assignment4

September 4, 2024

1 Getting started with Generative-AI

Submitted By: Jenish Twayana

Submitted Date: 4th September, 2024

1.1 Exercise 4: Building a custom weather agent tool

Objective:

Create a conversational agent in Langchain that can perform the following task:
Provide the current temperature for Kathmandu.

Pre-requisites:

API Key = bd5e378503939ddaee76f12ad7a97608
url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"

1.1.1 Import the necessary packages and libraries

[709]:
```
import os
import requests
from langchain_openai import ChatOpenAI
from langchain_core.prompts import ChatPromptTemplate, MessagesPlaceholder
from langchain.agents.format_scratchpad.openai_tools import␣
  ↪format_to_openai_tool_messages
from langchain.agents.output_parsers.openai_tools import␣
  ↪OpenAIToolsAgentOutputParser
from langchain.agents import AgentExecutor
from langchain.agents import tool
```

1.1.2 Set up the environment variables for LangChain Tracing

It logs the tracing data in the LangSmith web interface.

[710]:
```
os.environ["LANGCHAIN_TRACING_V2"]="true" # enables the tracing
os.environ["LANGCHAIN_ENDPOINT"]="https://api.smith.langchain.com"
os.environ["LANGCHAIN_API_KEY"]=os.getenv("LANGCHAIN_API_KEY")
os.environ["LANGCHAIN_PROJECT"]="assignment-4" #project name in the LangSmith␣
  ↪platform
```

1.1.3 Initialise the Chat Model

[711]:
```
llm = ChatOpenAI(model="gpt-3.5-turbo", temperature=0)
```

1.1.4 Custom Tool for fetching Weather Data

[712]:
```
@tool
def get_weather_data(city: str) -> str:
    """Calls the Weather API and return the weather data
    Args:
        city: str
    Returns:
        str
    """
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={os.
↪getenv('WEATHER_API_KEY')}&units=metric"
    response = requests.get(url)
    return str(response.json())
```

1.1.5 Custom Tool for fetching Address Data

[713]:
```
@tool
def get_city_name(location: str) -> str:
    """Calls the Location API and returns the address data
    Args:
        location: str
    Returns:
        str
    """
    url = f"https://nominatim.openstreetmap.org/search?
↪q={location}&format=json&limit=1"

    headers = {
    'User-Agent': 'MyGeocodingApp/1.0 (your-email@example.com)'
}

    response = requests.get(url, headers=headers)
    if(len(response.json()) > 0):
        return response.json()[0]

    return "City not found"
```

1.1.6 List of Custom Tools

[714]:
```
tools = [ get_city_name, get_weather_data ]
```

1.1.7 Prompt Template for the Chat Model

[715]:
```
prompt = ChatPromptTemplate.from_messages(
    [
        (
            "system",
            """
            You are very powerful weather data assistant equipped with multiple␣
↪tools.
              Here is the detailed instruction:
              1. Call the Weather API to get the weather data of the city.
              2. If the Weather API returns valid response with weather data␣
↪then, return the weather data in given output format.
              3. If the Weather API returns no weather data then, call the␣
↪Location API to get the city name.
              4. If the Location API returns a valid address then, extract only␣
↪the city name from it.
              5. Call the Weather API again to get the weather data of the␣
↪extracted city.
              6. If the Weather API returns valid response with weather data␣
↪for the extracted city then, return the weather data in given output format␣
↪for the extracted city.
              7. If the Location API returns no valid city name or the Weather␣
↪API cant get the weather data of the city then, return the response
              saying the weather data of the city is not available.
              The desired output format for different scenarios are given below:
              ###
              #Scenario where the weather data of the city or extracted city is␣
↪available:
              Here is the weather data of the city <city_name>:
              <weather_data_in_bullet_form (dash separated)>
              #
              #Scenario where the weather data of the city or extracted city is␣
↪not available, or city name is not valid:
              The weather data of the city <city_name> is not available.
              #
              ###
              """,

        ),
        ("user", "{input}"),
        MessagesPlaceholder(variable_name="agent_scratchpad"), # sequence of␣
↪messages that contain the previous agent tool invocations and the␣
↪corresponding tool outputs
    ]
)
```

1.1.8 Bind the Custom tools to the Chat Model

[716]:
```
llm_with_tools = llm.bind_tools(tools)
```

1.1.9 Initialise the Agent

[717]:
```
agent = (
    {
        "input": lambda x: x["input"],
        "agent_scratchpad": lambda x: format_to_openai_tool_messages(
            x["intermediate_steps"]
        ),
    }
    | prompt
    | llm_with_tools
    | OpenAIToolsAgentOutputParser()
)
```

1.1.10 Initialise the Agent Executor

[718]:
```
agent_executor = AgentExecutor(agent=agent, tools=tools, verbose=True)
```

1.1.11 Execute the Agent with the input

[719]:
```
output = list(agent_executor.stream({"input": "Describe the weather in Nagarkot.
 ↪"}))
```

```
> Entering new AgentExecutor chain…
```

```
Invoking: `get_weather_data` with `{'city': 'Nagarkot'}`
{'coord': {'lon': 85.5167, 'lat': 27.7}, 'weather': [{'id':
803, 'main': 'Clouds', 'description': 'broken clouds', 'icon': '04d'}], 'base':
'stations', 'main': {'temp': 22.11, 'feels_like': 22.28, 'temp_min': 22.11,
'temp_max': 22.11, 'pressure': 1004, 'humidity': 73, 'sea_level': 1004,
'grnd_level': 804}, 'visibility': 7000, 'wind': {'speed': 4.12, 'deg': 210},
'clouds': {'all': 75}, 'dt': 1725450687, 'sys': {'type': 1, 'id': 9201,
'country': 'NP', 'sunrise': 1725407901, 'sunset': 1725453358}, 'timezone':
20700, 'id': 1283018, 'name': 'Nagarkot', 'cod': 200}###
Here is the weather data of the city Nagarkot:
- Weather: Clouds - broken clouds
- Temperature: 22.11°C
- Feels Like: 22.28°C
- Minimum Temperature: 22.11°C
- Maximum Temperature: 22.11°C
- Pressure: 1004 hPa
- Humidity: 73%
- Wind Speed: 4.12 m/s
- Visibility: 7000 meters
- Sunrise: Time not available
- Sunset: Time not available
###

> Finished chain.
```

1.1.12 Display the final output

```
[720]: print(output[-1]['output'])
```

```
###
Here is the weather data of the city Nagarkot:
- Weather: Clouds - broken clouds
- Temperature: 22.11°C
- Feels Like: 22.28°C
- Minimum Temperature: 22.11°C
- Maximum Temperature: 22.11°C
- Pressure: 1004 hPa
- Humidity: 73%
- Wind Speed: 4.12 m/s
- Visibility: 7000 meters
- Sunrise: Time not available
- Sunset: Time not available
###
```