
# LLM-based Tool Learning

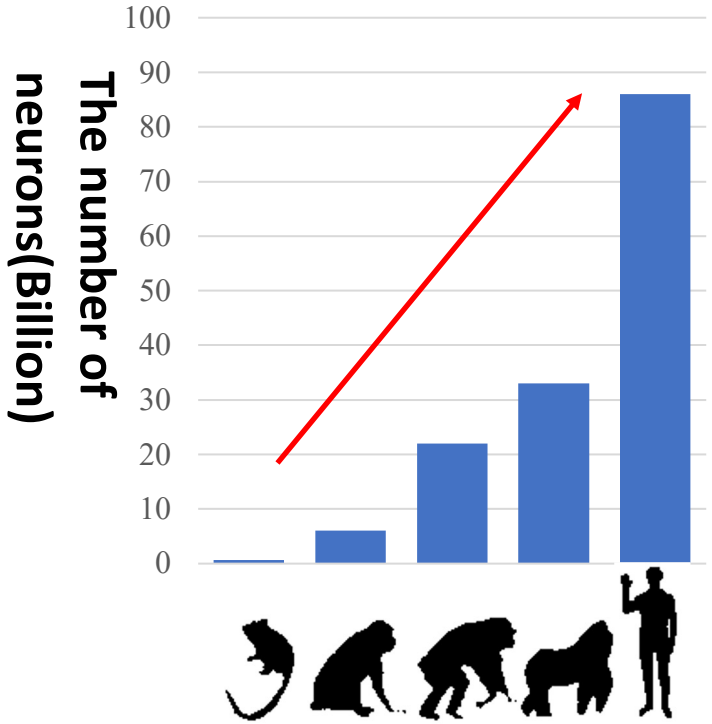
Yankai Lin

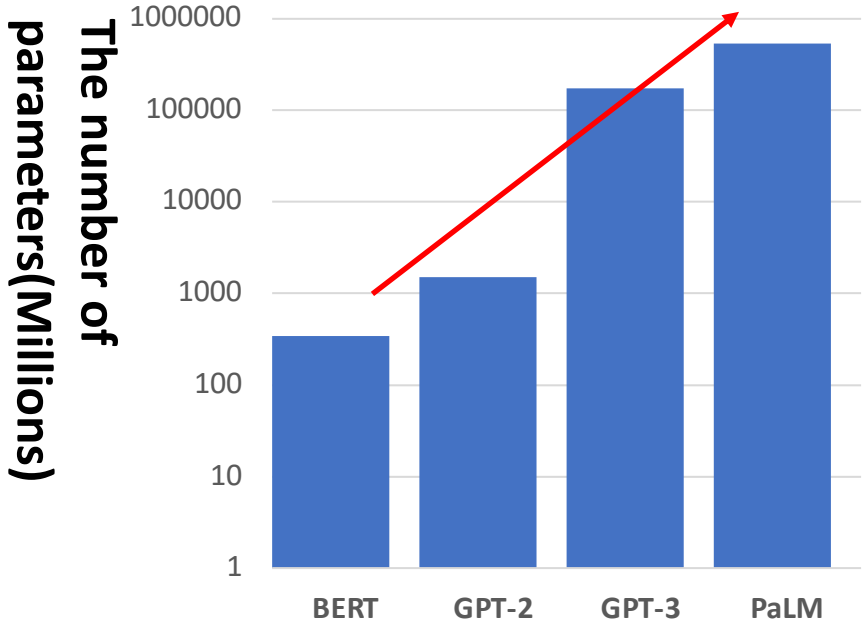
yankailin@ruc.edu.cn

THUNLP

# Individual Intelligence Emergence

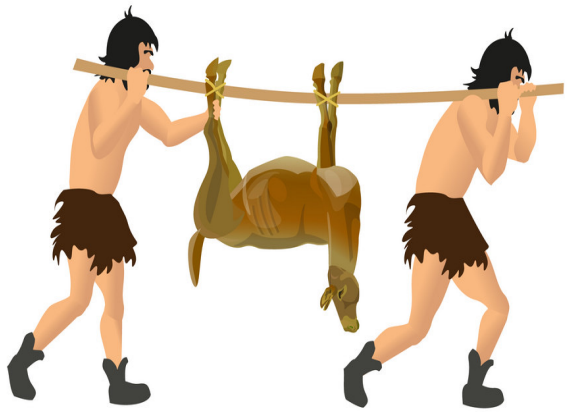
- Increasing the number of neurons leads to **the emergence of intelligence in biological individuals**
- Increasing the number of parameters leads to **the emergence of intelligence in large models**



