Assignment 4: Retrieval-Augmented

Generation with LangChain

NYCU\_313581016

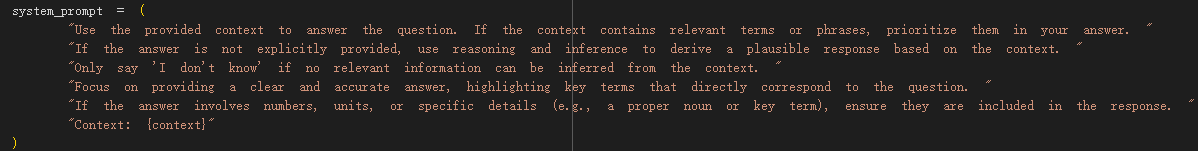
1. Please describe the details of your implementation for the RAG system
   1. What’s in your RAG system?

使用 Llama 3.2 1B 作為LLM，並選擇 Jina AI 的Embedding Model來生成文檔的向量表示。貓咪相關的資料被轉換為向量並存儲在 Chroma 中。檢索過程使用 similarity score threshold 的方式，設置score threshold=0.8，只有相似度高於該threshold的文檔才會被選取，除此之外，System Prompt也有修改，讓模型更精確的回答問題。

* 1. Which retrieval model you use?

MODEL 使用llama3.2:1b，而EMBED\_MODEL 則是使用jinaai/jina-embeddings-v2-base-en。

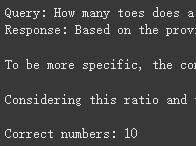
* 1. What’s your prompt?



說明：讓模型根據上下文來回答問題，強調提供清晰且準確的回應，並在需要時包含數字、單位或具體細節

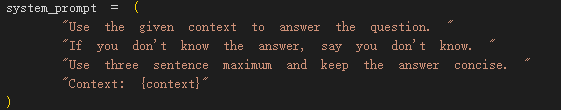
* 1. What’s new in your code in comparison with the code from our lab course?

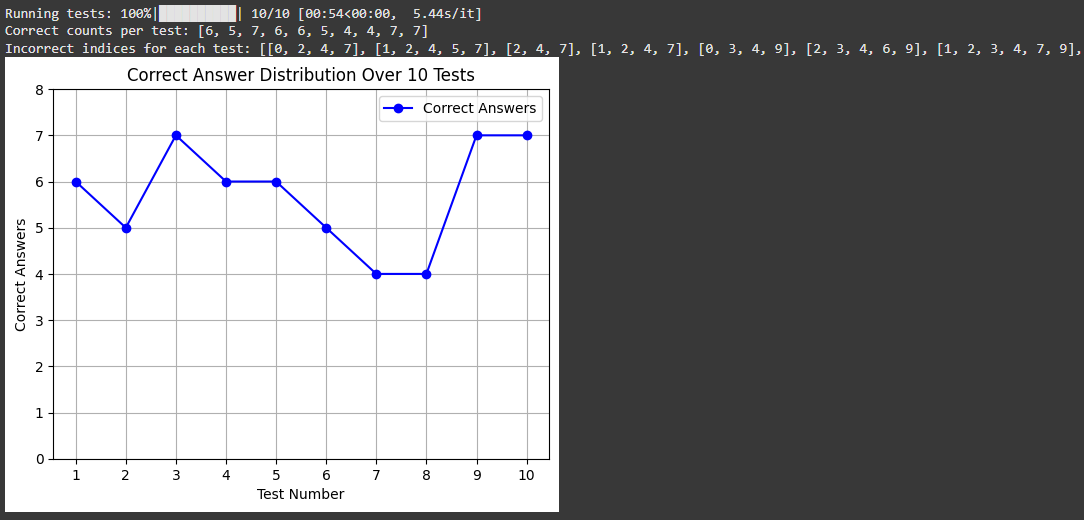
修改過 embedding model、retriever、system\_prompt，並添加分布圖的相關程式碼以說明差異(下面會有多張分布圖)。

