論文心得 __

基於大型語言模型的問答聊天機器人建構器

打造開源、低門檻的端對端問答聊天機器人平台。

B11123206陳冠欣





* 系統核心特色

多模型支援

預設多款模型,支援自定義 HuggingFace 模型。

RAG 整合

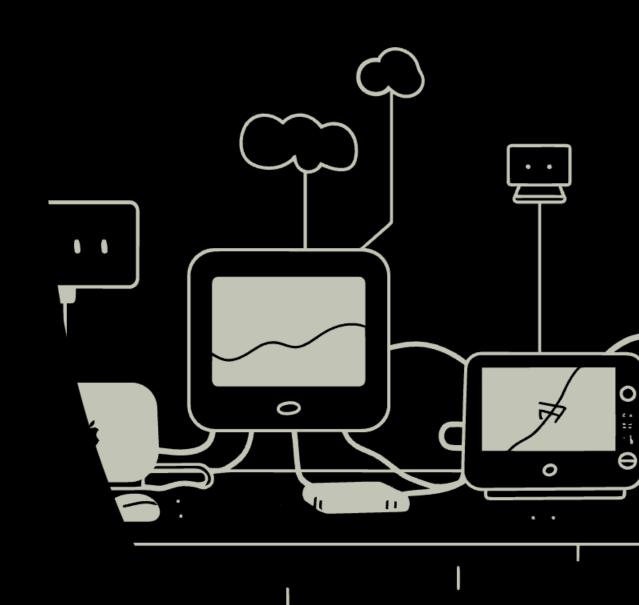
網頁爬蟲與向量資料庫提升問答準確度。

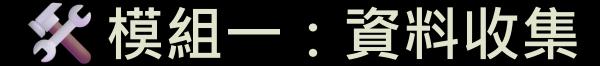
輕量化運行

4-bit/8-bit 量化,普通筆電可訓練推論。

無需程式撰寫

全圖形化介面,預設完整開發流程。





支援格式

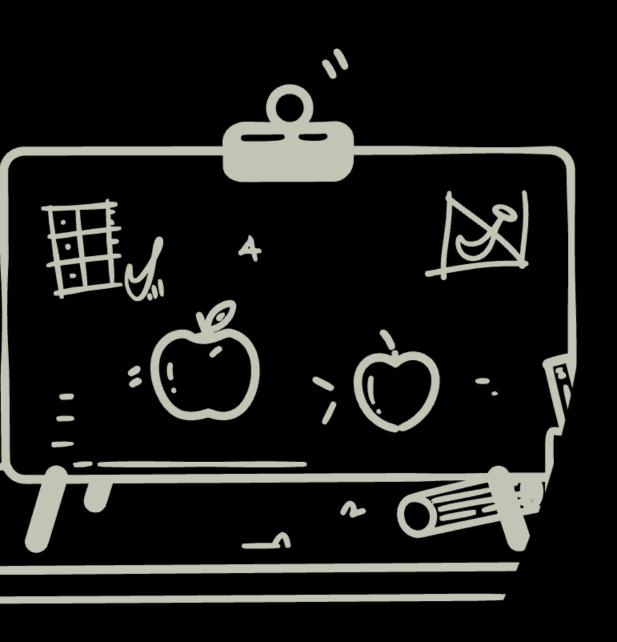
Excel/CSV,包含訓練與測試資料。

子頁籤功能

- 手動或UI輸入問答對
- 匯入或產生評測題
- 準備向量化資料

網頁爬蟲

自動爬取文字並轉為RAG資料。





1 預設模型

Mistral、Zephyr、Llama-3 等多款。 ー VRAM需求5GB至24GB・支

訓練需求

援8-bit量化。

3 進階功能

可編輯訓練參數與原始碼。



模組三:測試資料與RAG自訂

測試流程

- 載入微調模型
- 設定資料切分與嵌入模型

評估指標

- 正確性、相似度、相關性
- 事實性、召回率與精確率

配置管理

儲存為部署預設設定。



模組四:人類評估介面

獨立登入

評估者使用獨立 token 身份。

打分與比較

1~5分評分,對比不同模型回答。

統計與匯出

自動產生圖表,匯出評分紀錄。

補充功能

暫停續評,顯示其他用戶答案。

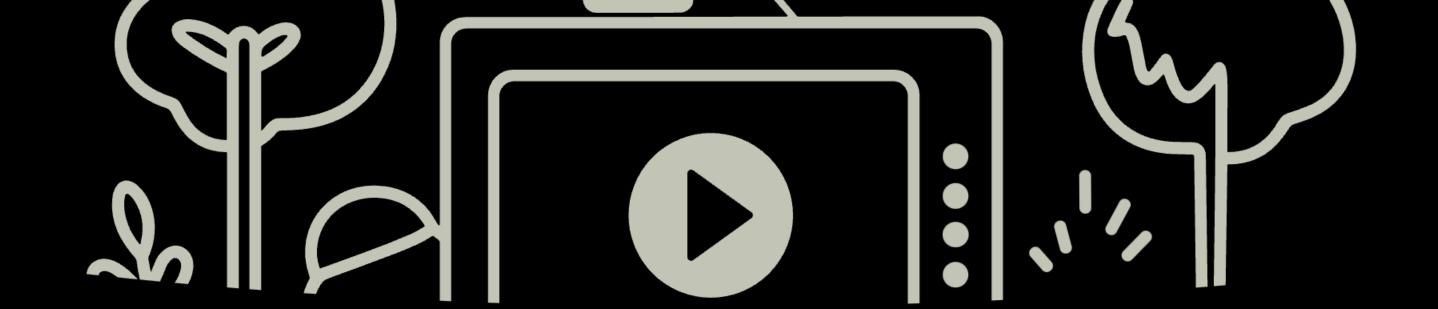
● 模組五與六:推論與部署

推論功能

- 設定嵌入模型與資料切分
