

GenAI RAG

Author : 歐軒佑

前言

在本次作業中，我們的目標是設計基於 RAG 架構的論文問答系統，不同於傳統 LLM，RAG 系統會先根據問題從資料庫中檢索出相關內容，再由 LLM 根據這些檢索結果進行回答，以提升回答的準確性。

本報告將會討論不同的資料前處理、chunk 切割策略、向量檢索方法、prompt 設計等策略對於 RAG 的影響，以最佳化 evidence 的 rougeL 分數以及答案的準確率。


資料前處理

■ 初步切割文本

考慮論文結構有 Abstract、Introduction、Experiment 等章節，透過使用三個換行符號 (\n\n\n) 來初步分割章節，保留章節的完整性，避免進行 chunk 時破碎化。

■ 選取 Embedding Model

嘗試多個 Hugging Face 上常見的 Embedding Model，並使用 public dataset 其中 40 筆資料去做測試，可以發現 `intfloat/e5-large-v2` 無論在 RougeL 或是 Accuracy 上都有很好的表現，因此我將選用它作為此專案的 Embedding Model。

Embedding Model	Rouge-L	Accuracy
<code>sentence-transformers/all-MiniLM-L6-v2</code>	0.198	0.30
<code>BAAI/bge-large-en-v1.5</code>	0.215	0.35
<code>thenlper/gte-large</code>	0.189	0.325
 <code>intfloat/e5-large-v2</code>	0.225	0.45

■ Chunking Size & Overlap

在初步分段之後，我使用 SpacyTextSplitter 作為文本切割工具，Spacy 具有 NLP 處理能力，可以盡可能在句子的邏輯邊界處進行切割，降低語意破碎的問題。

接著將 Chunk_Size 參數設為 512 字元，Overlap 為 256 字元，以避免 chunk 長度過長降低 RougeL 分數，一定比例的 Overlap 可以處理關鍵資訊剛好位於段落邊界時的情況。詳細實驗內容會放在後面。

檢索方法

■ Dynamic Top-k

在檢索階段，初步設定一個較大的 K 值，確保檢索過程涵蓋相關的資料，並記錄這 K 個 chunks 的相似度分數，接著根據設立的相似度門檻，動態篩選出符合門檻的 chunks 作為實際輸入模型的 context，並確保至少保底取 4 個。

動態的 K 值調整能夠去自適應不同領域跨度的問題，根據檢索分數動態選擇最適合的 chunk 數量，若相關段落少，就會自動壓低 K 值，反之則會增加。

■ Reranker

在初步檢索後，進一步去使用 Reranker 來對已檢索出的 chunks 進行篩選，對於每一組 (query, chunk)，計算一個新的 relevance 分數，也就是對 chunks 進行第二次評估，最大化輸入的品質，可以結合 Dynamic Top-k 的方式，去挑選排序後前 K 個 chunks 作為 input。

本專案會測試兩種 reranker，bge-reranker-large 以及 prompt-based 的 Llama-3B-Instruct，後者的運作方式是將初步檢索後的數個 chunks 丟入 Instruct 模型，讓模型評斷該段落是否與 query 相關並評分，再根據評分篩選出最後要選擇的 chunks，可以根據排名，抑或是使用門檻值來做選擇。

```
"""You are a document analysis expert responsible for scoring the relevance between a paragraph and a question.
Please strictly follow the rules below:
- Based on whether the paragraph can directly support answering the question, assign an integer score (0 or 1).
- 0 means completely irrelevant; 1 means partly or fully supports the answer.
- Strictly output only one integer (no period, unit, or explanation), directly as the score.
Question: {question}

Paragraph:
{doc.page_content}

Please output a single integer score directly, for example: 1
Your Answer: """
```

▲ 我所使用的Prompt

Prompt 技巧

■ 限制模型作答範圍及格式

我將使用 meta-llama/llama-3.2-3b-instruct 作為 Generator，並制定一系列的指令來讓模型遵循著回答，其中包含加入CoT、限制詞彙使用等方式，相比於單純叫模型根據文本回答，正確率大概能提升5 ~ 10%。

Base your answer solely on the provided context.

- ▶ 指示模型不要自由發揮，所述內容必須要有 context 支持，防止 hallucination

Start with brief 1.inference, keep your explanation based on provided content.