* 1. best score for the ten questions

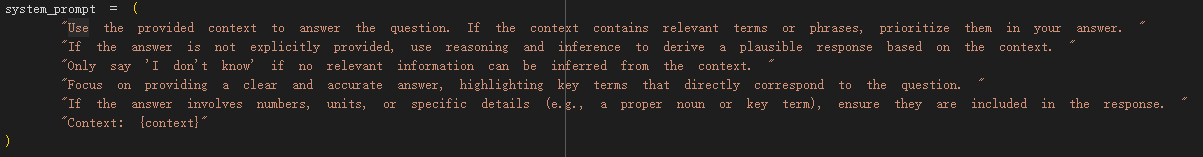
1. Please provide analysis for the RAG performance using different prompts

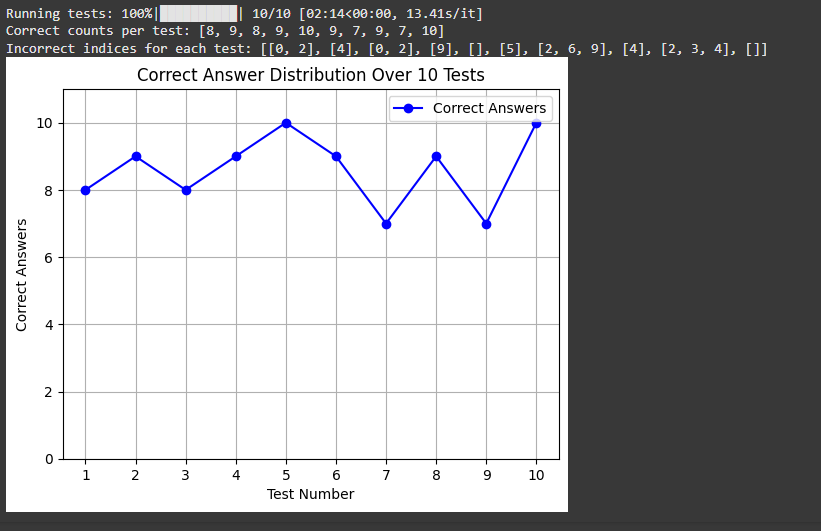
Prompt-1





Prompt-2





說明：以上是兩個 prompt 的分析，在其他參數固定下，我將兩個 prompt 分別做了 10次測試，並記錄錯誤的 index 分布，可以發現Promp-1 每次測試的正確答案數量較為不穩定，範圍從 4 到 7 不等，每次測試的錯誤答案都分佈在不同的問題，錯誤的數量較多，且範圍較廣，表示在多數測試中模型未能準確回答問題。而Promp-2 我調整了內容，希望模型針對 ”關鍵字” 回答，可以看到正確答案數量普遍提高，範圍從 7 到 10 不等，大多數測試的正確答案數量穩定在 8 個或以上，相比Promp-1，錯誤的數量大大減少。另外我也發現問題 2、4 是容易錯的題目，我想是因為這兩個答案的 answers 是由多個單字組成的，要exact match 比較困難。綜合以上分析，我們可以得出prompt 的設計對 RAG 系統的性能有著重要影響，適當的 prompt 可以顯著提高模型的回答準確性。

1. Please compare the RAG performance with different retrieval models and the performance without using RAG

|  |  |
| --- | --- |
| all-mpnet-base-v2-embedding-all | without using RAG |
| C:\Users\丁驥\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\39D406A.tmp | C:\Users\丁驥\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\833F0408.tmp |

說明：從測試結果來看，RAG sytem在使用不同Embadding model進行檢索時，性能有顯著差異，以下是我的觀察與比較：

* 使用 MPNet 模型的效果

當使用 all-mpnet-base-v2-embedding-all 時，正確答案的數量大多落在 4 到 6 個之間，表現明顯不如 Jina Embeddings(正確答案數量穩定在 8 個或以上)。這可能是因為 MPNet 的語義匹配在這些特定的文檔和問題結構下並不如 Jina 模型的表現強，導致檢索到的文檔相關性較低，影響最終回答的準確性。

* 不使用 RAG 的效果

當完全不使用檢索機制（即不使用 RAG）時，正確答案的數量僅在 0 到 3 之間，表現很糟糕。說明了在缺乏檢索輔助的情況下，語言模型無法有效處理問題並生成準確的答案，特別是當答案需要依賴外部文檔資訊時。

