



Agents

From ReAct to {insert most recent paper}



Agenda

- What are agents?
- Why use agents?
- Typical implementation
- ReAct
- Challenges with agents
- Memory
- Recent papers/projects

What are agents?

- Use LLM as the “reasoning engine”
- Non-deterministic sequence of actions

Why use agents?

- Connect LLM to outside sources of data/computation
 - Search
 - APIs
 - Database
 - Calculators
 - Run code
- Recover from errors, handle multi-hop tasks

Example: SQL query

- Chain: Natural Language -> SQL Query -> Result -> Natural Language
- What if error in SQL Query?
- What if multiple queries need to be made?

Typical Implementation

Pseudocode:

- Choose tool to use
- Observe output of the tool
- Repeat until stopping condition met

Stopping condition:

- LLM determined
- Hardcoded rules

ReAct

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1a) Standard

Answer: iPod



(1b) CoT (Reason Only)

Thought: Let's think step by step. Apple Remote was originally designed to interact with Apple TV. Apple TV can be controlled by iPhone, iPad, and iPod Touch. So the answer is iPhone, iPad, and iPod Touch.

Answer: iPhone, iPad, iPod Touch



(1c) Act-Only

Act 1: Search[Apple Remote]

Obs 1: The Apple Remote is a remote control ...

Act 2: Search[Front Row]

Obs 2: Could not find [Front Row]. Similar: ...

Act 3: Search[Front Row (software)]

Obs 3: Front Row is a discontinued media center software ...

Act 4: Finish[yes]



(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.

Act 1: Search[Apple Remote]

Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

Act 2: Search[Front Row]

Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

Thought 3: Front Row is not found. I need to search Front Row (software) .

Act 3: Search[Front Row (software)]

Obs 3: Front Row is a discontinued media center software ...

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

Act 4: Finish[keyboard function keys]



Challenges with agents

- Getting them to use tools in appropriate scenarios
- Getting them to not use tools when not needed
- Parsing LLM Output to tool invocation
- Getting them to “remember” previous steps taken
- Getting them to incorporate long observations
- Getting them to stay focused
- Evaluation

Challenge #1: Getting them to use tools in appropriate scenarios

- Instructions
- Tool descriptions in prompts
- Tool retrieval before hand
- Few shot examples
- Fine tuned model (toolformer)

Challenge #2: Getting them to not use tools when not needed

- Instructions
- “Human Input” tool

Challenge #3: Parsing LLM Output to tool invocation

- JSON/Typescript schema
- Retrying/fixing logic
- Output Parsers!

```
# Define your desired data structure.
class Joke(BaseModel):
    setup: str = Field(description="question to set up a joke")
    punchline: str = Field(description="answer to resolve the joke")

# You can add custom validation logic easily with Pydantic.
@validator('setup')
def question_ends_with_question_mark(cls, field):
    if field[-1] != '?':
        raise ValueError("Badly formed question!")
    return field
```

```
# Set up a parser + inject instructions into the prompt template.
parser = PydanticOutputParser(pydantic_object=Joke)
```

Challenge #4: Getting them to “remember” previous steps taken

- List of steps in memory
- Fetch previous steps with retrieval

Challenge #5: Getting them to incorporate long observations

- Parsing the long output
- Storing long output and doing retrieval

Challenge #6: Getting them to stay focused

- Reiterate the objective
- Separate planning/execution step

Challenge #7: Evaluation

- Evaluate end result
- Evaluate intermediate steps
 - Correct action
 - Correct action input
 - Correct sequence of steps
 - Most efficient sequence of steps

Memory

- Memory of User <> AI interactions
- Memory of AI <> Tool interactions
- Used to achieve personalization

AutoGPT

- Long term memory of Agent <> Tool interactions

<https://github.com/Significant-Gravitas/Auto-GPT>

BabyAGI

- Long term memory of Agent <> Tool interactions
- Separate planning / execution steps

<https://github.com/yoheinakajima/babyagi>

CAMEL

- Simulation environment

<https://www.camel-ai.org/>

Generative Agents

- Memory
 - Time weighted
 - Importance weighted
 - Relevancy weighted
- Reflection step
- Simulation environment

<https://arxiv.org/abs/2304.03442>