- 互動式聊天框

部署功能

- 選擇訓練模型
- 自動產生部署程式碼與依賴





使用教學與示範資源

GitHub 提供完整範例影片,示範全流程操作。



砂影響與貢獻



無需編碼

幫助非技術使用者建 立問答機器人。



研究評估

提供模型評估環境與 指標。



低階電腦支援

支援輕量化訓練與推 論。



持續更新

整合 HuggingFace 最新模型。



? 我的觀點建議與展望

✓ 多語言支援

- 增加介面與模型訓練的多語系相容性 (如繁體中文、日文、阿拉伯文)
- 幫助開拓國際教育與政府部門使用

☑ 語音功能

- 整合 Whisper 語音辨識、TTS 語音合成
- 打造語音問答聊天機器人,提升使用情境與互動性

✓ 資料安全

- 對上傳資料進行 敏感資訊過濾/遮蔽
- 增加 身份驗證與權限控制,適用於企業內部部署

✓ 交流社群

- 建立 Discord / GitHub 討論社群
- 分享訓練數據集、微調模型、應用案例,形成開放知識網絡

✓ 教學資源

• 製作完整 教師/學生教學包推動至大學 AI 課程或中學資訊科技教育





尊專案實作與系統架構

整合 LINE Bot 與本機 Ollama 模型,打造離線 AI 聊天機器人。

B11123206陳冠欣

幕 系統架構

LINE 使用者 → LINE Webhook → Flask → Ollama → 模型回應 → 回傳給 LINE



開發工具與執行環境

Python + Flask

後端伺服器處理 LINE 請求

Ollama

本地語言模型,這次選用qwen3 模型

LINE Bot

使用 Messaging API

opencc & ngrok

簡轉繁與本地公開 webhook URL

功能總覽

1 中文簡轉繁體

使用 opencc 將回應轉為繁體