- ▶ 使用 Chain of Thought (CoT) 讓模型整理、講解出推理解答的過程

After the Step1:inference, Step2:check and reflect whether the inference is correct

- ▶ 檢查推理內容有無瑕疵，進一步提升答案的品質

At the end, output "Final Answer:"

- ▶ 總結出最後的答案

■ 根據 Llama-3.1-8b-Instruct的回饋進行修改

Using a confident, concise sentence based directly on the context.

- ▶ 避免被視為不確定答案而直接被判0分

Direct using words or phrase in context whenever possible.

- ▶ 避免有時特定的名詞、形容詞的變化體會被視為錯誤，被評為0分

■ 最後的結果

```
CHAT_TEMPLATE_RAG = (
    """You are answering a question based on the provided context below. Strictly follow these rules:

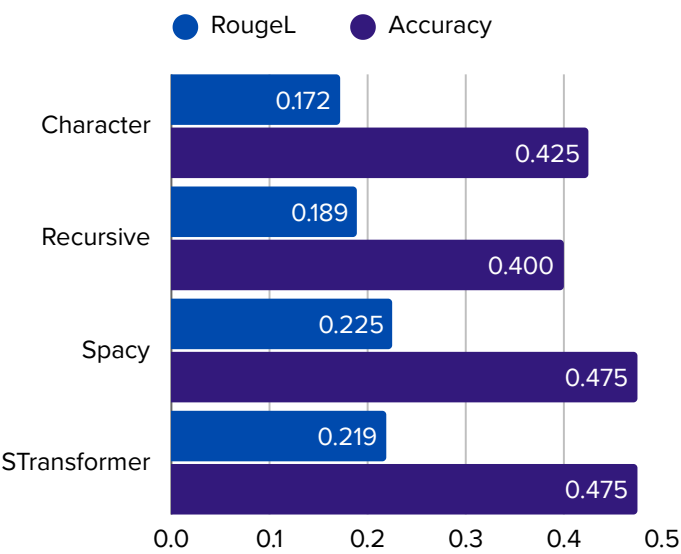
- Base your answer solely on the provided context.
- Start with brief 1.inference, keep your explanation based on provided content.
- After the Step1:inference, Step2:check and reflect whether the inference is correct
- At the end, output "Final Answer:"
- Using a confident, concise sentence based directly on the context.
- Direct using words or phrase in context whenever possible.
```

研究與實驗

■ 實驗 Split & Chunk

本實驗主要針對三個變數進行測試及分析：Text Splitter 類型、Chunk Size、Chunk Overlap。透過這三個因素的調整，觀察對於 Test metric 性能的影響。

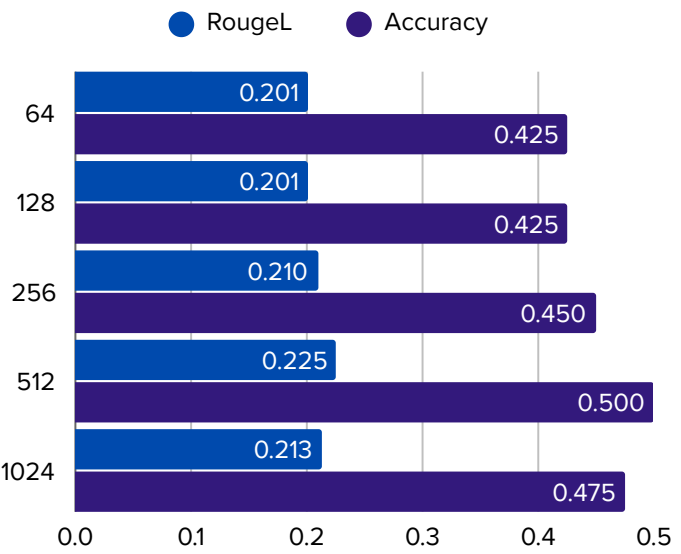
□ Splitter 類型



測試不一樣的 Splitter 對於 RAG 的影響，固定 chunk size = 512、Overlap = 256，用 40 筆資料測試 5 次並平均分數。

