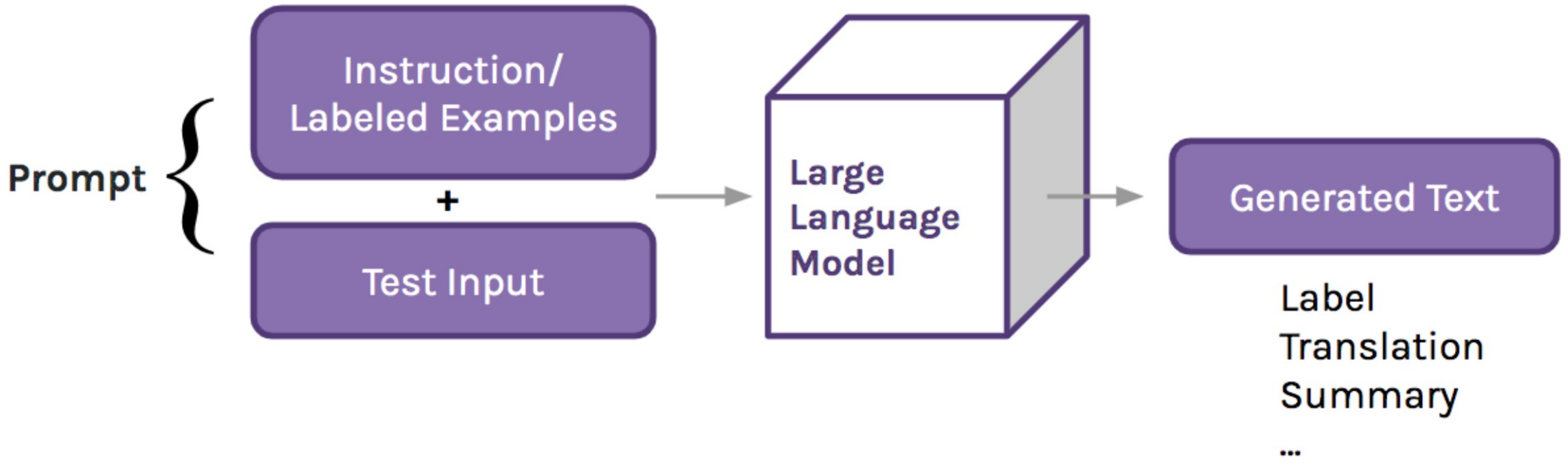
# Language Models as Agents

Yinxu Tang

# Background

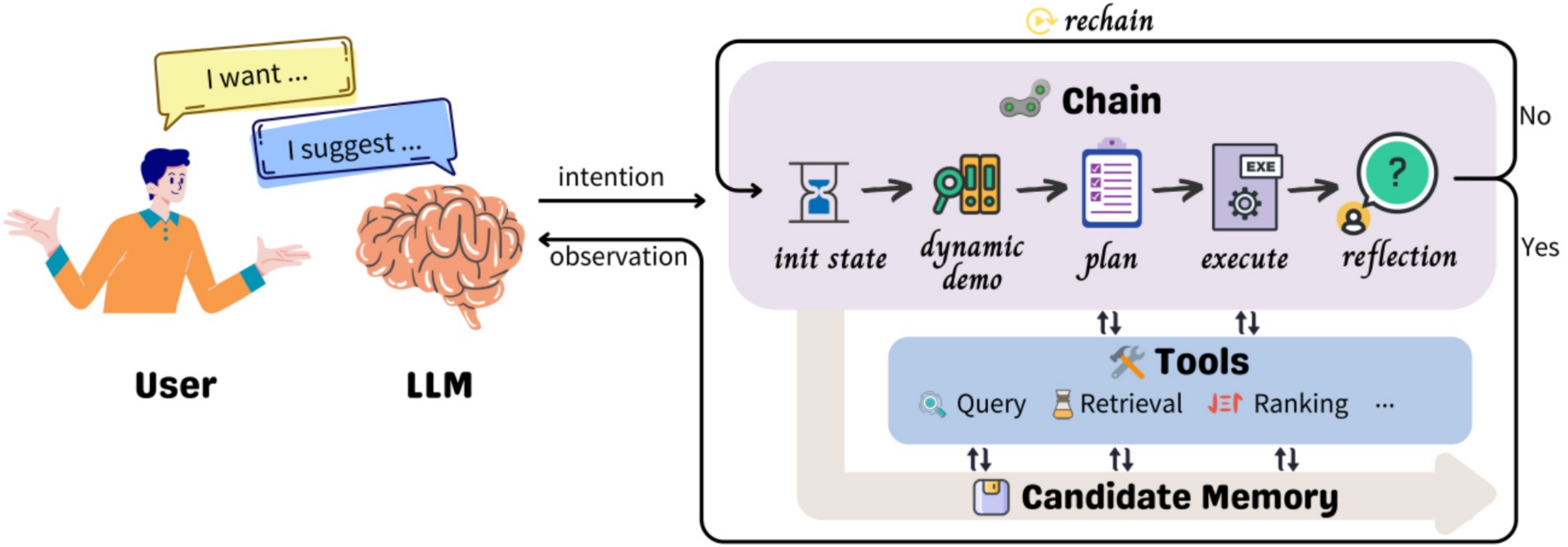
- Language tasks (*e.g.*, label, translation, summary)



*Flow chart of how in-context learning works with LLMs.*

# Background

- LLM-based agents
  - Built on the actual language models
  - Plan tasks, respond to queries, or solve problems with a range of tools



*An example framework.*

# Language Models as Agents

I. Reasoning and Acting

II. Use of External Tools

# Language Models as Agents

I. Reasoning and Acting

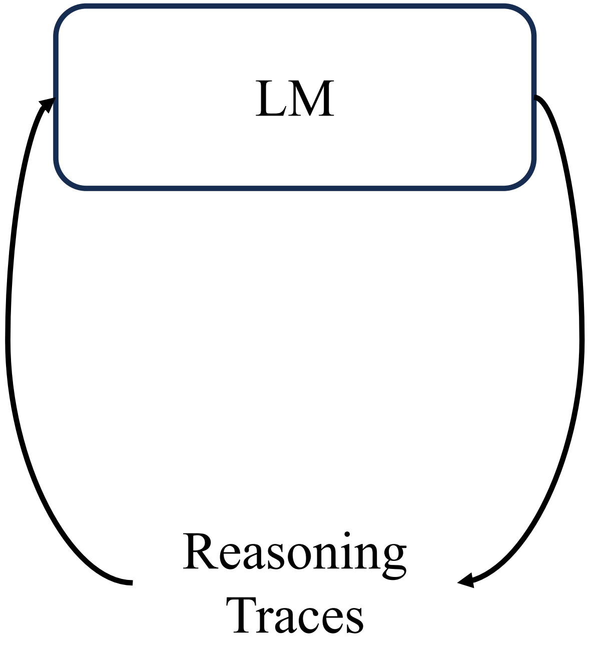
II. Use of External Tools

# Reasoning and Acting

- Language models (LM) get better at reasoning and acting, separately.

# Reasoning

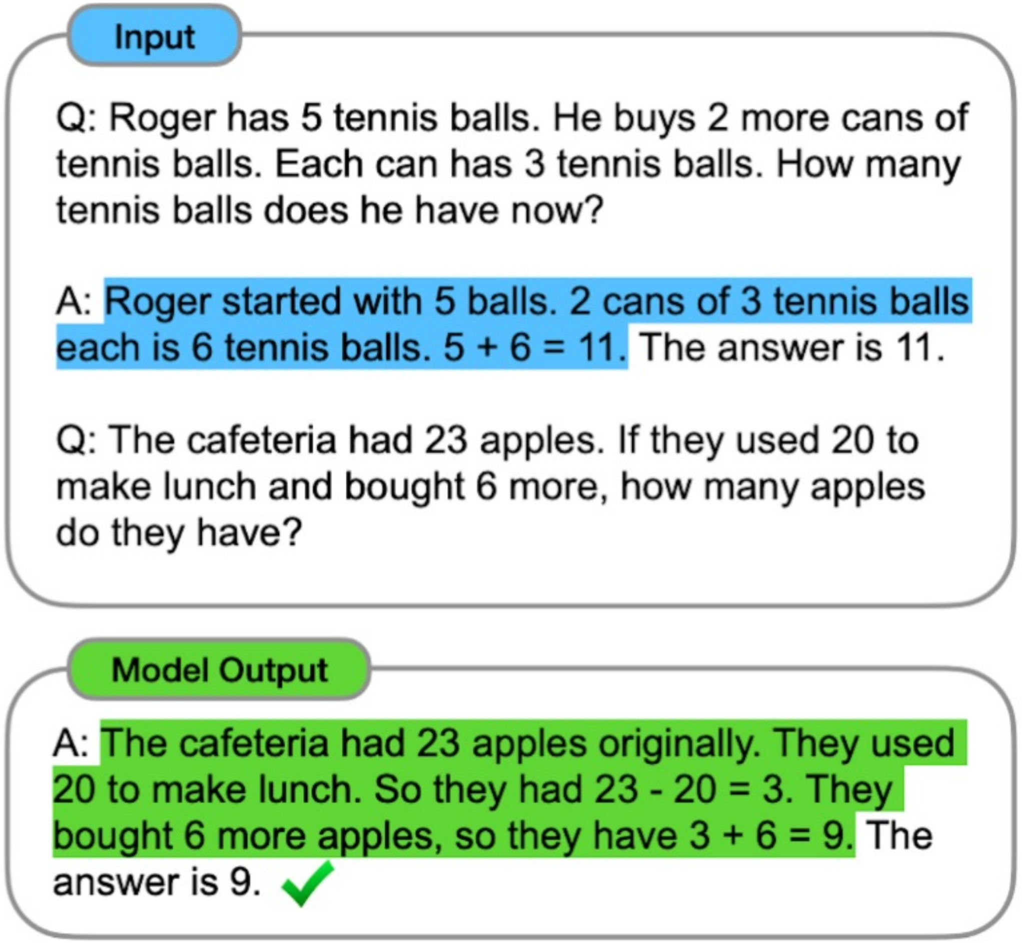
- Self-conditioned reasoning traces



LM

Reasoning Traces

*e.g.*, Chain-of-thought, Self-consistency



Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

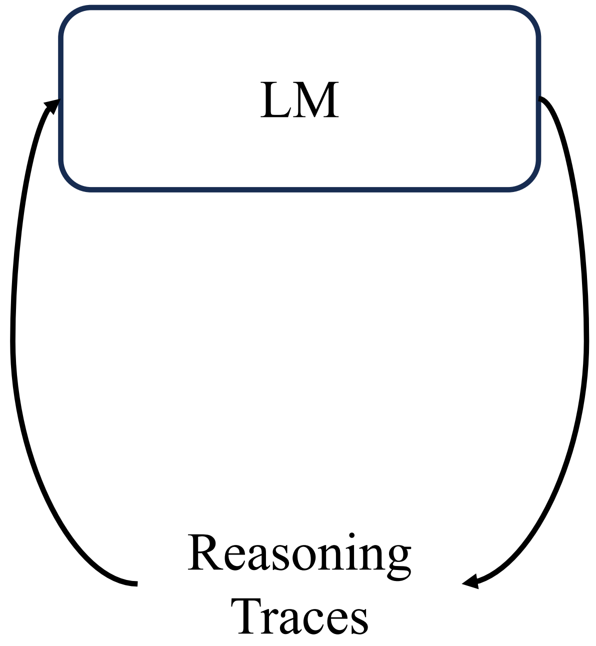
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

# Reasoning

- Self-conditioned reasoning traces

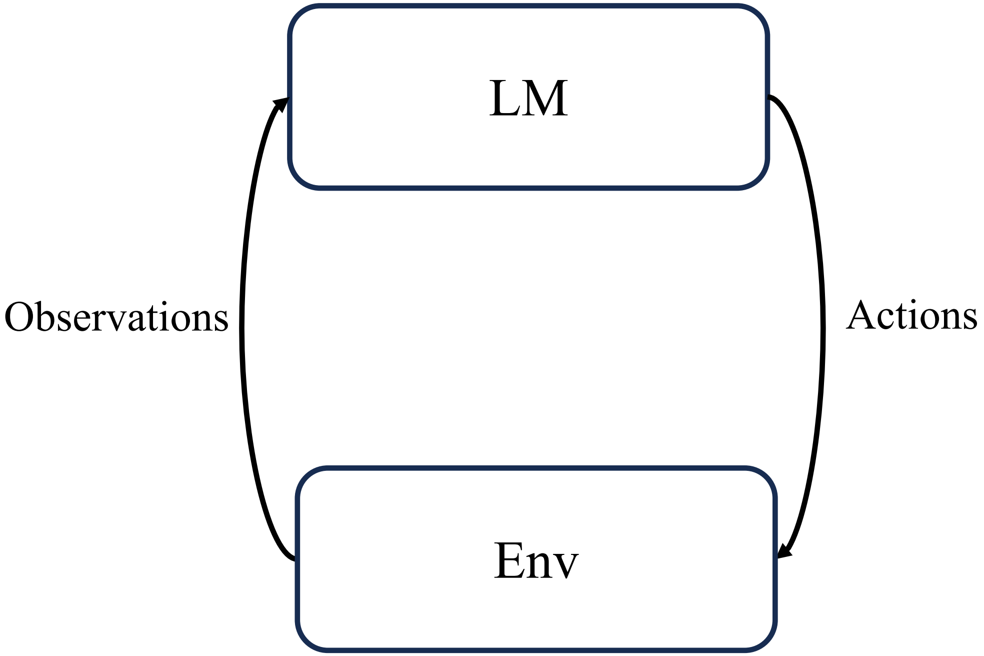


- Hallucination
- Error propagation

*e.g.*, Chain-of-thought, Self-consistency

# Acting

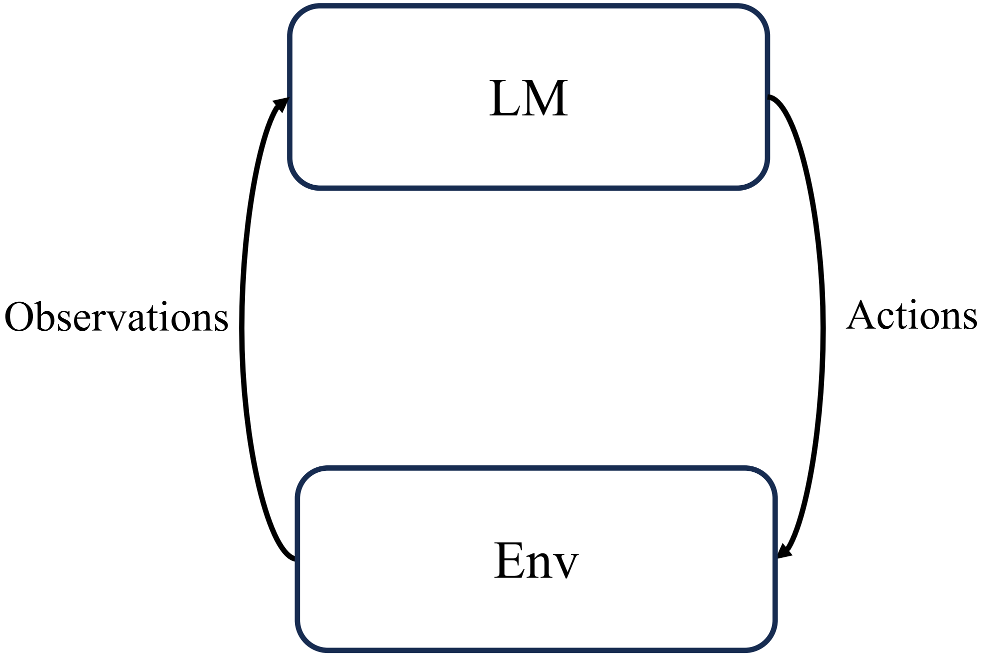
- Interaction with external environments
- Generate text actions/plan