# Human Intelligence and Artificial Intelligence

- Guess: Artificial intelligence is likely to follow the same developmental path as human intelligence

| | | | | |
|---|---|---|---|---|
| **Development** |  |  |  |  |
| **Human Intelligence** | **Small brain capacity** | **Big brain capacity** | **Tool Use** | **Collaborative labor** |
| **Arttificial Intelligence** | **Small model** | **Big model** | **Autonomous Agents** | **Multi-Agents** |

# Tool Intelligence

- Tools extends human capabilities in productivity, efficiency, and problem-solving
- Humans have been the **primary agents** in tool use throughout history
- Question: can **artificial intelligence** be as capable as humans in tool use?

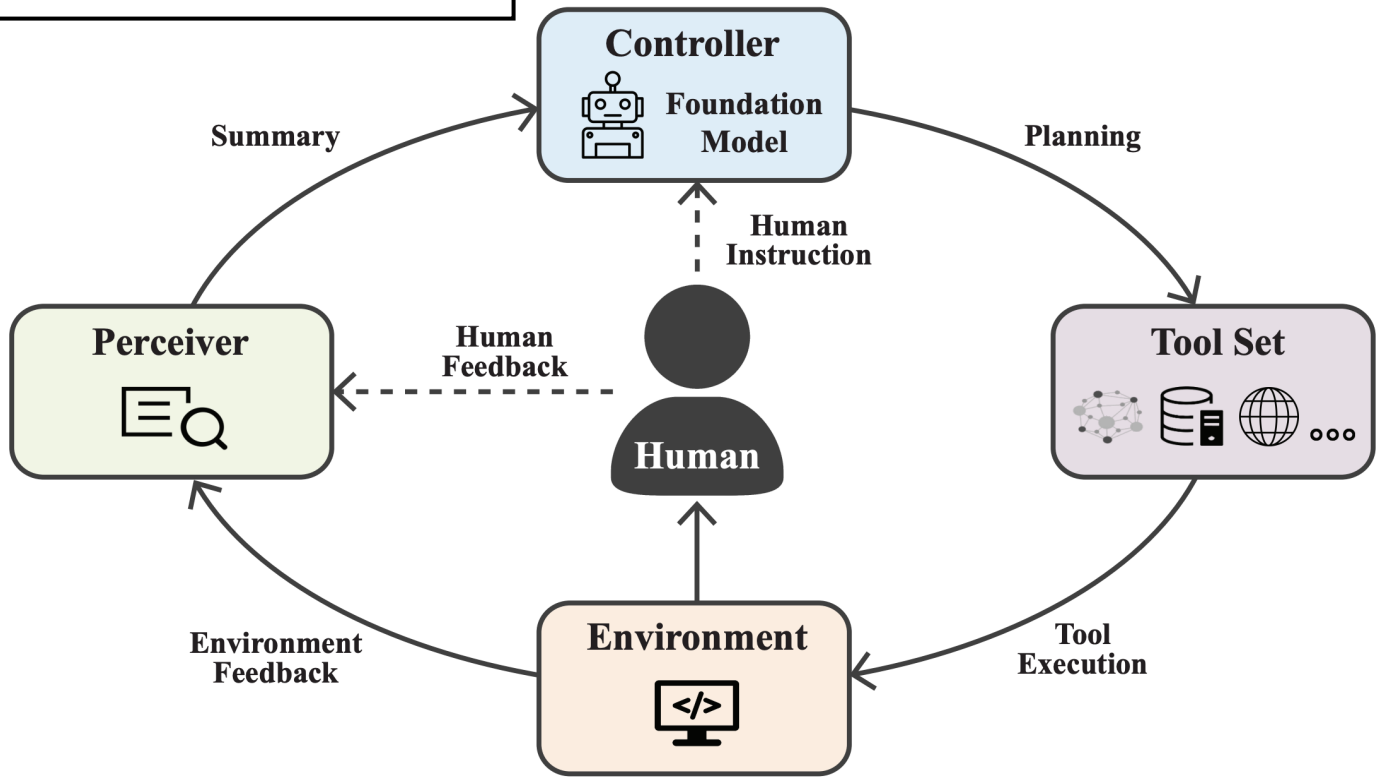


# Framework

THUNLP

# Framework

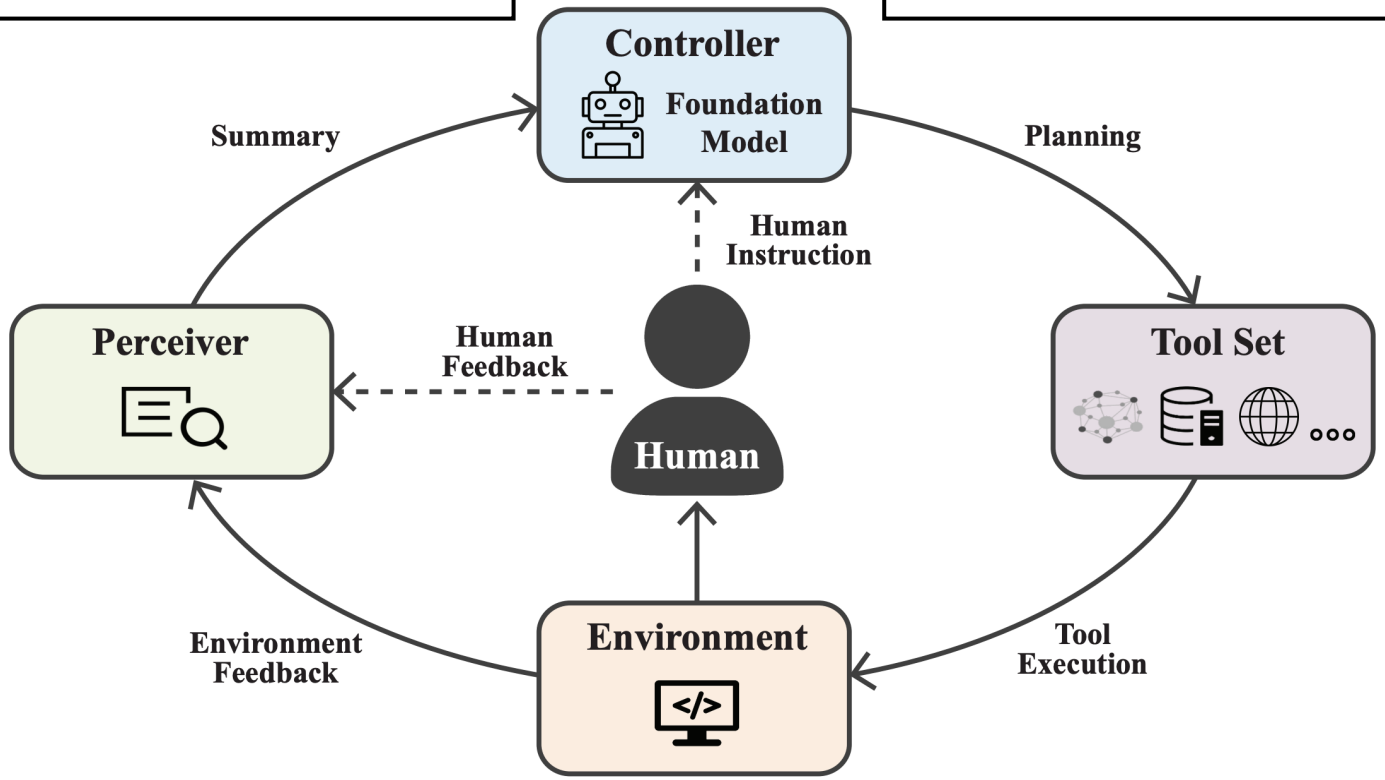
Controller provides feasible plans to fulfill user requests



