mollie

Introduction to LLM Agents

Ana Chaloska
Maria Bader, Ph.D.

18 September 2024

PyData

Who are we?

Ana Chaloska
Data Scientist
Mollie

Maria Bader
Data Scientist
Mollie

Workshop goals

coding / theory

- Understand how LLMs can **perform tasks** beyond text generation
- Grasp the **fundamental components** of an LLM Agent
- **Build an LLM Agent** with LangChain and/or the OpenAI chat completion API

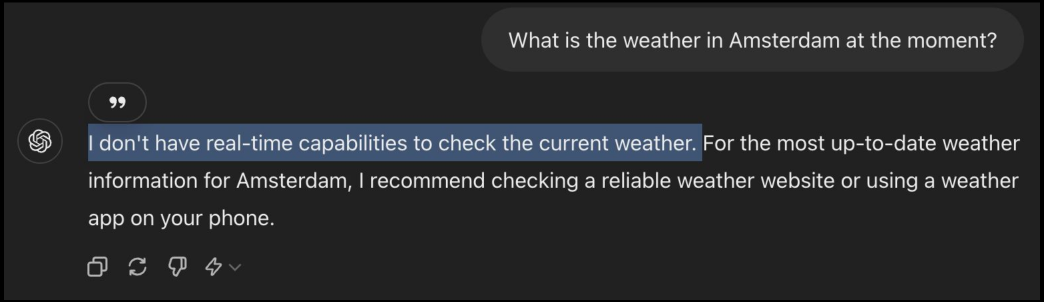
What LLMs can do

- **Question-answering** (e.g. chatbots)
- Text **summarization**
- **Sentiment** analysis
- Generate **code**
- **Translate** text
- Generate **embeddings** (BCC Classifier)
- etc

What LLMs can’t do

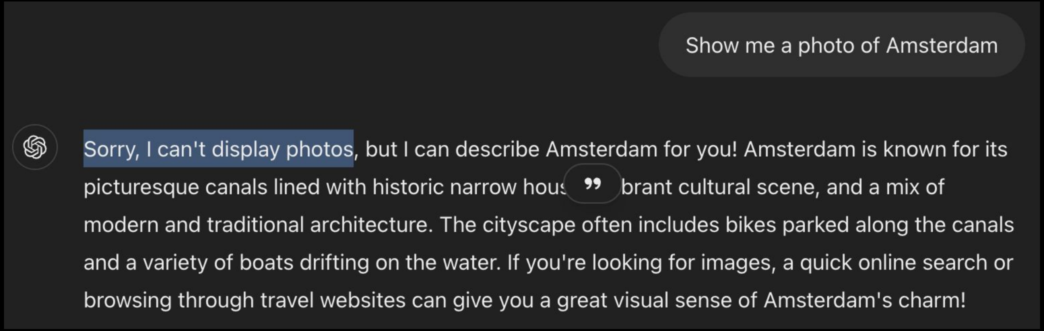
- Usually **computational/mathematical** tasks
- Awareness about **real-time** information
- Provide **non-text output**

What LLMs can’t do



What is the weather in Amsterdam at the moment?

I don't have real-time capabilities to check the current weather. For the most up-to-date weather information for Amsterdam, I recommend checking a reliable weather website or using a weather app on your phone.



Show me a photo of Amsterdam

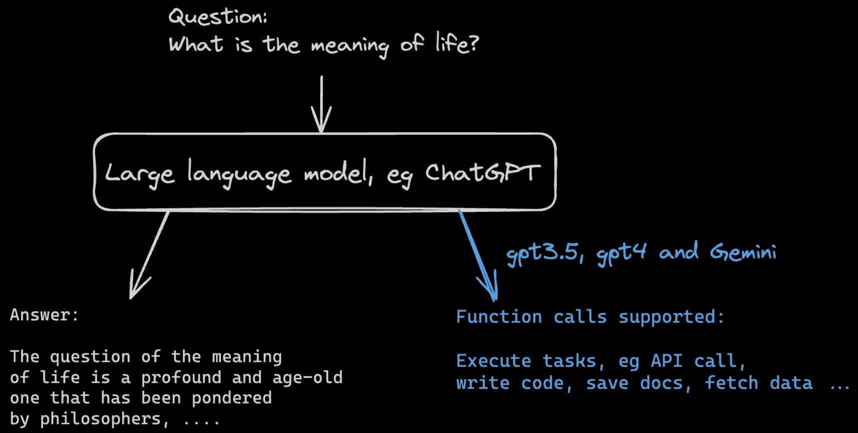
Sorry, I can't display photos, but I can describe Amsterdam for you! Amsterdam is known for its picturesque canals lined with historic narrow hous brant cultural scene, and a mix of modern and traditional architecture. The cityscape often includes bikes parked along the canals and a variety of boats drifting on the water. If you're looking for images, a quick online search or browsing through travel websites can give you a great visual sense of Amsterdam's charm!