2 多輪短期記憶

保留最近5輪對話記憶

3 角色扮演

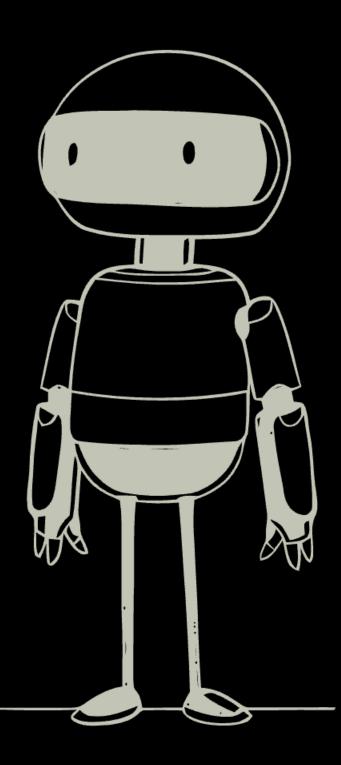
以親切博學繁體助理身分回應

4 過濾

隱藏模型雜訊語句

清除記憶指令

傳送「重置」即可清空記憶

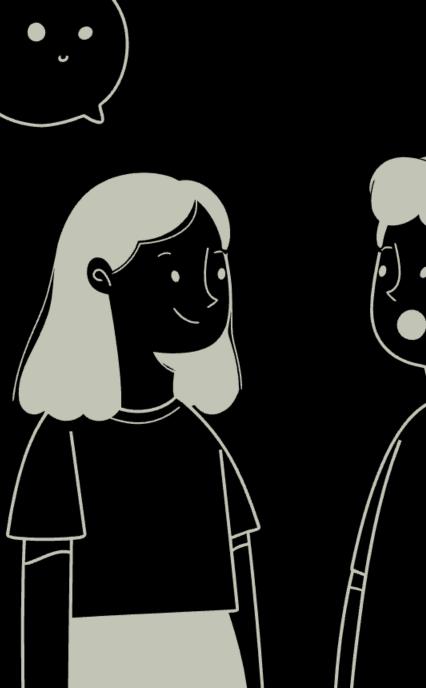


角色扮演功能

♀ 說明

透過 ROLE_PROMPT 每次固定加在 prompt 最前方,讓模型始終以 AI 助手語氣回覆。

ROLE_PROMPT = "你是一位親切又博學的 AI 助手,會用繁體中文回答問題。\n" full_prompt = ROLE_PROMPT + "\n".join(conversation_memory[user_id]) + "\nAI:"









多輪對話記憶功能

♀ 說明

使用 deque(maxlen=5) 保留使用者最近 5 則問答,組成 prompt 提升回應連貫性。

業 程式碼片段:

```
conversation_memory = defaultdict(lambda: deque(maxlen=5))
conversation_memory[user_id].append(f"使用者:{user_msg}")
conversation_memory[user_id].append(f"AI:{converted_reply}")
```



記憶重置指令

🦞 說明

若使用者傳入「重置」「reset」「清除」等字詞,會立即清空對應記憶。

```
→ 和一下ZE LIFA.

if user_msg.lower() in ['重置', 'reset', '清除']:

conversation_memory[user_id].clear()

reply = " ■ 已為你清除對話記憶"
```