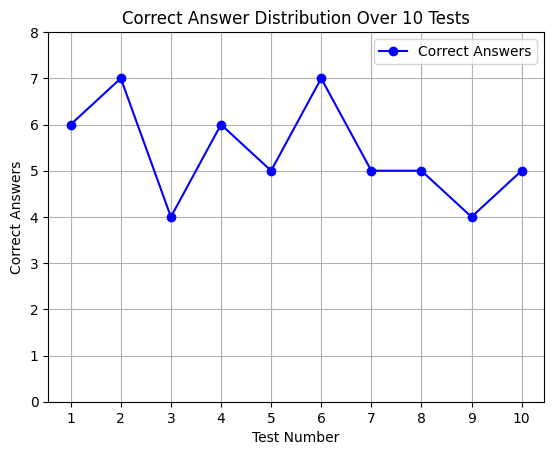
根據以上說明，我想可以說明RAG 的重要性以及不同的retrieval models 會影響到結果。

1. Anything that can strengthen your report

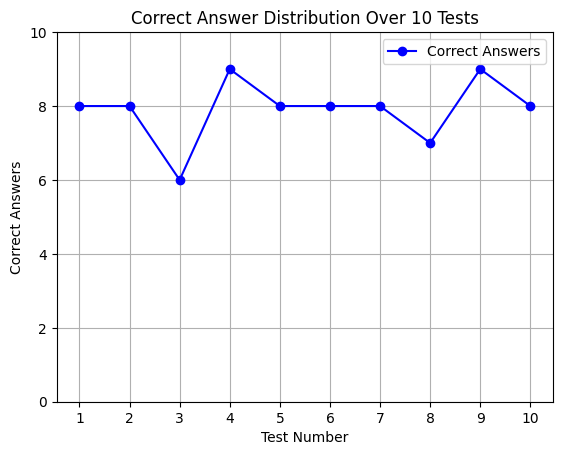
我想討論 Retriever 對於結果的影響，因此用不同的方法與參數進行比較，以下是相關結果。

* **search\_type="mmr"**

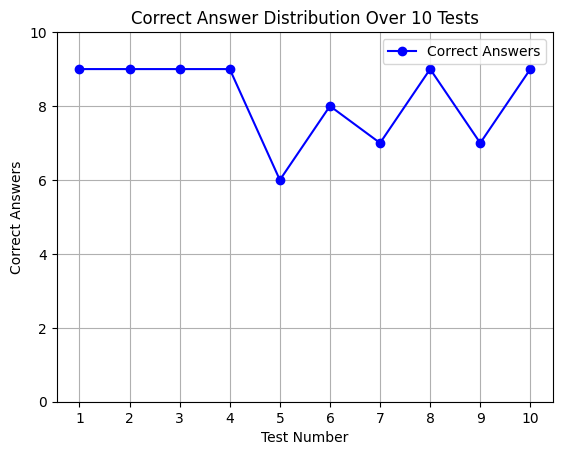
1. search\_kwargs={"k": 3, "fetch\_k": 5}