Can we help LLMs?

Yes.

How?

Nowadays there are LLMs that handle not only text but also **function calls** (logic in a function, API etc).


Question:
What is the meaning of life?
Large language model, eg ChatGPT
gpt3.5, gpt4 and Gemini

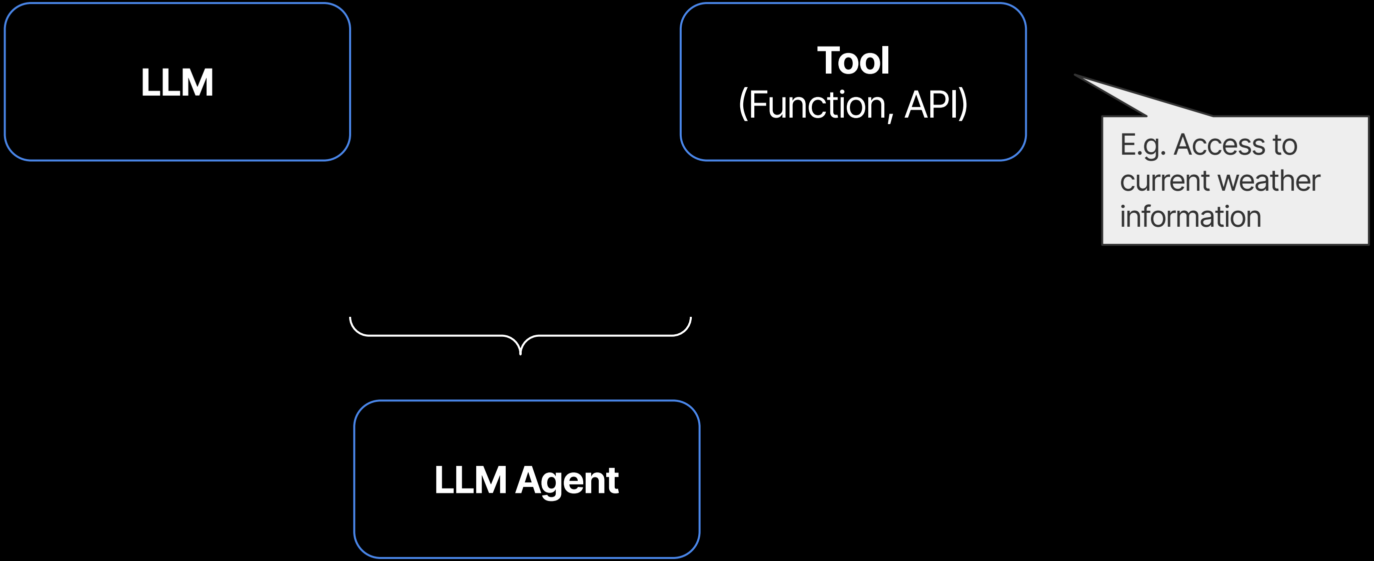
```
Answer:

The question of the meaning
of life is a profound and age-old
one that has been pondered
by philosophers, ....
```

```
Function calls supported:

Execute tasks, eg API call,
write code, save docs, fetch data ...
```

