

LLM 大型語言模型 介紹與企業結合

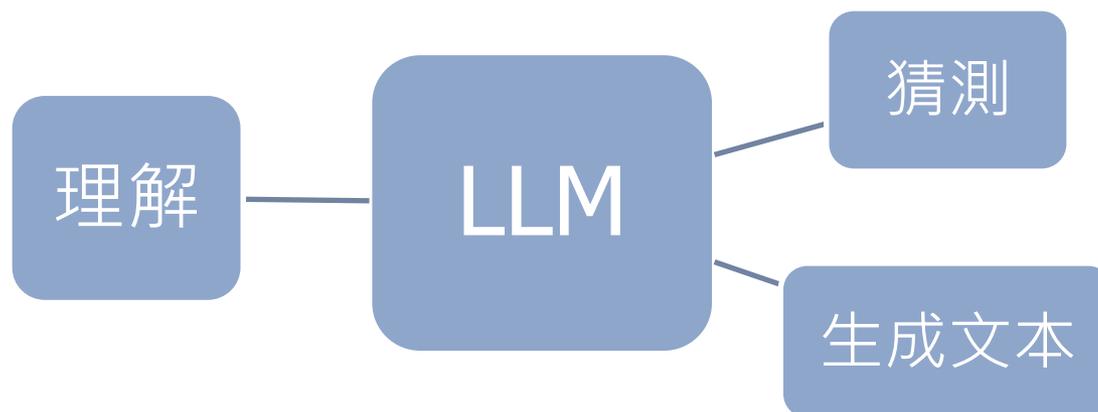
報告者：許嫚軒

指導教授：邱銘傳 教授

背景介紹

What

- 什麼是LLM (Large Language Model) ?
 - LLM是一種深度學習模型，目的為理解和生成人類語言。
 - 可以猜測文本接下來的內容，並透過上下文判斷文意。



What

- LLM是具有**超過 1,000 億個參數**的NLP系統

- 是NLP領域的應用，專注於高質量的文本生成和理解，如自動寫作、聊天機器人
- 藉由大量的文本訓練(新聞、書籍、網頁等)
- 通常基於先進的深度學習架構，如Transformer。
- 例如GPT系列、BERT、T5等大型預訓練模型。

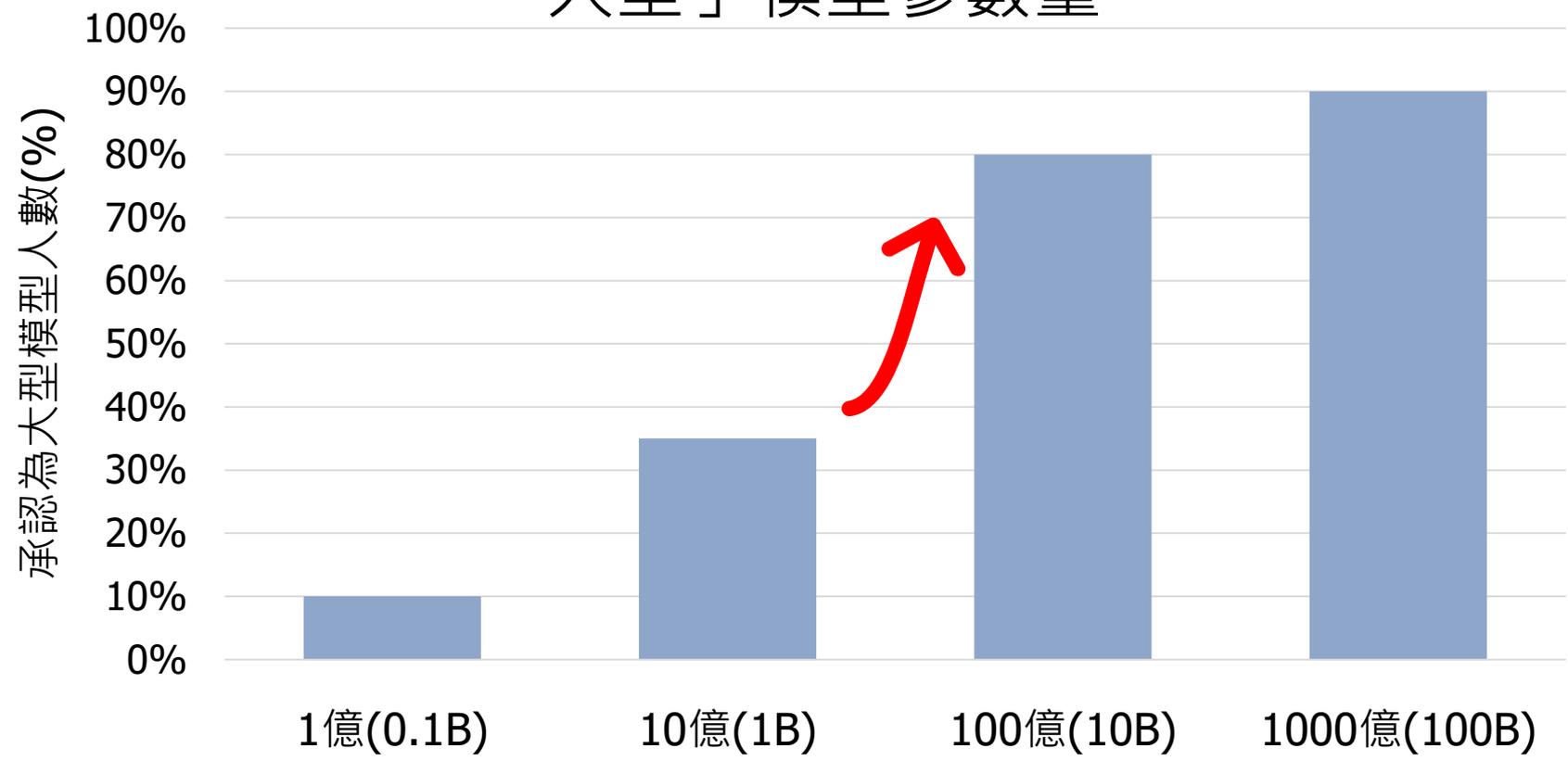
What - 「大型」？

- 多少參數稱作「大型」？
 - 2023年3月為止
 - 有10%人認為1億(0.1B)參數可稱為大型語言模型
 - 有35%人認為10億(1B)參數可稱為大型語言模型
 - 有80%人認為100億(10B)參數可稱為大型語言模型
 - 有95%人認為1000億(100B)參數可稱為大型語言模型

參數量越多，模型理論上可捕捉更細微的語言差異和結構，產生更彈性的回答。

What - 「大型」？

「大型」模型參數量



[2]

Ex.

Llama-2 7B

Llama-2 13B

ChatGPT-3.5

What - 「大型」？

- **54B** 是模型湧現能力的門檻
- **湧現能力(emergent capability)**：
指的是大語言模型(如GPT)在生成文
本時所展現的創造性和創新性。



What - 「大型」？

- 記憶體估計：
- 每個參數通常需4byte儲存。
- 100億參數的模型大約需要40GB記憶體來存儲參數。
- 運行時還需要額外記憶體需求。

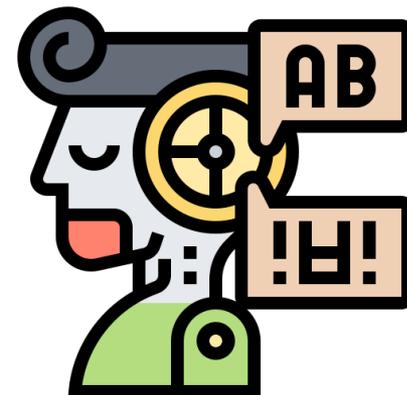


What – NLP vs. LLM

- **NLP (自然語言處理)**

- 使電腦理解、解釋和生成人類語言的技術。
- 詞性標注到複雜的對話和問答系統。
- 包括規則式方法和機器學習的方法。
- 例如語音識別系統Siri、

自動翻譯工具Google翻譯。



What – NLP vs. LLM

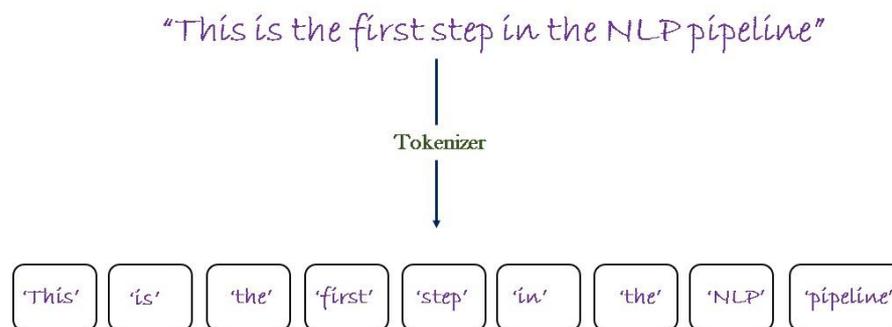
- LLM(大型語言模型)和
NLP(自然語言處理)的差別？

大型語言模型(LLM)是自然語言處理
(NLP)領域的其中一種應用



What - token

- Token –
 - 文本處理的基本單位
 - 可以是單詞、子詞 (subword)、字符 (character) 等
 - 分詞器 (**tokenizer**) 決定幾個字母組成幾個token
 - 例如GPT-3採用的Byte Pair Encoding (BPE)



What - token

- GPT-3.5 語言模型：每次輸出最高上限為 4096個 token(約2000中文字或3400個英文詞)

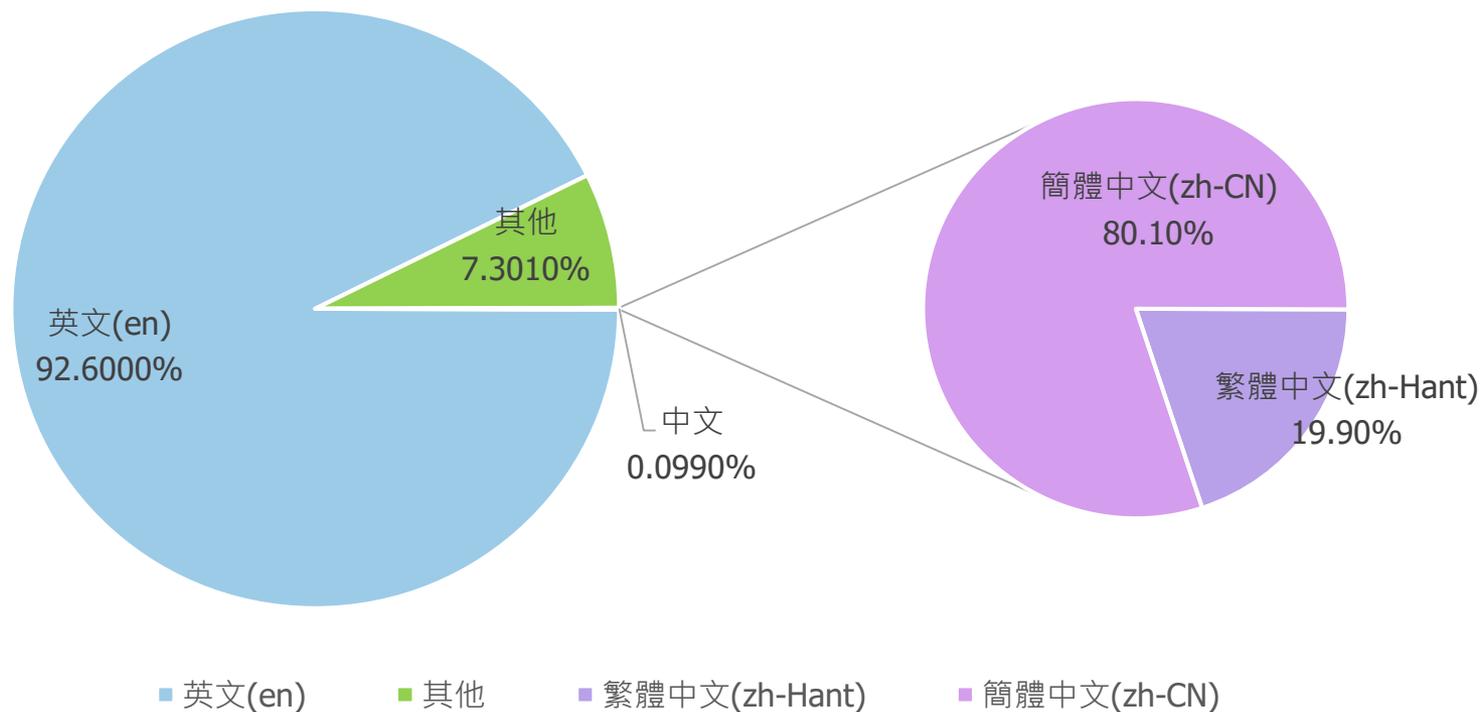
	繁體中文(字元)	英文(單詞)
每個字耗費的平均 token數	2.03	1.25
每 1 個 token 的平均字數	0.49	0.84

What - token

<p>GPT-3.5 & GPT-4 GPT-3 (Legacy)</p> <div style="border: 1px solid #ccc; padding: 10px; min-height: 200px;"> <p>我想吃漢堡</p> </div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> Clear Show example </div> <table border="0" style="margin-top: 10px;"> <tr> <td>Tokens</td> <td>Characters</td> </tr> <tr> <td style="font-size: 2em;">9</td> <td style="font-size: 2em;">5</td> </tr> </table> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>我 [?] [?] [?] [?] [?] [?] [?] [?]</p> </div>	Tokens	Characters	9	5	<p>GPT-3.5 & GPT-4 GPT-3 (Legacy)</p> <div style="border: 1px solid #ccc; padding: 10px; min-height: 200px;"> <p>I want to eat burger</p> </div> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> Clear Show example </div> <table border="0" style="margin-top: 10px;"> <tr> <td>Tokens</td> <td>Characters</td> </tr> <tr> <td style="font-size: 2em;">5</td> <td style="font-size: 2em;">20</td> </tr> </table> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>I want to eat burger</p> </div>	Tokens	Characters	5	20
Tokens	Characters								
9	5								
Tokens	Characters								
5	20								

What - token

Open AI預訓練資料庫



What – chatbot ?

