Analytics Vidhya

Hands-On: LLM Advanced Operations

Instructor

Dipanjan Sarkar

Head of Community & Principal AI Scientist at Analytics Vidhya

Google Developer Expert - ML & Cloud Champion Innovator

Published Author

Outline

- Tracking LLM Costs
- Caching
- Streaming

Tracking LLM Costs

- Most commercial LLMs charge you based on number of tokens
- LangChain enables you to track your token usage for LLM calls
- Currently this implementation is only for Open AI LLMs

```python
from langchain_community.callbacks import get_openai_callback
from langchain_openai import ChatOpenAI

chatgpt = ChatOpenAI(model_name="gpt-3.5-turbo",
                     temperature=0)

prompt = """Explain Generative AI in one line"""

with get_openai_callback() as cb:
    response = chatgpt.invoke(prompt)
    print(response.content)
    print(cb)

# Output
# Generative AI is a type of ......
# Tokens Used: 50
#   Prompt Tokens: 15
#   Completion Tokens: 35
# Successful Requests: 1
# Total Cost (USD): $9.25e-05
```

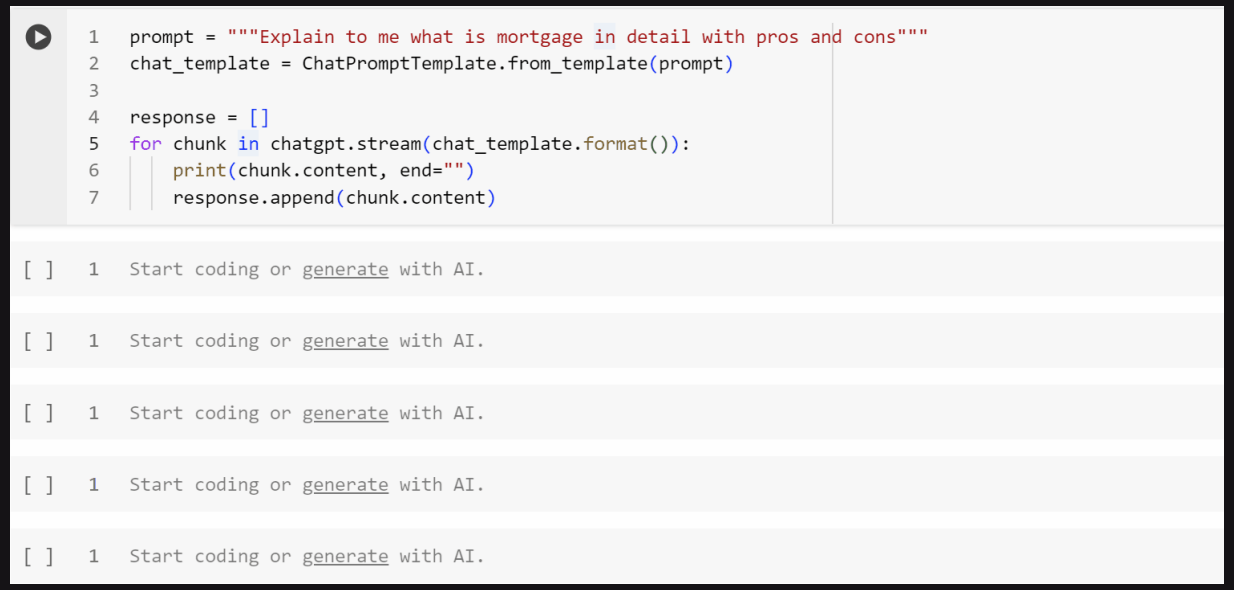
Caching

- LangChain provides an optional caching mechanism when interfacing with LLMs
- Saves cost by reducing the number of LLM calls, if you're often requesting the same prompt multiple times
- Response time is much faster when requesting with the same prompt

```python
from langchain_openai import ChatOpenAI
from langchain.cache import InMemoryCache
from langchain.globals import set_llm_cache
from langchain_core.prompts import ChatPromptTemplate

chatgpt = ChatOpenAI(model_name="gpt-3.5-turbo",
                     temperature=0)

# create memory cache
set_llm_cache(InMemoryCache())

# The first time, it is not yet in cache, so it should take longer
prompt = """Explain to me what is mortgage"""
chat_template = ChatPromptTemplate.from_template(prompt)
chatgpt.invoke(chat_template.format())
# Output
# Wall time: 2.27 s
# AIMessage(content='A mortgage is a type of loan....

# Let's now send the same prompt to the LLM
# The cache should give the response back rather than the LLM
# Response time should also be much faster
chatgpt.invoke(chat_template.format())
# Output
# Wall time: 2.23 ms
# AIMessage(content='A mortgage is a type of loan....
```

Streaming

- LangChain has its LLM APIs implement the Runnable interface, which comes with implementations of methods for streaming like stream and astream
- This gives all LLMs basic support for streaming
- Enables us to stream LLM responses live rather than waiting to get the full response before showing it



Thank You

Analytics Vidhya