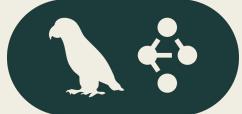


Learning LangGraph



Course by **Matteo Falcioni**

Mail: ***matteo.falcioni3@unibo.it***

GitHub: ***<https://github.com/MatteoFalcioni>***

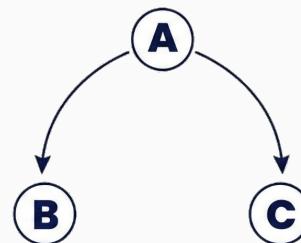
Unibo, DIFA
Science City Lab

Table of Contents

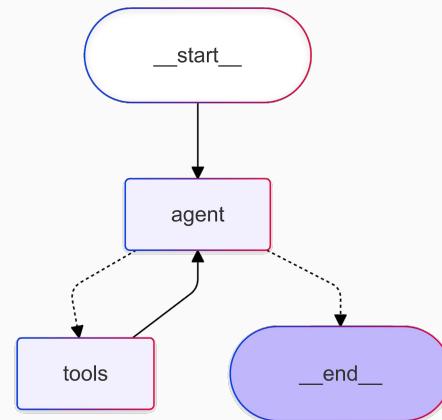
- | First Part (fundamentals) | Second Part (advanced) |
|--|--|
| • 1. Introduction | • 10. Supervised Graph |
| • 2. A Simple Graph | • 11. Hierarchical Agent Teams |
| • 3. LLMs & Agents in LangGraph | • 12. Project: Multimodality - Vision Agents |
| • 4. State Schema & Reducers | • 13. Project: Multimodality - Voice Agents |
| • 5. The Command Primitive | • 14. Project: Towards <i>UrbIA</i> - complexifying our supervised graph |
| • 6. Agentic Graph, with memory | • 15. Project: RAG Agent |
| • 7. Agents Collaboration | • 16. Project: <i>Your Suggestions</i> |
| • 8. Middleware: llm summarization, human in the loop, todo list | • 17. Building a Production Ready Agent: <i>UrbIA</i> as an example of how to deploy and scale |
| • 9. Deep Agents | |

LangChain & LangGraph

LangGraph is part of the LangChain ecosystem, but their structure is inherently different

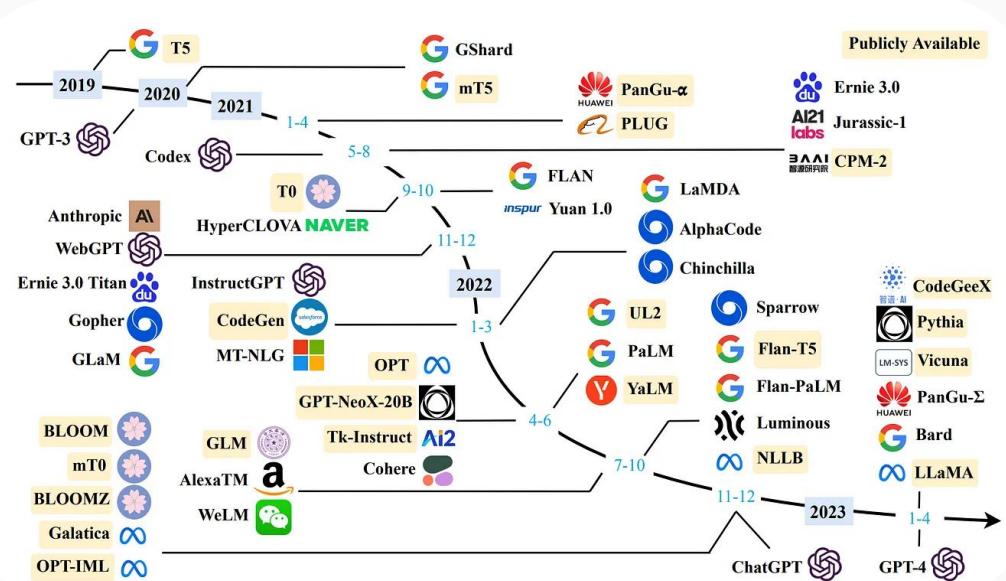
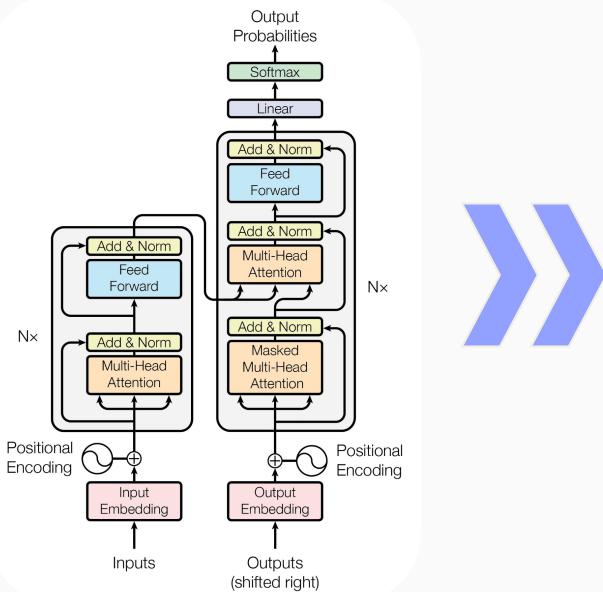


