Agents: The final boss of LLMs

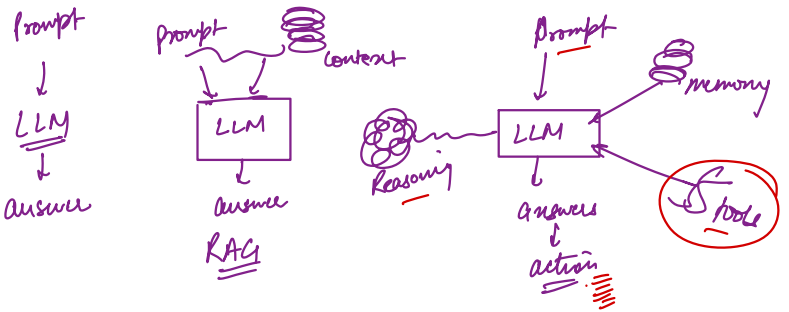
↳ final level of your LLMs
↳ Serpai
↳ complex & autonomous tasks
↳ go beyond your conversation or information retrieval.

How agents are better?

1. Autonomy and Task Execution → multi-step process
   ↳ various tool / system
2. Dynamic Decision-Making
3. Integration with Tools and APIs → Math, Map
4. Contextual Understanding and Memory

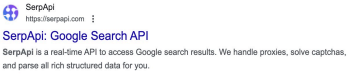
your inputs change → change your strategy

RAG →



Tools provide LLMs with a way to interact with the real world and access information beyond their training data.

Eg 1: Tools like **search engines** (Serpapi)allow LLMs to fetch the latest news, research, or any other relevant data.

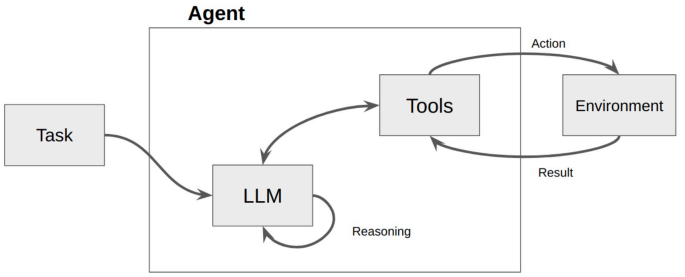


Tools can enable LLMs to execute tasks like sending emails, booking appointments, or making calculations.

Eg 2: Specialized tools can extend an LLM's abilities, such as using a code interpreter for mathematical operations or a database for information retrieval

BUT How do they work?

- Receive input from the user / goal
- Reasoning → Language model analyses i/p & determine the set of actions reqd to achieve the goal.
- Action Plan → LLM output → plan a seq of actions
- Action execution → Calling APIs, searching the web, interacting with another system.
- Feedback → feedback is fed to the LLM back → adjust plan of actions



Agent vs. Chain?

Chain is a subsequence of actions to take, always in a hardcoded manner. This is the crucial difference between an agent and a chain. **While in the agent, *the reasoning model* may choose other actions (from given tools) to fetch specific data**, *the chain will always take the same path that we have chosen.*

Components of an Agent?

Language Models

Tools and Actions

Memory

Types of tools?

- Search & Retrieve
- API interactions
- Code execution / Calculations
- Data storage or memory
- Automation / workflow tools
- Data Query / Analysis
- Content generation
- Viz & reporting

https://python.langchain.com/v0.2/docs/integrations/tools/