# Framework

Controller provides feasible plans to fulfill user requests

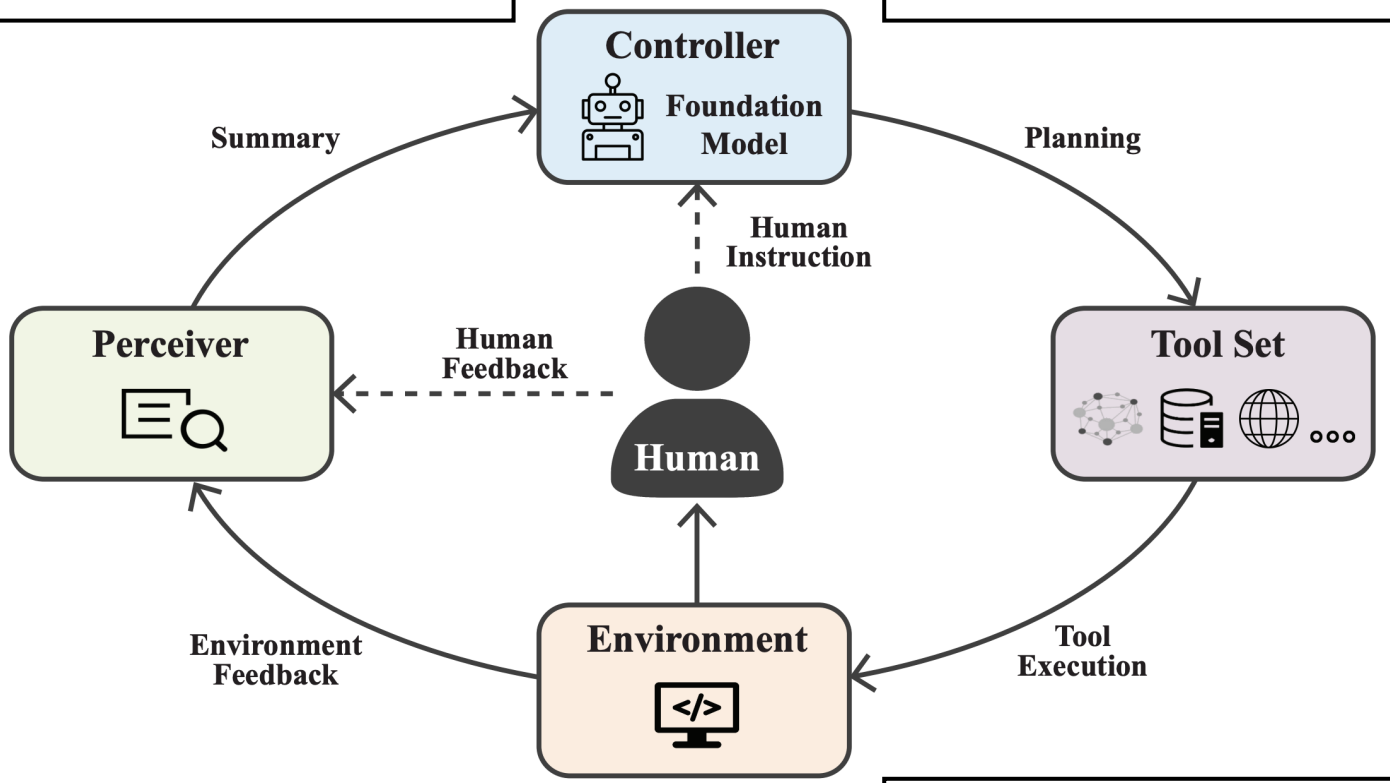
Tool Set: a collection of tools with different functionalities