LangGraph allows us to have **loops** (→ agents with tools)



Large Language Models - LLMs

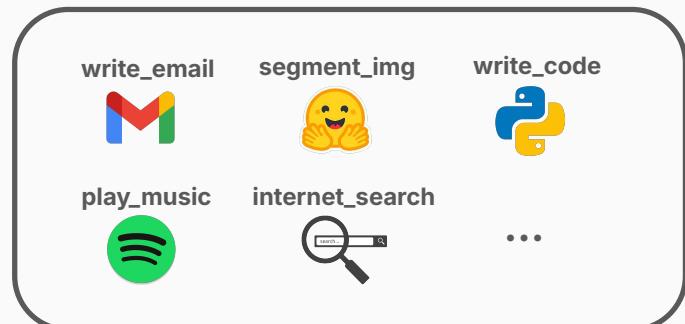
Attention is all you need!
(2017)



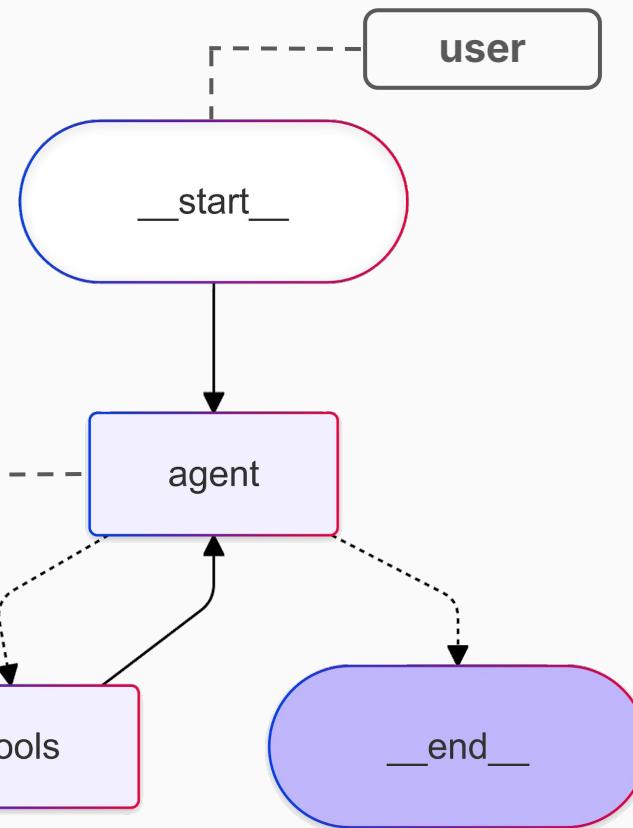
LLMs >> AI AGENTS

Can we make an LLM perform complex tasks,
other than just chatting?

Tools = arms



LLM = brain



Tools Examples

Usual tool structure

```
59 @tool
60 def read_sources_tool(
61     | runtime: ToolRuntime
62 ) -> Command:
63 """
64 Get the sources used in the analysis.
65 """
66 state = runtime.state
67 sources = state["sources"]
68 sources_str = "\n".join([f"- {source}" for source in sources])
69
70 return Command[tuple[()]](
71     | update = {
72         "messages" : [ToolMessage(content=f"Sources: {sources_str}", tool_call_id=runtime.tool_call_id)],
73     }
74 )
```

Tools w/ API calls

```
1 from tavily import TavilyClient
2 import os
3 from langchain_core.tools import tool
4
5 # ----- Internet Search Tool -----
6
7 @tool
8 def internet_search(query):
9     """Search the internet for information"""
10    tavily_client = TavilyClient(api_key=os.environ["TAVILY_API_KEY"])
11    return tavily_client.search(query)
```

Resources

- Course Notebooks on GitHub: <https://github.com/MatteoFalcioni/Learning-LangGraph>
- LangChain Academy: [LangChain Academy](#)
- LangGraph Graph API: [Graph API overview - Docs by LangChain](#)
- LangChain Docs: [LangChain overview - Docs by LangChain](#)
- LangChain Reference: [Home | LangChain Reference](#)
- LangChain OpenTutorial:  [The LangChain Open Tutorial for Everyone](#)

For any information or doubts don't hesitate to contact me at matteo.falcioni3@unibo.it