LLM Agents



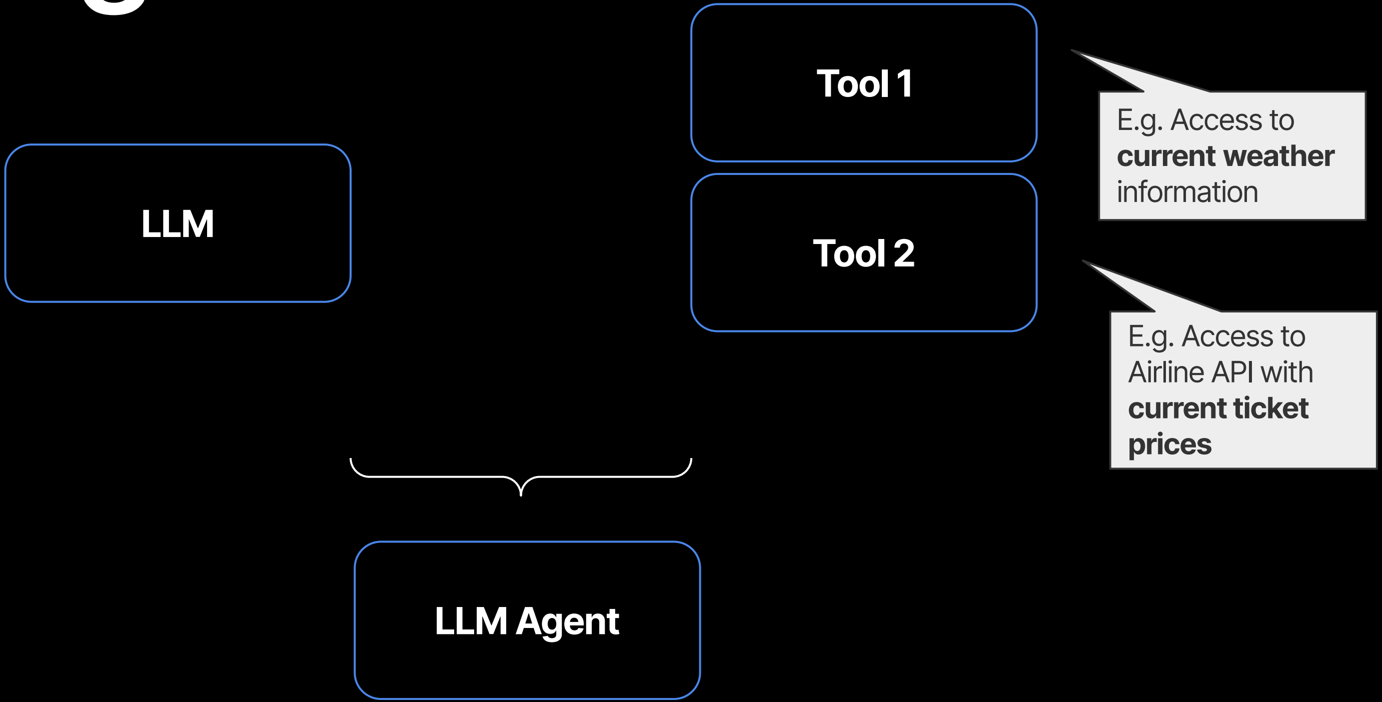
LLM

Tool
(Function, API)