1. search\_kwargs={"k": 5, "fetch\_k": 7}

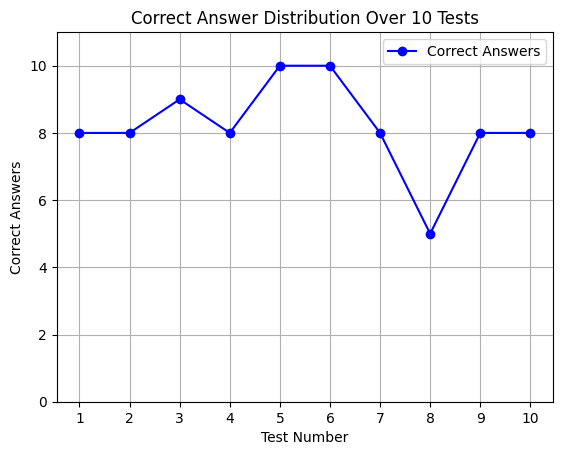


1. search\_kwargs={"k": 7, "fetch\_k": 9}

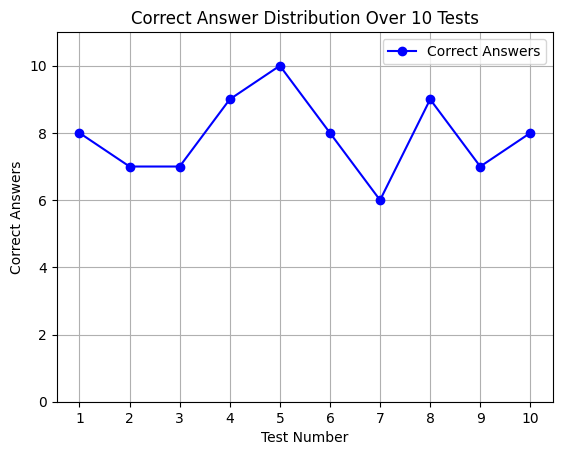


說明：根據以上結果可以知道，隨著參考的資料越多，正確率也會越高，但其運行的時間也會越久，像是(2)運行的時間幾乎是(1)的兩倍。而從(2)到(3)的準確度提升幅小於從(1)到(2)。因此，我想當資料量大的時候，如何在 ”時間” 與 ”正確性” 取的平衡是需要仔細考慮的。

1. search\_kwargs={'k': 6, 'lambda\_mult': 0.25}



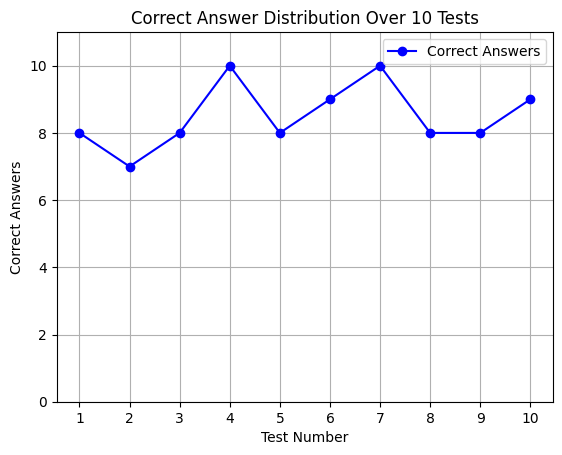
1. search\_kwargs={'k': 8, 'lambda\_mult': 0.25}



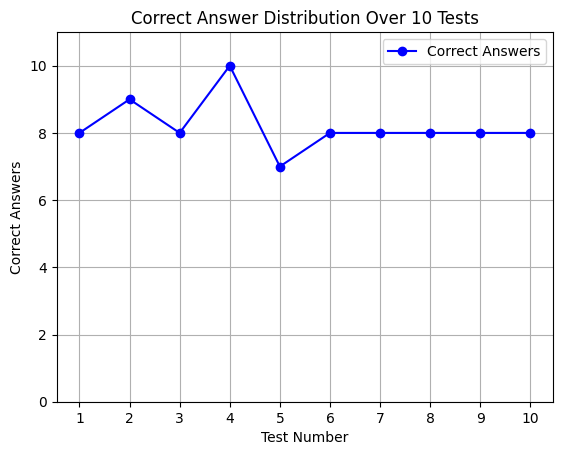
說明：同樣是MMR ，但我們調整 'lambda\_mult'，讓模型更關注文檔之間的多樣性。可以發現這樣的效果相當不錯，在(4)中，除了一個特殊值特別低外，其餘的數值都是 8 以上，甚至有兩個是全對的，因此可以說如果用 MMR ，lambda\_mult 對於這個 RAG system 是一個重要的參數。

* **search\_type=" similarity\_score\_threshold "**

1. search\_kwargs={"score\_threshold": 0.8}



1. search\_kwargs={"score\_threshold": 0.5}



說明：可以看到 score\_threshold 也是一個不錯的方法，在(1)中，甚至比前面所有的 MMR 都有更好的效果與穩定度，因此最後也是採用 score threshold 作為這次作業的Retriever。

附錄

|  |  |
| --- | --- |
| Environment | |
| Running environment | Colab |
| Python version | Colab |
| GPU(s) you used | T4 |

* 使用 GPT 進行繪圖( matplotlib 相關程式碼 )