*e.g.*, Saycan, WebGPT

# Acting

- Interaction with external environments
- Generate text actions/plan



LM

Env

Observations

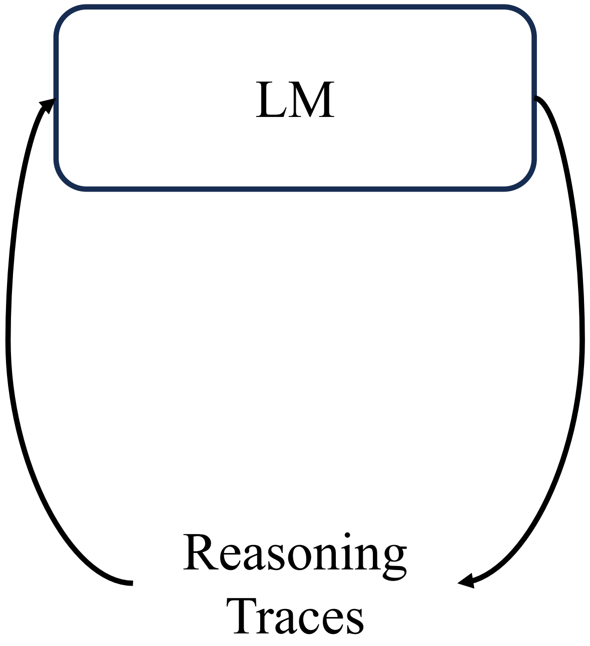
Actions

- Generalization
- Transfer

*e.g.*, Saycan, WebGPT

# ReAct: Reasoning and Acting

- ReAct[1]: generate both



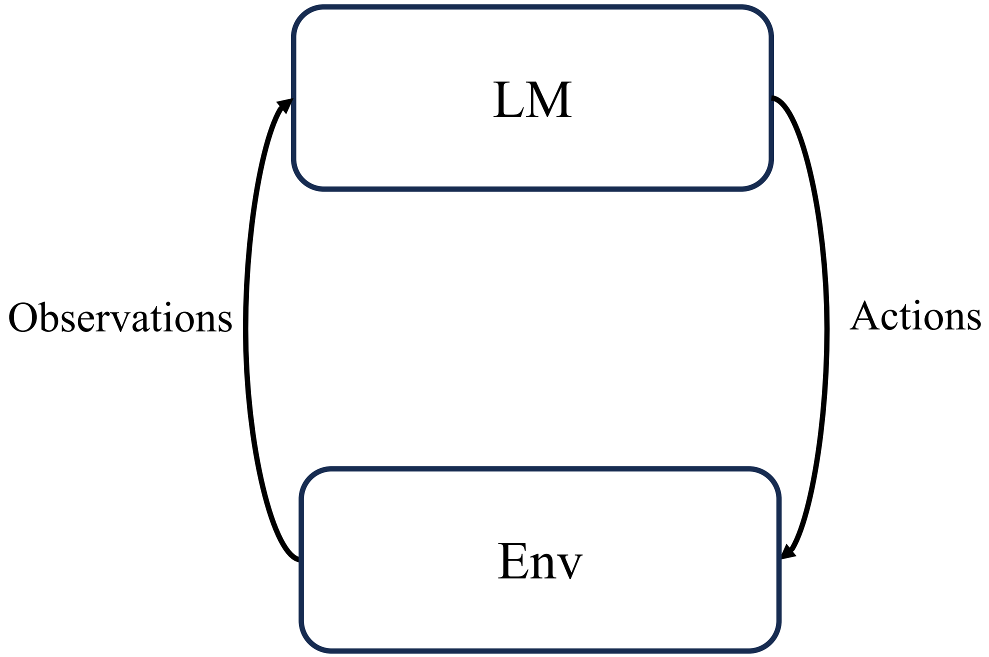
LM

Reasoning Traces

*e.g.*, Chain-of-thought, Self-consistency

Reasoning only

+



LM

Observations

Actions

Env

*e.g.*, Saycan, WebGPT

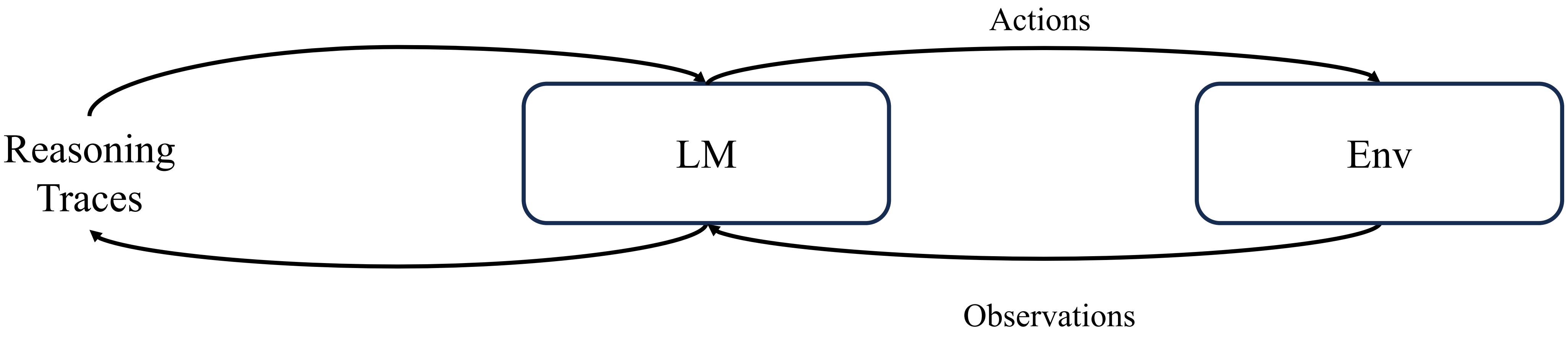
Acting only

---

*[1] Yao S, Zhao J, Yu D, Du N, Shafran I, Narasimhan K, Cao Y. "React: Synergizing reasoning and acting in language models." In NeurIPS Foundation Models for Decision Making Workshop, 2022.*

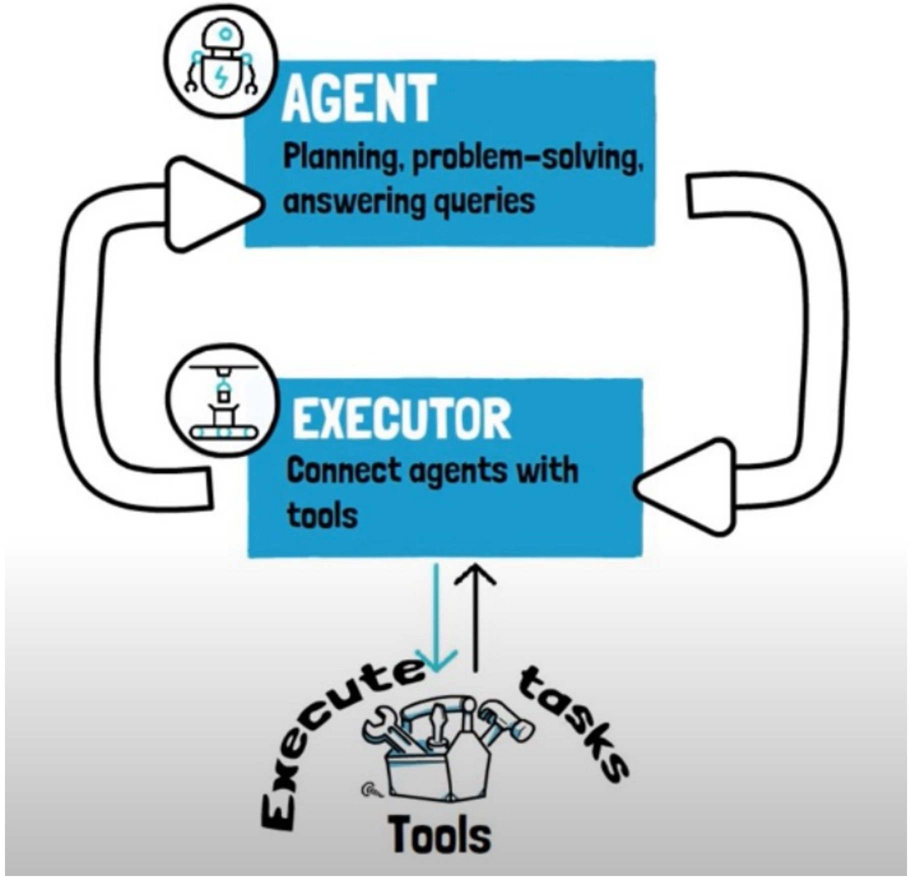
# ReAct: Reasoning and Acting

- ReAct: generate both
  - **Reasoning traces** (to update **internal** model belief)
  - **Actions** (to obtain **external** environment feedback)
- Reasoning guides acting, acting supports reasoning



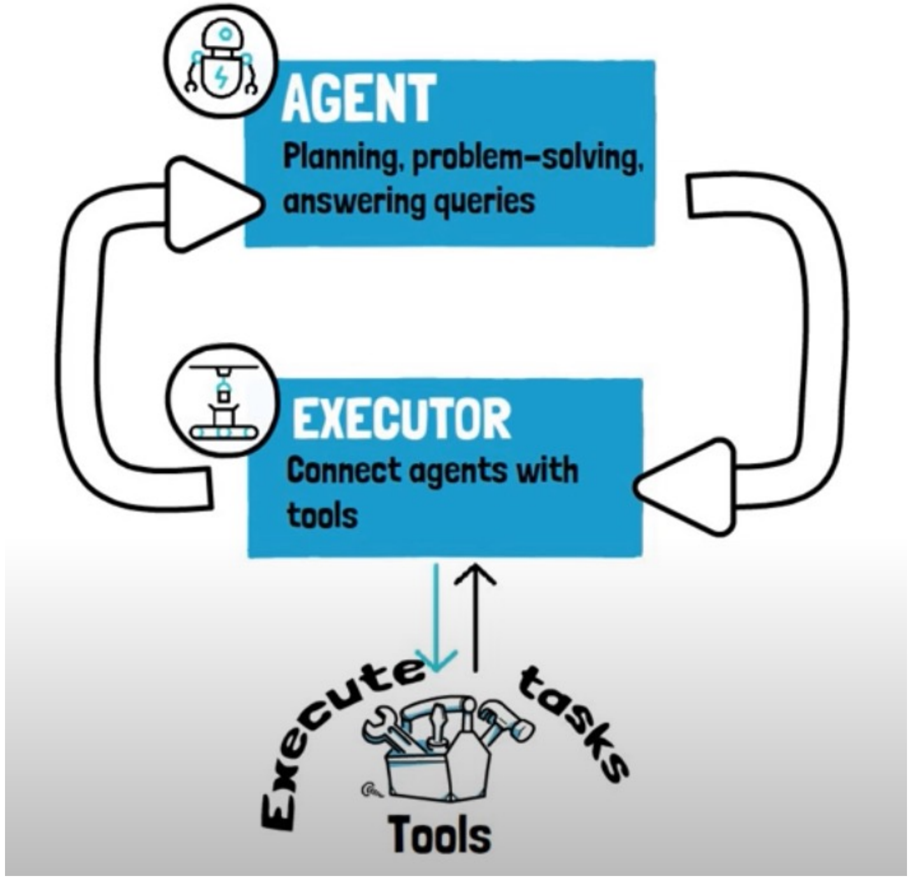
# ReAct: An Example of Order Assistant

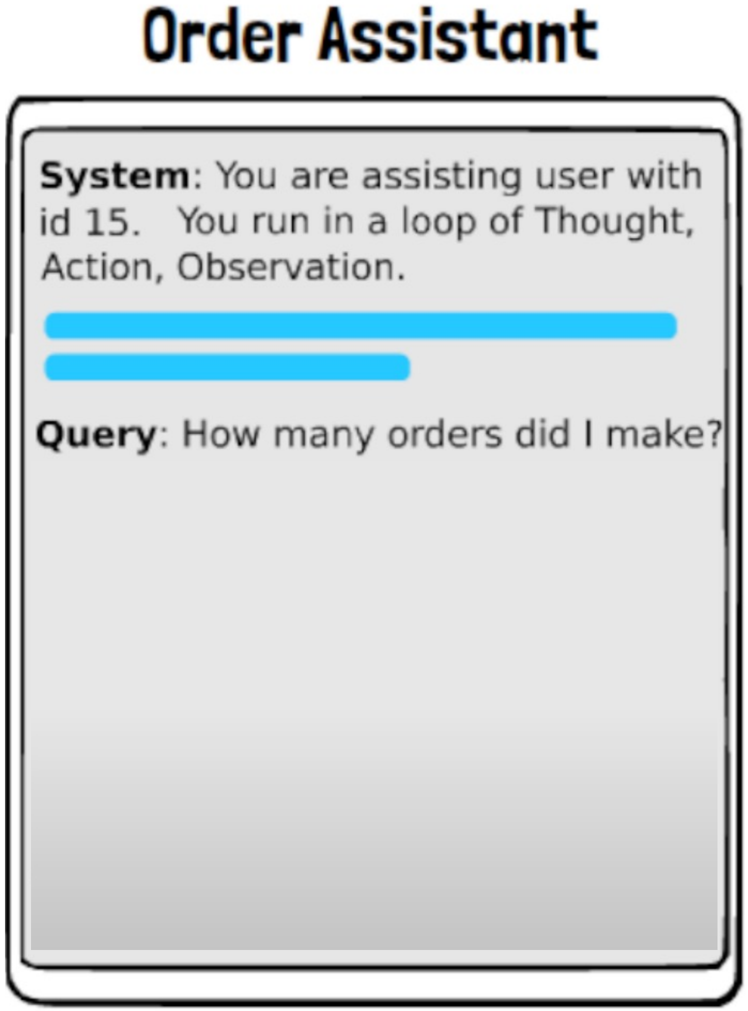
- Building blocks


AGENT
Planning, problem-solving, answering queries
EXECUTOR
Connect agents with tools
Execute tasks
Tools