# Framework

Controller provides feasible plans to fulfill user requests

Tool Set: a collection of tools with different functionalities

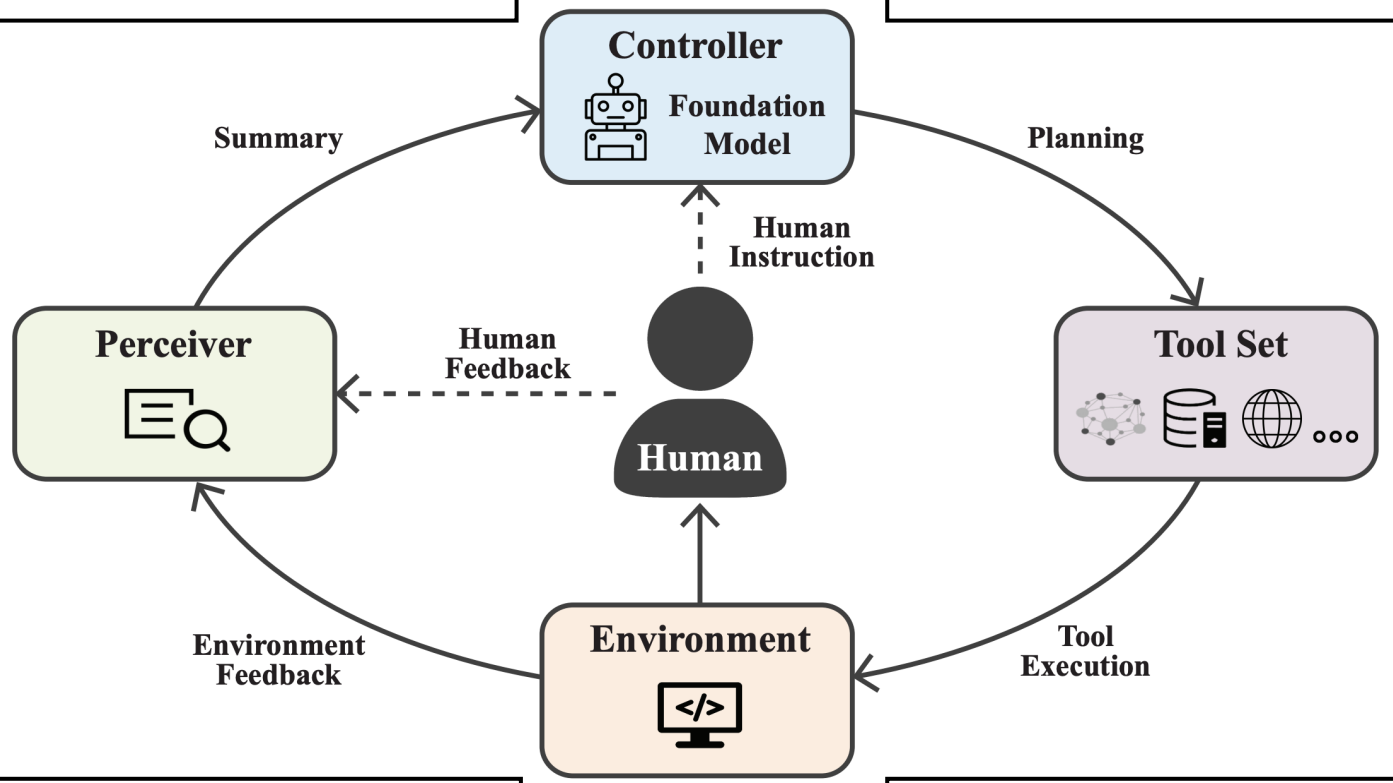



Environment provides the platform where tools operate

# Framework

Controller provides feasible plans to fulfill user requests

Tool Set: a collection of tools with different functionalities




The perceiver summarizes feedback to the controller

Environment provides the platform where tools operate

# Framework

- Controller $\mathcal{C}$ generates a plan $a_t$

Feedback History Instruction

$$p_{\mathcal{C}}(a_t) = p_{\theta_{\mathcal{C}}}(a_t \mid x_t, \mathcal{H}_t, q)$$