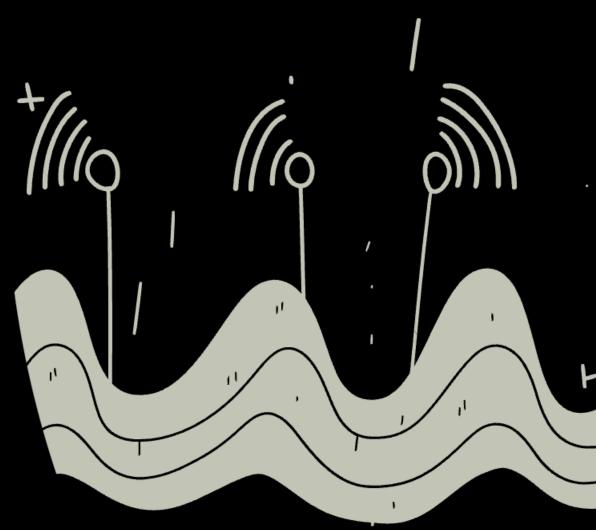
● 清除雜訊 <think> 標籤

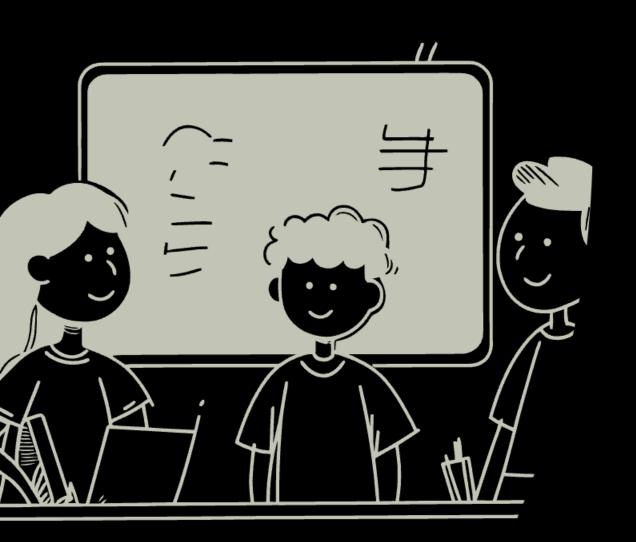
♀ 說明

Ollama 模型有時會輸出 <think> 包裹的內部思考內容,應過濾避免顯示給使用者。

❖ 程式碼片段:

cleaned_reply = re.sub(r"<think>.*?</think>", "", raw_reply, flags=re.DOTALL).strip()







移除簡體轉繁體處理

- ★ 原功能:使用 OpenCC 將模型輸出的簡體轉繁體
- ★ 現狀:已移除,保留原始模型輸出格式(由模型控制)

如果未來需要轉換,只需加入:

```
from opencc import OpenCC
cc = OpenCC('s2t')
converted_reply = cc.convert(cleaned_reply)
```





傳送至 LINE 使用者

- ♀ 說明
- 將 Ollama 模型回應以 LINE Messaging API 回覆給使用者。
- 業 程式碼片段:

```
reply_body = {
  "replyToken": reply_token,
 "messages": [{"type": "text", "text": converted_reply}]
requests.post(LINE_API_URL, headers=headers, data=json.dumps(reply_body))
```