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps


AGENT
Planning, problem-solving, answering queries
EXECUTOR
Connect agents with tools
Execute tasks
Tools

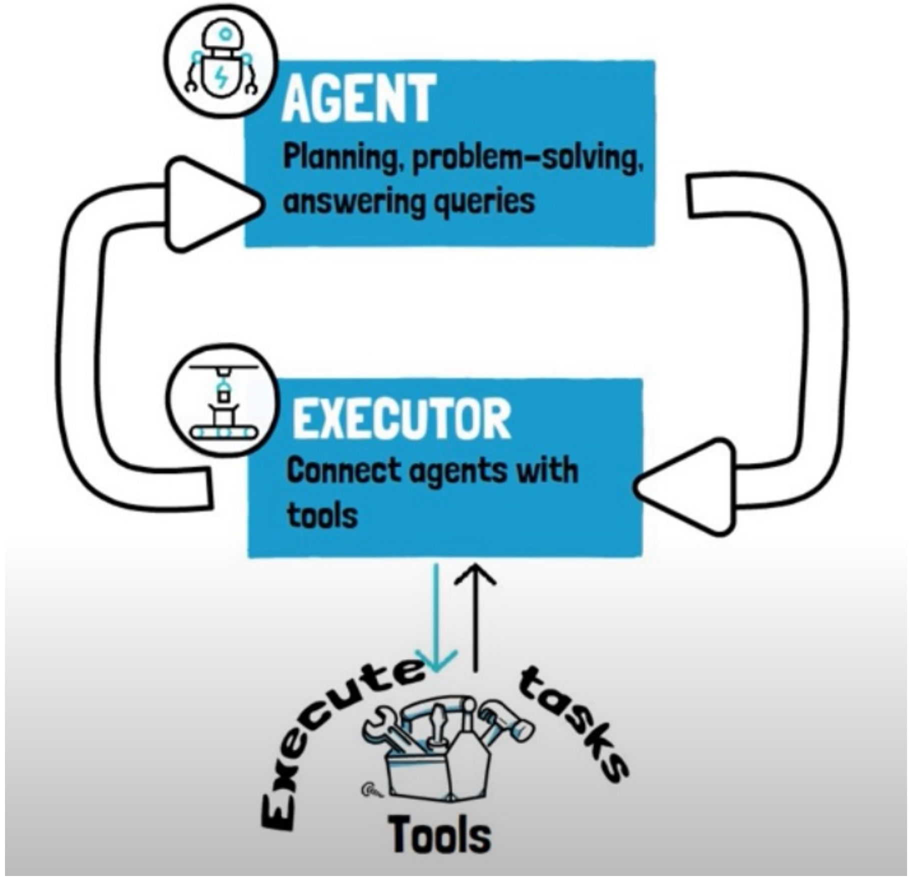


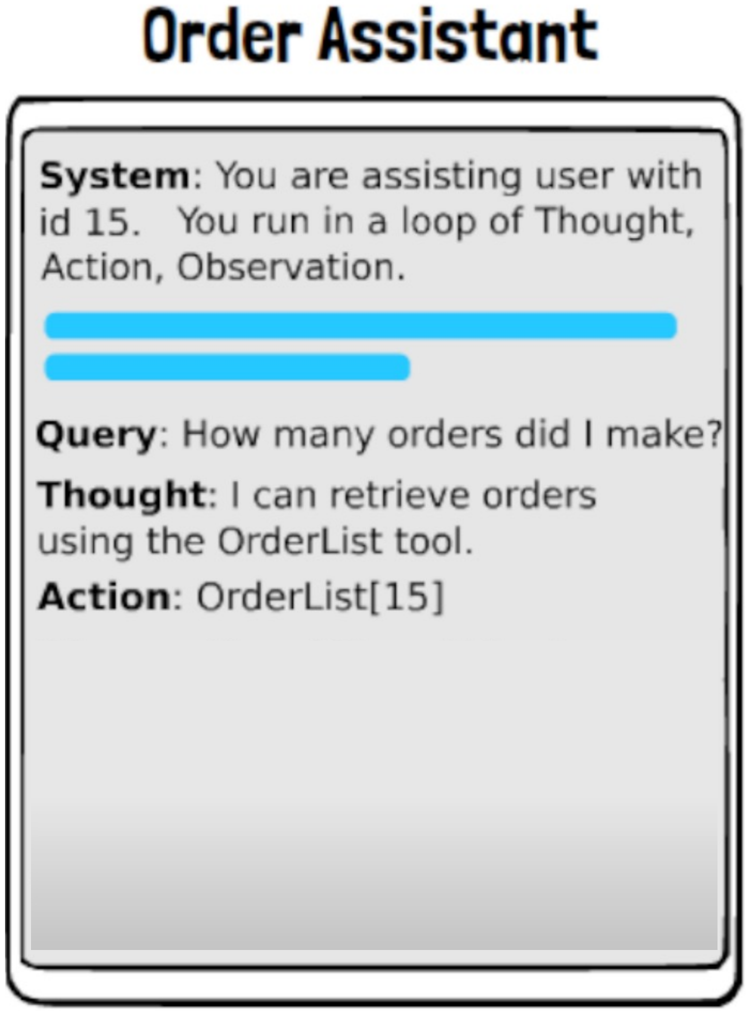
**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps


AGENT
Planning, problem-solving, answering queries
EXECUTOR
Connect agents with tools
Execute tasks
Tools



**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

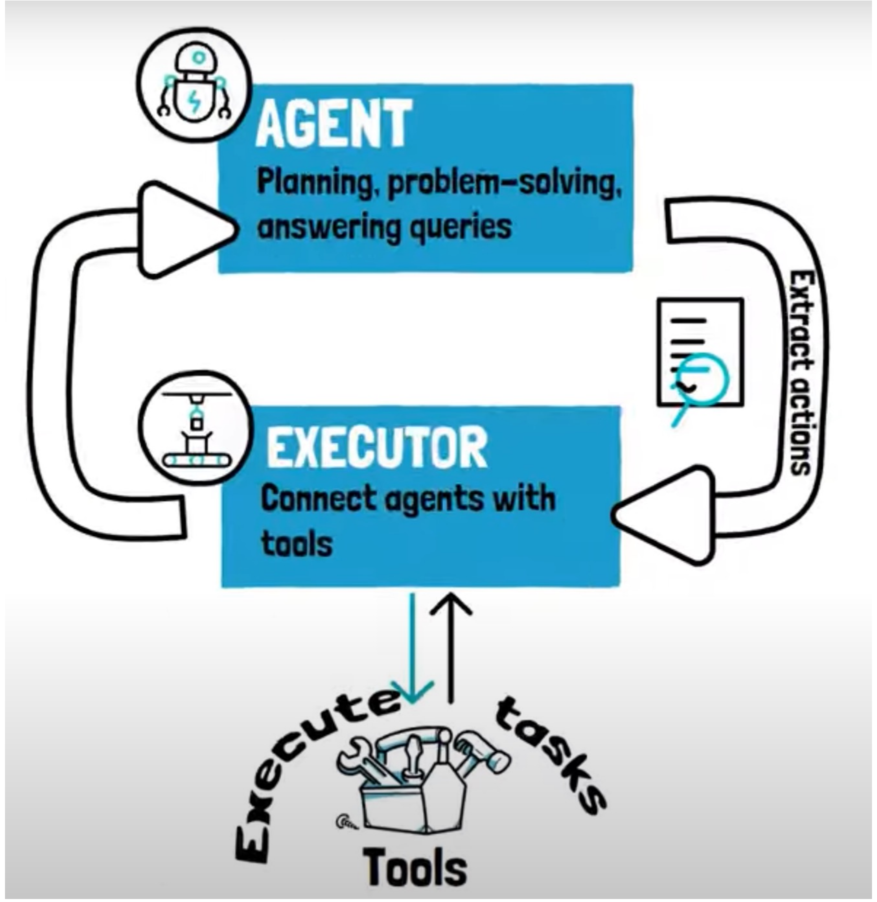
**Query**: How many orders did I make?

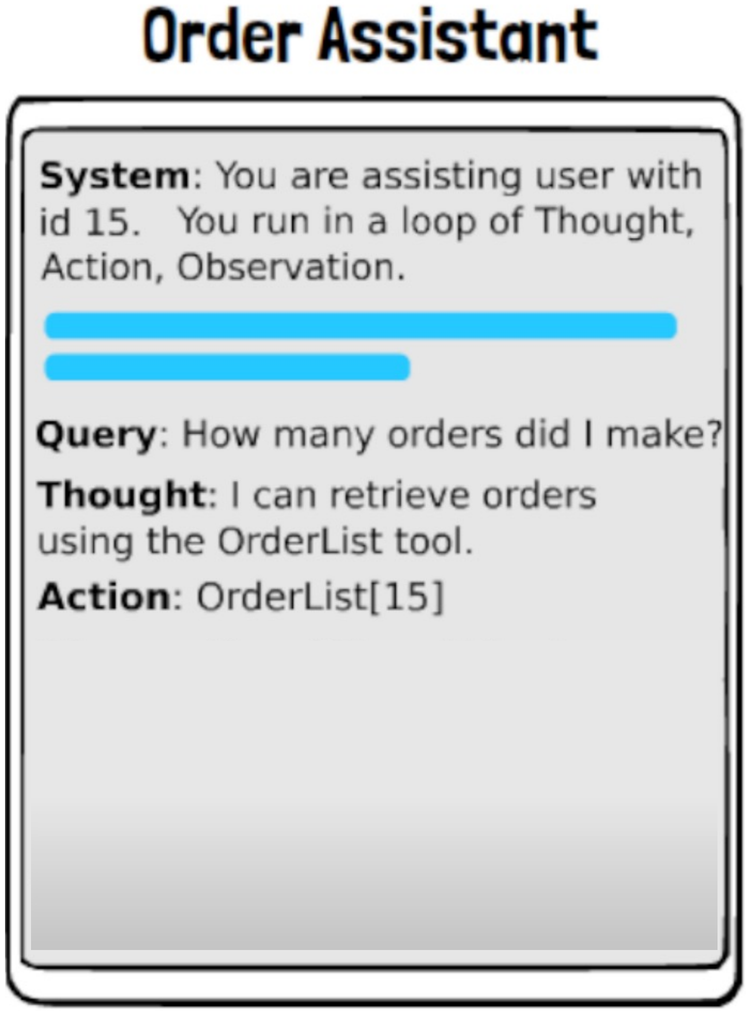
**Thought**: I can retrieve orders using the OrderList tool.

**Action**: OrderList[15]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps


AGENT
Planning, problem-solving, answering queries
Extract actions
EXECUTOR
Connect agents with tools
Execute tasks
Tools



**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

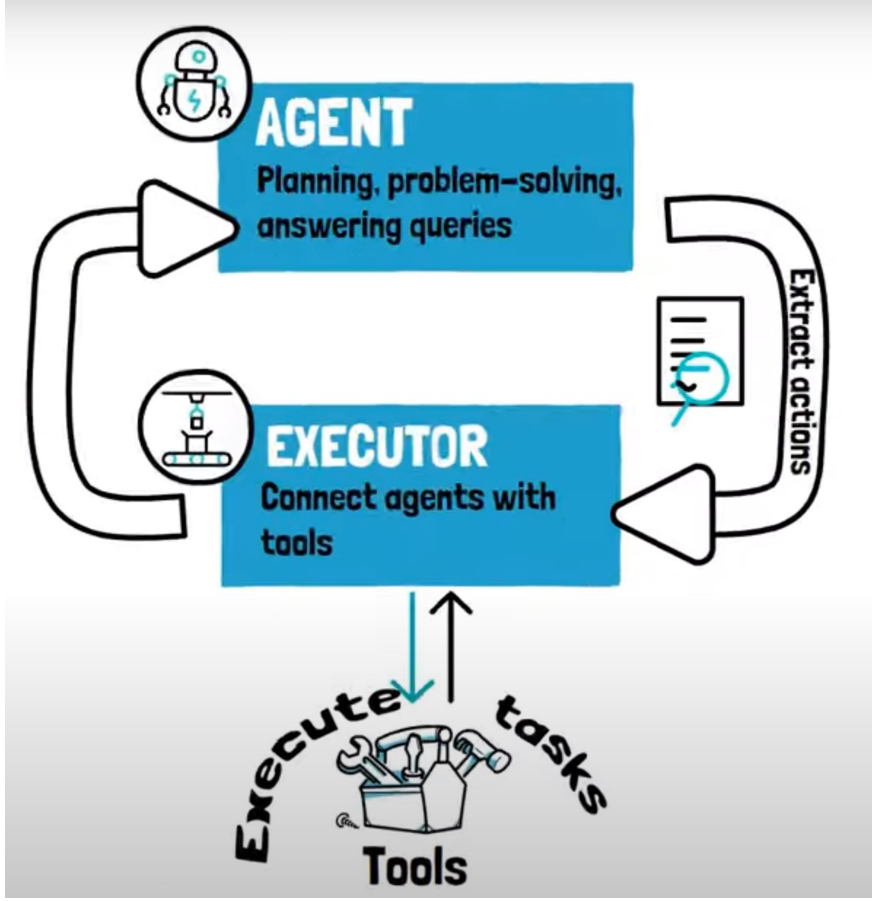
**Query**: How many orders did I make?

