Conversational memory for LLM

2023년 9월 18일 월요일   오전 9:53

YT LangChain Basic Conversation Chatbot with Memory Demo.ipynb - Colaboratory (google.com)Conversational Memory for LLMs with Langchain | Pinecone

Conversational memory : like a chatbot -> response to multiple queries

ConversationChain

{history} => conversational memory

Prompt 예시

Hugging face prompt 만들어 보기

```
Out[8]:
The following is a friendly conversation between a human and an AI. The AI is talkative and provides lots of specific details from its context. If the AI does not know the answer to a question, it truthfully says it does not know.
Current conversation:
{history}
Human: {input}
AI:
```

출처: <https://www.pinecone.io/learn/series/langchain/langchain-conversational-memory/>

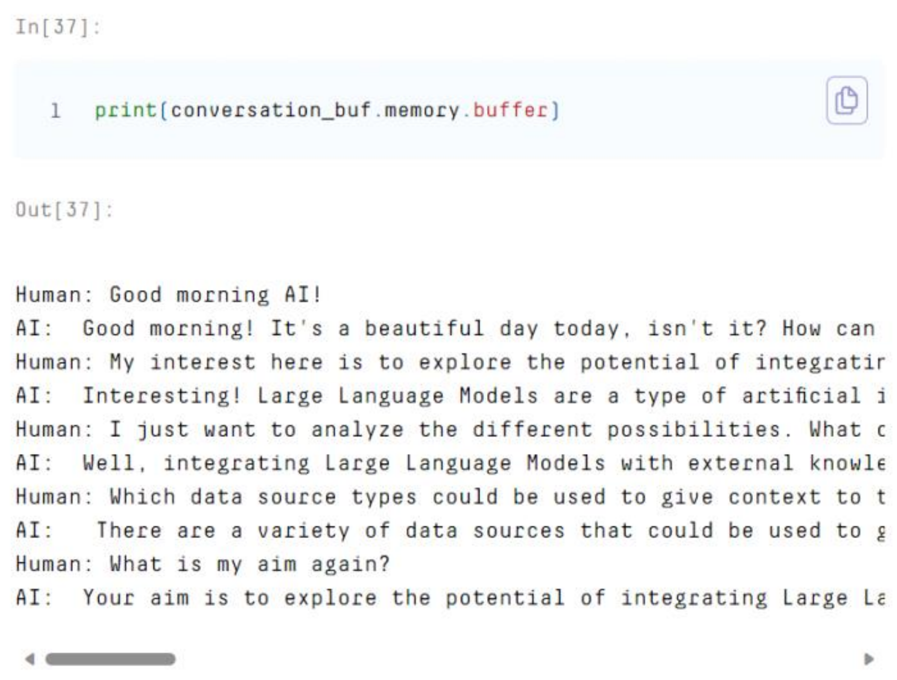
conversationBufferMemory

```
fromlangchain.chains.conversation.memory importConversationBufferMemory
conversation_buf =ConversationChain(llm=llm,memory=ConversationBufferMemory())
```

출처: <https://www.pinecone.io/learn/series/langchain/langchain-conversational-memory/>

```
conversation_buf("Good morning AI!")
```

출처: <https://www.pinecone.io/learn/series/langchain/langchain-conversational-memory/>


In[37]:

```
print(conversation_buf.memory.buffer)
```

Out[37]:

```
Human: Good morning AI!
AI:  Good morning! It's a beautiful day today, isn't it? How can
Human: My interest here is to explore the potential of integratir
AI:  Interesting! Large Language Models are a type of artificial i
Human: I just want to analyze the different possibilities. What c
AI:  Well, integrating Large Language Models with external knowle
Human: Which data source types could be used to give context to t
AI:   There are a variety of data sources that could be used to g
Human: What is my aim again?
AI:  Your aim is to explore the potential of integrating Large La
```

화면 캡처: 2023-09-18 오후 3:55

설정한 Conversation_buf 사용할수록 history가 쌓인다

하지만 token을 많이 사용한다 high cost token 신경 써야 한다

conversationSummaryMemory

Token 사용량을 계속 사용하는것을 피하기 위해 사용

from langchain.chains.conversation.memory import ConversationSummaryMemory

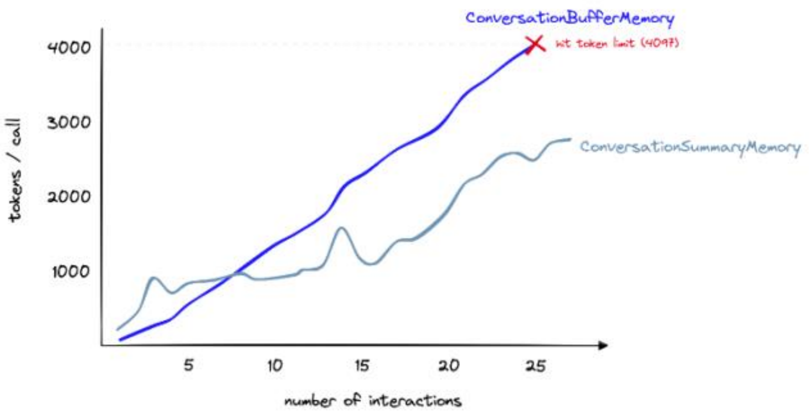
```
conversation = ConversationChain(
	llm=llm,
	memory=ConversationSummaryMemory(llm=llm)
)
```

-> summarization은 llm이 필요 하므로 인자로 llm 삽입한다

`print(conversation_sum.memory.buffer)`

출처: <https://www.pinecone.io/learn/series/langchain/langchain-conversational-memory/>

위에서 conversationbufmemory에서 하던것 처럼 똑같이 하였다 그리고
모든 내용이 나왔던 결과에 반해 summary사용하면 한문장으로 요약하여 나온다



Token count (y-axis) for the buffer memory vs. summary memory as the number of interactions (x-axis) increases.

화면 캡처: 2023-09-18 오후 4:00

하지만 여전히 token의 한계는 존재 한다

conversationBufferWindowMemory

Buffermemory에서 했던것과 유사한데 다만 모두 기억하지 않고 정해진 숫자 범위 안에서 기억한다

코랩 정리해놓음