Agent types

Action agents

Agents use an LLM to determine which actions to take and in what order. An action can either be using a tool and observing its output, or returning a response to the user. Here are the agents available in LangChain.

Zero-shot ReAct

This agent uses the ReAct framework to determine which tool to use based solely on the tool's description. Any number of tools can be provided. This agent requires that a description is provided for each tool.

Note: This is the most general purpose action agent.

Structured input ReAct

The structured tool chat agent is capable of using multi-input tools. Older agents are configured to specify an action input as a single string, but this agent can use a tools' argument schema to create a structured action input. This is useful for more complex tool usage, like precisely navigating around a browser.

OpenAI Functions

Certain OpenAI models (like gpt-3.5-turbo-0613 and gpt-4-0613) have been explicitly fine-tuned to detect when a function should be called and respond with the inputs that should be passed to the function. The OpenAI Functions Agent is designed to work with these models.

Conversational

This agent is designed to be used in conversational settings. The prompt is designed to make the agent helpful and conversational. It uses the ReAct framework to decide which tool to use, and uses memory to remember the previous conversation interactions.

Self ask with search

This agent utilizes a single tool that should be named `Intermediate Answer`. This tool should be able to lookup factual answers to questions. This agent is equivalent to the original self ask with search paper, where a Google search API was provided as the tool.

ReAct document store

This agent uses the ReAct framework to interact with a docstore. Two tools must be provided: a `Search` tool and a `Lookup` tool (they must be named exactly as so). The `Search` tool should search for a document, while the `Lookup` tool should lookup a term in the most recently found document. This agent is equivalent to the original ReAct paper, specifically the Wikipedia example.

Plan-and-execute agents

Plan and execute agents accomplish an objective by first planning what to do, then executing the sub tasks. This idea is largely inspired by BabyAGI and then the "Plan-and-Solve" paper.