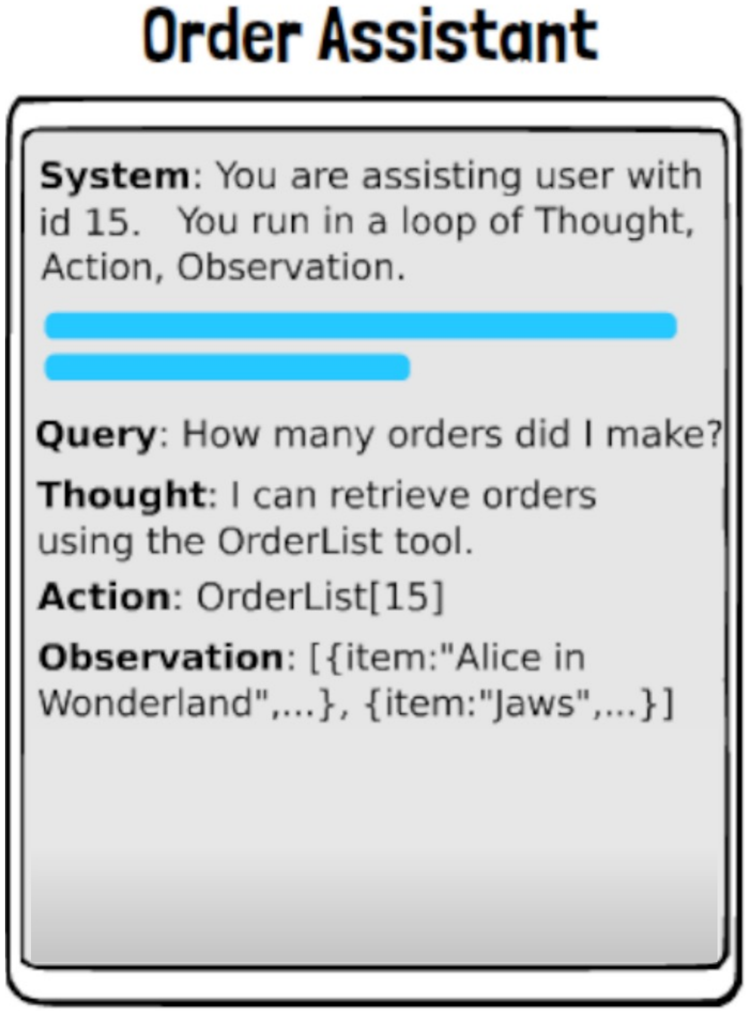
**Thought**: I can retrieve orders using the OrderList tool.

**Action**: OrderList[15]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps


AGENT
Planning, problem-solving, answering queries
Extract actions
EXECUTOR
Connect agents with tools
Execute tasks
Tools


Order Assistant

**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

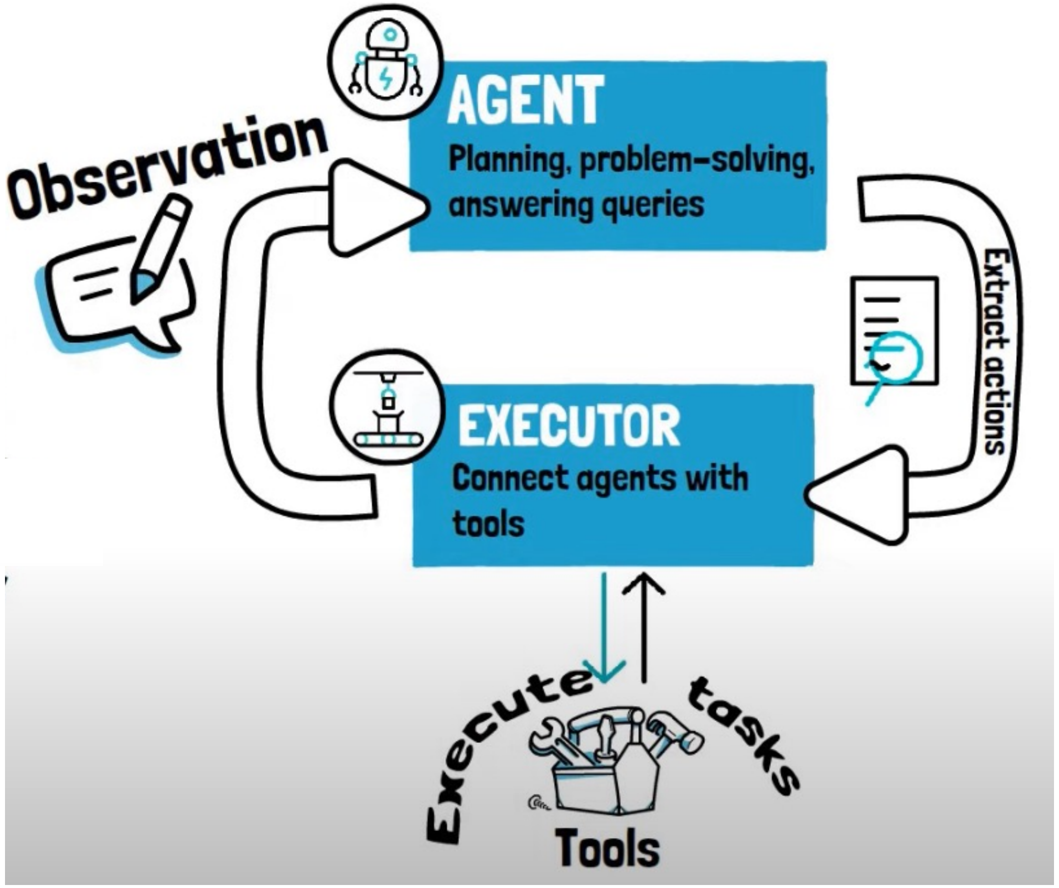
**Thought**: I can retrieve orders using the OrderList tool.

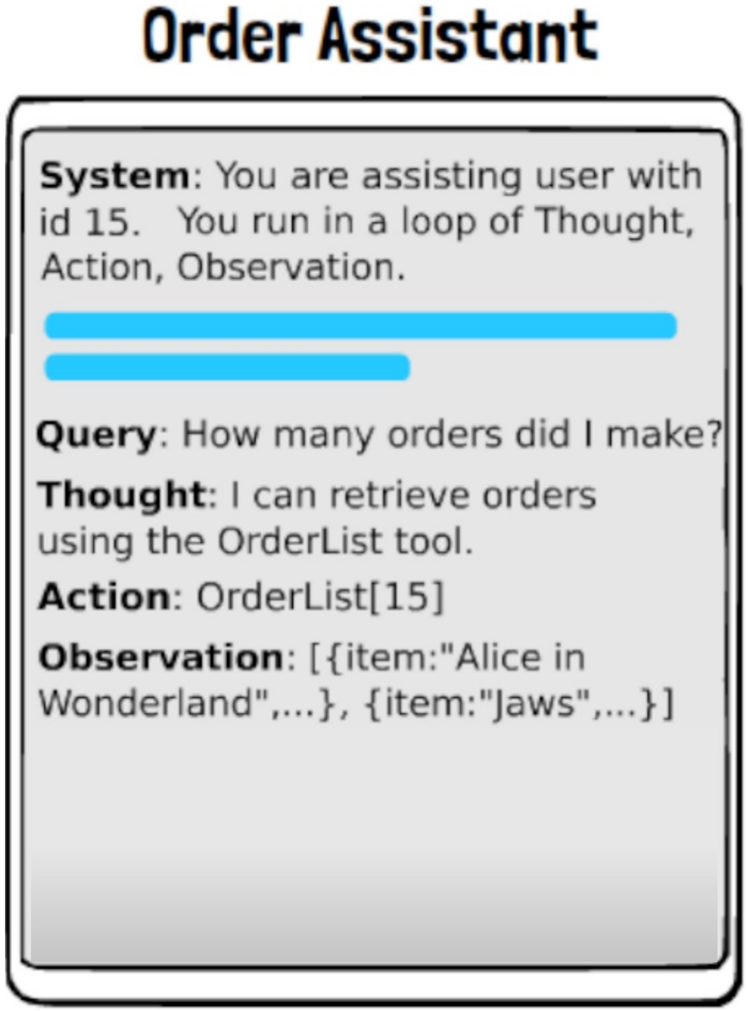
**Action**: OrderList[15]

**Observation**: [{item:"Alice in Wonderland",...}, {item:"Jaws",...}]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps


Observation
AGENT
Planning, problem-solving, answering queries
Extract actions
EXECUTOR
Connect agents with tools
Execute tasks
Tools


Order Assistant

**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

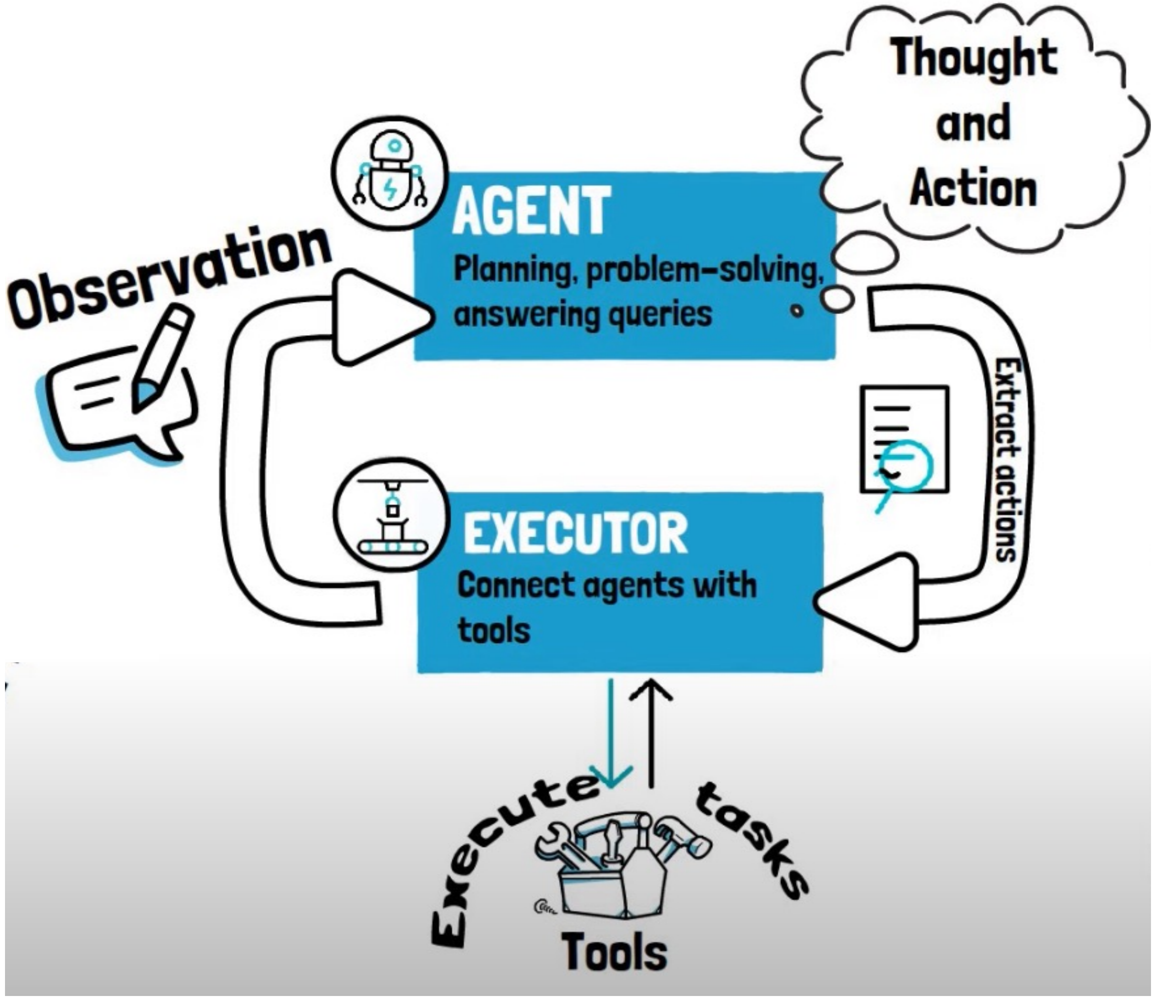
**Thought**: I can retrieve orders using the OrderList tool.

