人工智慧大型語言模型實作應用
LLM

Dr. Steve Lai
Mathison Intelligence
2025/09/13、14 @ iSpan

Resume

- Steve Lai (賴昭榮)
  - 現職：AI 應用研發資深工程師
- 學經歷：
  - Founder and CEO at **Mathison Intelligence**（麥錫森智能）
    - ✦ plusForm — 線上課程品牌
  - 台灣大學資訊工程博士
  - 中央大學資訊工程碩士

➡加入 plusForm 討論群

➡AI 科普相關

➡Youtube 

Discord 

LINE openChat 

課程目標

LLM Chatbot

- 理解大型語言模型（LLM）的基本概念與應用。
- 不使用框架軟體框架，如：LangChain, llamaIndex，直接開發 LLM Application 的優劣與技巧。
- 強化對 Prompt Engineering 的理解與應用。
- 學習如何使用自有資料微調（Fine-tune）Llama 或其他模型，打造專屬的 Chatbot。
- 探索 RAG（檢索增強生成）技術，提升 Chatbot 的回答準確性。
- 能夠自行開發並部署一個完整的 LLM Chatbot 系統。

Python Quiz

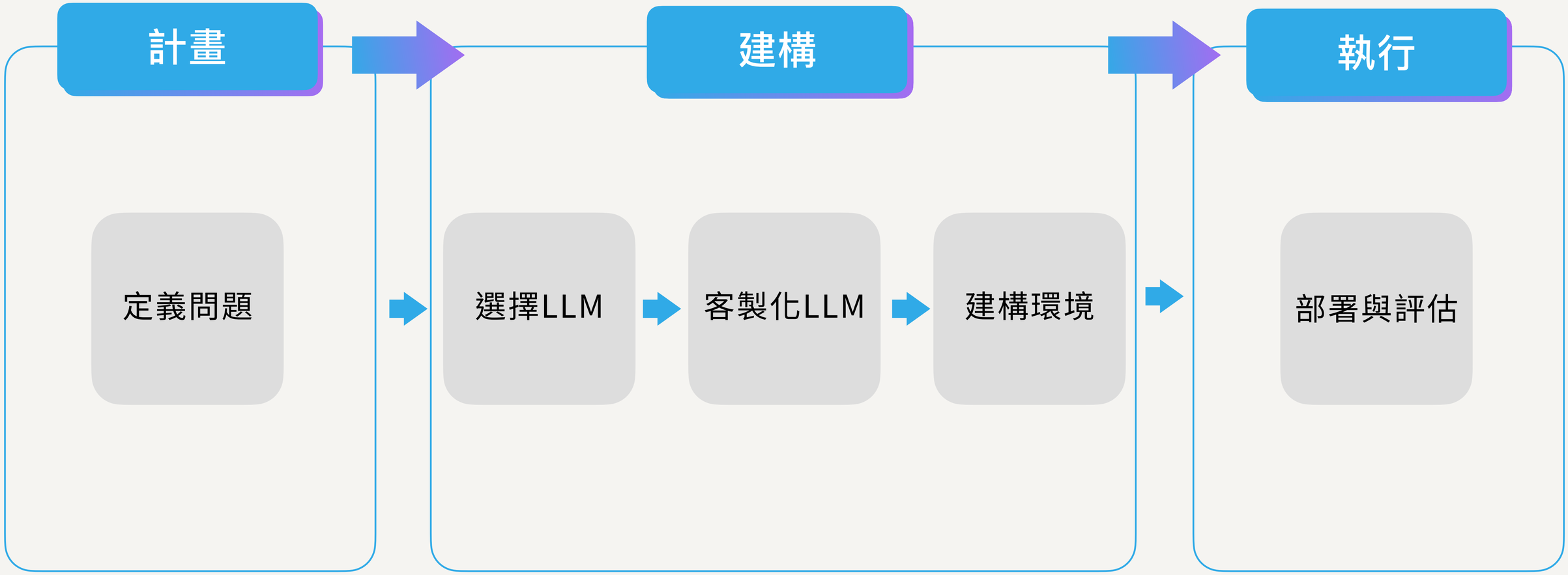
大型語言模型（LLM）基本概念

- 什麼是 LLM？ 一個超強的文字接龍與歸納大師)
- LLM 的核心原理： Tokens, Embeddings, Transformer 架構
- 市場主流 LLM 模型巡禮：
  - 閉源模型：OpenAI GPT 系列 (GPT-4/4o/5/o1)、Claude、Gemini
  - 開源模型：Meta Llama 系列 (Llama 2/3/4)、gpt-oss、gemma