E.g. Access to current weather information

LLM Agent

LLM Agents



LLM

Tool 1

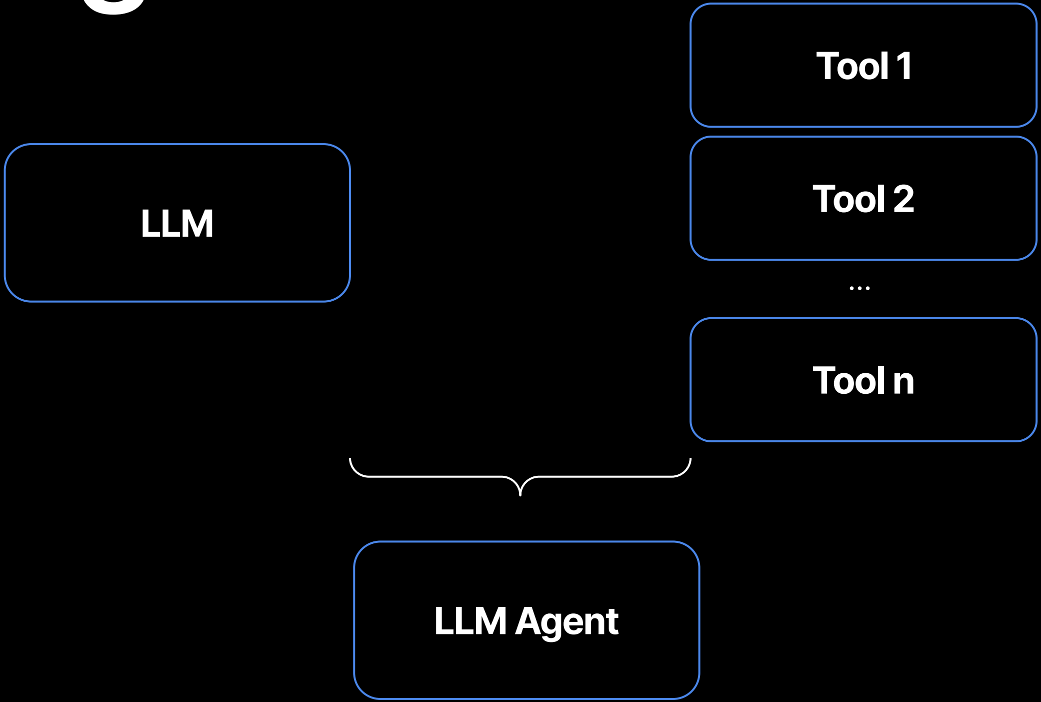
E.g. Access to **current weather** information

Tool 2

E.g. Access to Airline API with **current ticket prices**

LLM Agent

LLM Agents



LLM

Tool 1

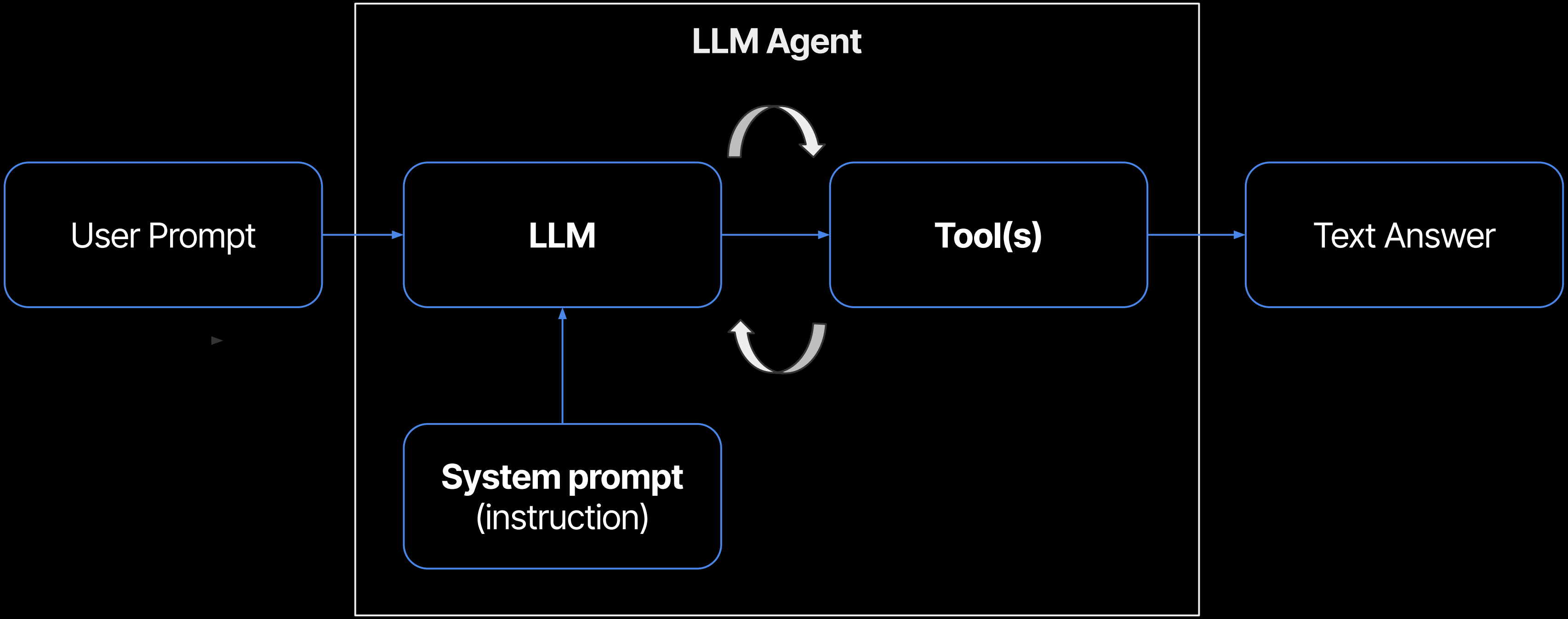
Tool 2

...