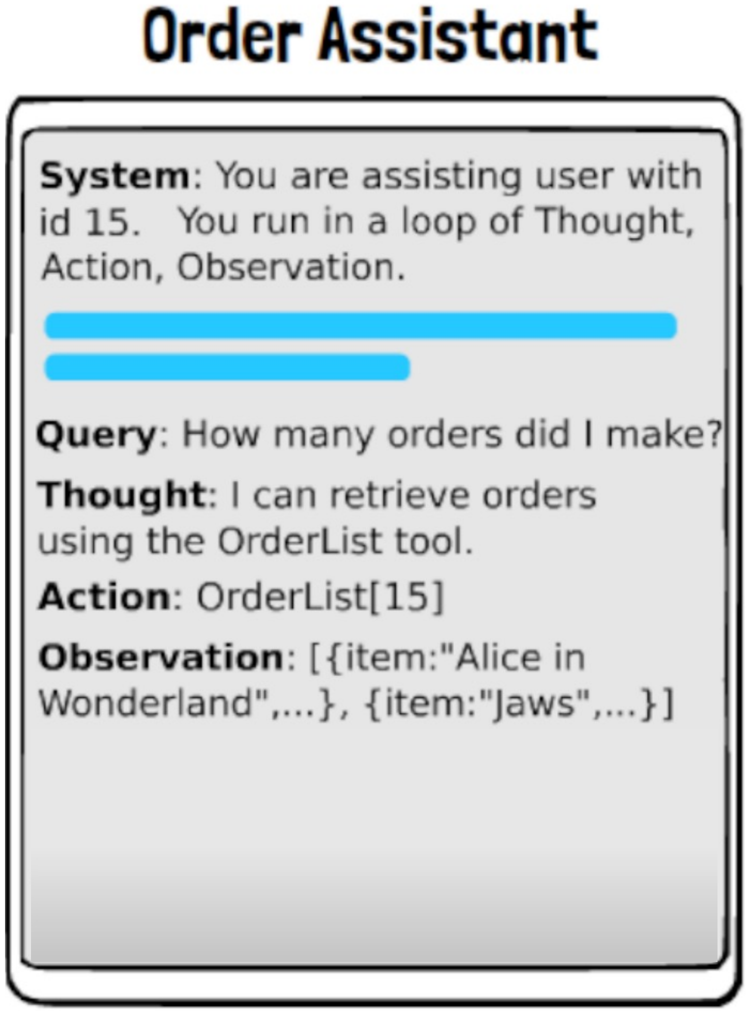
**Action**: OrderList[15]

**Observation**: [{item:"Alice in Wonderland",...}, {item:"Jaws",...}]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps




Order Assistant

**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

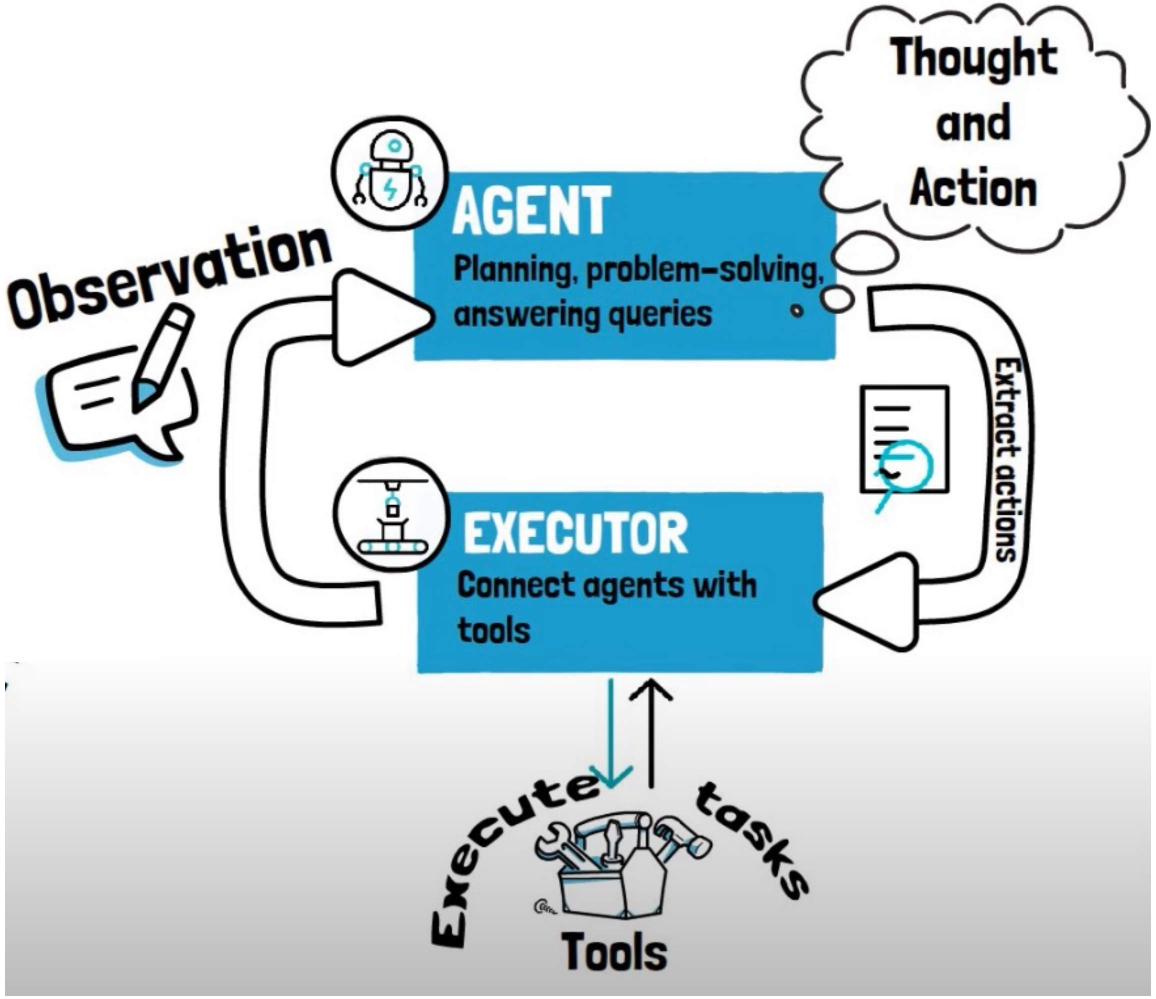
**Thought**: I can retrieve orders using the OrderList tool.

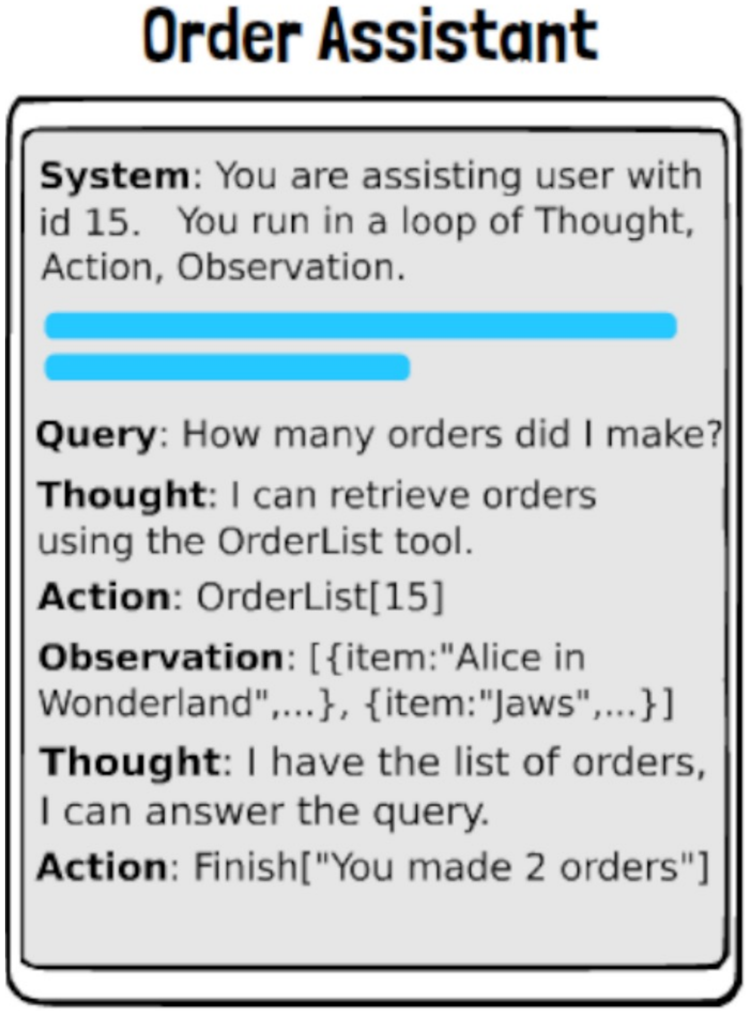
**Action**: OrderList[15]

**Observation**: [{item:"Alice in Wonderland",...}, {item:"Jaws",...}]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps





Order Assistant

**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

**Thought**: I can retrieve orders using the OrderList tool.

**Action**: OrderList[15]

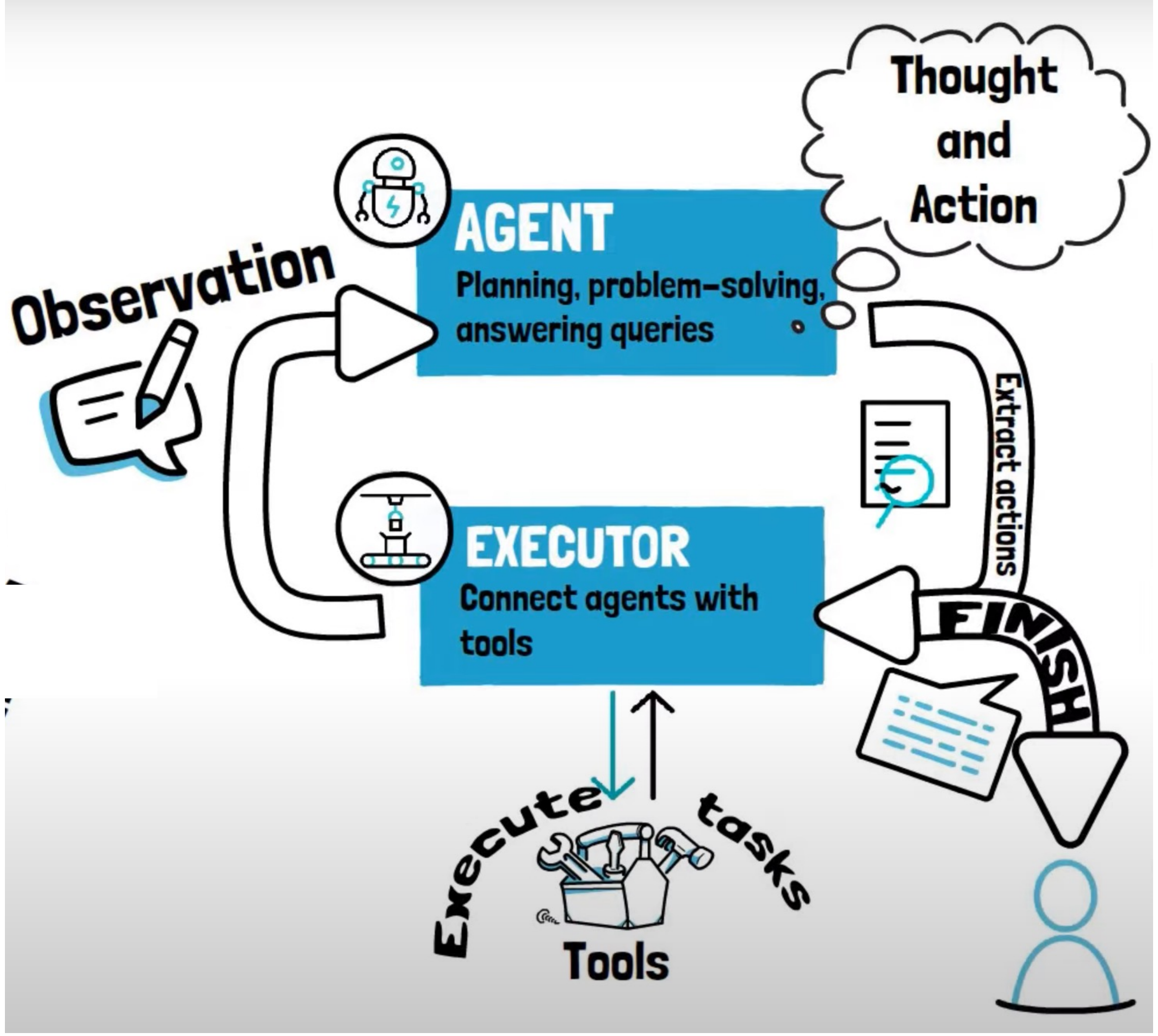
**Observation**: [{item:"Alice in Wonderland",...}, {item:"Jaws",...}]

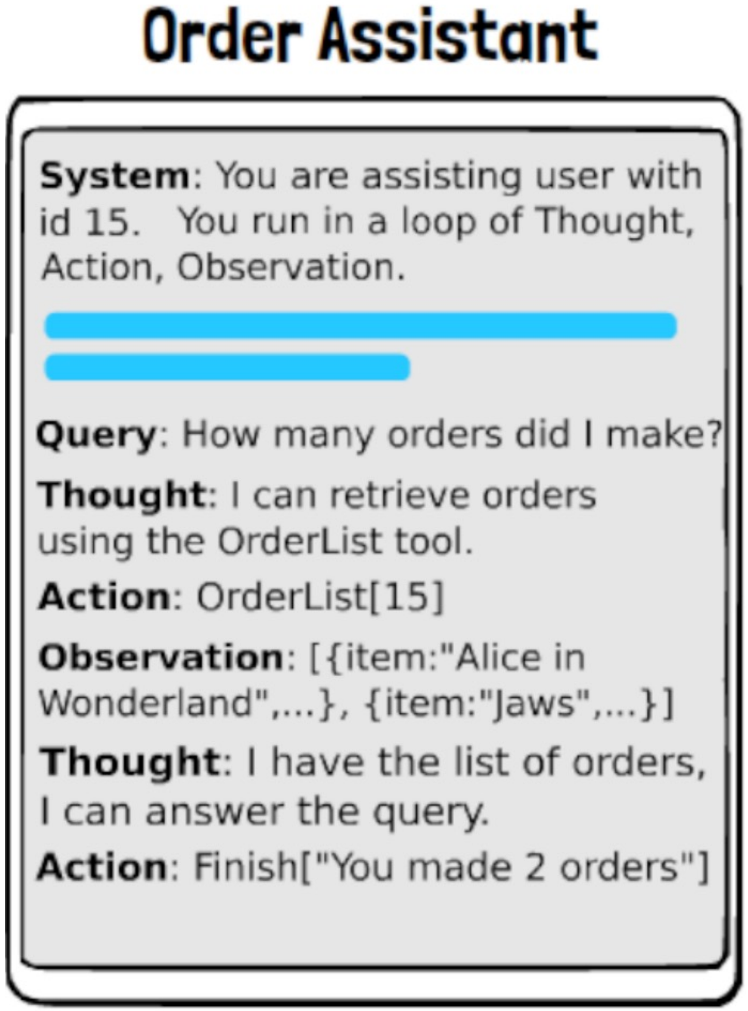
**Thought**: I have the list of orders, I can answer the query.