開源與閉源的選擇考量

- 成本：API費用 vs 本地部署成本
- 隱私：資料安全性需求
- 效能：推理速度與準確度平衡
- 客製化：是否需要微調

Building an LLM application



Copyright © 2024 Steve Lai & Mathison Intelligence. All rights reserved.

Building an LLM application

- 定義問題
  - 單一且夠大的問題
- 模型選擇
  - 商用 licensed
  - 模型大小
  - 模型效能

環境設置

- 環境指南
  - Python 與 VS Code (或 Jupyter Notebook) 安裝確認
  - Google Colab
- 申請 OpenAI API Key
- 開源模型的寶庫：Hugging Face
  - AI 模型的 GitHub)

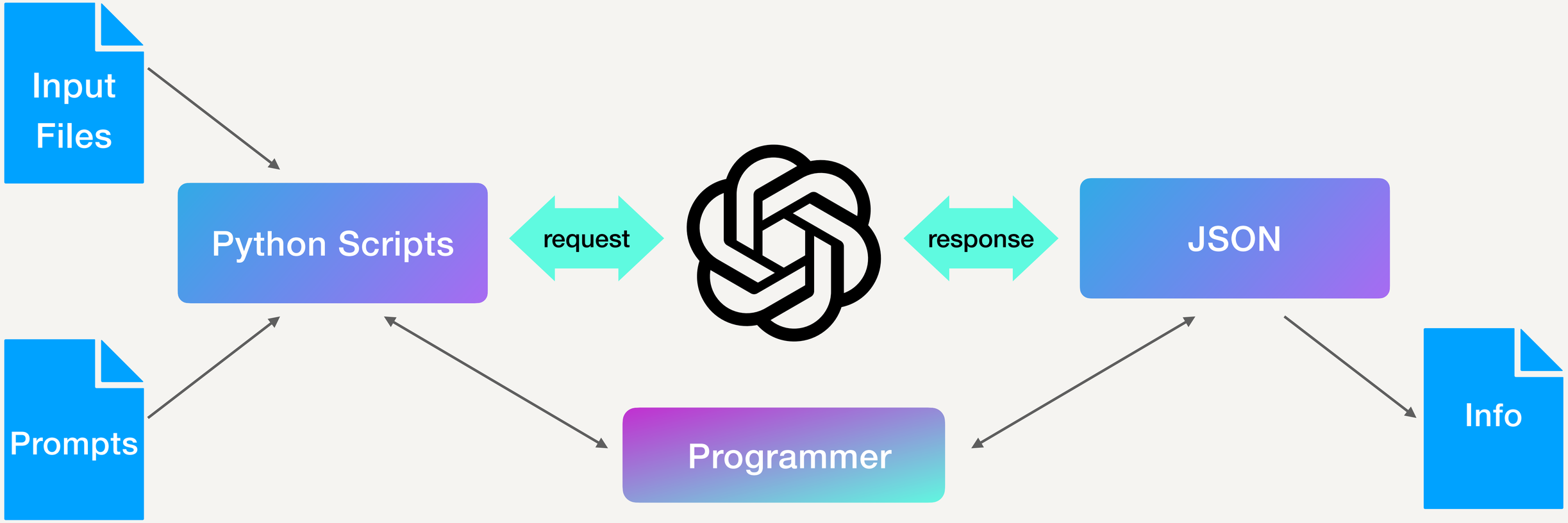
本地端語言模型

- 使用 transformers 函式庫載入與運行模型
  - pipeline vs. AutoModelForCausalLM + AutoTokenizer
- ollama 使用模型
- Live Demo: 載入模型並進行對話

OpenAI API設置與使用

- 訪問 platform.openai.com
- 註冊/登入帳號
- 前往 API Keys 頁面
- 創建新的 API Key
- 安全保存 Key（只顯示一次）
- 察看免費額度

