AI 驅動的多向量語意分析架構

1. 架構概述

本研究提出一套以 AI 為基礎的語意分析架構，旨在對資料庫欄位的實際資料內容與其在應用程式中的使用情境進行語意分析，並轉換為向量嵌入後分別儲存於多個語意向量資料庫中。可根據使用者的自然語言輸入進行語意推論，整合多源語意資訊，並自動產生對應的 SQL 查詢語句，以實現智能化、可解釋的資料存取流程。

---

2. 語意資料建模架構

整體架構由五個語意向量庫組成，分別承載不同語意層級的資訊：

2.1 向量資料庫 A – 欄位內容語意（Field Content Semantics）

- 從每個資料庫欄位中擷取具代表性的樣本值（例如前 1000 筆紀錄）。
- 將這些值的分布轉換成自然語言描述。
- 使用如 bge-base-zh-v1.5 的模型進行向量嵌入（embedding）。
- 將向量儲存至 Vector DB A，以表示該欄位本身的資料語意。

程式碼範例：

```
from sentence_transformers import SentenceTransformer

sample_values = ['2024-01-01', '2024-01-15', '2024-02-01']
description = f"This column contains dates like: {',
'.join(sample_values[:3])}"

model = SentenceTransformer("bge-base-zh-v1.5")
vector = model.encode(description)
```

欄位語意關聯判斷

為了判斷不同欄位之間是否具有語意上的關聯，本方法採用多策略進行相似性評估。這些方法可支援後續任務，例如同義欄位偵測、結構比對、自動提示組裝等。

策略一：Cosine Similarity 餘弦相似度

- 計算每對欄位之間的向量餘弦相似度。

- 若相似度高於閾值（例如 0.85），則視為具語意關聯。

```
from sklearn.metrics.pairwise import cosine_similarity

similarity = cosine_similarity([vector_field_A], [vector_field_B])[0][0]
if similarity > 0.85:
    print("發現潛在語意關聯")
```

策略二：Clustering 分群分析

- 對所有欄位的向量執行無監督式分群（如 KMeans 或 DBSCAN）。
- 屬於同一群集的欄位視為語意相近。

```
from sklearn.cluster import DBSCAN

clusters = DBSCAN(eps=0.3, min_samples=2).fit(all_field_vectors)
labels = clusters.labels_
```

策略三：語意與規則混合判斷（Hybrid Strategy）

結合語意向量與輕量級規則，提升關聯判斷精度：

規則條件	效果
欄位名稱相似度高（如 Jaccard）	分數加權
資料型別相同	分數加權
欄位說明文字接近	分數加權
欄位值分布相似	分數加權
來自同一資料表或 schema	分數加權

```
def score_relation(vec1, vec2, name1, name2, type1, type2):
    score = cosine_similarity([vec1], [vec2])[0][0]
    if jaccard_similarity(name1, name2) > 0.6:
        score += 0.05
    if type1 == type2:
        score += 0.05
    return score
```

進階策略：LLM 語意比對輔助

可進一步使用本地大型語言模型（LLM）來判斷兩欄位說明是否為語意等價或相近。

```
prompt = f"""
欄位一描述: "儲存訂單建立時間的欄位"
欄位二描述: "交易初始化日期"

這兩個欄位語意是否相關？請回答「是」或「否」，並簡要說明理由。
"""
result = local_llm(prompt)
```

2.2 向量庫 B － 程式語意用途語意庫

- 靜態分析應用程式碼中所有 CRUD 操作。
- 自動擷取資料欄位在邏輯中的使用語境（如查詢條件、更新依據等）。
- 將其轉換為自然語言敘述後產生語意向量。
- 儲存至向量資料庫 B，以建構欄位的功能語意模型。

程式碼範例：

```
code_snippet = "order = db.query(Order).filter(Order.created_at >=
'2024-01-01')"
prompt = f"描述以下程式碼中欄位的使用語意：\n{code_snippet}"
summary = local_llm(prompt)
embedding = model.encode(summary)
```

2.3 向量庫 C － 語意整合與推理層

- 對向量庫 A、B、D、E 中欄位語意向量進行比對與融合。
- 建立欄位間的語意關聯關係（如同義欄位、實體對映、資料血緣）。
- 結果儲存至向量資料庫 C，作為 AI 推論與查詢生成的語意圖譜依據。

程式碼範例：

```
from sklearn.metrics.pairwise import cosine_similarity

similarity = cosine_similarity([vector_a], [vector_b])[0][0]
if similarity > 0.85:
    print("Strong semantic relation found")
```

2.4 向量庫 D - View 結構語意庫（可擴充模組）

- 擷取資料庫中 View 的結構與描述性語意（如 Join 關係與聚合邏輯）。
- 將其向量化後存入向量庫 D，以便查詢時納入分析推理。

程式碼範例：

```
view_description = "This view aggregates sales data by region and
month."
vector_view = model.encode(view_description)
```

2.5 向量庫 E - Stored Procedure 語意庫（可擴充模組）

- 分析 Stored Procedure 的邏輯流程與參數意圖。
- 提取語意敘述並進行向量化處理。
- 儲存於向量資料庫 E，用以建構作業流程層的語意背景。

程式碼範例：

```
sp_description = "Stored procedure calculates quarterly revenue for a
given region."
vector_sp = model.encode(sp_description)
```

---

3. 查詢流程

3.1 使用者自然語言輸入

例：「請分析 2024 年第一季到第四季的接單量」

3.2 查詢語句語意向量化

```
query_vec = model.encode("請分析 2024 年第一季到第四季的接單量")
```

3.3 多語意向量庫查詢比對

```
results_A = vector_db_A.search(query_vec)
results_B = vector_db_B.search(query_vec)
results_D = vector_db_D.search(query_vec)
results_E = vector_db_E.search(query_vec)
```

3.4 組合語境提示並交由 LLM 推論

```
prompt = f"""
使用者問題：「請分析 2024 年第一季到第四季的接單量」

【向量庫 A】：
{results_A}

【向量庫 B】：
{results_B}

【向量庫 D】：
{results_D}

【向量庫 E】：
{results_E}

請推論：
1. 需要使用哪些欄位？
2. 如何定義 2024 年各季的時間範圍？
3. 接單量該如何計算？
4. 請產生對應的 SQL 查詢語句。
"""
response = local_llm(prompt)
```

3.5 自動產出 SQL 查詢語句

```
SELECT DATE_TRUNC('quarter', created_at) AS quarter,
       SUM(amount) AS total_orders
FROM orders
WHERE created_at BETWEEN '2024-01-01' AND '2024-12-31'
  AND status = 'completed'
GROUP BY quarter
ORDER BY quarter;
```

---

4. 技術組件

組件類別	描述
語意嵌入模型	採用 BGE（如 bge-base-zh-v1.5）進行文本語意向量化
大型語言模型（LLM）	使用 LLaMA3、Mistral 等本地模型進行自然語言理解與推理

組件類別	描述
向量資料庫引擎	FAISS（本地）、Qdrant、Weaviate（雲端/容器化部署）
程式語意擷取工具	靜態分析（AST）、LLM 摘要器、正則運算式與規則擷取模組

---

5. 應用場景

- **自然語言 SQL 查詢助理（AI SQL Copilot）**
- **跨系統欄位語意對映與關聯辨識**
- **無文件系統語意探索與欄位理解**
- **資料治理與語意血緣分析**
- **自動化報表腳本生成與維運工具**

---

6. 創新價值

- **多源語意整合能力：** 同時解析資料內容、程式語意與資料庫物件結構。
- **自然語言查詢入口：** 降低使用門檻，支持業務與資料人員直接查詢。
- **查詢生成自動化：** 節省人工 SQL 撰寫與邏輯推敲時間。
- **高擴充性：** 可模組化納入 View、Stored Procedure 等其他物件。
- **語意血緣透明化：** 強化資料治理與後設資料分析機制。

---

7. 未來發展方向

- 導入知識圖譜與本體論（Ontology）結構，提升語意推理能力
- 支援中、英、越等多語語意模型切換
- 擴展至異質資料源與非結構化資料整合
- 納入使用者互動回饋機制並進行 RLHF（人類回饋強化學習）微調