答中我先餵進所有的 initial_evidences，prompt 如下圖：

```
prompt = (
    "You are a natural language processing (NLP) expert.\n"
    "Answer the following question based only on the provided evidence. "
    "If an evidence is not relevant to the question, simply ignore it. "
    "Use only the relevant evidence to compose your answer.\n"
    "Provide your answer in a concise paragraph (2-3 sentences).\n"
    "The evidence is sourced from a research paper.\n\n"
    f"Paper title:\n{title}\n\n"
    f"{evidence_blocks}\n\n"
    f"Question:\n{question}\n\n"
    "Answer:"
)
```

先告知他是一個 NLP 專家，再請他依照提供的 initial_evidences 回答，如果認為某些 evidence 不符合題意，直接跳過即可，且要在 2~3 句話之間回答完成，並提供給他論文標題、initial_evidences、問題，請他生成 initial_answer。

得到 initial_answer 後，再將 initial_evidences 跟 initial_answer 餵進 LLM，請他告訴我我的 initial_answer 是根據哪些 initial_evidence 回答的，prompt 如下圖：

```
ATTRIBUTION_PROMPT = """
You are a natural language processing (NLP) expert.

You are given a list of evidences and an answer.

Please carefully read the answer and the evidences.

Task:
- Identify which evidence(s) directly support the answer.
- Return a list of the evidence numbers that are relevant.

Format:
Evidence used: [list of numbers]

Example:
Evidence used: [1, 3, 5]

____

Evidences:
{evidence_list}

Answer:
{answer}
"""
```

一樣告訴他是一個 NLP 專家，再仔細讀完所有 initial_evidences 後，判斷哪些與答案直接相關，並給他一個範例知道如何輸出相關 evidence 的 list，作為我的 final_evidences。


得到 final_evidences 後，再用第一個 prompt 與 final_evidences 餵進 LLM 中，得到我的 final_answer，最終上傳檔案即用 final_evidences 與 final_answer。

這樣的設計先讓 LLM 看過較多 evidences 後，判斷哪些對回答問題有幫助，去蕪存菁後，得到較好的 evidences 並回答更好的 answer。

二、 研究與實驗

1. 模型選用：

在 LLM 模型選用上，我使用 FINGU-AI/Chocolatine-Fusion-14B (雖然名字寫 14B 但實際上是使用 8.37B，如附圖：


 Safetensors ⓘ

Model size 8.37B params

Tensor type

F32 · FP16 · U8

 Chat template

 Files info

還請助教留意一下)，選用原因單純是在 hugging face 的 open_llm_leaderboard 中找參數量小於 9B 的最好模型，且該模型在 Multistep Soft Reasoning (MuSR) 上獲得不錯的成績，代表在邏輯推演中有很好的表現，能清楚判斷 evidence 與問題之間的關聯。

原本有想尋找 7B 以下的模型，但效果不佳，詳情於 3. 額外技巧解釋。

2. Split and Chunk:

(1) 測試 chunk_size:

chunk_overlap = 32, retrieve_top_k = 10 (dense 與 sparse 皆找 10 個)
橫軸為 chunk_size，縱軸為 public_set 的 paper 編號

	64	128	256	512
0	0	1	1	1
1	0	0	1	0
2	0	1	1	1
3	1	1	1	1
4	0	0	1	1

測試方法為在 public_set 的 paper 0~4 中，retrieve 回來的資料是否含有正確答案所需的關鍵字，有的為 1，沒有的為 0。

判斷當 chunk_size 太小時容易漏掉資料，太大的話對 embedding 的編譯與搜尋資料太大，造成混亂，因此選 256。

(2) 測試 chunk_overlap:

chunk_size = 256, retrieve_top_k = 10 (dense 與 sparse 皆找 10 個)

橫軸為 chunk_overlap，縱軸為 public_set 的 paper 編號

	16	32	64	128
0	1	1	1	1
1	0	1	1	1
2	0	0	1	1
3	1	1	1	1
4	0	0	1	1

測試方法為在 public_set 的 paper 0~4 中，生成答案是否合理，有的為 1，沒有的為 0。

判斷當 overlap 過小時，即使抓到關鍵字，仍有可能將關鍵字與不相關的片段視為一個 chunk，導致答案錯誤，而太高時可能都 retrieve 到類似的 chunk，因此雖然 64 與 128 在此次實驗效果相同，我選擇 64。

(3) 測試 retrieve_k:

chunk_size = 256, chunk_overlap = 64

橫軸為 retrieve_k，縱軸為 public_set 的 paper 編號

	5	6	7	8	9	10
0	1	1	1	1	1	1
1	0	0	1	1	1	1
2	1	1	1	1	1	1
3	1	1	1	1	1	1
4	1	1	1	1	1	1

測試方法為在 public_set 的 paper 0~4 中，retrieve 回來的資料是否含有正確答案所需的關鍵字，有的為 1，沒有的為 0。

判斷當 retrieve 太少東西時，可能會使真正有幫助的資訊沒有抓到，因為單純的相似度高與能否回答答案仍有一段距離，相似度低

的也有可能對回答有幫助，而當 retrieve 太多的時候，可能會使 LLM 混淆，因此即使 7~10 即可，我選擇 8 (前測出來似乎效果不錯)。

3. 額外技巧:

首先，先利用 chunk 與文句的對照，只要有將關鍵字抓到，就可以還原整個句子，如在 public 文章 1 中原本抓到的字句僅為:

1) the corrected version of the same FCE training set on which the system is trained (450K tokens), and 2) example sentences extracted from the English Vocabulary Profile (270K tokens)..

語意並不明確，導致 LLM 無法判斷原意，利用文句對照可抓到更完整的句子，增加容錯率。

在利用 LLM 得知用了哪些 initial_evidences 時，我原先是直接一條一條直接問他哪個好，但其實單一條看的話很容易被判斷成與問題不相關，因此才有後來的設計，先回答一次在找相關 evidences，一次看到多個才能互相參照。

在模型選用上原本想用更小的模型，但更小的模型最大的問題就是每次輸出都不照我的格式來，導致我的多次問答設計無法執行，效果極低，才選用大一點的模型。

三、心得

對於這次的實驗我學到很多 RAG 相關的技巧與設計，但學到更多的是挑對模型真的很重要，原本用小的根本甚麼都做不出來，相當氣餒，幸好用了大一點的就大幅改善，讓我重拾信心。