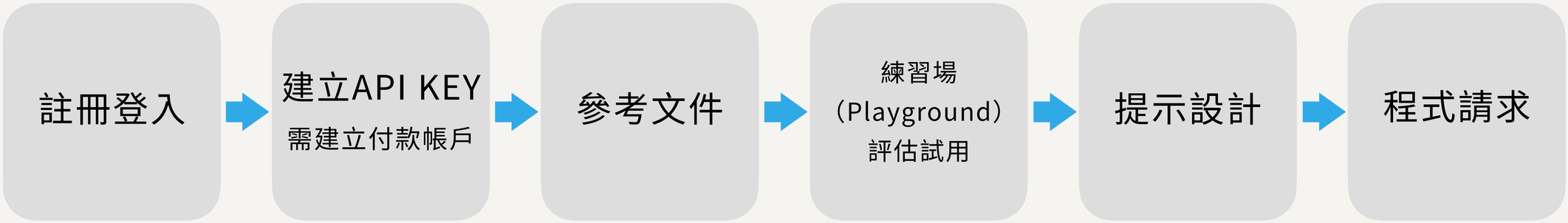
LLM API Usage Flow



ChaGPT logo, 丁志仁, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons

Copyright © 2024 Steve Lai & Mathison Intelligence. All rights reserved.

OpenAI API Use Steps



Copyright © 2024 Steve Lai & Mathison Intelligence. All rights reserved.

不使用框架的優點

- 更深入理解： 掌握底層運作原理
- 靈活性更高： 可完全客製化流程
- 效能優化： 避免不必要的抽象層
- 除錯更容易： 直接控制每個步驟

提示工程的原則

- 了解你的目標：定義目標，並瞭解透過提示取得資訊、創意還是解決問題？
- 保持清晰簡潔：避免過於複雜或模糊的提示。清晰的提示能有更準確的回應。
- 重視上下文：提供足夠的背景資訊以便 LLM 理解情境，但避免不必要的資訊。
- 嘗試和迭代：根據你得到的回應，嘗試不斷的改善提示。迭代是找到最有效措辭的關鍵。
- 考慮你的受眾：根據會與 AI 回應互動或受眾來調整你的提示。
- 評估和調整：不斷評估提示的有效性，並準備隨時進行調整。

Prompts Engineering Principles

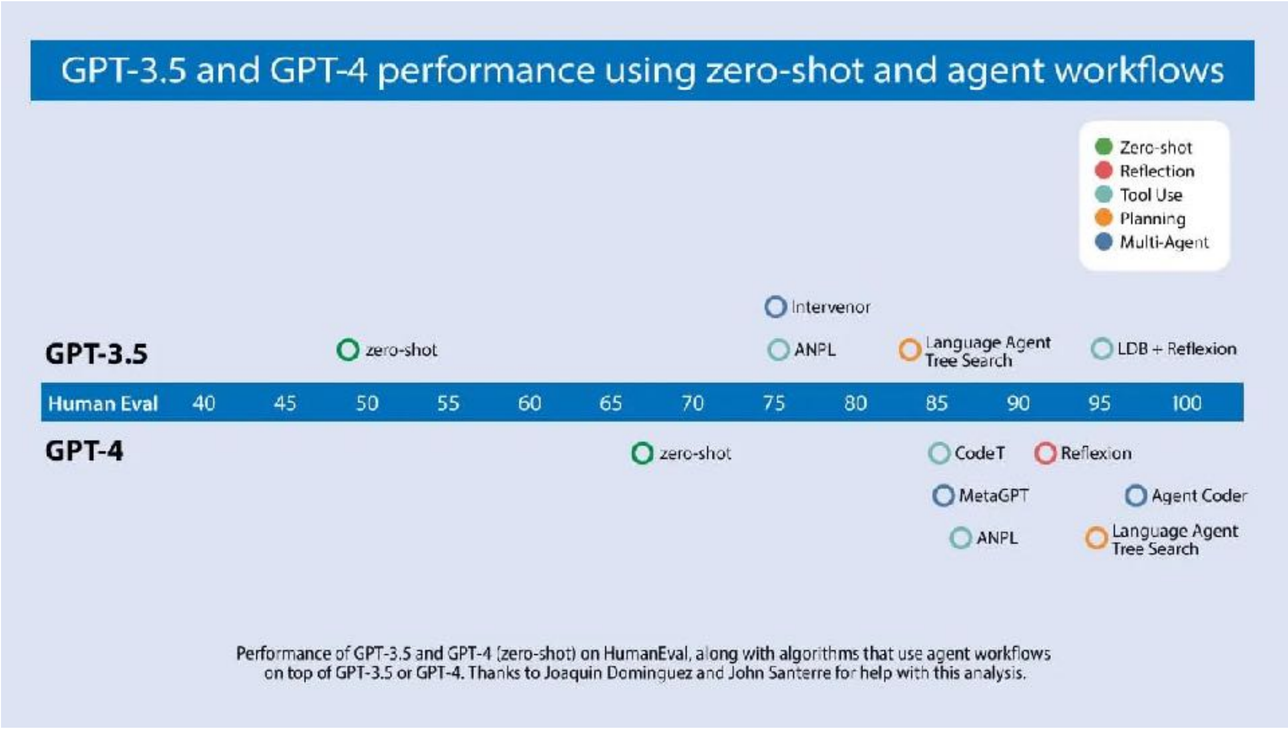
- 兩個prompt的原則
  - 寫出清楚明確的指示
  - 給模型"思考"的時間
- 寫出清楚明確的指示
  - 使用界定符號清楚指示輸入的不同部分
  - 要求結構性的輸出 ，e.g., JSON, HTML
  - 要求模型檢查條件是否滿足
- 給模型思考的時間
  - 指定完成任務所需的步驟
    - 指定回答格式
  - 指示模型製定出自己的解決方案，在模型匆忙下結論之前