Tool n

LLM Agent

LLM Agents



LLM Agent

User Prompt

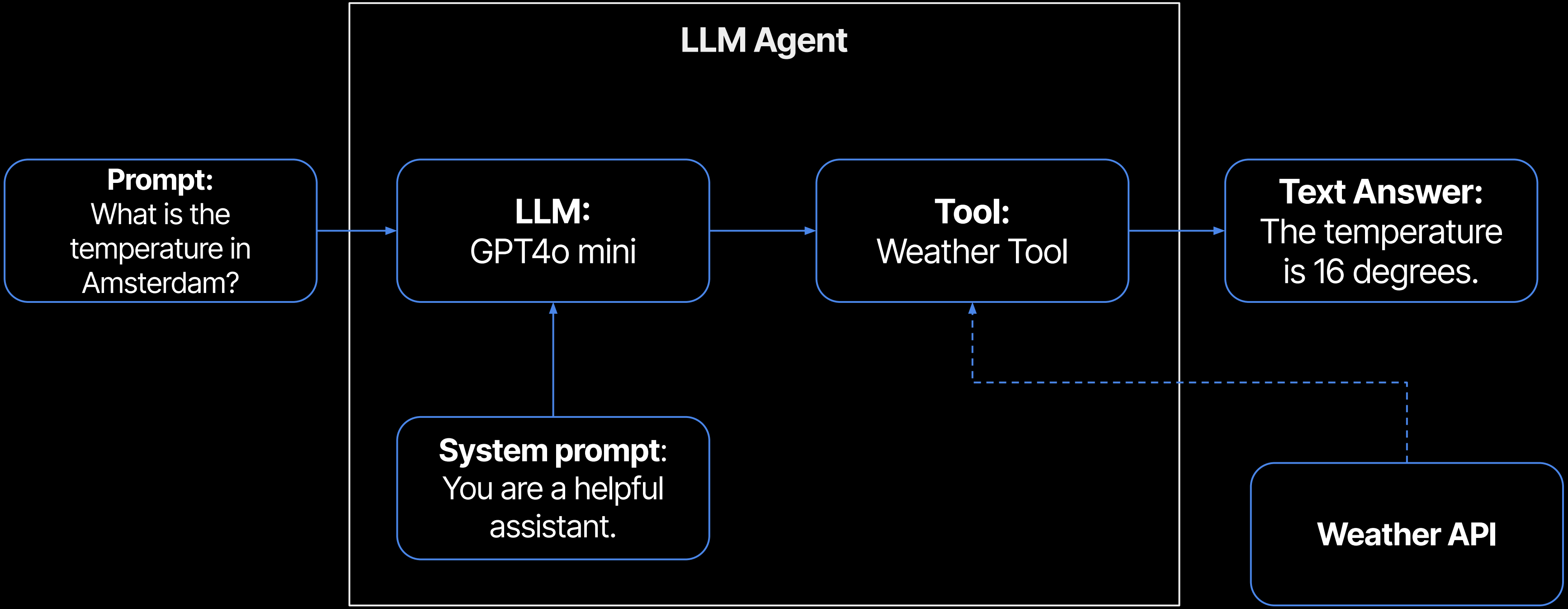
LLM

Tool(s)

Text Answer

System prompt
(instruction)

LLM Agents



LLM Agent

Prompt:
What is the temperature in Amsterdam?

LLM:
GPT4o mini

System prompt:
You are a helpful assistant.