**Action**: Finish["You made 2 orders"]

# ReAct: An Example of Order Assistant

- Loop: thought-action-observation steps





**Order Assistant**

**System**: You are assisting user with id 15. You run in a loop of Thought, Action, Observation.

**Query**: How many orders did I make?

**Thought**: I can retrieve orders using the OrderList tool.

**Action**: OrderList[15]

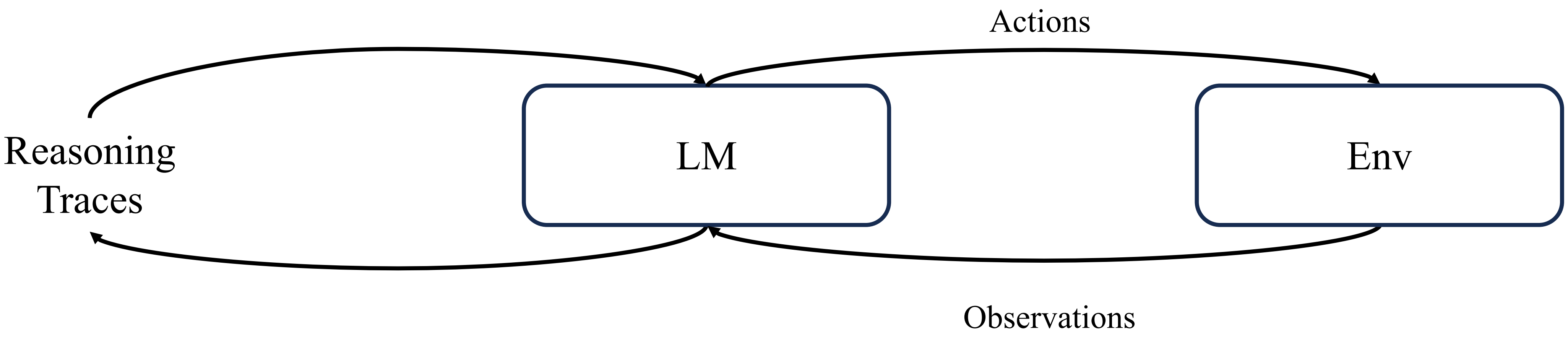
**Observation**: [{item:"Alice in Wonderland",...}, {item:"Jaws",...}]

**Thought**: I have the list of orders, I can answer the query.

**Action**: Finish["You made 2 orders"]

# ReAct: Key Challenges

- Learning a policy is challenging:
    - Mapping from context to current action is implicit.
    - Requiring extensive computation.



# ReAct: Key Methods

- Introducing language space into action space
  - Augmented action space = action space ∪ language space
  - One action ∈ action space → observation feedback
  - One action ∈ language space → a thought or a reasoning trace, no effect on the external environment, no observation feedback
    - Compose useful information by reasoning over the current context
    - Update the context to support future reasoning or acting

- Unlimited language space
  - A frozen large language model, PaLM-540B
    - Be prompted with few-shot in-context examples
    - Generate both domain-specific actions and free-form language thoughts

Reasoning Tasks

- Generate thoughts and actions alternately
- Include multiple thought-action-observation step

Decision-making Tasks

- Thoughts appear sparsely in the most relevant positions of a trajectory
- Decide the asynchronous occurrence of thoughts and actions for model itself

# ReAct: Experiments and Results

- **Tasks**: Question answering, Fact verification, Text game, Web interaction
- **Learning**: Prompting, fine-tuning
- **Model**: PaLM-540B, GPT-3

# ReAct: Experiments and Results – Reasoning Tasks

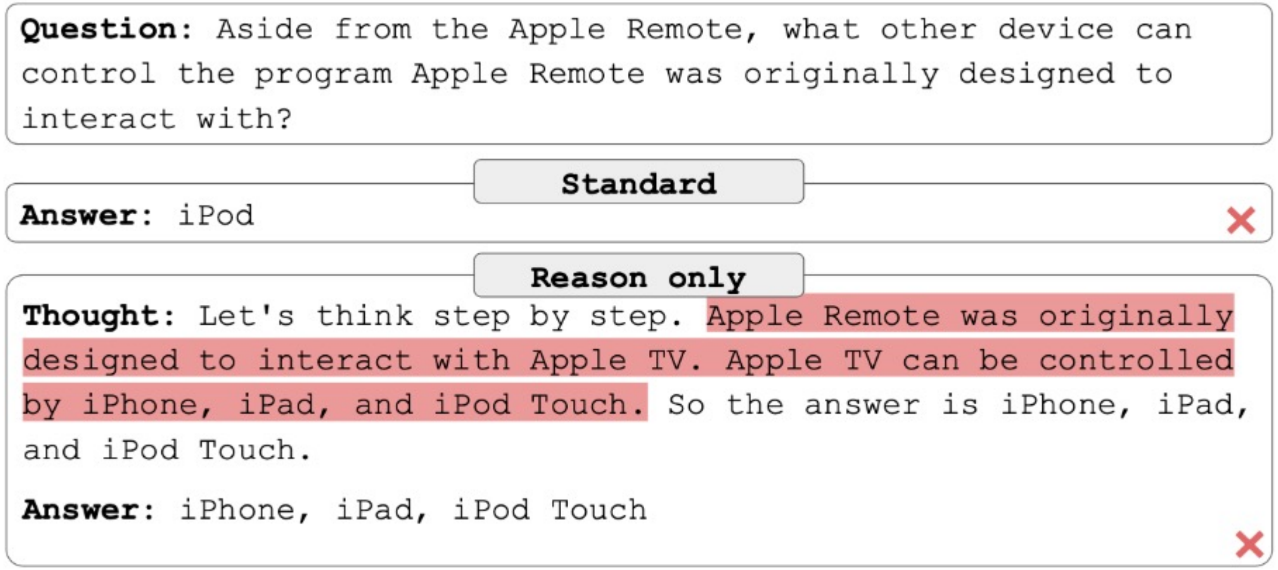
- **Tasks**: Question answering (HotPotQA), Fact verification (FEVER)
    - Open domain setup: interact with a Wiki API for knowledge reasoning
- **Action Space**: search[entity], lookup[string], finish[answer]

*PaLM-540B prompting results on HotpotQA and Fever.*

| | HotpotQA (EM) | | FEVER (accuracy) | |
|---|---|---|---|---|
| Standard | 28.7 | 6 samples | 57.1 | 3 samples |
| Reason-only (CoT) | 29.4 | | 56.3 | |
| Act-only | 25.7 | | 58.9 | |
| Best ReAct method | 35.1 | | 64.6 | |
| Supervised SoTA | 67.5 | 140k samples | 895 | 90k samples |

# ReAct: Experiments and Results – Reasoning Tasks

- An example of HotpotQA without acting
    - Prone to misinformation



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

**Standard**

**Answer:** iPod ✗

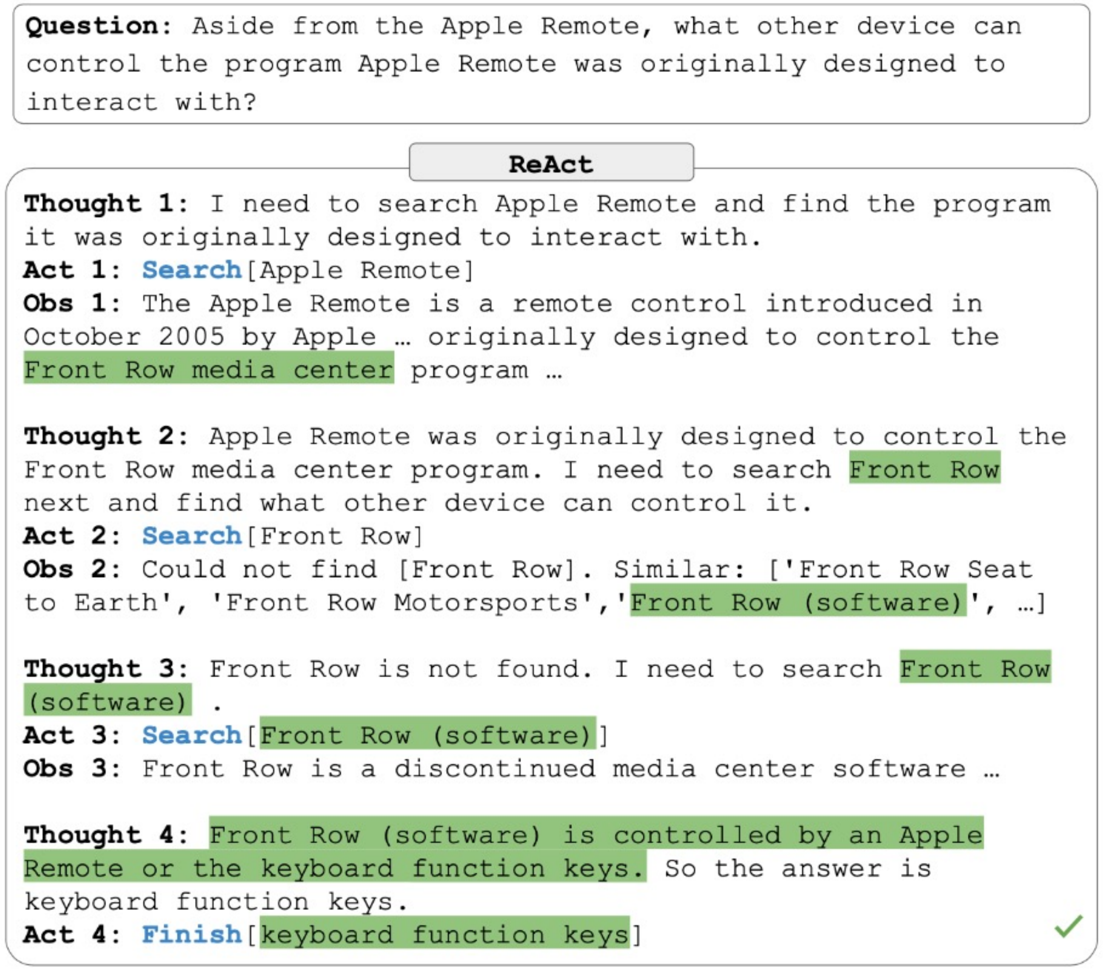
**Reason only**

**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 ✗

# ReAct: Experiments and Results – Reasoning Tasks

- An example of HotpotQA with reasoning and acting
  - Interpretable, factual, exception handling



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

**ReAct**

**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 …

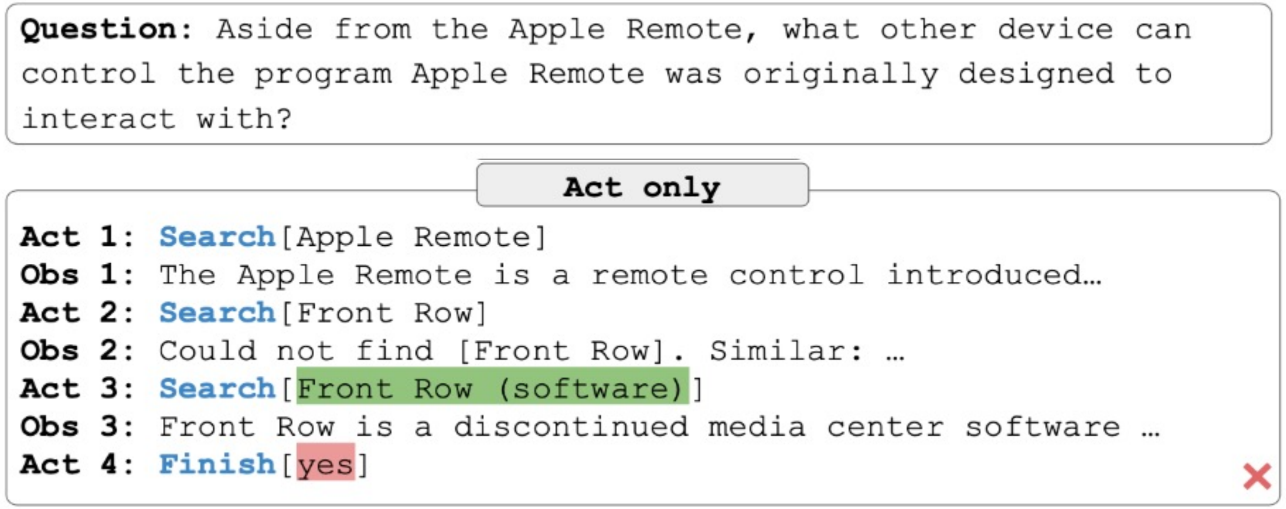
**Thought 2:** Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
**Act 2:** Search[Front Row]
**Obs 2:** Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports','Front 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] ✓

# ReAct: Experiments and Results – Reasoning Tasks

- An example of HotpotQA with only acting
    - Unable to synthesize final answer