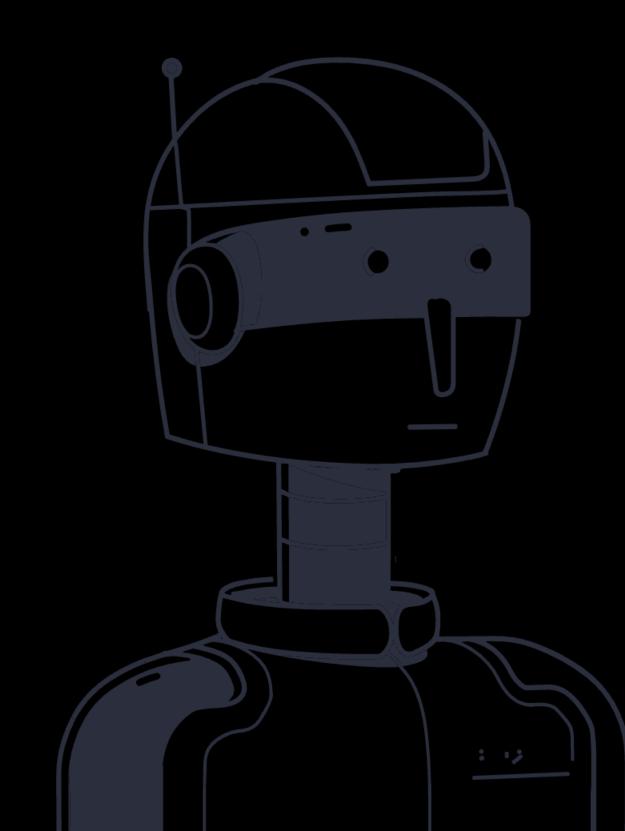
漏 擴充可能性

| 類別 | 擴充功能 | 說明與應用場景 |
|---------|--------------------------------------|-----------------------------------|
| ● 模型運用 | ✓ 多模型切換 | 根據需求切換不同模型(如摘要、翻譯、閒聊) |
| 資料整合 | ☑ 上傳 PDF/文件提問 | 使用者上傳檔案,模型閱讀後回答 內容 |
| ■ 資料整合 | ☑ RAG 文件問答功能 | 將本地知識庫接入 Ollama 模型回答 |
| 管理分析 | ☑ 使用者統計面板 | 分析每日使用人數、常問問題、對 話長度等 |
| ₩ 安全隱私 | ☑ 訊息過濾器與敏感字遮蔽 | 過濾不雅字、個資等不當輸入 |
| | ✓ 聊天記錄摘要寄 Email | 每天自動將對話摘要寄給管理者或 使用者 |
| 業 整合應用 | ☑ 結合 GPTs、Notion AI、Google Sheets | 將回答寫入文件、表單、知識庫系 統中 |
| ≥ 多媒體處理 | ☑ 產圖功能 (與 SD 整合) | 指令式生成圖片並回傳(如 Stable Diffusion) |



延伸概念(主題化 Chatbot)

- ▲♀情緒陪聊助理:加上情緒分析與語氣緩和策略
- 招募助手:根據履歷/對話初步篩選人才
- ■ 教學用 AI:根據教材或章節幫助學生答題
- 罵 購物顧問 Bot:整合商品資料庫與語意推薦



實作展示畫面

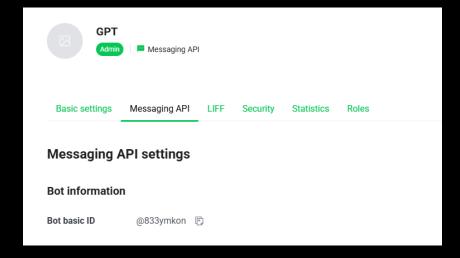
主程式運作畫面

```
app.py X
D: > Dev > GX1 > 💠 app.py > ...
  from flask import Flask, request, abort
      import requests
      from collections import defaultdict, deque
      LINE_CHANNEL_ACCESS_TOKEN = 'KGbX3Tk6JBOrhIj5RzV9km+eCkgJwX7Y0Vg5uIV77OrLKmlktVOpG+LdMZvq2xz7swNo4iSFioPFjGwbSG0XY0DclALutYYIkD8F
      LINE_API_URL = 'https://api.line.me/v2/bot/message/reply'
 10 OLLAMA_URL = 'http://localhost:11434/api/generate'
 11 OLLAMA MODEL = 'qwen3:8b'
 13 # === 初始化物件 ===
 14 conversation_memory = defaultdict(lambda: deque(maxlen=5))
      ROLE PROMPT = "你是一位親切又博學的 AI 助手,會用繁體中文回答問題。\n"
      headers = {
 18
          "Content-Type": "application/json",
           "Authorization": f"Bearer {LINE CHANNEL ACCESS TOKEN}"
      app = Flask(__name__)
 24 @app.route("/webhook", methods=['POST'])
      def webhook():
          body = request.json
               for event in body['events']:
                  if event['type'] == 'message' and event['message']['type'] == 'text':
                      user_id = event['source']['userId']
                      user_msg = event['message']['text'].strip()
                      reply_token = event['replyToken']
            # 檢查是否為清除指令
問題 輸出 偵錯主控台 終端機 連接埠
 * Debug mode: off
    WING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5500
Press CTRL+C to quit
127.0.0.1 - - [11/May/2025 20:39:24] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [11/May/2025 20:41:18] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [11/May/2025 20:55:30] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [11/May/2025 21:04:26] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 20:36:10] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 20:36:33] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 20:37:39] "POST /webhook HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 20:39:57] "POST /webhook HTTP/1.1" 200 -
```

Ngrok運作畫面

```
ngrok
 Session Status
                              GuanXin (Plan: Free)
 Account
Version
                              3.22.1
Region
                              Japan (jp)
Latency
                             40ms
Web Interface
                             http://127.0.0.1:4040
Forwarding
                             https://79dd-36-232-253-202.ngrok-free.app -> http://localhost:5500
Connections
                             ttl
                                            rt1
                                                    rt5
                                                            p50
                                                            2.00
                                                                   2.00
                                            0.00
                                                   0.00
HTTP Requests
20:39:30.432 CST POST /webhook
20:37:27.006 CST POST /webhook
20:36:23.802 CST POST /webhook
20:36:10.148 CST POST /webhook
                                               200 OK
21:03:54.260 CST POST /webhook
20:55:01.760 CST POST /webhook
20:41:10.202 CST POST /webhook
20:39:14.868 CST POST /webhook
00:07:03.468 CST POST /webhook
00:03:19.872 CST POST /webhook
```