情緒分析

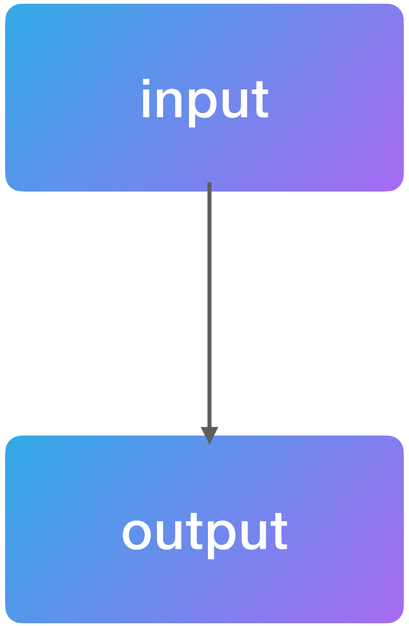
- 原始回答會一個句子回覆
- 指定回答的方式（Positive or Negative）
- 列出有的情緒
- 辨認文章是否出現某情緒
- 從文章萃取資訊、整理成JSON格式
- 一次多個任務
- 所有問題一次問是可以的，只要條列清楚問題，並加上清楚的指示。

Agentic Patterns Prompting

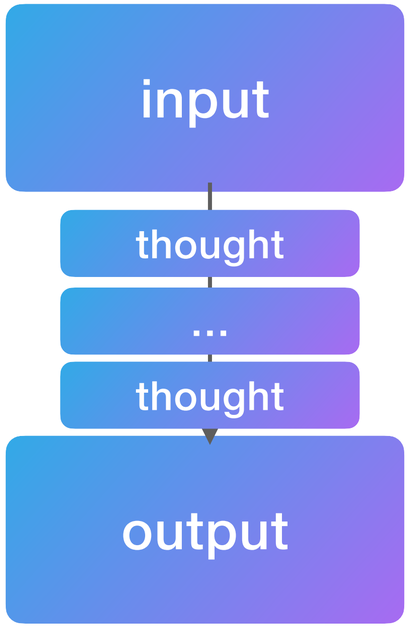
- Reflection
  - LLM 自己審視自己回覆的結果並改善
- Tool Use
  - 讓 LLM 擁有像是網路收尋、程式執行或其他可以取得資料、資料分析或是採取行動
- Planning
  - LLM 採取更細部的步驟，有計畫的一步一步去達成目標
- Multi-agent
  - 許多的 agent 一起合作，互相討論或是互相辯論後產生出結果



Chain of Thought（CoT）



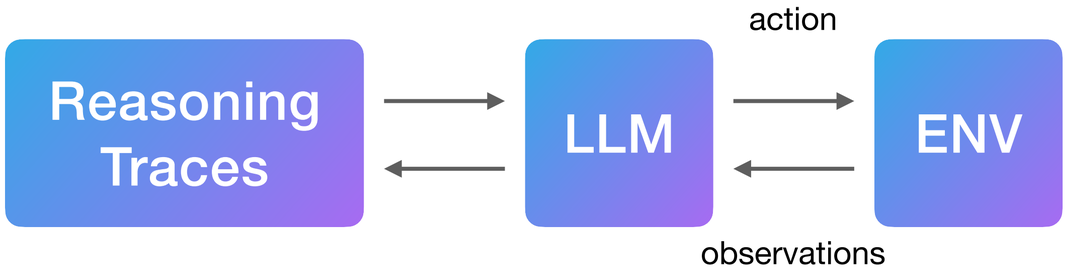
Standard



CoT

ReAct Prompting

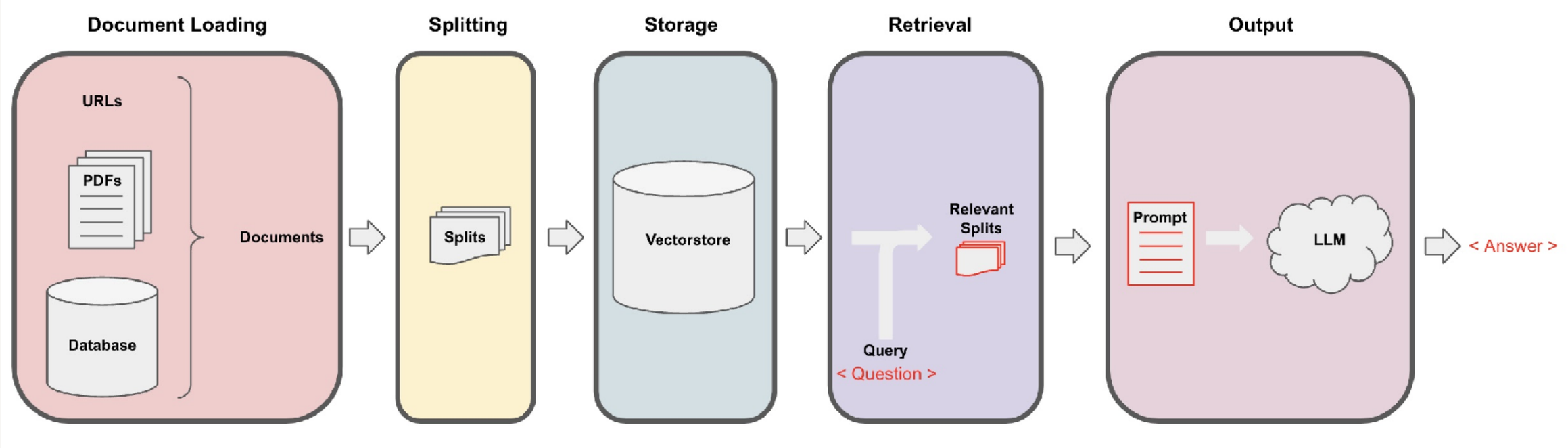
- Reasoning + Acting
  - 結合推理與行動的提示方法
- 推理 (Reasoning)
  - 理解和分解任務
  - 多次的推論和判斷，逐步接近問題的解決方案
- 行動 (Acting)
  - 在推理過程中，可執行行動
  - 如查詢資料庫、訪問 API、與其他系統合作



ReAct Prompting Application

- 互動問答系統：
  - 查詢外部資料庫或執行多步操作的問答系統
- 任務導向對話系統
  - 例如預訂航班、處理客服請求等
- 複雜問題求解
  - 科學研究、數學推理等

Retrieval Augmented Generation（RAG）



Copyright © 2024 Steve Lai & Mathison Intelligence. All rights reserved.

Source Code

- https://github.com/lzrong0203/iSpan_LLM09