可以看出 Spacy Splitter 在 RougeL 上有明顯的優勢，這是因為 Spacy 擁有 NLP 的能力，可以夠根據語言的斷句結構進行語義切分，chunk size 是動態變化的，能有效保留每個 chunk 的語意完整性與連貫性。

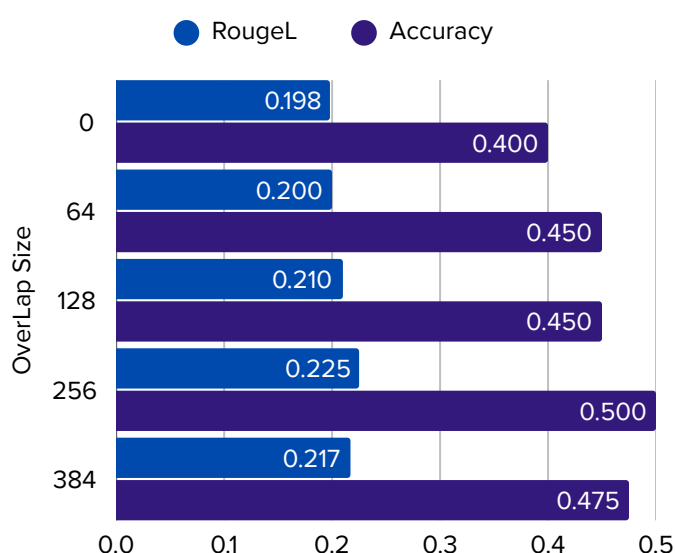
□ Chunk Size



測試 chunk size 設定對於 RAG 的影響，固定 chunk overlap 為 size 的一半 以及其他變因，用 40 筆資料測試 5 次並平均分數，可以發現 chunk size = 512 時會有最佳表現，RougeL 為 0.225，Accuracy 為 0.475。

整體趨勢顯示 chunk size 適中時，也就是接近真實 evidence 長度時，能有助於提升回答品質，並拉高 RougeL 分數。

Overlap



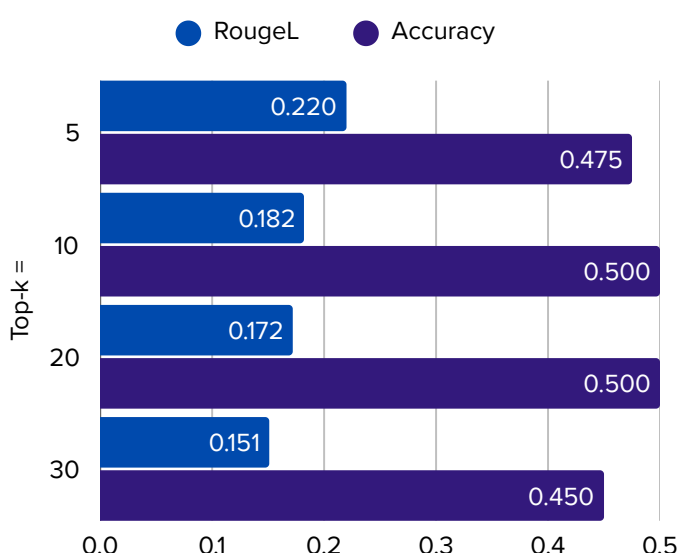
探討不同 chunk overlap 設定對於檢索與作答表現的影響，固定 chunk size = 512 以及 Prompt、Splitter 等變因，用40筆資料測試 5 次並平均分數。

可以發現 Overlap = 256，也就是約為 chunk size 一半時有突出的表現，RougeL 為 0.225，Accuracy 為0.475。

額外技巧 - 實驗 Dynamic-k & Reranker

針對 RAG的檢索階段，比較 Dynamic-k 以及 Reranker 後排序機制，評測是否會提升效能及正確率，本實驗會去比較靜態 Top-k (k = 5、10、20、30)、Dynamic-k、Reranker、Dynamic-k + reranker 在 RAG 上的表現。