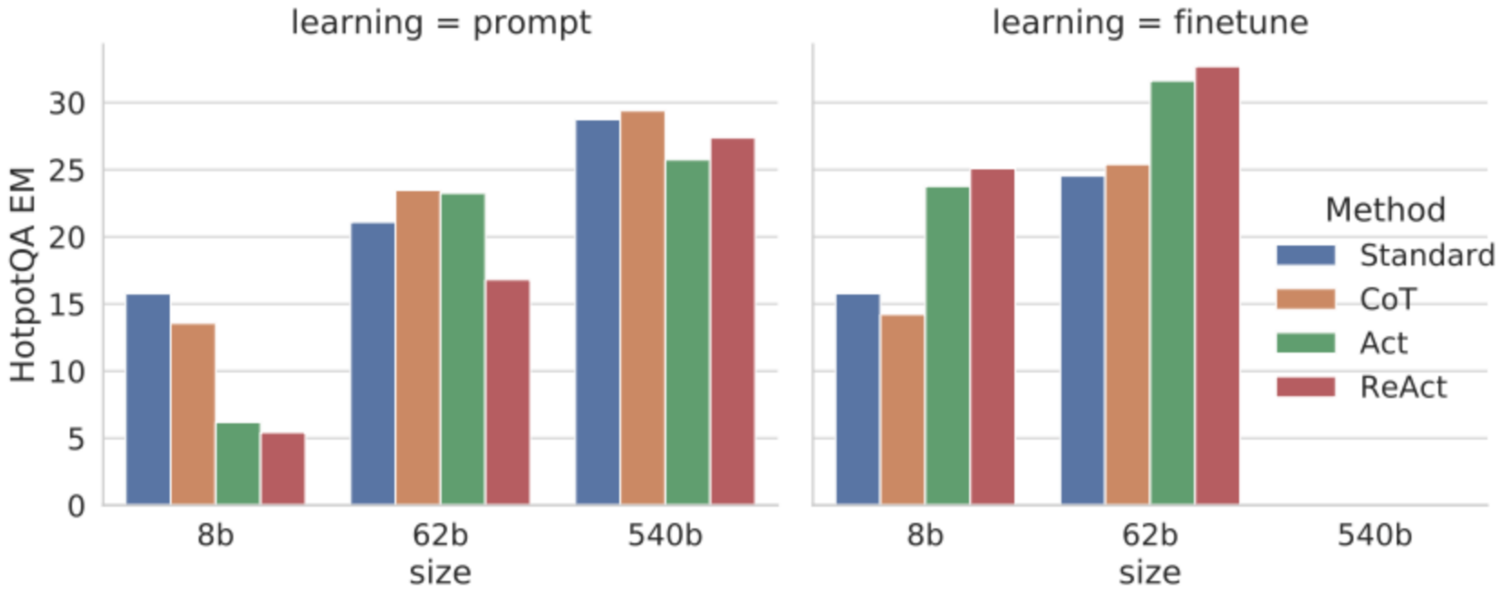
**Question:** Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

**Act only**

**Act 1:** Search[Apple Remote]
**Obs 1:** The Apple Remote is a remote control introduced…
**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] ✗

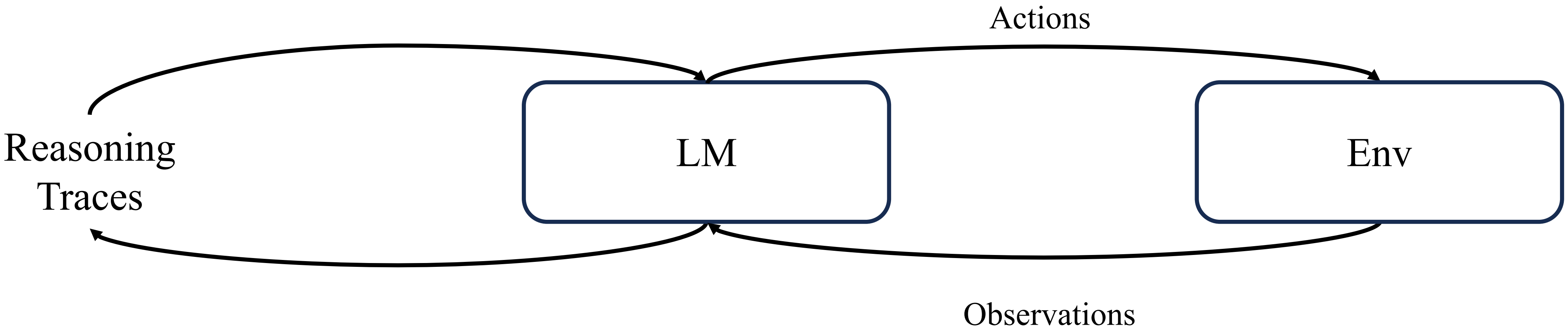
# ReAct: Experiments and Results

- **Learning**: Prompting → Fine-tuning
- Prompting: only works with LLMs, limited learning support
- Fine-tuning: promising
  - ReAct finetuned small LMs > ReAct prompted large LMs
  - ReAct finetuning is better than other formats across model sizes



# ReAct: Summary

- **ReAct**: Let LMs generate both reasoning traces and actions
- **Synergy**: Reasoning guides acting, acting supports reasoning
- Benefits: flexibility, generalization, alignment



# Language Models as Agents

I. Reasoning and Acting

II. Use of External Tools

# Introduction to Tools

What other name is Pittsburgh known by?

↓

↓

The Steel City

3435*235/9

↓

↓

89691.67

∅

↓

↓

Tuesday,
March 5, 2024

# Problems in Large Language Models

*Who is the current President of the United States?*

As of my last update in January 2022, the current president of the United States is Joe Biden. However, please note that my information might be outdated, so it's a good idea to verify with the latest sources for the most current information.

*What is the result of 3435 * 235 / 9 ?*

The result of the expression 3435×235/9 is approximately 88208.33388208.333. ✗

*What day of the week is it today?*

Today is Friday. ✗

# Previous Works

- Human annotation
  - Change something → expensive in time and cost
  - Pretty limited
  - *e.g.,* LaMDA, WebGPT, Internet-Augmented Dialogue Generation

- Prompting
  - More flexible & less expensive
  - Particular to a specific domain/tool
  - *e.g.,* ReAct, PAL

# Toolformer

- Toolformer[2]
  - No human annotation
  - Automatic
  - Inexpensive
  - Self-supervised
  - Prompting in one of the steps

*[2] Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., ... & Scialom, T. "Toolformer: Language models can teach themselves to use tools." in Proceedings of NeurIPS, 2024*

# Response in Large Language Models

*Who is the current President of the United States?*

As of my last update in January 2022, the current president of the United States is Joe Biden. However, please note that my information might be outdated, so it's a good idea to verify with the latest sources for the most current information.

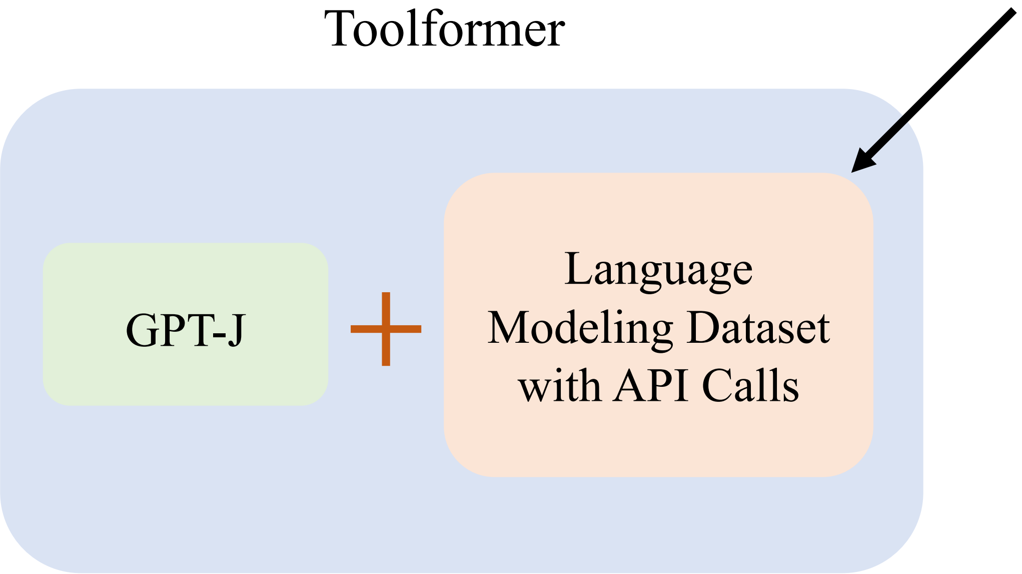
# Training Toolformer

- Train a model to learn:
  - Which tool to use
  - When to use it
  - How to use it

*The result of 3435 * 235 / 9 is* _____

The result of 3435 * 235 / 9 is <u>[Calculator (3435 * 235 / 9) → 89691.66 ]</u>

# Training Toolformer



Toolformer

GPT-J

+

Language Modeling Dataset with API Calls

*Need: New Dataset with API Calls*

# Creating the Training Dataset

Language Modeling Dataset

Pittsburgh is also known as the Steel City.

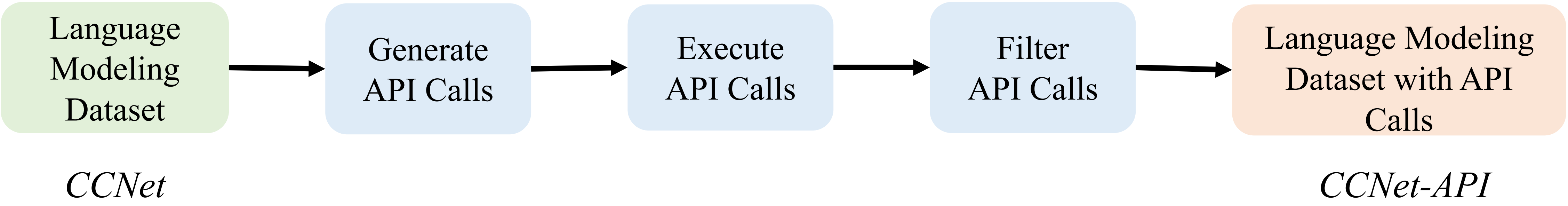
*CCNet*

Language Modeling Dataset with API Calls

Pittsburgh is also known as **[QA(What other name is Pittsburgh known by? → the Steel City)]** the Steel City.

*CCNet-API*

# Stages of Creating the Training Dataset



Language Modeling Dataset

Generate API Calls

Execute API Calls

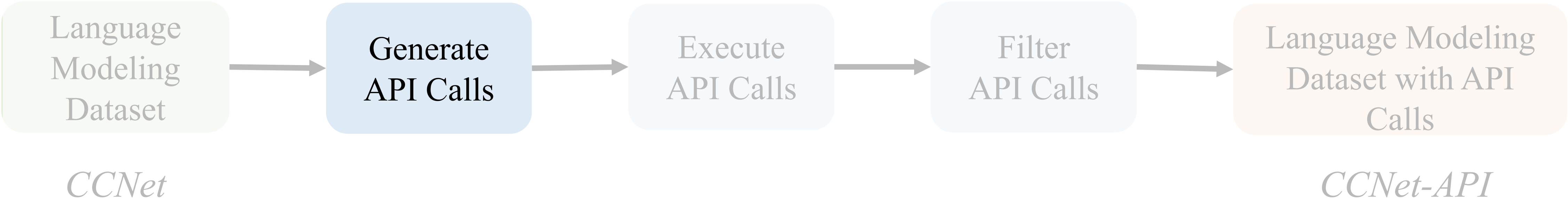
Filter API Calls

Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

# Generate API Calls

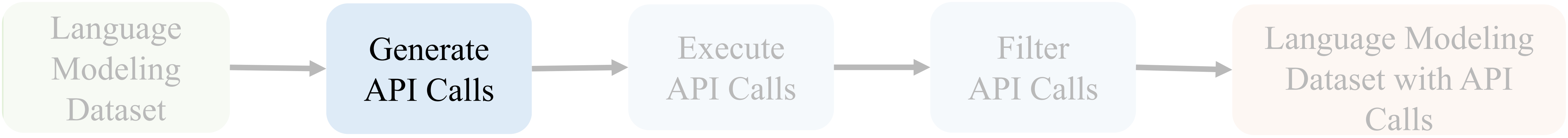

Language Modeling Dataset
Generate API Calls
Execute API Calls
Filter API Calls
Language Modeling Dataset with API Calls
CCNet
CCNet-API

*Prompting the Model to Generate API Calls*

**Input:** Joe Biden was born in Scranton, Pennsylvania.

**Output:** Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

# Generate API Calls



Language Modeling Dataset → Generate API Calls → Execute API Calls → Filter API Calls → Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

*You task is to […]*

**Input:** Pittsburgh is also known as the Steel City.

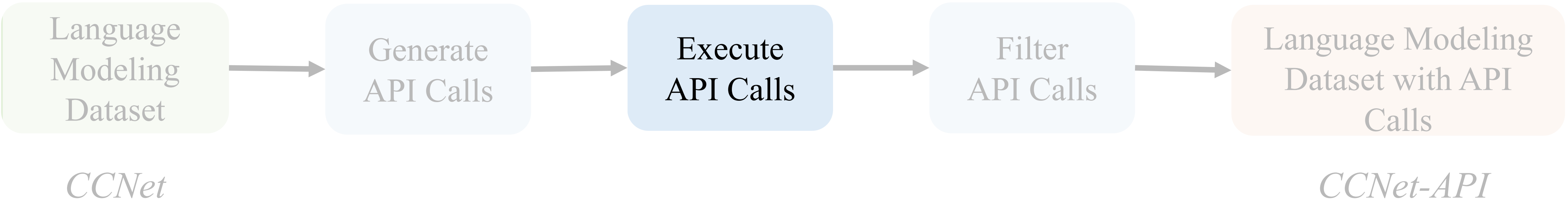
**Output:**

Pittsburgh is also known as [QA("In which state is Pittsburgh?")] the Steel City.

Pittsburgh is also known as [QA("What other name is Pittsburgh known by?")] the Steel City.

Pittsburgh is also known as [QA("What is the second city in Pennsylvania?")] the Steel City.

# Execute API Calls



Language Modeling Dataset → Generate API Calls → **Execute API Calls** → Filter API Calls → Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

Pittsburgh is also known as [QA("In which state is Pittsburgh?")] the Steel City.

In which state is Pittsburgh? → ? → Pennsylvania

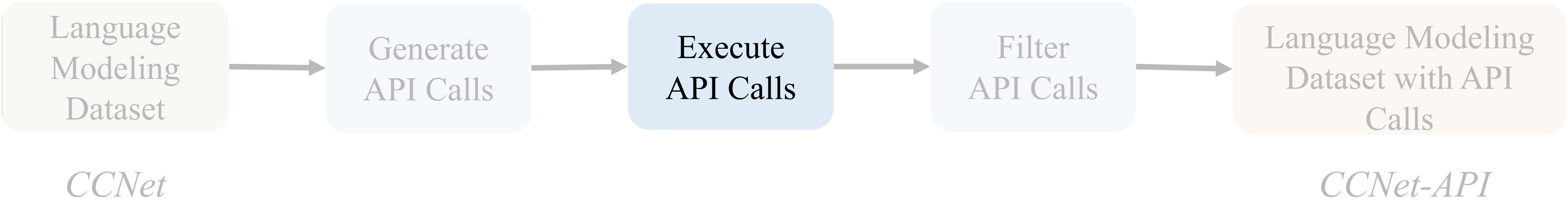
Pittsburgh is also known as [QA("What other name is Pittsburgh known by?")] the Steel City.

What other name is Pittsburgh known by? → ? → The Steel City

Pittsburgh is also known as [QA("What is the second city in Pennsylvania?")] the Steel City.

What is the second city in Pennsylvania? → ? → Pittsburgh

# Execute API Calls



Language Modeling Dataset

Generate API Calls

Execute API Calls

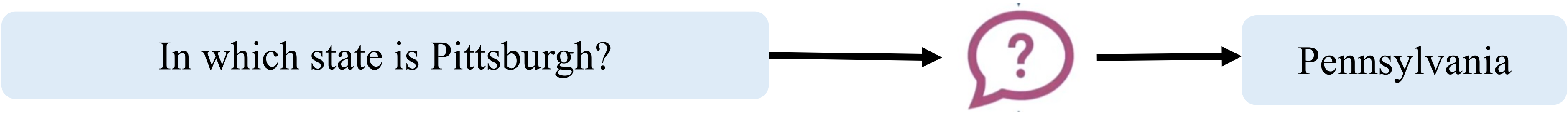
Filter API Calls

Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

Pittsburgh is also known as [QA("In which state is Pittsburgh?")] the Steel City.

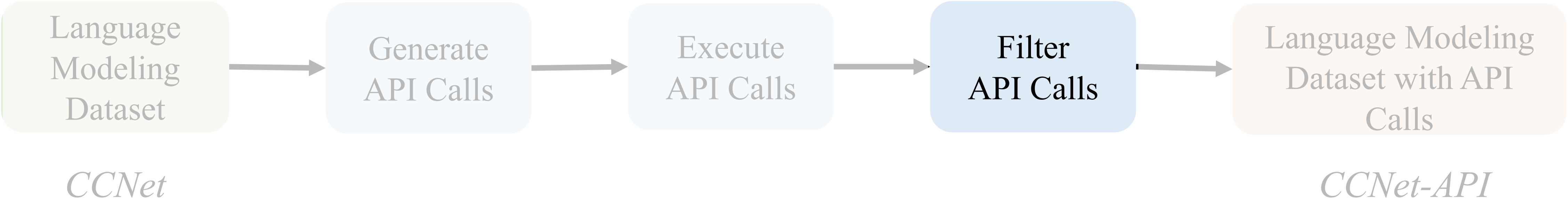


In which state is Pittsburgh?

Pennsylvania

Pittsburgh is also known as [QA("In which state is Pittsburgh?”) → Pennsylvania] the Steel City.

# Filter API Calls: Model-based Perplexity



Language Modeling Dataset

Generate API Calls

Execute API Calls

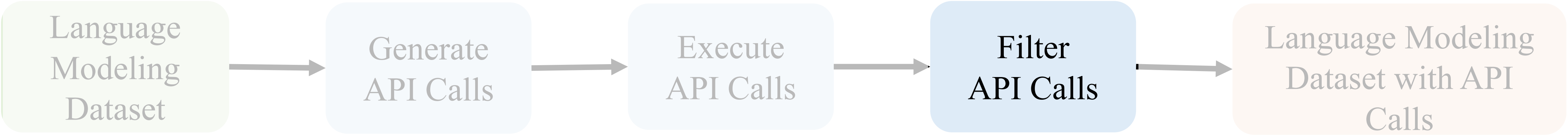
Filter API Calls

Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

# Filter API Calls: Model-based Perplexity


Language Modeling Dataset
Generate API Calls
Execute API Calls
Filter API Calls
Language Modeling Dataset with API Calls

*CCNet*

$L_\bullet(\text{PREFIX}) = -\log p(\text{the Steel City.} \mid \text{PREFIX})$

*CCNet-API*

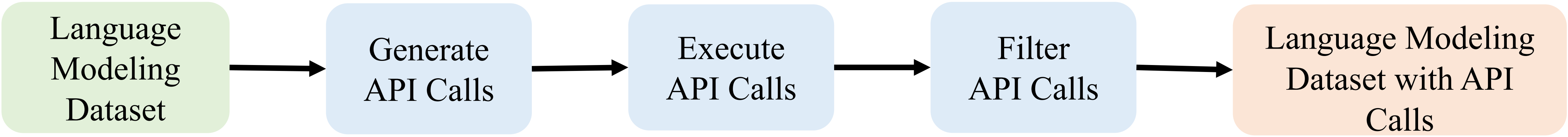
A. No API Call: $L_A$(Pittsburgh is also known as) = 2.5

B. Non-executed API Call: $L_B$(Pittsburgh is also known as [QA("What other name is Pittsburgh known by?") → ? ]) = 2.1

C. Executed API Call: $L_B$(Pittsburgh is also known as [QA("What other name is Pittsburgh known by?") → The Steel City]) = 0.8

$$\text{Usefulness} = \min(L_A, L_B) - L_C = \min(2.5, 2.1) - 0.8 = 1.3$$

# Creating the Training Dataset



Language Modeling Dataset

Generate API Calls

Execute API Calls

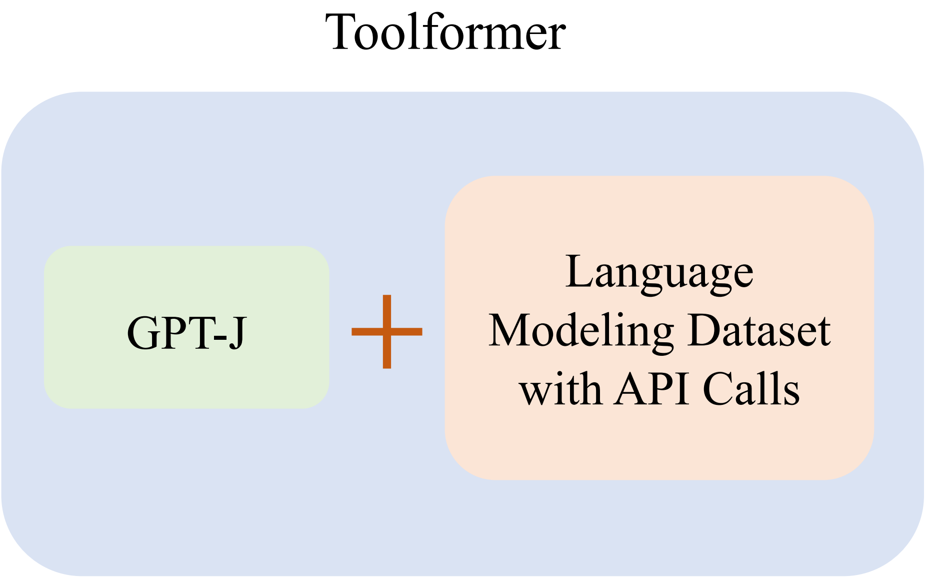
Filter API Calls

Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

# Finetuning Toolformer


Toolformer

GPT-J

+

Language Modeling Dataset with API Calls

# Toolformer Tools

Question Answering

Machine Translation System

Calculator

Wikipedia Search

Calendar

# Toolformer: Experiments and Results

**Datasets**:

- Fact Completion
    - "The theory of relativity was developed by _____"
- Question Answering
    - "In Greek Mythology, who is the goddess of spring growth?"
- Math Computations
- Multilingual Questions
    - Context is given in English, question is multilingual.
- Temporal Questions

# Toolformer: Experiments and Results

**Models to Compare**:

- GPT-J (6B)
- GPT-J + CC
- Toolformer
- Toolformer (disabled)
- GPT-3 (175B)

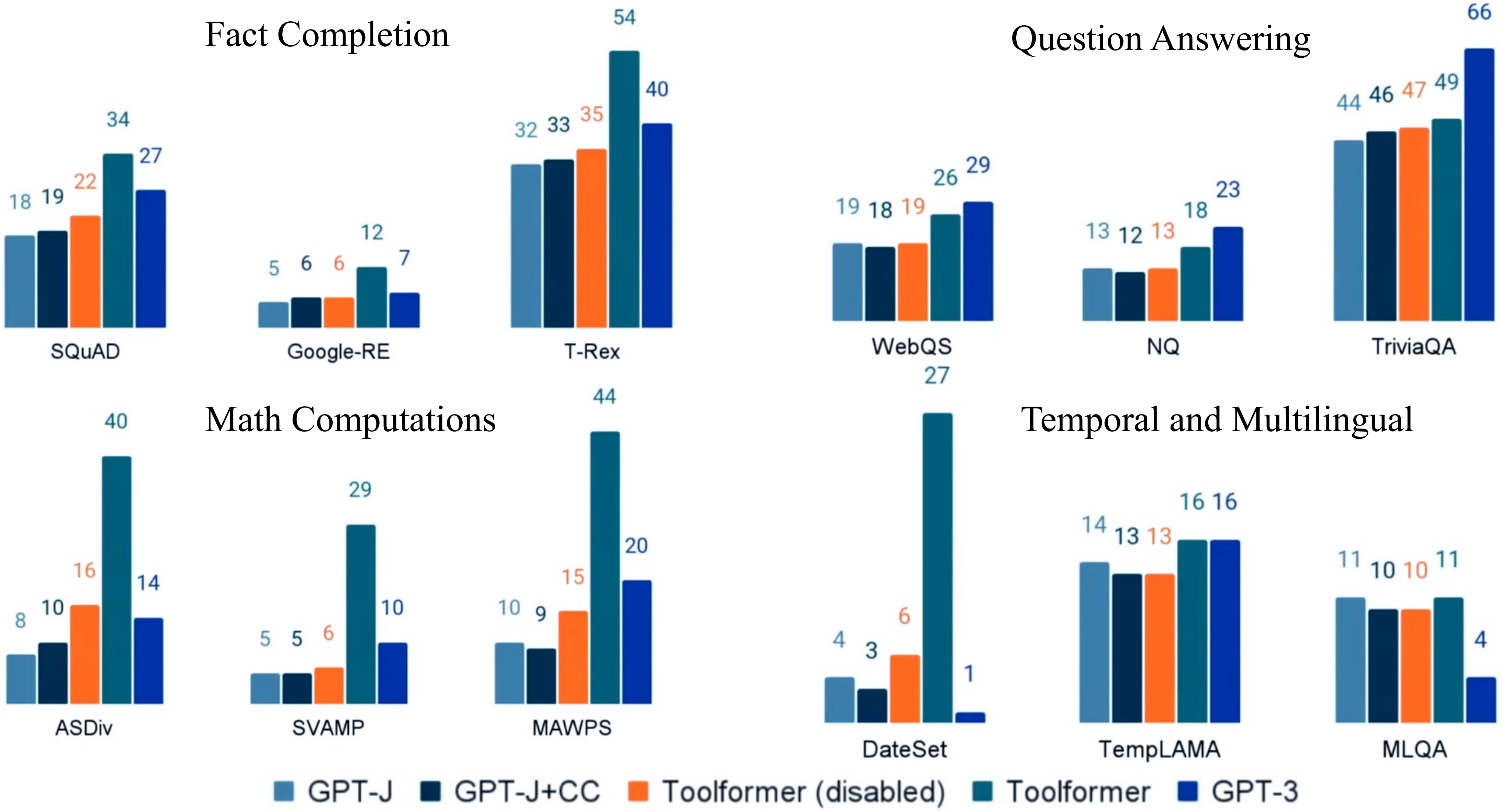
GPT-J + CC

GPT-J + Language Modeling Dataset

Toolformer

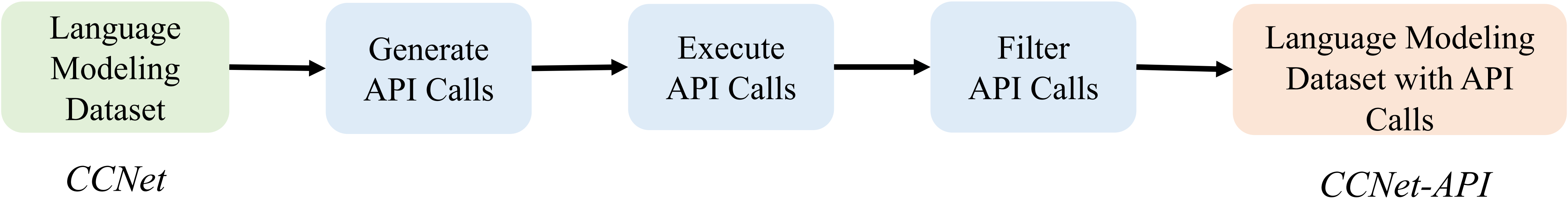
GPT-J + Language Modeling Dataset with API Calls

# Toolformer: Results in Different Datasets



# Summary

- Trained Toolformer by creating a new dataset embedded with API calls



Language Modeling Dataset

Generate API Calls

Execute API Calls

Filter API Calls

Language Modeling Dataset with API Calls

*CCNet*

*CCNet-API*

- Toolformer outperforms without losing core language modeling capabilities

# Thank you

Any Question?