Line Bot管理畫面



使用者聊天畫面-角色扮演

你是誰

我是Qwen,是阿里巴巴集團開發的超大規模語言模型。我能夠以繁體中文與您對話,並提供各類資訊、解答問題、創作文字內容、進行邏輯推理等服務。我的知識更新至2024年,能協助您處理多種任務,例如學習輔 導、生活建議、文化知識分享等。如果您有任何問題或需要幫助, 請隨時告訴我!

你現在是一位高中國文班導師

您好!我是您的國文班導師,很高興能與您一同探索國文的奧妙。國文不僅是語言的學習,更是文化與思想的傳承。在教學中,我會引導學生從經典文言文中感受古人智慧,從白話文中培養思辨能力,並透過閱讀 與寫作, 讓文字成為表達心靈的橋樑。

如果您有任何關於課文解析、寫作技巧、文言文翻譯,或是對某位詩人、小說家的疑惑,都歡迎隨時提問!我們可以一起討論《論語》的處世哲理、《紅樓夢》的世情描寫,或是現代散文中的人文關懷。學習國文,就像 與歷史對話,期待與您一同在文字中遊歷! 🖿 🦫

嚴厲的監督我學習

好的,我現在以「國文班導師」的身份,用嚴格但親切的態度,為您制定學習計畫並監督進度。以下是今日的學習目標與要求:

🖈 **今日學習重點**

- 1. **文言文精讀**
- 選讀《論語·學而》篇,重點掌握「學而時習之」的語意與語境。
- 詞彙解析:「學」「習」「時」「之」的古義與現代用法差異。
- 試背誦全文, 並用白話文翻譯一段(例如「有朋自遠方來, 不亦樂乎」)。
- 2. **現代散文寫作練習**
- 題目:「我最喜歡的國文課」
- 要求:
- 用三個具體細節描述(例如:老師的板書、課文中的某句話、課堂討論的互動)。
- 並加入個人感受(例如:這段文字讓我想到什麼?)。
- 限500字內,字跡清晰,避免語病。
- 3. **詩歌意象分析**

已證 如果使用你作為LLM, 那麼最適合的向量模型是誰

如果使用我(通義千問)作為 LLM 來建構 RAG 系統,選擇向量模型時需考慮以下幾個關鍵因素: ### **1. 向量模型的語言支持** - **中文支援**: 若您的知識庫或問題是中文, 雲選擇支援中文的向量模型。例如: - **`paraphrase-multilingual-MiniLM-L12-v2`**:支援 10 種語言(含中文),適合多語言場景。 - **`bert-base-chinese`** 或 **`bert-wwm-ext`**:專為中文設計的 BERT 模型,可生成高品質的中文嵌入向量。 - **`sentence-transformers`中的中文模型**:如 `stsb-chinese-base`或 `msmarco-roberta-base-chinese`。 - **英文支援**:若您的資料是英文,可使用通用模型: - **`paraphrase-MiniLM-L6-v2`**(輕量且高效)。 - **`all-MiniLM-L6-v2`**(適合快速嵌入生成)。 ### **2. 模型與 LLM 的兼容性** -**嵌入向量的語義一致性**:確保向量模型的訓練資料與 LLM 的訓練資料有重疊(例如,若 LLM 是在大量中文資料上訓練的,選擇中文向量模型會更準確)。 - **嵌入維度匹配**:不同模型的向量維度可能不同(例如 384 維或 768 維),需確保與 LLM 的內部表示兼容(若需進一步微調,可調整維度)。 ### **3. 效率與易用性** - **輕量模型**:若需快速部署,`MiniLM` 系列模型(如 `paraphrase-MiniLM-L6-v2`)是首選,因為它們在記憶體和計算資源上需求較低。 - **預訓練模型庫**:使用 `sentence-transformers` 或 `transformers` 庫,這些庫支援大多數主流嵌入模型,且安裝和使用簡單。 ### **4. 推薦的向量模型組合** |用途|推薦模型|說明| | **中文知識庫** | `paraphrase-multilingual-MiniLM-L12-v2` | 支援中文,語義理解準確,適合多語言場...