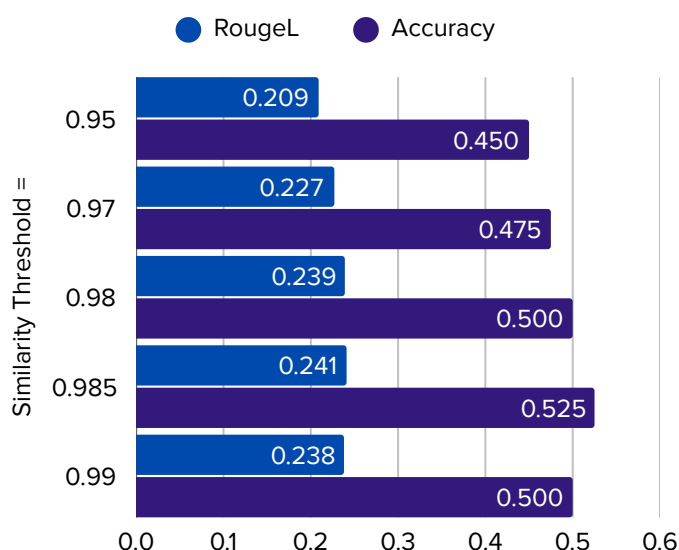
靜態 Top-k



先觀察固定 Top K 下的表現，由於 RougeL 的計算方式會隨著 K 的增加而使分母變大，因此越大的 K 值會造成分數降低。

此外，由於 chunk size 已經足夠大，資訊已經充足，在 accuracy 上並沒有太大的差別，其中 k = 30 較為不理想，個人推測是由於資訊過多使得模型想要涵蓋的更全面造成的反效果

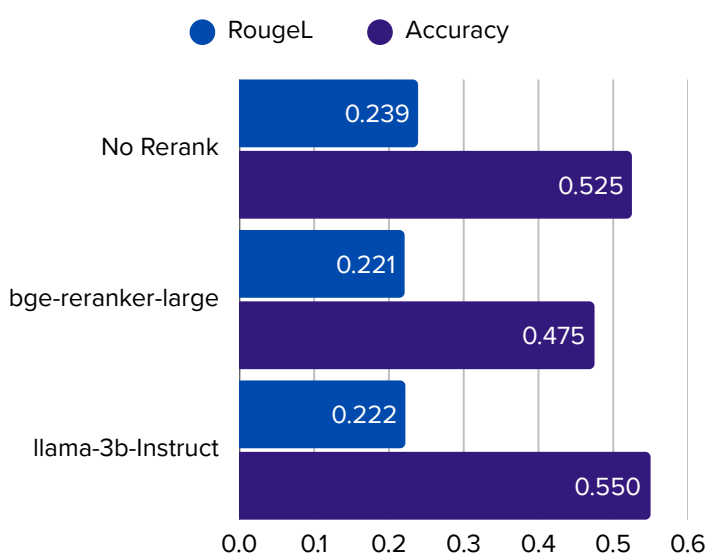
□ Dynamic-k



設定 $\text{initial_k} = 20$, $\text{min_k} = 3$ ，代表在每次檢索中，模型最多可以取回 20 個 chunk，最少則保證取回 3 個，測試設立不同的相似度門檻下對於 RAG 的影響。

從實驗結果可以觀察到，採用 dynamic-k 策略後，RougeL 分數整體有明顯提升趨勢，尤其是在適當的 Threshold 下，Accuracy 則沒有太大變化。

□ Dynamic-k + Rerank



這邊測試三種情況: 單純使用 dynamic-k、r bge-reranker-large、llama-3b-instruct (Prompt-based)。

bge-reranker 整體的表現不如預期，顯然的，e5-large-v2 的效果在此任務上還是比 reranker 版本的 bge-large 還要更好。

而 Prompt-based 的確實能讓模型的回答正確率提升，但模型可能會選出太多段落，因此犧牲 rougeL，這部分還要調整 prompt，讓模型去做更聰明的選擇。

■ 結論

經過多次實驗與比較，最後選擇使用 SpacySplitter、chunksize=512、overlap=256，並單獨使用 dynamic-k 方法，去拉高分數並達到 ROUGE-L 和 Accuracy 的平衡。

而在模型的選擇上，個人嘗試了 [Mistral-7b-instruct-v0.2](#)、[Llama-3.1-8b-instruct](#)、[Llama-3.2-3b-instruct](#)、[Llama 3.3 70B Instruct](#) 幾種，發現回答正確率都差不多，正確率都落在 0.45 左右，故選用 3B，也就是參數量少的模型作為 Gen