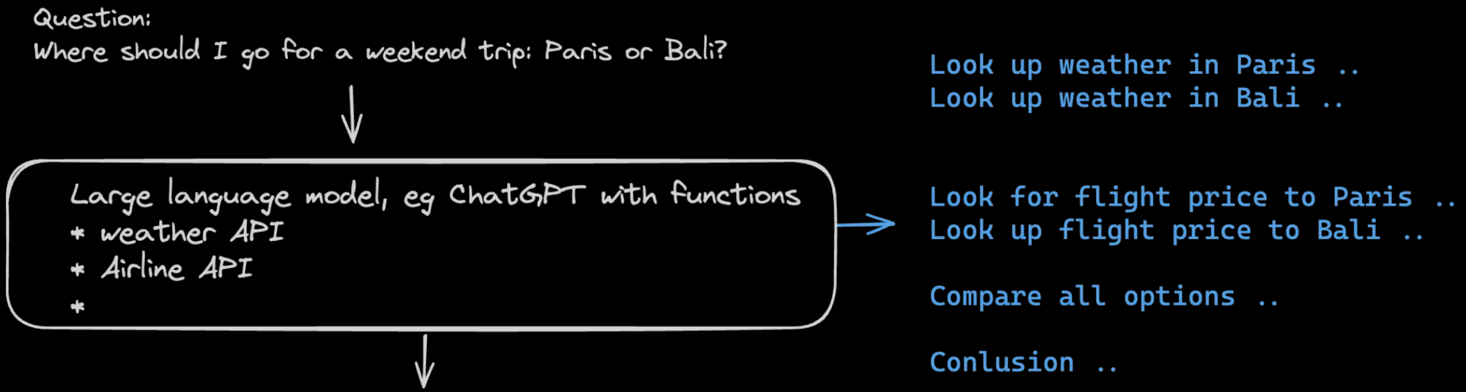
Tool:
Weather Tool

Weather API

Text Answer:
The temperature is 16 degrees.

LLM agents

LLM choses from a list of provided functions which actions to take to complete a task.



Question:
Where should I go for a weekend trip: Paris or Bali?

Large language model, eg ChatGPT with functions
- weather API
- Airline API
-

```
Look up weather in Paris ..
Look up weather in Bali ..

Look for flight price to Paris ..
Look up flight price to Bali ..

Compare all options ..

Conlusion ..
```

```
Answer:

The temperature in Bali is 30 degree, while in Paris it is 15 degrees.
Flight prices to Bali are in general much higher than to Paris. Since the
temperature in Paris is nice for a city trip, and the flights are cheaper,
Paris is a more popular destination for a short weekend trip.
```

Getting started with the workshop

In this workshop

You will implement an LLM agent, who has access to

- the wikipedia API,
- the virtual crossing weather API,
- the HuggingFace API to generate images.

You will develop a program (agent) that can help you make a **decision** about your future winter holiday.

Workshop setup

LangChain

Wrapper framework for developing LLM applications

Notebook 1

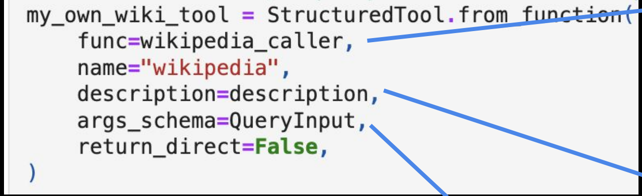
ChatGPT

ChatGPT 4o mini chat completion API

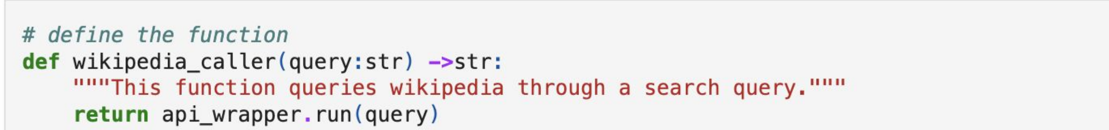
Notebook 2

Tools with Langchain

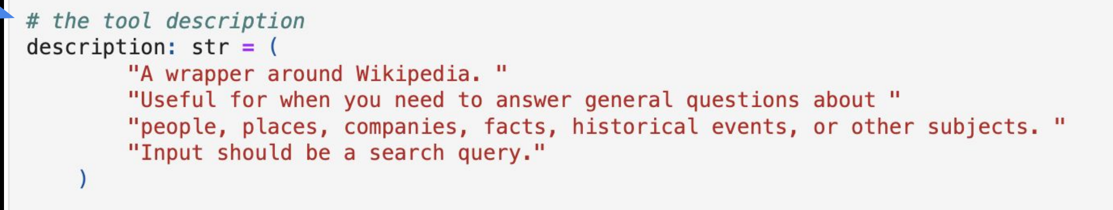
Tools are interfaces/APIs that an LLM can use to extract information from.



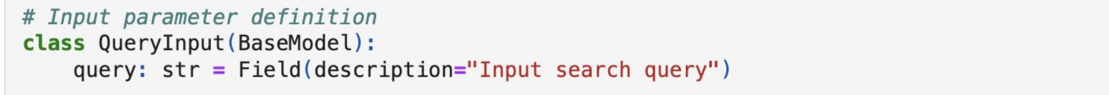
```
my_own_wiki_tool = StructuredTool.from_function(
    func=wikipedia_caller,
    name="wikipedia",
    description=description,
    args_schema=QueryInput,
    return_direct=False,
)
```



```
# define the function
def wikipedia_caller(query:str) ->str:
    """This function queries wikipedia through a search query."""
    return api_wrapper.run(query)
```



```
# the tool description
description: str = (
        "A wrapper around Wikipedia. "
        "Useful for when you need to answer general questions about "
        "people, places, companies, facts, historical events, or other subjects. "
        "Input should be a search query."
    )
```

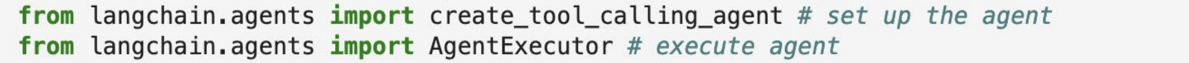


```
# Input parameter definition
class QueryInput(BaseModel):
    query: str = Field(description="Input search query")
```

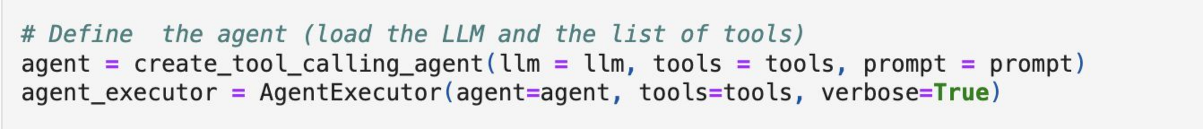
Function calling with LangChain

LangChain

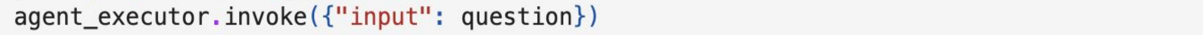
Allow and LLM to choose a sequence of actions from a list of tools.



```
from langchain.agents import create_tool_calling_agent # set up the agent
from langchain.agents import AgentExecutor # execute agent
```



```
# Define  the agent (load the LLM and the list of tools)
agent = create_tool_calling_agent(llm = llm, tools = tools, prompt = prompt)
agent_executor = AgentExecutor(agent=agent, tools=tools, verbose=True)
```



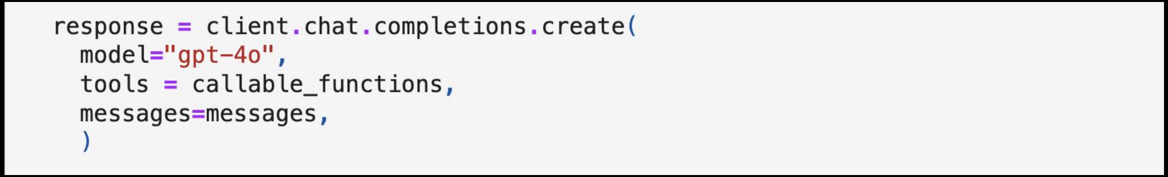
```
agent_executor.invoke({"input": question})
```

👉 [1_workshop_agent_with_langchain.ipynb.ipynb]

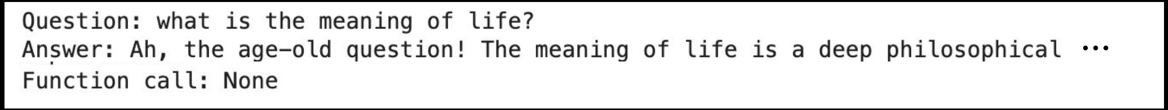
ChatGPT

Function calling with the chat completion API

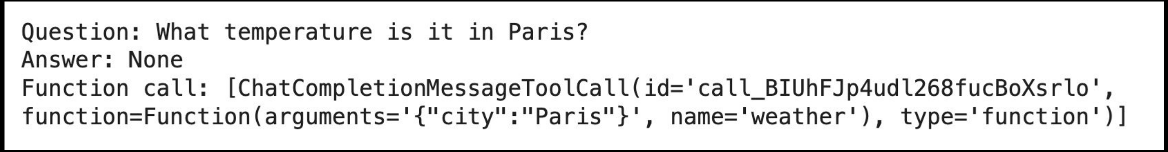
API returns either returns a string or a function call object



```
response = client.chat.completions.create(
  model="gpt-4o",
  tools = callable_functions,
  messages=messages,
  )
```



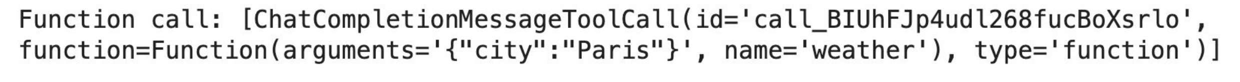
```
Question: what is the meaning of life?
Answer: Ah, the age-old question! The meaning of life is a deep philosophical …
Function call: None
```



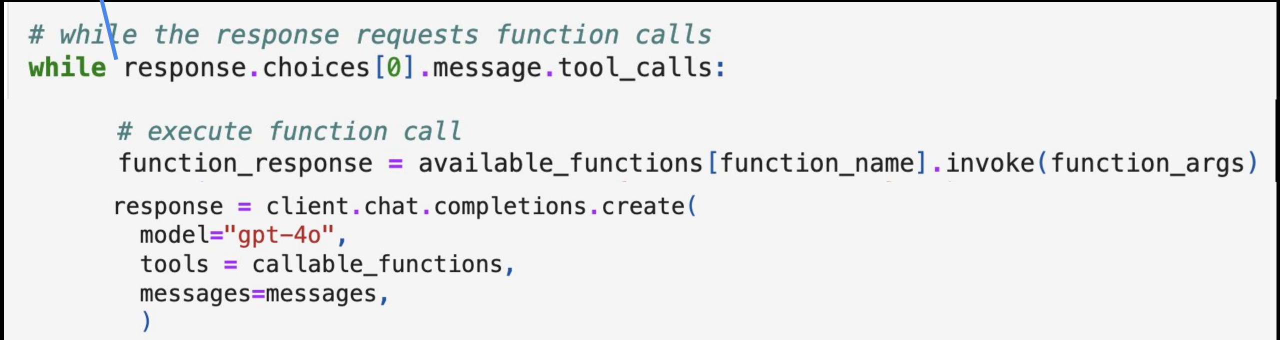
```
Question: What temperature is it in Paris?
Answer: None
Function call: [ChatCompletionMessageToolCall(id='call_BIUhFJp4udl268fucBoXsrlo',
function=Function(arguments='{"city":"Paris"}', name='weather'), type='function')]
```

Function calling with the chat completion API

Execute function calls until response is a text object



```
Function call: [ChatCompletionMessageToolCall(id='call_BIUhFJp4udl268fucBoXsrlo',
function=Function(arguments='{"city":"Paris"}', name='weather'), type='function')]
```



```
# while the response requests function calls
while response.choices[0].message.tool_calls:

    # execute function call
    function_response = available_functions[function_name].invoke(function_args)
    response = client.chat.completions.create(
      model="gpt-4o",
      tools = callable_functions,
      messages=messages,
      )
```

2_workshop_agent_with_chat_completion_endpoint.ipynb

Advantage of using the chat completions endpoint

ChatGPT

Added complexity gives freedom to tailor functionalities as needed

- Explicit function management allows for safeguards for hallucination of function names and arguments

- Dynamic update of messages objects or tool descriptions allows for additional formatting, checking and optimization

- Robustness towards library changes

Let’s get coding

Link to the **repo**

Link to the **slides**

Thank you

Scan to connect with
Ana Chaloska

Scan to connect with
Maria Bader

PyData

mollie