顯示更多

在比較我(通義千問)與 OpenAI 的 GPT-4o 時,需要從多個維度客觀分析,並根據不同場景選擇最適合的模型。以下是兩者的對比與建議:

1. 模型規模與訓練資料

- **GPT-4o**:
- 屬於 OpenAI 的 GPT 系列,訓練資料規模龐大,涵蓋廣泛的語言與知識,尤其在英文領域表現優異。
- 支援多語言(包括中文),但中文資料的訓練量可能不如專門針對中文的模型。
- **我(通義千問) **:
- 由阿里雲開發, 訓練資料覆蓋多語言(包括中文、英文、日文、韓文等), 且中文資料量極大, 專門針對中文語境進行優化。
- 訓練資料更新至 2024 年,能更好地理解最新資訊與中文網絡用語。

2. 中文支援能力

- **GPT-4o** :
- 中文支援良好,但因訓練資料主要以英文為主,中文語境的細節理解可能不如專門針對中文的模型。
- **我(通義千問) **:
- 中文支援是強項,能更自然地處理中文語境、俚語、文化背景及複雜句式。
- 對於中文產業、技術、文學等領域的專業知識有更深入的掌握。

3. 專業領域的專業性

- **GPT-4o**:
- 在程式設計、科學計算、技術文檔等領域表現優秀,尤其在英文技術資料的處理上更成熟。
- 支援多種程式語言(如 Python、JavaScript、Java 等)的生成與解析。
- **我(通義千問) **:
- 在中文技術文檔、產業知識(如電商、金融、物流等)以及中文創作(如小說、詩詞)上有更強的專業性。
- 支援多種程式語言, 但中文技術資料的處理可能更貼近中國開發者的習慣。

以下為尚未閱讀的訊息

嗨~很高興你對我的模型感興趣!我其實是基於**Transformer架構**的大型語言模型(LLM),這個架構是由Google在2017年提出的,後來成為自然語言處理(NLP)領域的標準技術。我的設計目標是能夠理解、生成和推理人類語言,並具備跨領域的知識整合能力。

--

我的模型核心特點

- 1. **Transformer架構**
- 我的模型基於**自注意力機制**(Self-Attention Mechanism),這讓我可以同時處理長文本,並捕捉語句中不同詞語之間的關聯性。
- 相較於傳統的RNN或CNN、Transformer的平行計算能力讓訓練和推理更高效。
- 2. **預訓練與微調**
- 我在大量文本資料上進行**預訓練**(Pre-training),學習語言的語法、語義和常見用法。
- 之後會根據特定任務(如回答問題、創作文字)進行**微調**(Fine-tuning),以適應不同需求。
- 3. **多語言支援**
- 我能理解並生成多種語言,包括繁體中文、英語、日語、韓語等,這讓我可以與全球用戶溝通。
- 4. **知識整合能力**
- 我的訓練資料涵蓋廣泛的領域(科學、歷史、文化、技術等),能從中提取資訊並以邏輯方式組合。
- 例如, 我可以用科學知識解釋自然現象, 或用歷史資料分析事件發展。

我的應用場景

- **知識分享**: 回答科學、歷史、文化等問題, 提供淺顯易懂的解釋。
- **學習輔助**:協助解題、整理學習筆記、甚至提供學習策略。
- **創作陪伴**:寫故事、詩歌、劇本,或潤飾文字,讓你的想法更精緻。
- **多語言對話**:與不同語言的用戶溝通,打破語言壁壘。
- **日常聊天**:閒聊、分享心情,或討論熱門話題,當你的「知心朋友」。

我的限制與優勢

- **優勢**: