RAG 與 Graph RAG 實作

專案資訊

項目：傳統 RAG 與 知識圖譜增強 RAG 實作與比較

環境：Python 3.12.7

套件：requests (2.32.3) langchain (0.3.19) langchain-openai (0.3.7) pinecone (6.0.1)
networkx(3.4.2) matplotlib (3.10.1)

安裝方式 : Poetry , requirements.txt

github : https://github.com/simongood/project_RAG_and_Graph.git

執行檔：

RAG : rag_query.ipynb

graph RAG : graph_query.ipynb

—---------- 實作說明 —----------

總結報告前言

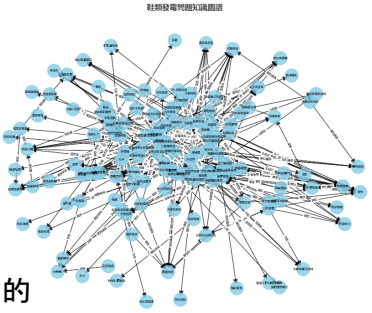
1. 技術文件 : 使用針對鞋子發霉技術文件
   由 claud ai 生成 (大小 : 14.7kb, 字數: 6493, 路徑: 'file\shoe-mold-technical-document.txt')
2. 目標 : rag, graph - rag 實作，包含 比較 5 種不同複雜度問題、3 種評分定義
3. 對 graph rag 的理解 :
   針對業務：主要作用在分析深度，所以在業務上要保證其必要性，若不需要深度關係則傳統資料庫即可解決
   效率原則：對關係分析沒有大影響的放在傳統資料庫，核心：設計成小而輕的載體
4. 所以此專案針對目標 : 建立於針對對鞋子有黴菌問題的客服系統

 # rag 實作說明

- 前製處理：文件 > 切割 > 向量化 > 存入向量資料庫
  1. 切割 : 300 一切，60 重疊，由於本文件多為短段落，通常會是段落截斷而不會遇到重疊文本
  2. 向量化 : 使用 gina-embbeding v3 進行 1024 維向量化
  3. 存入向量資料庫 : 使用 pinecone，總共儲存 32 筆段落資料
- 使用者提問：提問 > 向量化 > 比對向量資料庫 > 取 top3 放入 prompt 參考資料 > llm 進行回答

 # graph - rag 實作說明

- 提取實體與關係概念 : 從 100 題針對鞋子黴菌的問題集，提出實體與關係集合 (詳見備註1)
- 前置作業：提取文件知識圖譜實體關係 > 建置圖譜 > 圖譜可視化 > 將實體關係集合向量化至 pinecone
  1. 取出文件知識圖譜實體關係 : 將文檔切成不同大小，請 claud 比對已有的實體關係集合，取出最適合表示的知識圖譜資料 (路徑 : 'data\graph_data.json')
  2. 建置知識圖譜 : 小量資料 => 使用 networkx 存成 pkl 檔
  3. 可視化 : (路徑 : 'knowledge_graph.png')



- 使用者提問：提問 > llm 提取實體關係 > 實體關係向量比對校正
  > llm 生成新的知識圖譜比對 > 搜尋最佳知識圖譜路徑
  > 參考所有知識圖譜路徑給予回答
  1. 遇到難點：llm 無法準確抓取提問的實體與關係，無法配對知識圖譜
     解決方案 : 只要 llm 提出實體與關係，再經由向量資料庫比對出top3接近的
  2. 遇到難點：若原本提出 3 個實體，2 個關係，則向量比對後將生出 9 個實體 6 個關係
     若進行排列組合會有 9 x 8 x 6 個組合 => 組合氾濫
     解決方案 : 將 9 個實體 6 個關係給 llm 請其重新排列為可能的知識圖譜關係
  3. 待解決難點：已經盡量減少 llm 的 prompt 內容，但還是會有小機率生成錯誤訊息
     可能解決方案 : 將多種結構問題語句進行向量化，並提前設定好其實體與關係，若llm 出現錯誤則直接使用向量化搜尋已建好的問題語句，配對其實體與關係

—---------- 系統評估與比較分析 —----------

 # 1. 評估指標定義：

對 graph rag 傳統 rag 定義了三項最大的差異特性，由 prompt 給 claude 分別對其回答進行評測

1. 推理能力 : 辨別回答中所呈現的多樣性、複雜性
2. 背景訊息 : 是否提供理解答案所需的足夠背景和上下文
3. 資訊量匹配度 : 綜合考量焦點、簡潔度、額外資訊的必要性

 # 2. 兩個系統的回答進行評分（詳見路徑：grade.md）

		RAG			graph - RAG		
複雜度	**問題**	**推理能力**	**背景訊息**	**資訊量匹配度**	**推理能力**	**背景訊息**	**資訊量匹配度**
1. 基礎事實查詢 最簡單	鞋子發霉怎麼辦？	6	5	7	**8**	**8**	**9**
2. 比較查詢 中等複雜度	白色和黑色黴菌對鞋子的危害有什麼不同？	**7**	**8**	**7**	5	6	6
3. 多步驟處理查詢 中高複雜度	我的運動鞋裡外都有黴菌， 怎麼徹底清洗並防止再次發霉？	7	6	8	**8**	**7**	**9**
4. 專業知識與健康關聯查詢 高複雜度	長期穿著發霉的鞋子對足部健康有哪些特定風險？ 特別是對糖尿病患者來說。	**6**	5	6	5	**6**	**7**
5. 情境分析與多方案比較查詢 最高複雜度	我在熱帶高濕地區工作，需要長期在戶外穿著皮靴， 已經出現反覆發霉問題。 請比較不同防黴方案的效果、成本和實用性， 並給出最適合我情況的解決方案。	5	4	5	**8**	**7**	**9**

 # 3. RAG VS graph - RAG 優缺點比較

值得討論的點：

1. 綜觀：graph RAG 在大部分狀況下，推理能力、背景訊息、資訊量匹配度都優於 RAG
2. 第二點：RAG 較優，由於文本的切割段落剛好完整解釋了問題，這種情況通過 graph 反而造成反效果
3. 第四點：兩者表現皆較差，由於技術文件內並沒有提及糖尿病患者的健康風險
4. 第五點：高複雜度的差異，若文件內容涵蓋了問題解釋，則 graph RAG 在高複雜度問題下遠優於 RAG

結論優缺點：

1. RAG：適合當資料是一段短文本專注解釋一個問題時，因為本身即為全局，反而不需要做提取
   (由上第2點得出；ex：商品、電影簡介，由短文本即完整解釋了產品)
2. graph RAG：適合在長文本能解釋多項問題時，全局能夠回答各種不同複雜度的問題，實體關係的抽取提供了結構性的推理能力（ex: 技術文件、長篇文章)

 # 4. 應用方位討論（因實作方法不同於微軟 graph RAG，所以比較內容也包括了微軟，詳見備註1.）

	目標不交集的**短文本**	**長文本 - 專注業務範圍**	**長文本 - 專注完整解釋文本本身**
RAG	**O**		
我的 graph - RAG		**O**	
微軟 graph - RAG			**O**

 # 5. 改進建議

1. 成本：在 graph RAG 中，提取關係與實體，花費 llm 價格極高，若能使用自己微調的模型去做提取(館長)，不但可以在提取方面更符合需求，在背後提升其推理能力，更可以有效降低化費，相輔相成
2. graph 搜尋路線：可對使用者的問題依照不同複雜度去做設計
   ex：簡單複雜度對應到最短路徑，高複雜度對應到三條最短路逕
3. rag 補充知識點：graph 還是會有不滿意的回答，針對蒐集客戶問題不滿意回答的問題，建立短文件解釋，作為補充知識點搭配 graph RAG

---------- 備註 ----------

備註 **1.**

提取實體與關係概念：

1. 微軟 **graph - rag** 概念：將所有實體與關係提出，觀察 community 分布，得出結論
   需要多次反覆與llm確認，價格高昂
2. 我的 **graph - rag** 概念：從100題針對鞋子黴菌的問題集，提出實體與關係

	實體與關係數量	價格	速度	可回答範圍	實際上會被問到但能回答的題目	效益
微軟 graph	高	高		高	相同	
我的 graph			高		相同	高

3. 說明：目標為 ”針對業務” 及 “ 效率原則”，所以與微軟相反，我先得出我想要的結論(專注於回答哪個 community 的問題)，只將那個 community 實體與關係提出，對於需要的問題，不提取多餘的關係實體，提升了效率，也降低了成本。