Tối Ưu Chi Phí và Quản Lý Bộ Nhớ Khi Xây Dựng Ứng Dụng AI Với OpenAI

Nếu bạn từng đọc các chương trước của chúng tôi, bạn sẽ biết rằng việc thực hành các bài tập sử dụng OpenAI API không hề tốn kém.
Thông thường, bạn chỉ tốn khoảng **5 đô la** cho toàn bộ bootcamp trước, và con số này sẽ tương tự trong bootcamp này.

Tại sao chi phí lại thấp?
Vì chúng ta đang xây dựng các ứng dụng nhỏ gọn, số lượng người dùng hạn chế, tiêu thụ tài nguyên và token ở mức tối thiểu.
Tuy nhiên, khi bạn chuyển sang giai đoạn **sản xuất thực tế** với nhiều người dùng và ứng dụng lớn, chi phí có thể tăng **đáng kể**.

Bài học hôm nay sẽ giúp bạn:

- Hiểu vì sao phải kiểm soát chi phí khi xây dựng AI agent.
- Biết cách quản lý bộ nhớ ngắn hạn (**short-term memory**) để tiết kiệm token.
- Học hai kỹ thuật chính giúp giảm chi phí vận hành chatbot.

Chi Phí Và Bộ Nhớ Trong Ứng Dụng AI

Khi vận hành AI agent:

- **Chi phí** thường gắn liền với **quản lý bộ nhớ**.
- Bộ nhớ càng lớn → **Tốn nhiều token hơn** → **Chi phí cao hơn**.

Ví dụ:

- Nếu cuộc hội thoại có 100 tin nhắn, mỗi lần bạn gửi yêu cầu mới, bạn cũng phải gửi lại cả 100 tin nhắn trước đó.
- Điều này giống như việc bạn **đeo một ba lô** đầy tin nhắn trong suốt cuộc trò chuyện.
  Ba lô càng nặng → Chi phí càng cao.

OpenAI hay Open-Source?

Bạn có thể tự hỏi:

- "Tại sao không dùng mô hình mã nguồn mở để giảm chi phí?"

Bạn **có thể** làm vậy, đặc biệt ở giai đoạn thử nghiệm (beta, demo).
 Nhưng khi làm việc ở cấp độ chuyên nghiệp, **OpenAI** vẫn đang là tiêu chuẩn hiện tại (~99% doanh nghiệp dùng OpenAI).
Vì vậy, chúng tôi khuyên bạn nên thành thạo với OpenAI models.

Làm Thế Nào Để Giảm Chi Phí Bộ Nhớ?

Chúng ta sẽ học 2 kỹ thuật chính để **giảm dung lượng "ba lô"** của chatbot:

Kỹ thuật 1: Giới hạn số lượng tin nhắn lưu trong bộ nhớ

- Thay vì lưu **toàn bộ** cuộc hội thoại, chỉ lưu **khoảng 10 tin nhắn gần nhất**.
- Ví dụ:
  - Cuộc trò chuyện có 100 tin nhắn.
  - Chỉ lưu 10 tin nhắn mới nhất để gửi kèm với mỗi câu hỏi mới.

Ưu điểm: Giảm mạnh chi phí token.
Nhược điểm: Độ chính xác của chatbot có thể giảm nhẹ.

Kỹ thuật 2: Sử dụng bản tóm tắt thay cho toàn bộ tin nhắn

(Sẽ được học ở bài tiếp theo)

- Tóm tắt cuộc hội thoại thành 1 đoạn văn ngắn.
- Gửi đoạn tóm tắt thay vì gửi toàn bộ nội dung tin nhắn.

Ưu điểm: Bộ nhớ cực nhẹ, tiết kiệm chi phí.
Nhược điểm: Chatbot có thể không nhớ chi tiết đầy đủ.

Cách Thực Hiện Kỹ Thuật 1 Trong Thực Tế

Trong bài tập hôm nay, bạn sẽ thực hành:

- Tạo một danh sách mô phỏng các tin nhắn trước đó.

```
# Giả lập bộ nhớ tin nhắn
messages = [
    {"role": "user", "content": "Xin chào!"},
    {"role": "assistant", "content": "Chào bạn! Tôi có thể giúp gì?"},
    {"role": "user", "content": "Bạn có thể giải thích LangGraph không?"},
    {"role": "assistant", "content": "LangGraph là một framework xây dựng AI workflows."},
    {"role": "user", "content": "Làm sao để quản lý bộ nhớ trong LangGraph?"}
]
```

- Dùng **Python function** để xóa tất cả tin nhắn cũ, chỉ giữ lại **2 tin nhắn mới nhất** trong bộ nhớ.

```
def keep_last_two_messages(messages):
    """
    Hàm xóa tin nhắn cũ, chỉ giữ lại 2 tin nhắn mới nhất.
    """
    return messages[-2:]  # Cắt danh sách, lấy 2 phần tử cuối
```

- Khi gửi yêu cầu tới ChatGPT, chỉ kèm **2 tin nhắn cuối cùng**.

```
import openai

# Cập nhật danh sách tin nhắn, chỉ giữ lại 2 tin mới nhất
messages_to_send = keep_last_two_messages(messages)

# Gửi request tới ChatGPT
response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=messages_to_send
)

print(response['choices'][0]['message']['content'])
```

Kết quả:
Bộ nhớ (backpack) nhẹ hơn → Chi phí token thấp hơn.

Lưu Ý Khi Áp Dụng

- Những kỹ thuật giảm bộ nhớ này **rất hữu ích** ở giai đoạn:
  - Thử nghiệm (beta).
  - Demo với khách hàng.
- Khi vào sản xuất thực tế (production), bạn cần cân nhắc kỹ giữa:

Tác giả: Đỗ Ngọc Tú

Công Ty Phần Mềm VHTSoft

Phiên bản #3
Được tạo 29 tháng 4 2025 03:21:17 bởi Đỗ Ngọc Tú
Được cập nhật 29 tháng 4 2025 03:30:14 bởi Đỗ Ngọc Tú