- 和一般的chatbot有甚麼差別?
- ChatGPT是結合了LLM的chatbot
 - 分辨提問中的多個意圖
 - 結合知識庫
 - 產出更貼切的答案



Why-優缺點(1/5)

- LLM能為人類帶來的好處?
- **優點**

1.提高工作效率和生產力

- 幫助使用者彙整資料、解答問題以及撰寫程式，甚至提供更好的方案。Ex. 內容生成和靈感提供

2. 查找程式語言的bug或自然語言文章不通順處

- LLM能夠快速檢閱程式碼，提出可能的錯誤點。

Why-優缺點(2/5)

3. 功能不僅限於人類的語言

- 生物學的語言(如蛋白質、分子序列)理解、分析。

4. 提供更個人化的服務

- 語言模型可以通過處理大量數據來瞭解客戶行為和偏好，推動個人化。 Ex. 電子商務

5. 易於使用

- 只需以一般與人溝通方式下達指令，易於使用。

Why-優缺點(3/5)

- 缺點

1. 維護以及研發的經費龐大

- 建構基礎模型通花費數月的培訓時間和數百萬美元，且後續擴展與維護同樣常需要大量的資金。Ex.

ChatGPT

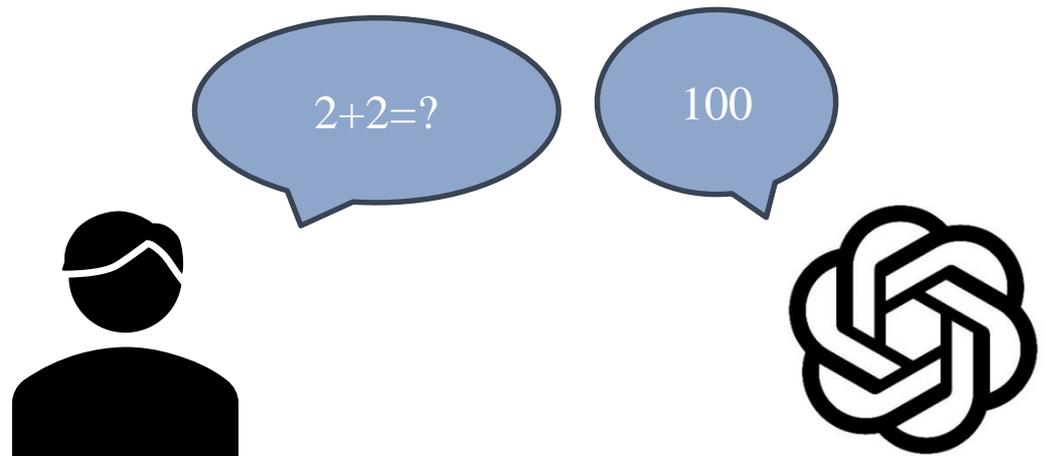
2. 可能產生有歧視的言論

- 若訓練數據集沒有經過檢查或修正，語言模型可能會做出種族主義或性別歧視的評論。Ex. Microsoft-Tay

Why-優缺點(4/5)

3. 錯誤資訊的產生(幻覺, Hallucination)

- 當模型的訓練量不夠，遇到他不知道的答案時，會自行編造出假的答案。
- 需有過程監督，給予正面回饋



Why- 過程監督(5/5)

3. 錯誤資訊的產生(幻覺, Hallucination)

– 過程監督(Process Supervision)：

- 不只在結果正確時給予正面回饋
- 在生成過程中，若推理過程正確，即給予正面回饋
- 若生成過程中發現模型產生誤導性的信息，可以進行干預，調整訓練數據或模型參數，以改善結果

Ex. ChatGPT開發的過程監督

When

- 發展歷史

年份	模型名稱	描述 / 創新點
2017	<u>Transformer</u>	引入了全注意力機制，去除了循環結構，大幅提升了訓練速度和效果。
2018	<u>BERT</u>	通過雙向Transformer的預訓練，提升了文本理解的深度和精度。
2019	<u>GPT-2</u>	通過更大的數據和模型規模，提升了生成文本的流暢性和多樣性。
2020	<u>GPT-3</u>	進一步擴大模型規模，達到了令人驚異的文本生成和多任務處理能力。
2021	<u>T5</u>	將各種NLP任務統一成文本到文本（text-to-text）的框架，提高了模型的通用性。

Where

LLM可以應用在哪裡？



創意發想



程式設計



文章整理



醫療方法



業務協助

Where - 創意發想



創意發想

- 廣告和營銷策略

提出獨特的廣告創意和營銷策略，幫助品牌以新穎的方式吸引目標受眾。Ex. [Influenxio](#) 圈圈

- 設計靈感

根據特定主題或風格提供設計靈感，例如時尚設計、建築設計，甚至是網頁和平面設計。Ex. [Pimento](#)

Where – 程式設計



程式設計

- 程式檢查、生成、接寫及建議

在編程過程中，根據已寫代碼的上下文，大型語言模型能夠智能推薦代碼補全選項或建議，提升開發效率。

- Copilot、Code-llama協助

專為程式生成而生的LLM大型語言模型，可從自然語言描述中生成或接寫代碼。

Where – 文章整理



文章整理

- 自動摘要生成：
 - LLM能夠讀取長篇文章並生成簡潔的摘要，幫助讀者快速掌握文章的主要內容。Ex. Microsoft - [UniLM模型](#)
- 關鍵詞提取：
 - 通過分析文章的內容，LLM能自動識別並提取關鍵詞，便於快速瞭解文章的核心話題和關注點。

Where – 醫療方法開發



醫療方法開發

- 疾病診斷輔助

- 分析病人的症狀描述和醫學記錄，提供初步診斷建議或識別可能的健康問題。 Ex. Google - [Med-PaLM 2](#)、智海系統（gHi system）

- 藥物相互作用預測

- 利用LLM分析藥物數據庫，預測潛在的[藥物相互作用](#)和副作用，從而提高治療的安全性。

Where – 醫療方法開法



醫療方法開發

- 醫院碳管理

- 分析醫院的能源消耗數據，識別能源浪費的區域，並提出改進建議以減少碳排放。
- 利用LLM分析產品碳足跡數據，為醫院提供綠色採購的建議，以購買低碳排放的產品和設備。

Where – 業務協助



業務協助

- 市場趨勢分析：

- 利用LLM分析大量數據和新聞來生成市場趨勢分析報告，幫助企業做出更有根據的決策。

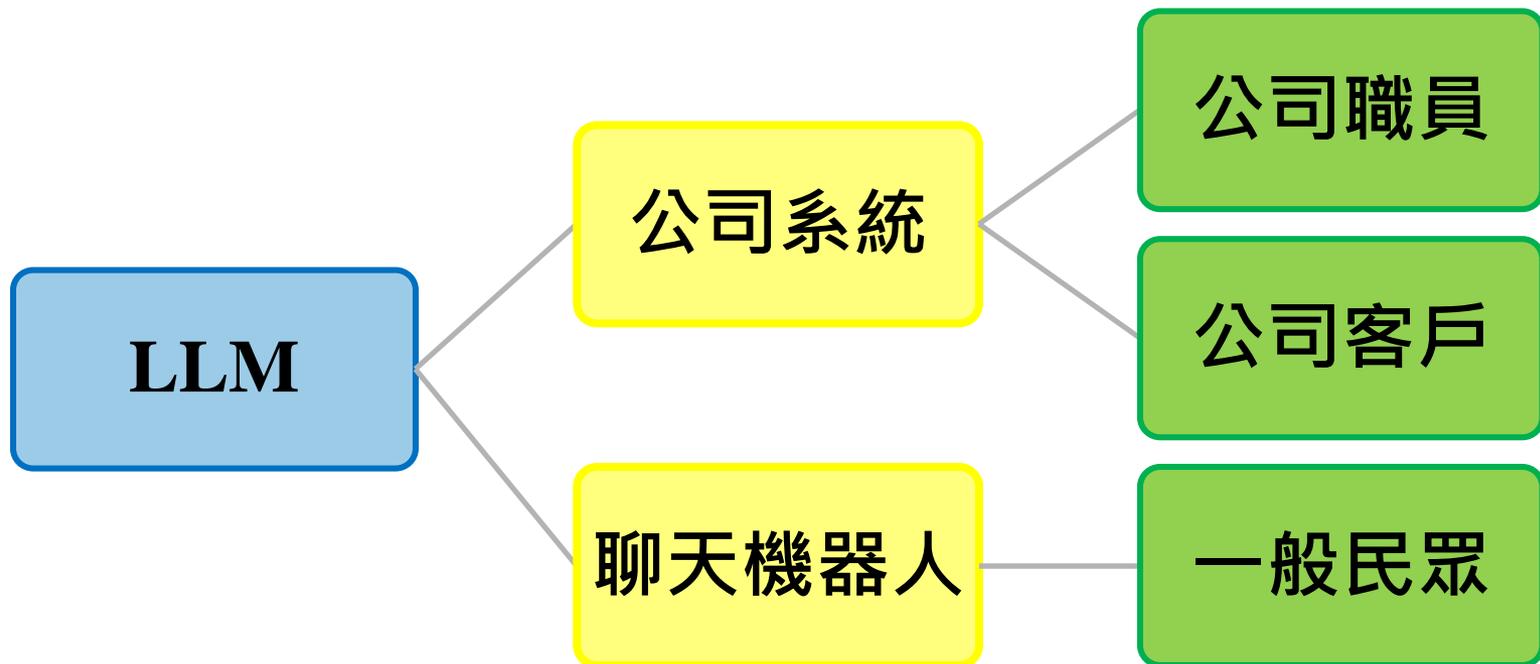
- 客戶服務和支援：

- 即時提供的客戶支援，個人化解答常見問題，並根據客戶的具體需求提供個人化建議，提升客戶滿意度。

Ex. [AIXPERT](#)

Who

- 所有人都用的到LLM



Who – for 台灣

- **TAIDE** (Trustworthy AI Dialogue Engine)
- 國科會打造台灣專用的LLM模型
- 於2023年6月發布7B模型
- 基於Meta - LLaMa模型建立，微調訓練繁體中文數據集。
- 強調其輸出的透明度和可靠性。

Who – for 台灣

- **TAIDE (Trustworthy AI Dialogue Engine)**
- 目的：
 - 理解和回應本地文化、語言、價值觀和習俗，旨在滿足台灣用戶的特定需求。
 - Trustworthy 可信任的
- 英業達、叻揚、凌群、群聯、台智雲、Ai3、國泰及玉山金控等企業也都取得模型並嘗試導入實際應用

Who – for 機密資料

- **AOAI(Azure Open AI) 平台**
- 檢索增強生成(RAG)方法，將自家知識管理系統的人資類基礎文件作為檢索對象
- 附上資料來源，來降低誤解風險

Who – for 機密資料

- **AOAI(Azure Open AI) 平台**

- **優點**

1. **易於整合**：可以與Azure生態系統中的其他服務和工具輕鬆整合，方便企業和開發者使用。
2. **安全性和合規性**：微軟Azure平台提供高標準的安全性和合規性，這對於處理敏感數據特別重要。
3. **高級語言模型技術**：結合Open AI的語言模型，如GPT-3，提供卓越的自然語言處理能力。

Who – for 機密資料

- **AOAI(Azure Open AI) 平台**

- **缺點**

1. **複雜性**：對於非技術用戶來說，配置和管理Azure Open AI可能需要一定的學習曲線。
2. **依賴於特定平台**：將業務緊密結合到Azure平台上可能會增加對特定供應商的依賴。
3. **企業客戶隱私**：同樣是由他人協助保管，雖有公司保障，仍不及本地模型安全。

WHO - 現有模型

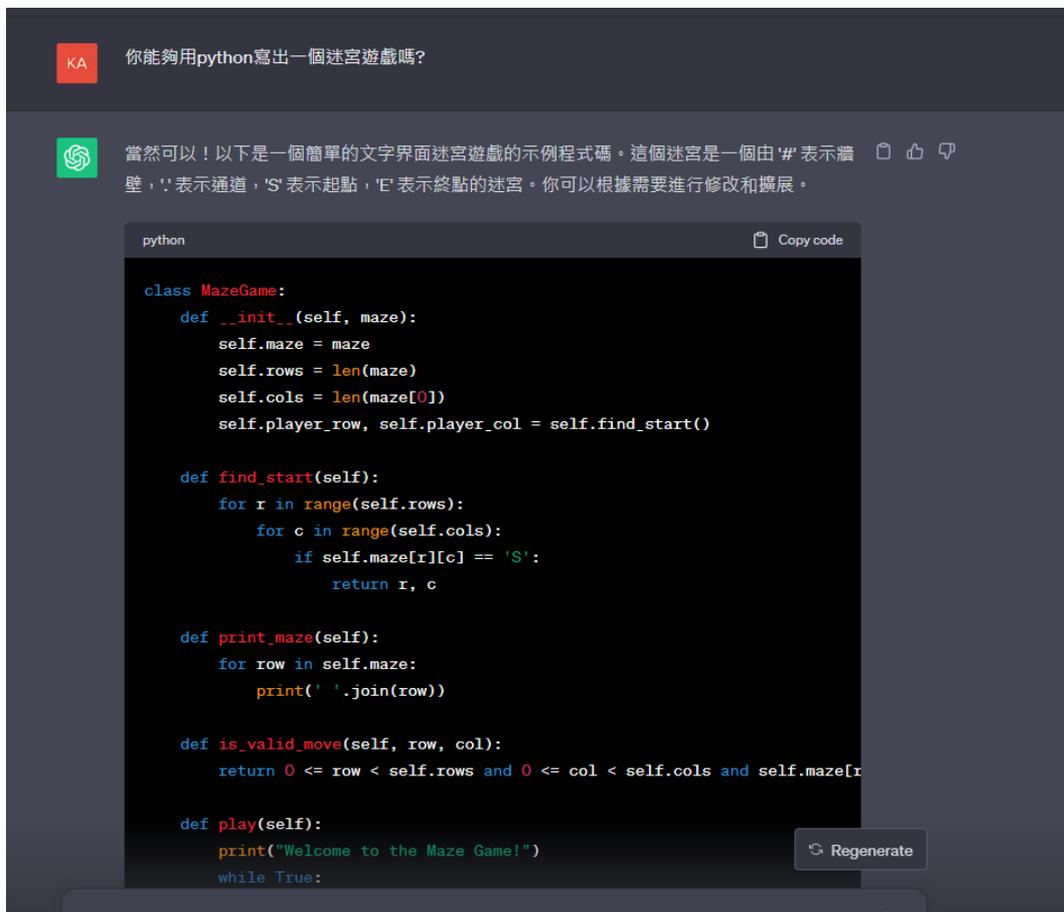
ChatGPT

- ChatGPT :
 - Open AI開發
 - 能理解和生成文本，廣泛用於對話、文章撰寫、問答等任務。
 - 目前使用最廣泛的免費模型為GPT -3.5，另可升級GPT -4

ChatGPT

ChatGPT

• ChatGPT



KA 你能夠用python寫出一個迷宮遊戲嗎?

當然可以！以下是一個簡單的文字界面迷宮遊戲的示例程式碼。這個迷宮是一個由 '#' 表示牆壁，'!' 表示通道，'S' 表示起點，'E' 表示終點的迷宮。你可以根據需要進行修改和擴展。

```
python Copy code  
  
class MazeGame:  
    def __init__(self, maze):  
        self.maze = maze  
        self.rows = len(maze)  
        self.cols = len(maze[0])  
        self.player_row, self.player_col = self.find_start()  
  
    def find_start(self):  
        for r in range(self.rows):  
            for c in range(self.cols):  
                if self.maze[r][c] == 'S':  
                    return r, c  
  
    def print_maze(self):  
        for row in self.maze:  
            print(' '.join(row))  
  
    def is_valid_move(self, row, col):  
        return 0 <= row < self.rows and 0 <= col < self.cols and self.maze[r]  
  
    def play(self):  
        print("Welcome to the Maze Game!")  
        while True:
```

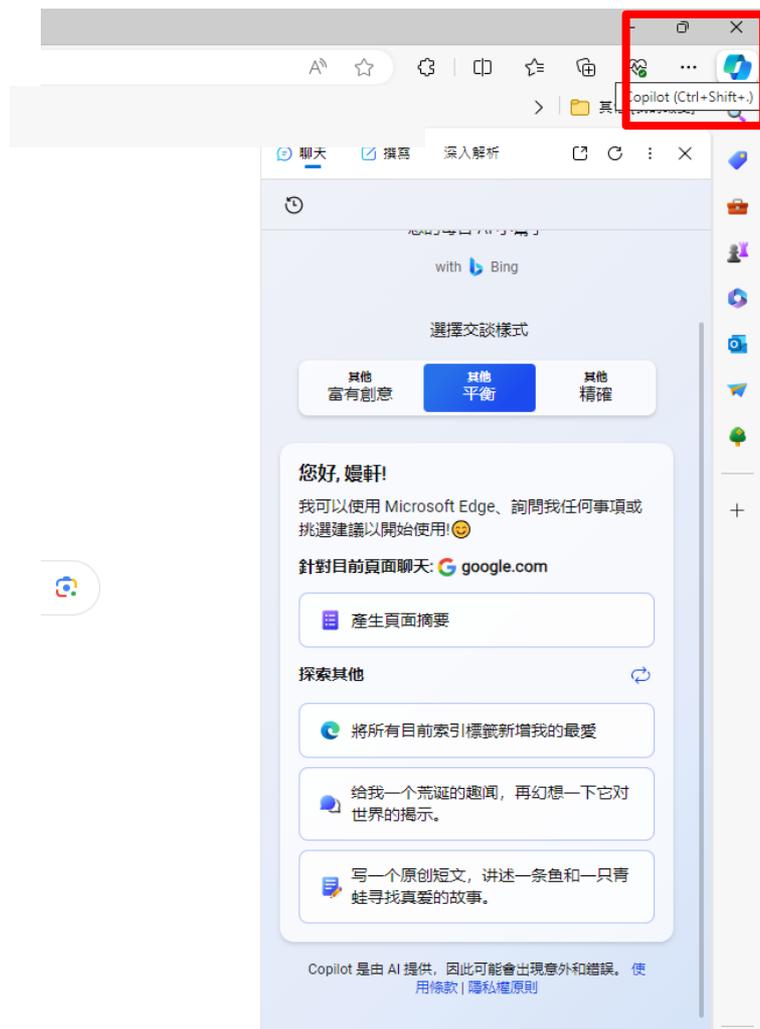
Regenerate

Bing Chat

- Bing Chat :
 - Microsoft 開發
 - 點擊Microsoft Edge瀏覽器右上角即可進入對話
 - 網頁導航輔助、資訊查詢、用戶互動
 - 使用GPT -4模型，可接收和回覆圖片等視覺查詢



Bing Chat



Bloom

- Bloom :
 - Hugging face開發
 - 支援多達 46 種語言，資料涵蓋新聞、書籍、教育資料、百科全書等多個領域致力於生成無偏見、無敵意的文本內容。
- 由聯發科的「MediaTek Research」聯發創新基地開發繁體中文版。(2023-4)



Bloom

A poem about the Large Hadron Collider by Alfred Tennyson.

Title: The Particle at the End of the Universe

The mindless swarm that fills the void,
The blind, unthinking, purposeless crowd
Of atoms, blindly rushing to their doom,
With never a thought of what they are or why,
Or whence they came, or whither they must go;
This is the end of all we see and know,

Hark! from the abyss there comes a sound
As of a mighty wind that shakes the ground,
A sound of rending, tearing, crashing sound,
As if the solid earth were rent in twain,
And all the elements in wild uproar
Were hurled together in a chaotic storm;

And then a pause, and then again
The sound as of ten thousand thunders rolled
Upon the startled ear of night,
Then silence

استخراج العدد العائلي في لغة بايثون:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        result = 1
    for i in range(1, n + 1):
        result *= i
    return result
```

Traduce español de España a español de Argentina
El coche es rojo - el auto es rojo
El ordenador es nuevo - la computadora es nueva
el bolígrafo es negro - lapicera es negra
la nevera - heladera
los zapatos - zapatillas
las gafas - anteojos

BLOOM
a BigScience Initiative

Input 176B params
Output 59 languages
Open-access

Llama-2

- Llama-2 (Low-Level Autonomously Managed Assets) :
 - Meta 開發
 - **完全開源、免費商用**
 - 目標是**自動化管理大量的低階資產**，如同服务器和硬體，並透過**實時模擬**來預測和解決問題。能預見性地避免硬體故障，大幅**提升數據中心的可靠性和效率**。

Llama 2

Llama-2-7B
Llama-2-13B
Llama-2-70B

Mistral 7B

- Mistral 7B :

- Mistral AI開發

- **完全開源**

- 擁有7B大型語言模型，效果與Llama2 13B模型並駕齊驅，被稱為目前最強的7B大型語言模型之一

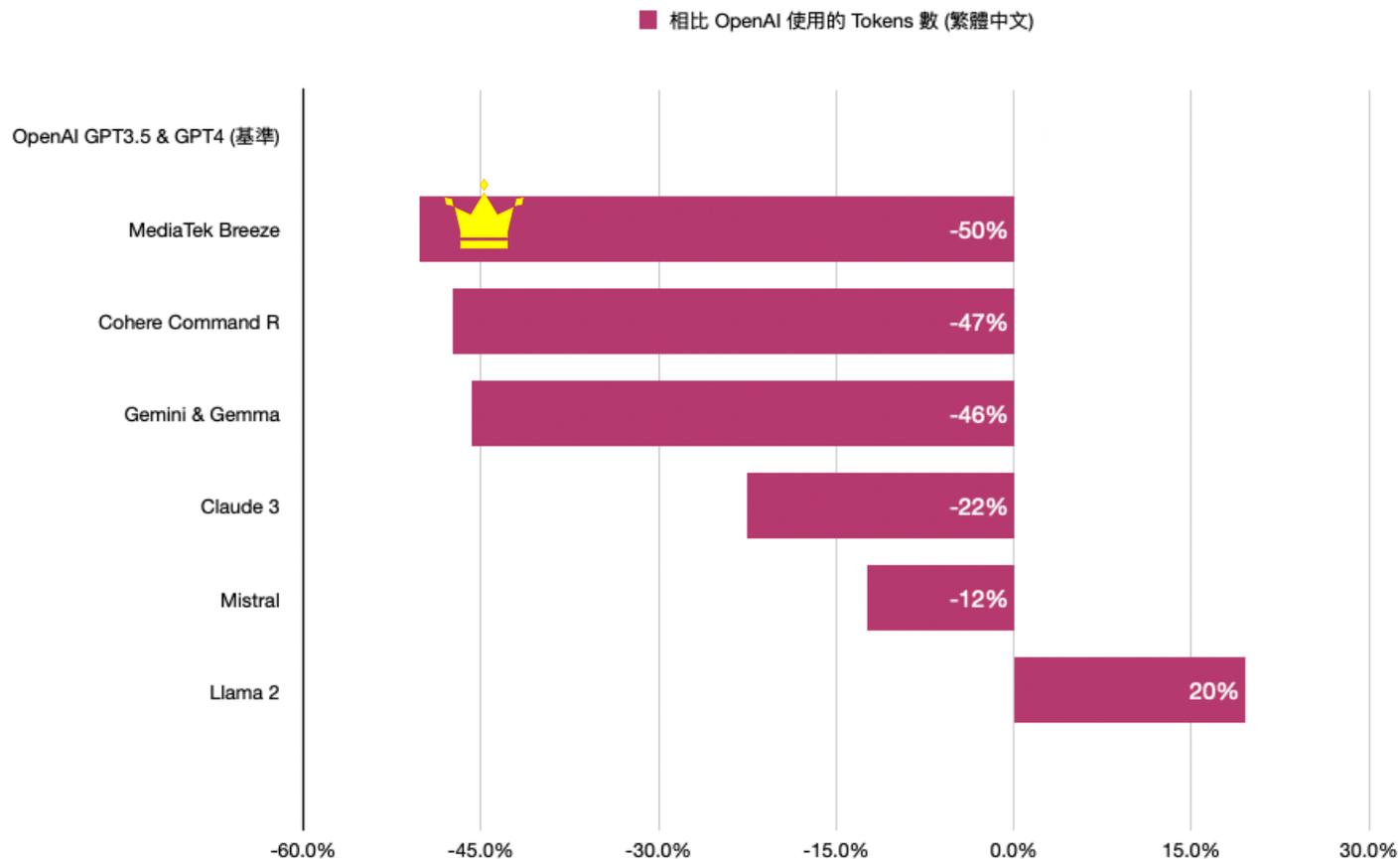
比較

模型	發布日期	開發者	開源	特點	參數量	上下文長度 (token)
GPT-3	2022-11	Open AI	X	ChatGPT, BingChat 應用 目前已經迭代到 GPT-4.5	175B	4096
Bloom	2022-07	Hugging Face	V	聯發科發展繁體中文 版Bloom-zh	176B	2048
Llama 2	2023-06	Meta	V	完全開源、 免費商用、 TAIDE繁中版	7,13,33, 65B	4096
Mistral 7B	2023-9	Mistral AI	V	完全開源、 免費商用	7B	4096-16k

繁體中文模型比較

模型	開發者	開源	微調來源
BreeXe	聯發科	X	Mistral 8x7B
Breeze 7B	聯發科	V	Mistral 7B
TAIDE 7B	國科會	V	Llama2
Bloom-zh	聯發科	V	BLOOM

繁體中文模型比較



硬體設備 - 比較

模型	最小VRAM需求	推薦GPU	微調VRAM需求
Llama – 2 7B	6GB	RTX 3060, GTX 1660, 2060, AMD 5700 XT, RTX 3050	24GB
Mistral 7B			
Llama – 2 13B	10GB	AMD 6900 XT, RTX 2060 12GB, 3060 12GB, 3080, A2000	64GB

How – 訓練方法

GPT Assistant training pipeline



How – 訓練方法

預訓練



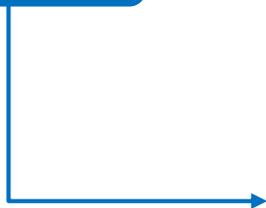
監督微調



獎勵建模



強化學習



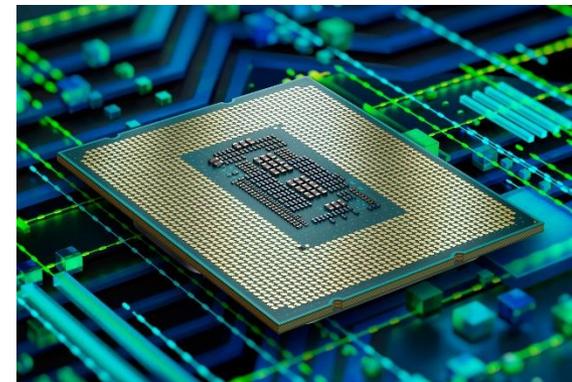
數據量及訓練所需資源巨大

How - 在CPU上預訓練的LLM

- ThirdAI的BOLT 2.5B
 - 主要依靠ThirdAI的動態稀疏性
 - LLM運行成本大大降低
 - 能力與GPT2-x1相當
 - BOLT 2.5B 使用串接10台雙路56核心CPU的伺服器，可處理**2B tokens/天**
 - GPT2-x1 使用128個V100 GPU可處理**17B tokens/天**
- Ex. 20分鐘用CPU轉換莎士比亞文本風格(44,000tokens)

How - 在CPU上預訓練的LLM

- BOLT 2.5B
- 優點
 - 降低算力需求，不須要很多顆昂貴的GPU才能達成
 - 可使用CPU微調
- 缺點
 - 上下文長度有限，限制理解和生成能力
 - 算力不足應付新課題，可發展性不足



監督微調 (Fine-tuning)

根據特定的應用需求來調整和優化模型的表現

- **特定目標：**

- 微調過程專注於特定的任務或應用，例如文本分類、情感分析、問答系統等。

- **小規模數據集：**

- 與預訓練相比，微調通常使用較小的、針對特定任務標注的數據集。

監督微調 (Fine-tuning)

根據特定的應用需求來調整和優化模型的表現

- **參數更新：**

- 在微調過程中，模型的權重和參數會根據特定任務的數據進行更新，以提高在該任務上的表現。

- **節省時間和資源：**

- 通過在已預訓練的基礎模型上進行微調，可以節省大量的訓練時間和計算資源。

Ex. 加入其他語言至NLLB翻譯模型

檢索增強生成(RAG)

- RAG(Retrieval-Augmented Generation)

- 目的：

- 克服知識局限性：

獲取並整合在**原預訓練數據之外**的資訊，透過外部知識庫檢
索相關信息

- 減少重新訓練消耗：

不需要一直更新資訊重新訓練，僅需利用RAG搭配長時間定
期進行fine-tuning

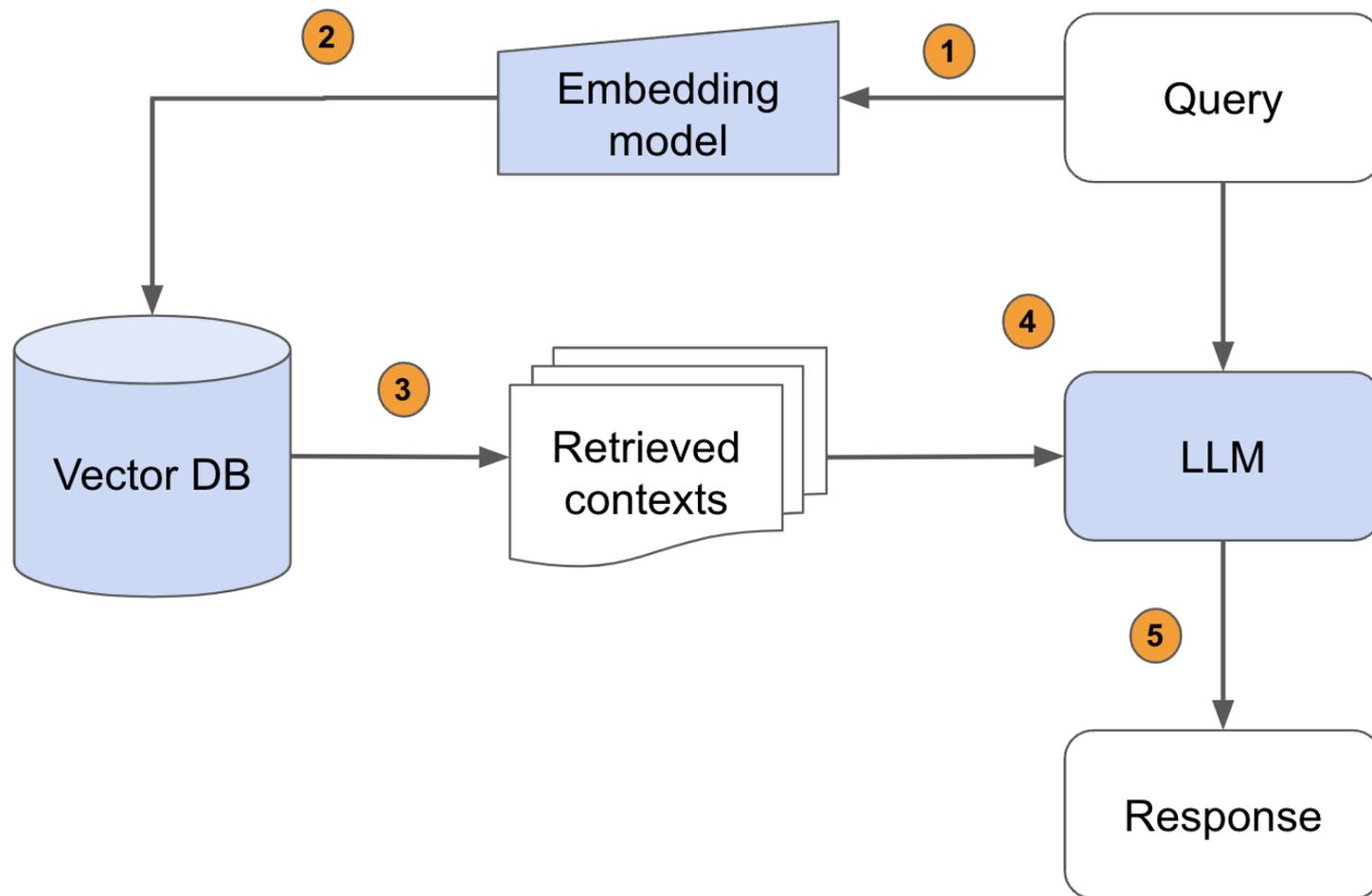
Ex. Azure AI 搜尋使用RAG方法

檢索增強生成(RAG)

- RAG(Retrieval-Augmented Generation)
- 功能：
 - 結合了搜尋檢索和生成能力的自然語言處理架構
 - RAG架構主要分為檢索器和生成器
 - 「大抄」的內容轉譯和整理，就是RAG過程。A4紙張，就是Embedding Vector Database 儲存庫



檢索增強生成(RAG)



檢索增強生成(RAG)

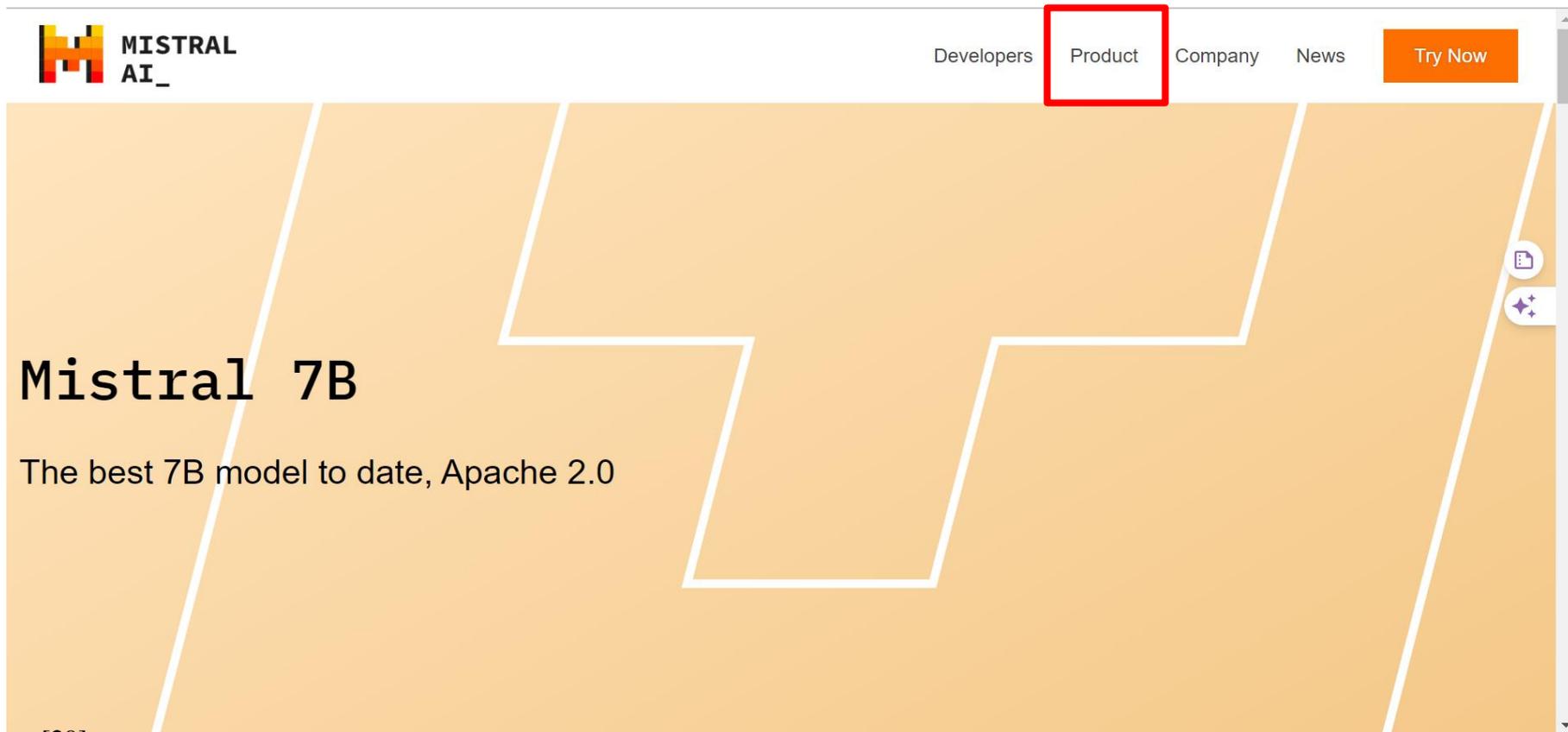
	微調模型	RAG
比喻	就像考試前認真讀書，考試 closed	就像考試 open book，帶筆記去考試，若筆記上有寫可以回答的很好
缺點	訓練模型需要花時間和計算成本，不可能隨時訓練更新資料	仍有 Token 長度限制 要用工具抓資料因此處理時間較長
優點	品質可能更好，這需要機器學習專業知識	不用擔心新資料更新

How – 如何使用？

- 開源模型直接下載(程式碼)
- [LM Studio](#)
- [LangChain](#)
- [Azure Open AI](#)

How – 開源模型下載

- Mistral 7B 為例



The screenshot shows the Mistral AI website. The top navigation bar includes the Mistral AI logo on the left, and links for 'Developers', 'Product', 'Company', and 'News' in the center. A red box highlights the 'Product' link. To the right of the navigation bar is an orange 'Try Now' button. The main content area has a light orange background with a white geometric pattern. The text 'Mistral 7B' is prominently displayed, followed by the tagline 'The best 7B model to date, Apache 2.0'. On the right side of the page, there are two small icons: a document icon and a star icon.

How – 開源模型下載(Locally)



Try Now

Mistral 7B in short

Mistral 7B is a 7.3B parameter model that:

- Outperforms Llama 2 13B on all benchmarks
- Outperforms Llama 1 34B on many benchmarks
- Approaches CodeLlama 7B performance on code, while remaining good at English tasks
- Uses Grouped-query attention (GQA) for faster inference
- Uses Sliding Window Attention (SWA) to handle longer sequences at smaller cost

We're releasing Mistral 7B under the Apache 2.0 license, it can be used without restrictions.

- [Download it](#) and use it anywhere (including locally) with [our reference implementation](#),
- Deploy it on any cloud (AWS/GCP/Azure), using vLLM [inference server and skypilot](#).
- Use it on [HuggingFace](#).



How – 開源模型下載

- Mistral 7B 為例(on Hugging Face)



Developers

Product

Company

News

Try Now

Our models.

MISTRAL 7B

A 7B dense Transformer, fast-deployed and easily customisable. Small, yet powerful for a variety of use cases. Supports English and code, and a 8k context window.

Licence: **Apache 2.0**

Download it

MIXTRAL 8X7B

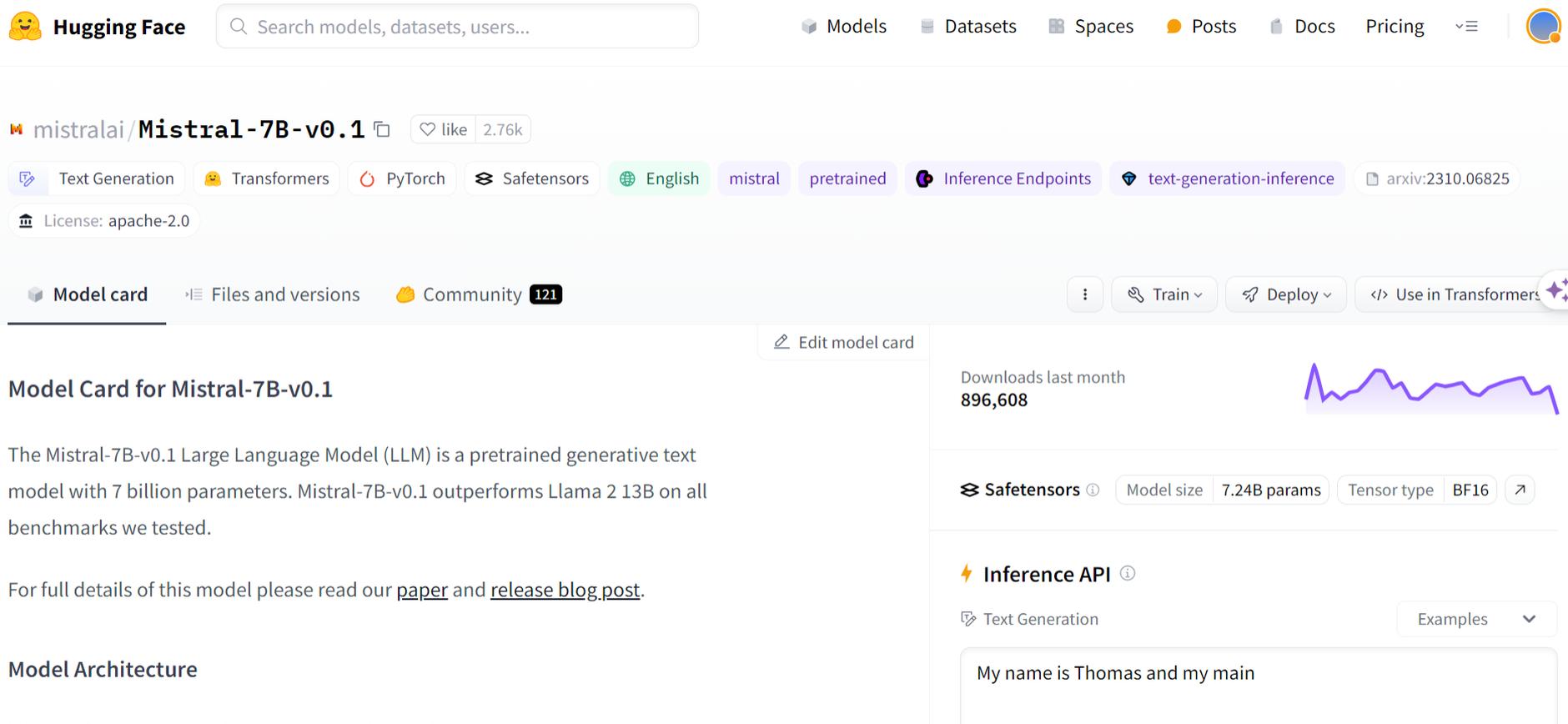
A 7B sparse Mixture-of-Experts model with stronger capabilities than Mistral 7B. Uses 12B active parameters out of 45B total. Supports multiple languages, code and 32k context window.

Licence: **Apache 2.0**

Download it

How – 開源模型下載

- Mistral 7B 為例(on Hugging Face)



Hugging Face Search models, datasets, users... Models Datasets Spaces Posts Docs Pricing

mistralai/**Mistral-7B-v0.1** like 2.76k

Text Generation Transformers PyTorch Safetensors English mistral pretrained Inference Endpoints text-generation-inference arxiv:2310.06825

License: apache-2.0

Model card Files and versions Community 121 Train Deploy Use in Transformers

Model Card for Mistral-7B-v0.1

The Mistral-7B-v0.1 Large Language Model (LLM) is a pretrained generative text model with 7 billion parameters. Mistral-7B-v0.1 outperforms Llama 2 13B on all benchmarks we tested.

For full details of this model please read our [paper](#) and [release blog post](#).

Model Architecture

Downloads last month: 896,608

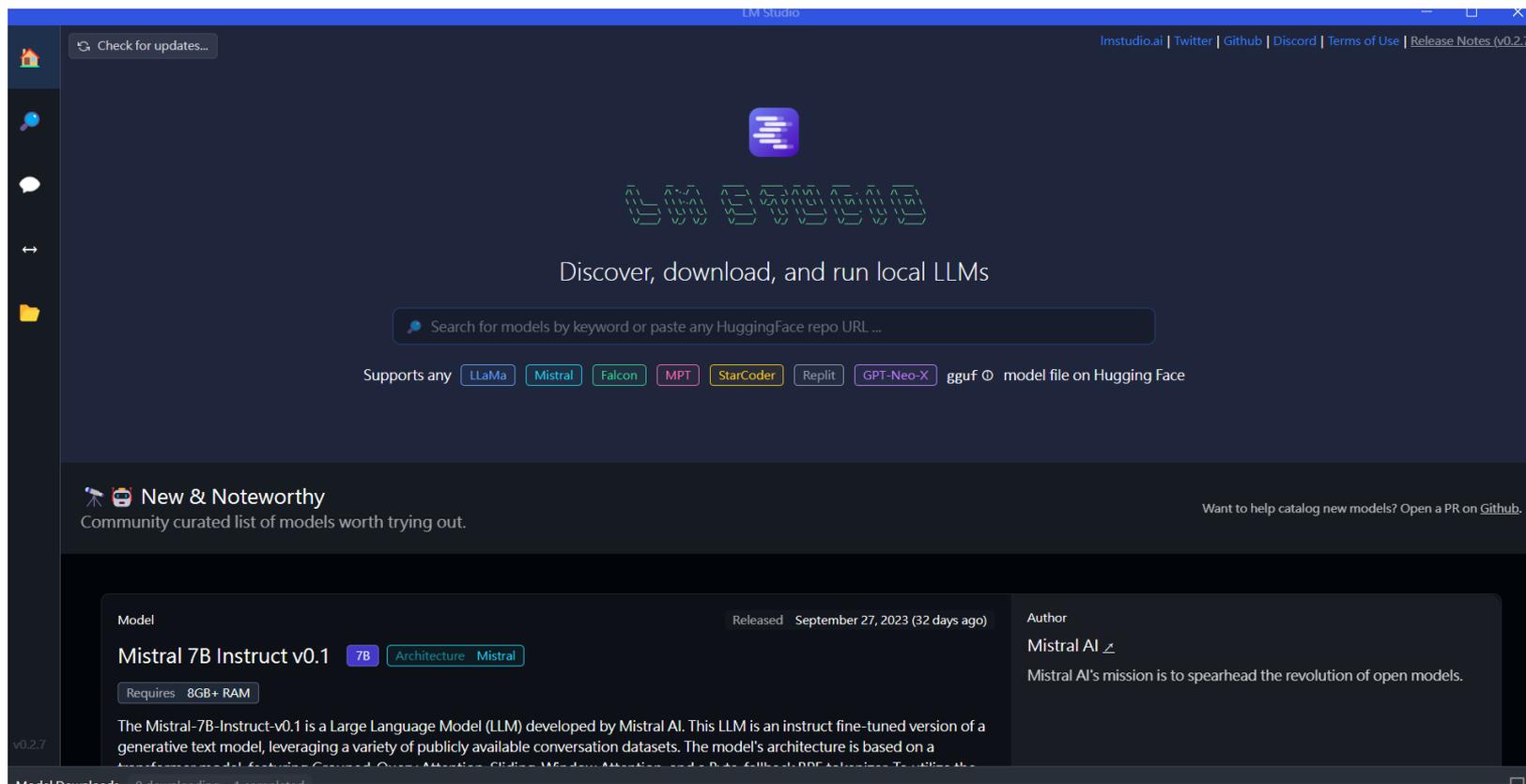
Safetensors Model size: 7.24B params Tensor type: BF16

Inference API Text Generation Examples

My name is Thomas and my main

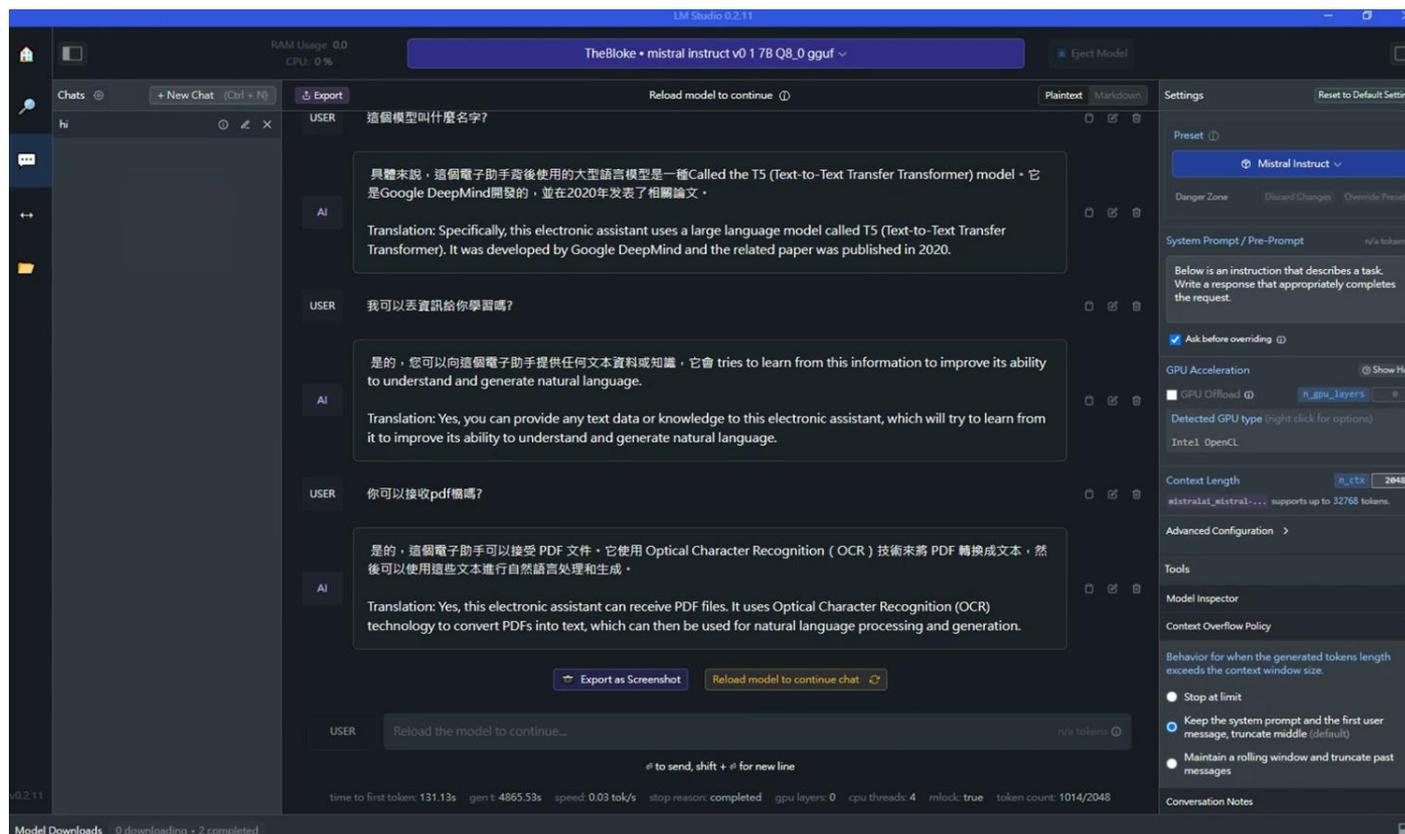
How – LM studio

- LM Studio - 語言模型的使用與管理介面



How – LM studio

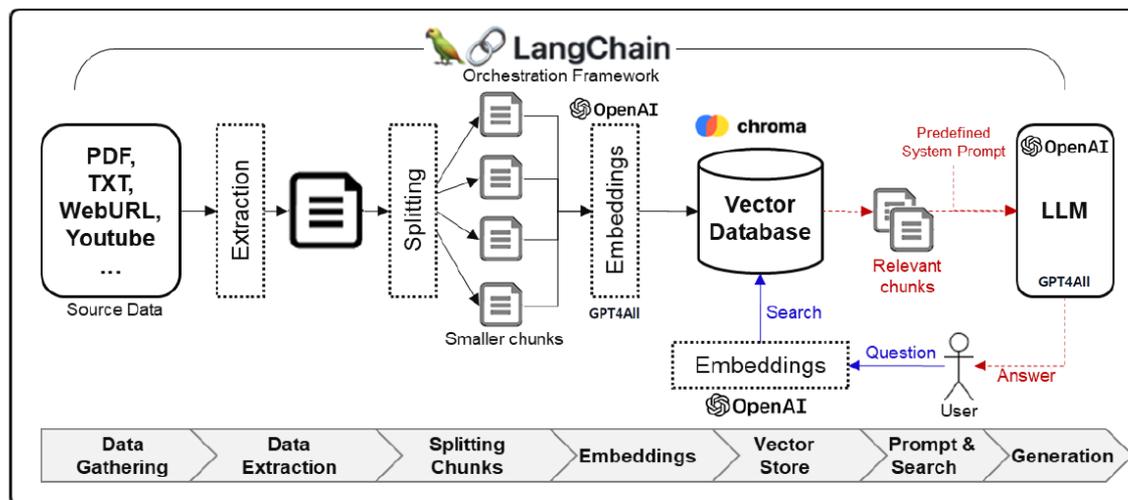
- LM Studio - 語言模型的使用與管理介面



How – LangChain 框架

- LangChain 框架

- 創新框架：改變開發由語言模型驅動的應用程式的方式
- 智能代理：使語言模型能夠與環境進行互動和自適應

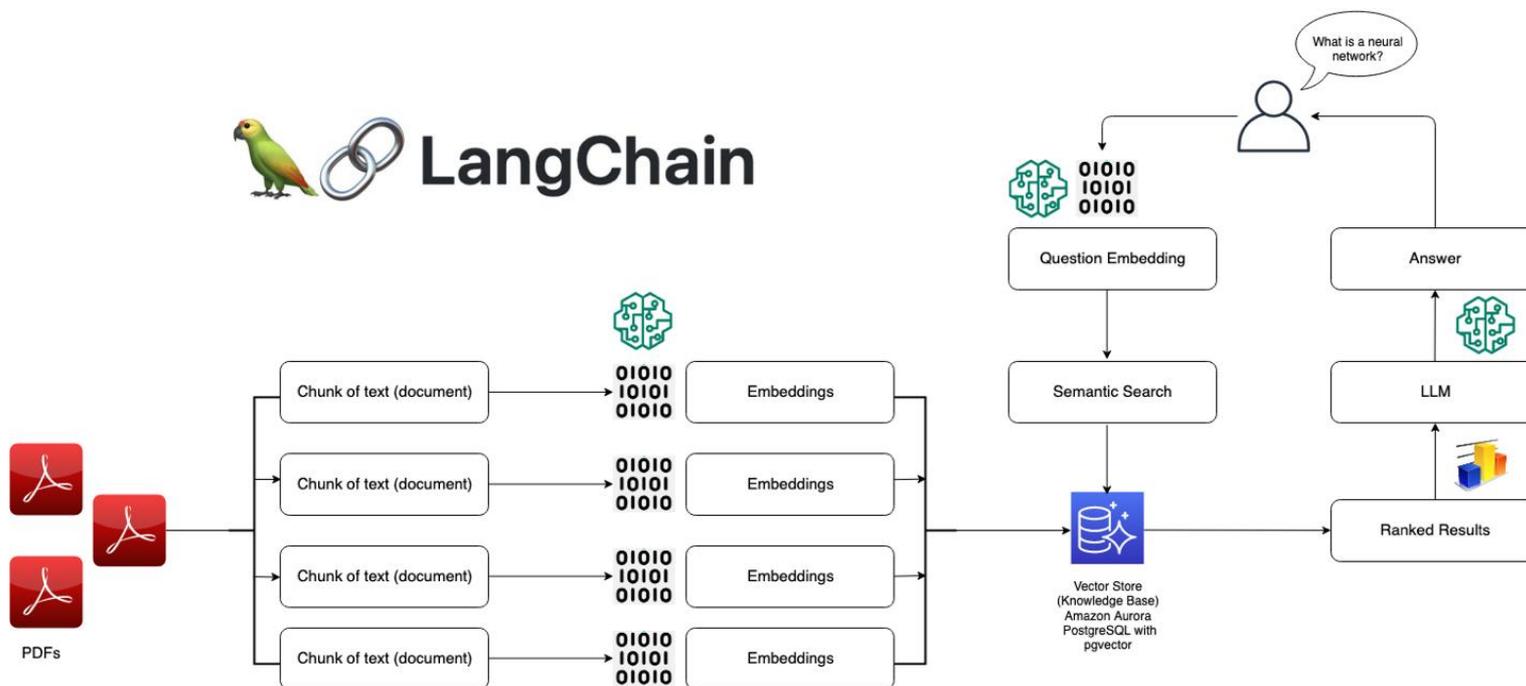


How – LangChain 框架

- LangChain 使用
 - 由 鏈(chain)與連結(link) 組成
 - Link是具體操作或任務，有明確目標。例如從某資料庫檢
索訊息、向LLM發送提示以生成文本、或對LLM輸出進行
後處理。
 - Chain是自動化行動序列。描述輸入到輸出的完整過程。



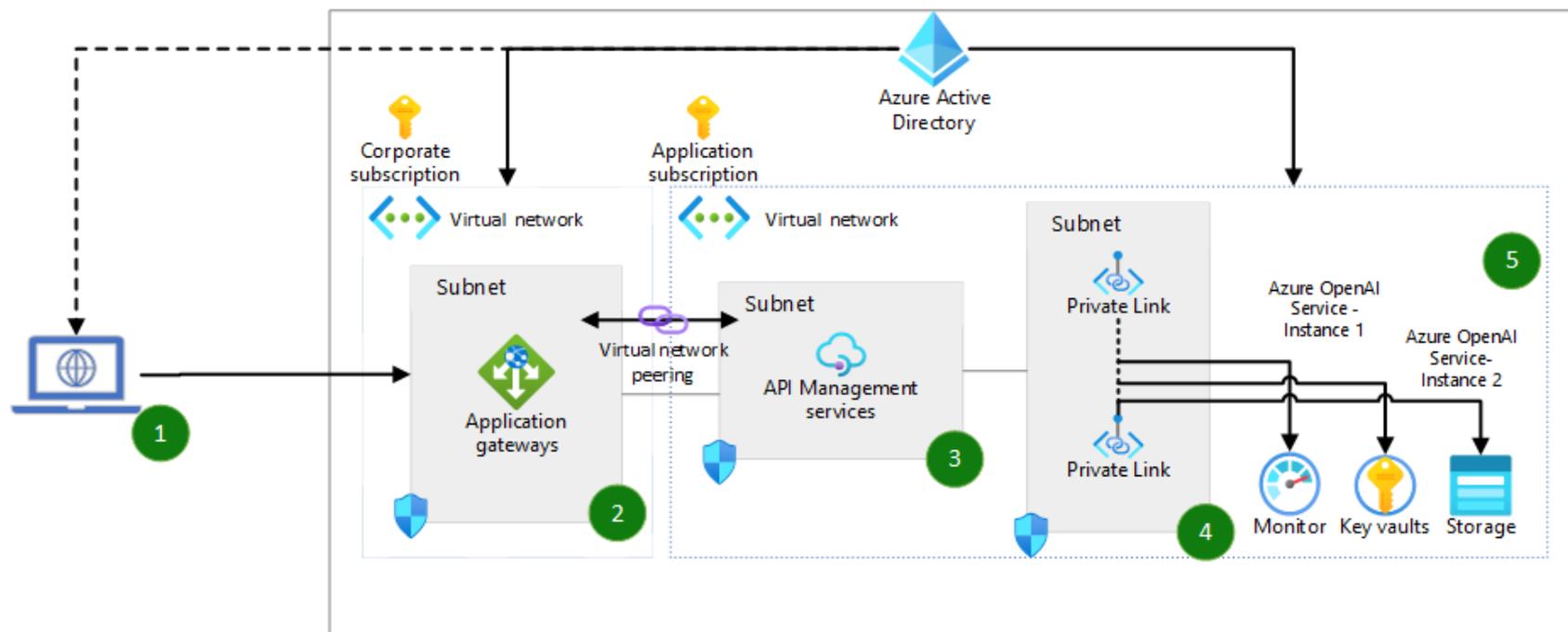
How – LangChain 框架



How - Azure Open AI (AOAI)

- AOAI
- 可進行多種程式的結合與呼叫，進而自動化完成複雜的步驟與程序
- 專為企業用戶設計的，在穩定性、可靠度等方面更有保障
- 高度安全性、存取管理、網路傳輸、數據保護、擴展容量，以及不容忽視的在地化技術支援、即時的問題排除

How - Azure Open AI (AOAI)



企業整合

企業整合 – 情境

在電子商務競爭激烈的環境中，A公司因為客戶需求過大導致回應緩慢，決定引入大型語言模型（LLM）技術改革客服流程。這項創新讓A公司能即時理解並回答客戶請求，大幅提升客戶滿意度。

同時，員工得以專注於更複雜的問題和創新項目，推動公司發展。

LLM技術的融合不僅迅速解決了回應時間長的問題，也為公司開闢了一條創新與效率兼顧的發展之路。

企業整合 – 目的

1. 提高業務效率：

自動化重複性工作，節省時間和資源。

2. 改善客戶服務：

整合多方功能，快速回答客戶的查詢。幫助企業分析客戶的反饋和社交媒體數據，以改進產品和服務。

Ex. [AIXPERT](#)

企業整合 – 目的

3. 內部知識管理：

整合內部資料到語言模型中，能更好地組織和檢索內部知識，促進知識共享和協作。

Ex. [Radar人力資源管理系統](#)

4. 可視覺化設計：

方便公司員工使用的API，不需要了解語言模型程式本身即可使用。

企業整合 – 技術

1. 資料隱私與安全性：

確保符合資料隱私法規，實施堅固的安全措施，以保護敏感的企業資料。 Ex. Intranet

2. 訂制化與微調：

使用公司特定的資料對語言模型進行微調，使其更符合業務需求。 Ex. 金融消費者問答

3. 可擴展性與性能：

確保語言模型能夠應對企業所需的大規模資料和工作負載，而不影響性能。

Ex. AWS、Google Cloud雲端運算

企業整合 - 應用

1.互通性：

確保語言模型能與企業內其他現有系統互動並共享資料。

2.使用者培訓與支援：

為將使用語言模型的員工提供足夠的培訓和支援，以最大程度地發揮其益處。

3.資料所有權和完整性：

在與外部AI提供者或數據來源合作時，明確界定資料所有權和授權協議。

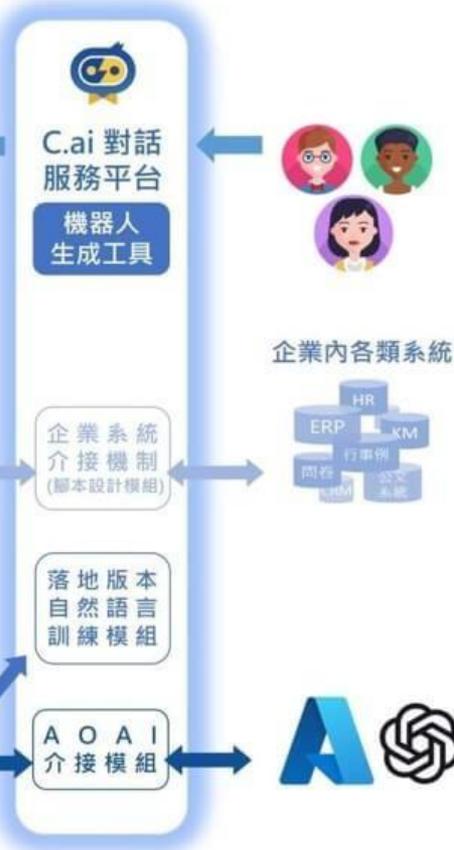
企業整合

可自主生成各領域機器人窗口



應用於多樣化服務場景

應用模式	產	銷	人	發	財	資
通知/回應	生產進度 異常提醒 客訴通知	訂單進度 異常提醒 客訴通知	打卡提醒 防疫機制 考勤狀態 相關審核	研發進度 日誌填報 客訴通知	報支提醒 決策數據 客訴通知	待辦/審核 二段認證 VPN Pass 行程提議
資料隨手查	品項/專業明細 生產進度 客訴處理	品項/專業明細 訂單進度 客訴處理	假/勤資訊 課程/履歷	KM/紀錄 專案資訊 客訴處理	請簽紀錄	待辦/審核 通訊錄 KM/資源
隨手填單	待辦派工 衛安查核	待辦派工 客訴	請假 加班 訓練 打卡	待辦派工 衛安查核	費用報支 購易購採購	待辦/審核 預約資源 各類申請 問題叫修
QA	工作知識	官網客服 售後服務 工作知識	HR 規章 工作知識	工作知識	請採報支 工作知識	系統客服 工作知識



未來挑戰及發展

現行發展

個性化和可解釋性使模型更好地適應個人用戶的偏好和需求，以及提供更清晰解釋模型決策和生成結果的方式。

1

更深層次的理解和生成能力

對多模態數據（如文字、圖像、聲音）的整合能力

Ex. MM-LLM

2

縮小模型規模

在相同的參數規模下生成更高水準的回應。

Ex. SLM(小型語言模型)

3

現行發展 – MM-LLM

- MM-LLM(Multimodal Large Language Model)
- 多模態大型語言模型
 - 利用大型語言模型來執行多模態任務。例如圖像分析、進行視覺文字辨識
 - ChatGPT-4 plus版本就是基礎的MM-LLM
 - 未來可應用於如影片字幕生成、圖像回答問題等領域



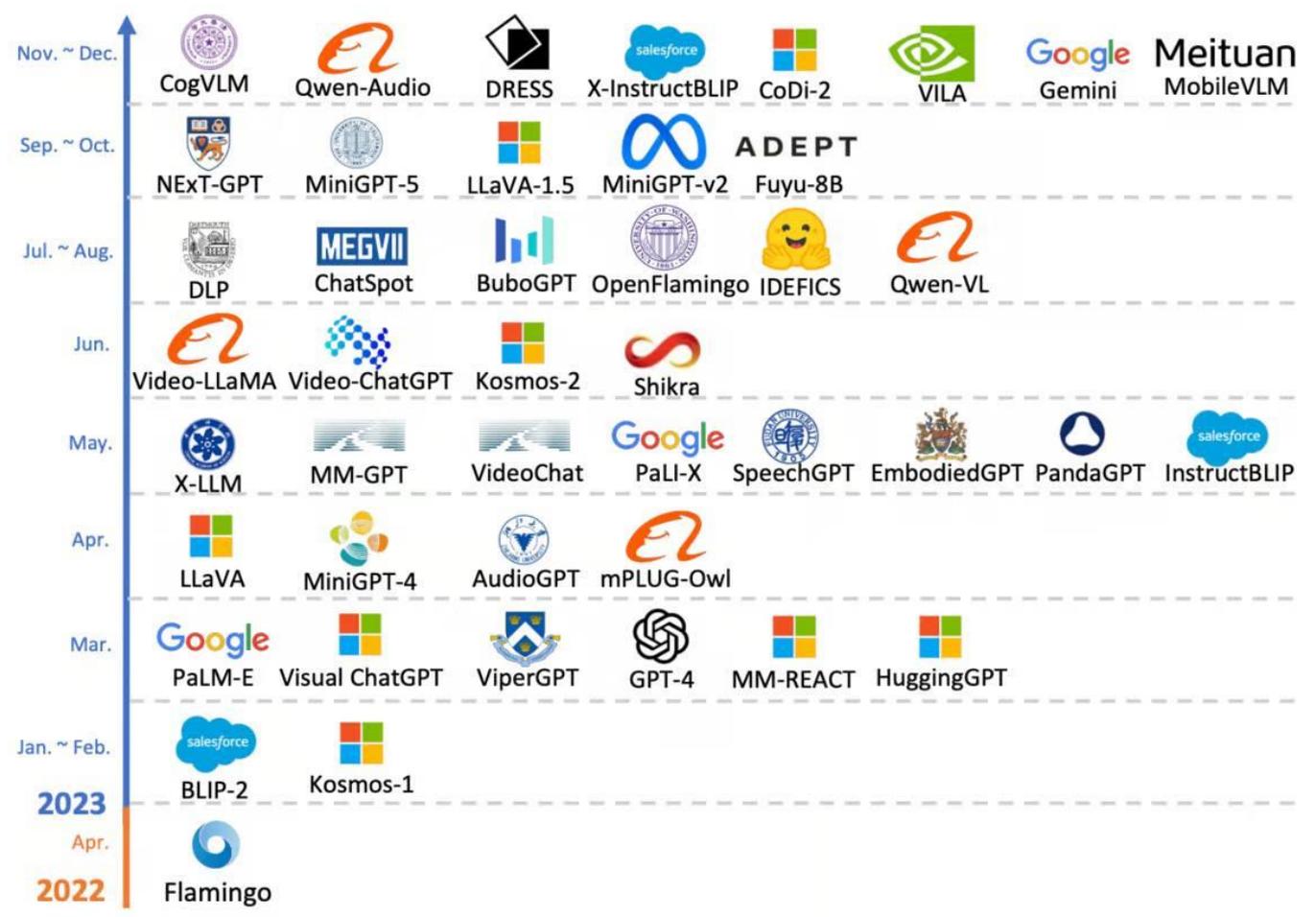
現行發展 –MM-LLM



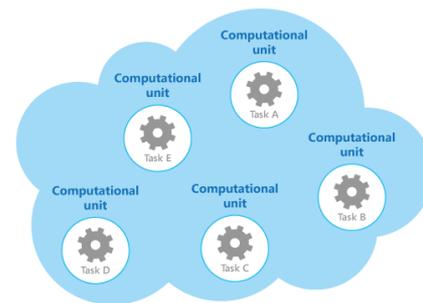
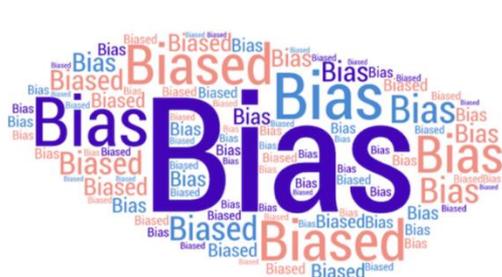
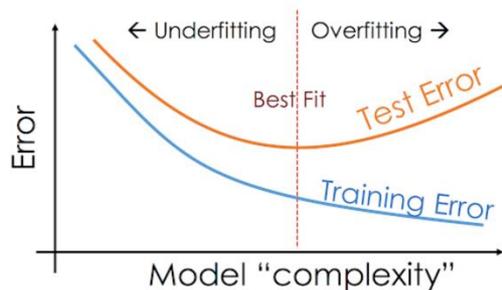
Gemini

- Gemini Pro By Google
 - 展現在多種資訊類型（文字、程式碼、音訊、影像和視訊）上的多模態AI模型能力。
- Codi-2 By Microsoft
 - 專為支援任意模態輸入和輸出而設計的端對端多模態模型，減輕級聯繫統中傳播的錯誤。

MM-LLM



LLM挑戰



過度擬合

LLM可能在處理有限的數據時產生過度擬合問題，影響模型的穩定度。

倖存者偏差

例如偏見和歧視，LLM學習自大量文本，可能存在歧視性言語和鮮有的觀點。

計算資源

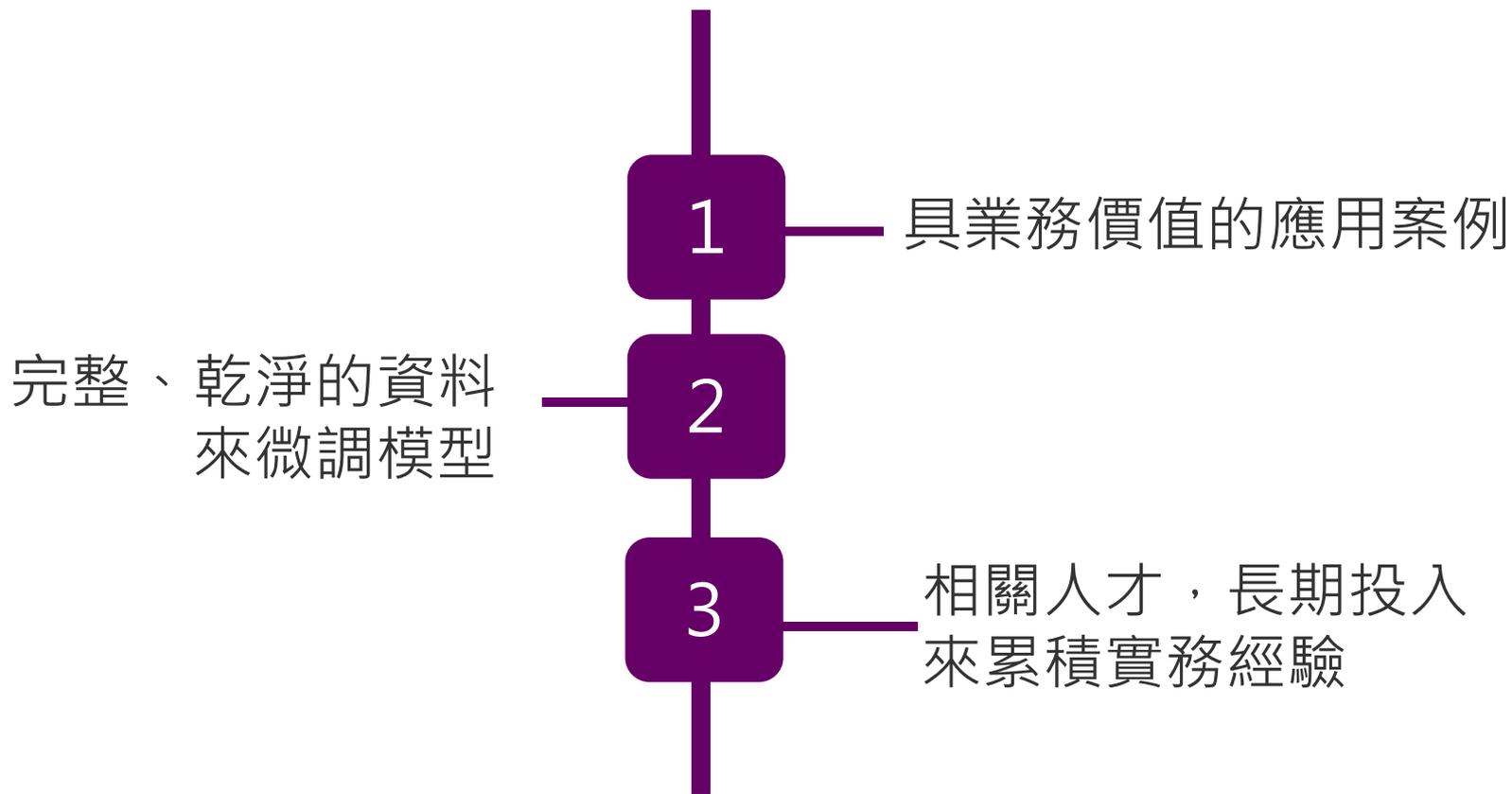
LLM需要大量的計算資源和時間進行訓練和運行，對環境造成負擔。

LLM挑戰

- LLM如何驗證
 - Perplexity (困惑度)
 - BLEU (Bilingual Evaluation Understudy) 分數
 - ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
分數
 - 生成文本的人工評分



企業整合LLM之挑戰



最新模型

• ChatGPT -4o

- 高級自然語言處理：ChatGPT-4.0 可以處理多種語言任務，包括問答、翻譯、文本生成等。
- 強大的理解能力：它能理解上下文和複雜的語言結構，提供更準確和相關的回應。
- 多用途應用：適用於客戶服務、教育、內容創作和個人助理等多種應用場景。
- 持續學習與改進：透過不斷的訓練和更新，ChatGPT-4.0 能夠提升其性能和準確性。

<https://www.youtube.com/watch?v=jay7kise3PI>

最新模型

- ChatGPT -4o



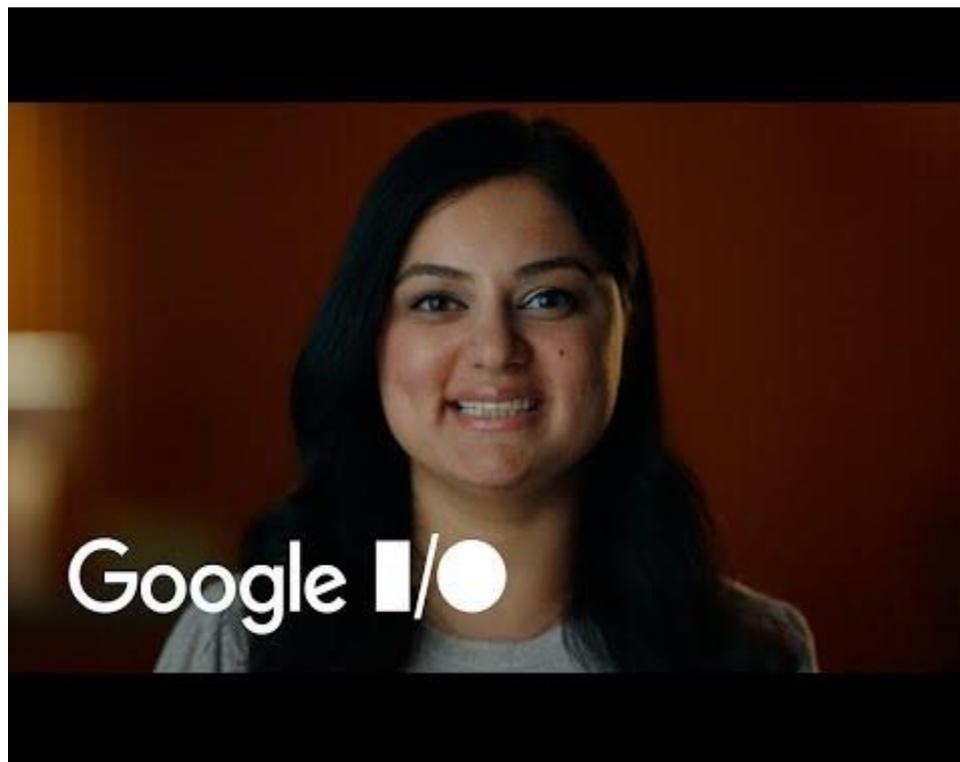
<https://www.youtube.com/watch?v=jay7kise3PI>

最新模型

特性	ChatGPT 3.5	ChatGPT 4	ChatGPT 4o
發布時間	2022年	2023年	2024年5月
語言理解能力	高	更高	最佳
上下文理解深度	約4,000 tokens	約8,000 tokens	約16,000 tokens
訓練數據集規模	約570GB	約1.2TB	約2.5TB
輸入/輸出模態	文字	文字、圖片	文字、圖片、音訊、影片

最新模型

- Gemini 1.5 pro



<https://www.youtube.com/watch?v=cogrxfRvWw>

參考資料(1/4)

1. <https://www.linkedin.com/pulse/llm-revolution-how-ai-language-models-transforming-lives-ahmed-jawed>
2. <https://zhuanlan.zhihu.com/p/650458297>
3. <https://platform.openai.com/tokenizer>
4. <https://www.ithome.com.tw/news/160094>
5. <https://www.markreadfintech.com/p/llm>
6. <https://www.cup.com.hk/2023/06/30/ai-hallucination/>
7. https://blog.csdn.net/v_JULY_v/article/details/134765564
8. <https://aitools.fyi/zh/compare/opencopilot-vs-code-llama>
9. <https://blog.infuseai.io/taide-model-introduction-f14d1334bf17>
10. <https://www.ithome.com.tw/news/160094>
11. <https://www.ithome.com.tw/news/160092>
12. <https://taide.tw/index>
13. <https://vocus.cc/article/63bd8866fd897800013a0fe0>

參考資料(2/4)

14. <https://www.inside.com.tw/article/31639-what-is-LLM>
15. <https://www.kocpc.com.tw/archives/489933>
16. <https://vocus.cc/article/63bd8866fd897800013a0fe0>
17. <https://www.inside.com.tw/article/31639-what-is-LLM>
18. <https://blog.infuseai.io/mistral-7b-introduction-2f6857f6982b>
19. <https://zhuanlan.zhihu.com/p/639276066>
20. <https://github.com/eugeneyan/open-llms>
21. <https://techdiylife.github.io/big-model-training/deepspeed/LLM-state-of-GPT.html>
22. <https://reurl.cc/Y009vn>
23. <https://medium.com/thirdai-blog/introducing-the-worlds-first-generative-llm-pre-trained-only-on-cpus-meet-thirdai-s-bolt2-5b-10c0600e1af4>
24. <https://blog.infuseai.io/rag-retrieval-augmented-generation-introduction-a5854cb6393e>

參考資料(3/4)

25. https://docs.aws.amazon.com/zh_tw/sagemaker/latest/dg/jumpstart-foundation-models-customize-rag.html
26. <https://www.anyscale.com/blog/a-comprehensive-guide-for-building-rag-based-llm-applications-part-1>
27. <https://blog.infuseai.io/rag-retrieval-augmented-generation-introduction-a5854cb6393e>
28. <https://aws.amazon.com/tw/what-is/langchain/>
29. <https://mistral.ai/news/announcing-mistral-7b/>
30. <https://the-walking-fish.com/p/lmstudio/>
31. <https://lmstudio.ai/>
32. <https://aws.amazon.com/tw/what-is/langchain/>
33. <https://www.readfog.com/a/1706407494707941376>
34. https://www.gss.com.tw/eis/271-eis109/3510-eis109_6
35. <https://learn.microsoft.com/zh-tw/azure/architecture/ai-ml/openai/architecture/log-monitor-azure-openai>

參考資料(4/4)

36. https://www.digitimes.com.tw/tech/dt/n/shwnws.asp?id=0000668958_GH0LKR404F00FY9AXP7F5
37. <https://www.linkedin.com/pulse/from-llms-mlms-unlocking-advanced-machine-ashish-bhatia/>
38. https://arxiv.org/pdf/2401.13601.pdf?fbclid=IwAR1EFCRrqrctcL70vfZCMw81GHAA8g3a2HcJuy-ttHbvdpWUL_JCSfDR46Xo
39. <https://medium.com/sherry-ai/數據會說話-淺談倖存者偏差-survivorship-bias-75ccbceeda5>
40. <https://ihower.tw/blog/archives/11933?fbclid=IwAR16ZYhYnRQDDMUgD6thDtuXlQyrlkijoXluclpzoFAnOQzmUnwl0eBBDpY>

Thank You for Your Listening

LLM 與 企業整合應用 作業

使用大型語言模型回答內部資料

回家作業

1. 建立環境
2. 利用pypdf 讀取特定資料
3. 使用預設的embedding
4. 連結Google雲端，使用Breeze7B.gguf模型
5. 給LLM一些提示前綴
6. 與LLM對話至少**五組問答**，內容皆須關於給予之資料

回家作業

Text



Tokens



Embedding



Model

回家作業

1. 建立環境

安裝langchain等必要環境

```
%pip install -U langchain-community  
%pip install pymupdf  
%pip install sentence-transformers  
%pip install chromadb  
%pip install llama-cpp-python
```

```
from langchain.document_loaders import PyMuPDFLoader  
from langchain.text_splitter import RecursiveCharacterTextSplitter  
from langchain.vectorstores import Chroma  
from langchain.embeddings import HuggingFaceEmbeddings  
from langchain.llms import LlamaCpp  
from langchain.chains import RetrievalQA
```

回家作業

2. 利用pypdf 讀取特定資料

改成自己的檔案

```
loader = PyMuPDFLoader("Virtual_characters.pdf")  
PDF_data = loader.load()
```

```
text_splitter = RecursiveCharacterTextSplitter(chunk_size=100, chunk_overlap=5)  
all_splits = text_splitter.split_documents(PDF_data)
```

回家作業

3. 使用預設的embedding

```
# Embed and store the texts
# Supplying a persist_directory will store the embeddings on disk
persist_directory = 'db'
model_name = "sentence-transformers/all-MiniLM-L6-v2"
model_kwargs = {'device': 'cpu'}
embedding = HuggingFaceEmbedders(model_name=model_name, model_kwargs=model_kwargs)

vectordb = Chroma.from_documents(documents=all_splits, embedding=embedding, persist_directory=persist_directory)
```

回家作業

4. 連結Google雲端

```
from google.colab import drive  
drive.mount('/content/drive')
```

回家作業

4. 將Breeze7B.gguf模型 以捷徑方式連結至自己雲端

複製雲端路徑

```
from langchain.callbacks.manager import CallbackManager
from langchain.callbacks.streaming_stdout import StreamingStdOutCallbackHandler
from langchain_community.llms import LlamaCpp
#使用Breeze7B模型GGUF檔案
#https://drive.google.com/file/d/1vo3XDH56JBJLNvH8RyoDcbMzAV80mZm3/view?usp=sharing
#可以下載或使用捷徑連結到檔案位置
model_path = "/content/drive/MyDrive/Model/Breeze-7B-Instruct-v0.1-Q4_K_M.gguf"

llm = LlamaCpp(
    model_path=model_path,
    n_gpu_layers=100,
    n_batch=512,
    n_ctx=2048,
    f16_kv=True,
    callback_manager=CallbackManager([StreamingStdOutCallbackHandler()]),
    verbose=True,
)
```

回家作業

5. 給LLM一些提示前綴

```
from langchain.chains import LLMChain
from langchain.chains.prompt_selector import ConditionalPromptSelector
from langchain.prompts import PromptTemplate

DEFAULT_LLAMA_SEARCH_PROMPT = PromptTemplate(
    input_variables=["question"],
    template="""<<SYS>>
    You are a helpful assistant eager to assist with providing better Google search results.
    <</SYS>>

    [INST] Provide an answer to the following question in 150 words. Ensure that the answer is informative, \
    relevant, and concise:
    {question}

    [/INST]""",
)

DEFAULT_SEARCH_PROMPT = PromptTemplate(
    input_variables=["question"],
    template="""You are a helpful assistant eager to assist with providing better Google search results. \
    Provide an answer to the following question in about 150 words. Ensure that the answer is informative, \
    relevant, and concise: \
    {question}""",
)

QUESTION_PROMPT_SELECTOR = ConditionalPromptSelector(
    default_prompt=DEFAULT_SEARCH_PROMPT,
    conditionals=[(lambda llm: isinstance(llm, LlamaCpp), DEFAULT_LLAMA_SEARCH_PROMPT)],
)

prompt = QUESTION_PROMPT_SELECTOR.get_prompt(llm)
prompt
```

回家作業

6. 與LLM對話至少**五組問答**，內容皆須關於給予之資料

```
retriever = vectordb.as_retriever()

qa = RetrievalQA.from_chain_type(
    llm=llm,
    chain_type="stuff",
    retriever=retriever,
    verbose=True
)
```

Q:

```
##變更為與自己檔案相關的問題，至少五組對答
query = "Tell me about Alison Hawk's career and age"
qa.invoke(query)
```

A:

```
> Finished chain.
{'query': "Tell me about Alison Hawk's career and age",
 'result': 'Alison Hawk is a 28-year-old female researcher known for her genius and mysterious intentions.'}
```

回家作業

繳交

1. 五組問答截圖
2. .ipynb檔
3. 輸入的.pdf文檔

Thank You for Your Listening