- Problem
  - Planning: divide the user query into sub-tasks
  - Tool Use: use the appropriate tool to solve sub-task
  - Memory: manage the working history
  - Profile: manage the user preference

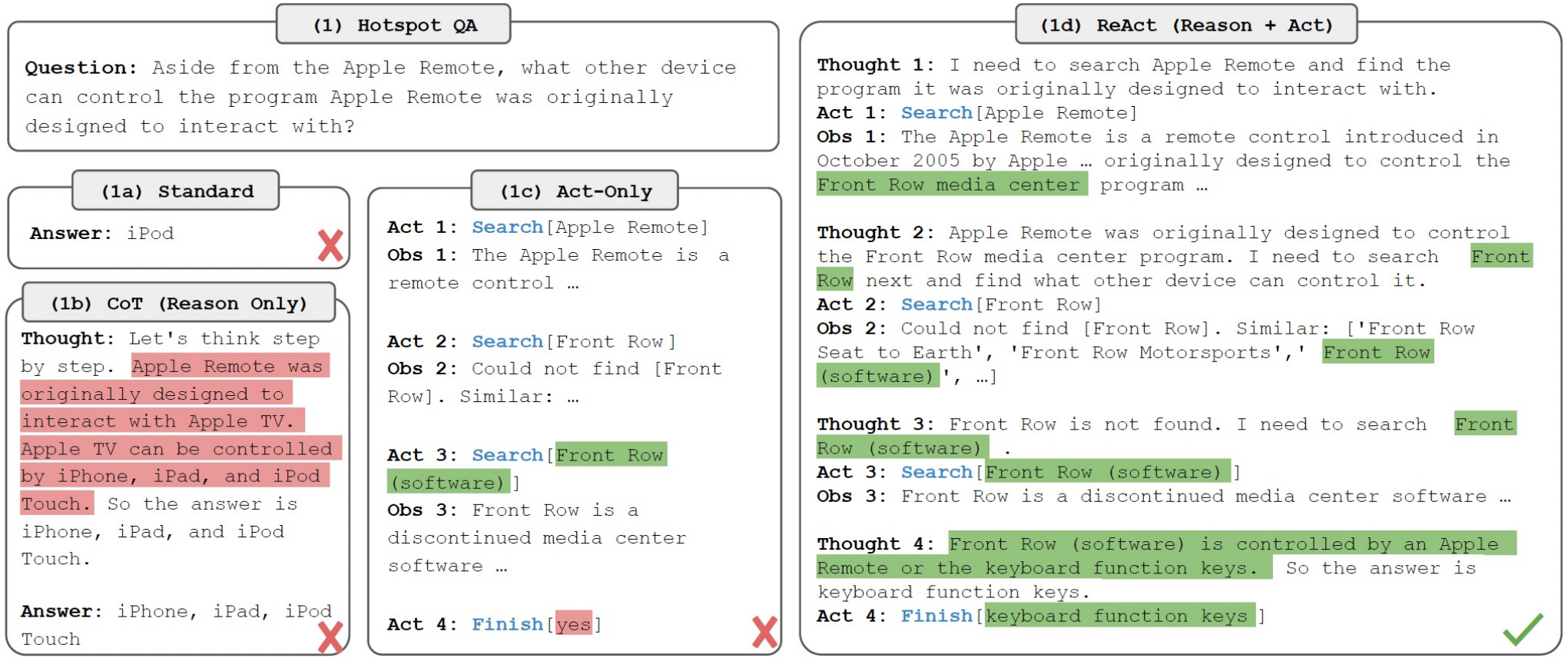


# Planning

THUNLP

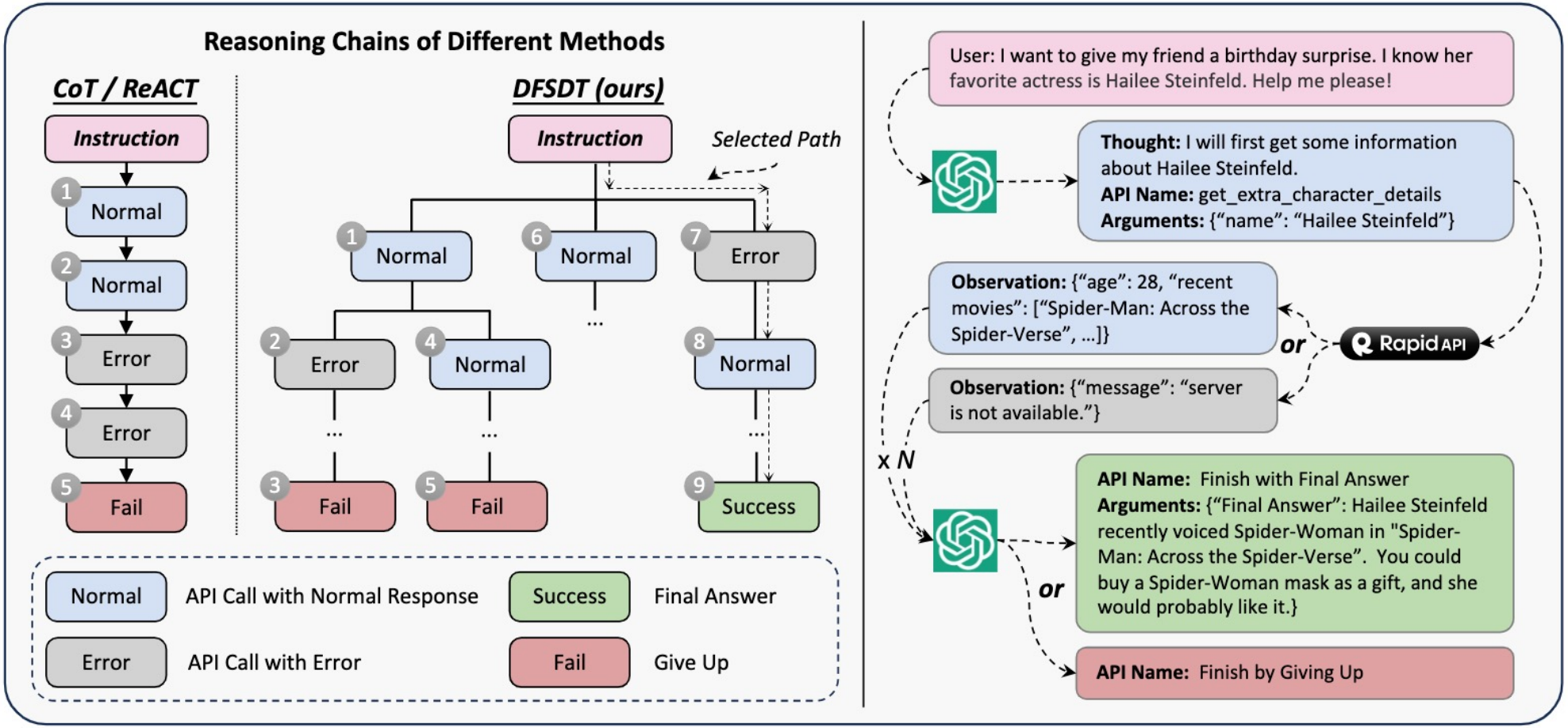
# Planning with Feedback

- ReAct



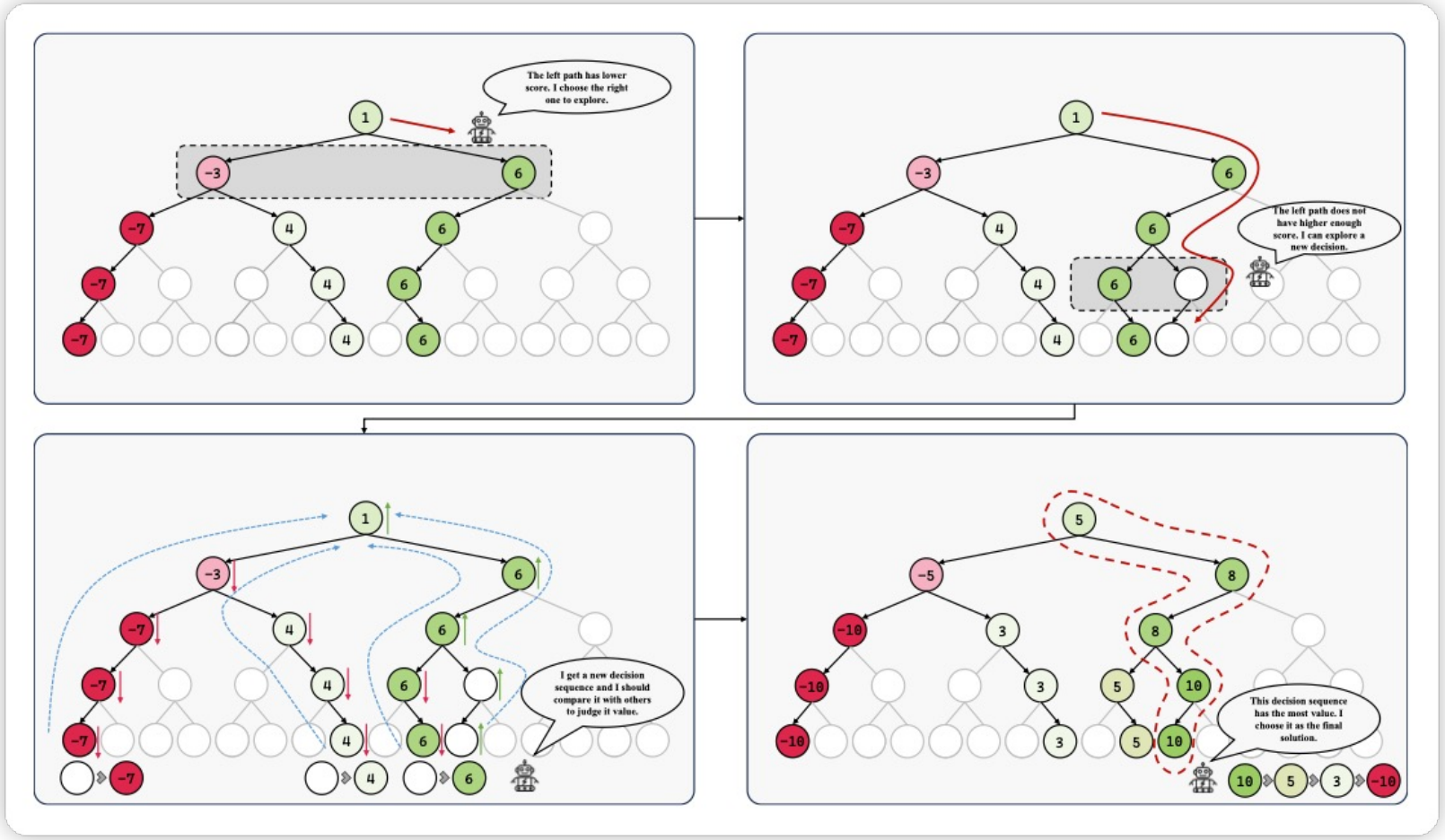
# Planning with Feedback

- DFSDT



# Planning with Feedback

- RADAgent



# Planning with Feedback

- RADAgent
  - ELO Tree Search
  - Forward: Explore based on node scores
  - Backward: Update node scores using the ELO rating system
- Elo Rating System
  - Assumes that each player's skill level follows a Gaussian distribution, and each game is a sample. The expected win rate between two players is:

$$P(d_i) = \frac{\exp(\frac{v_i}{\tau})}{\sum_j \exp(\frac{v_j}{\tau})},\ d_i \in \{d_1, d_2, \cdots, d_n\}$$

  - The ELO scores are dynamically adjusted according to actual game outcomes:

$$\tau_d = \tau_0 * \frac{1}{1 + \sqrt{\ln(M_d + 1)}}$$

# Planning with Feedback

- RADAgent

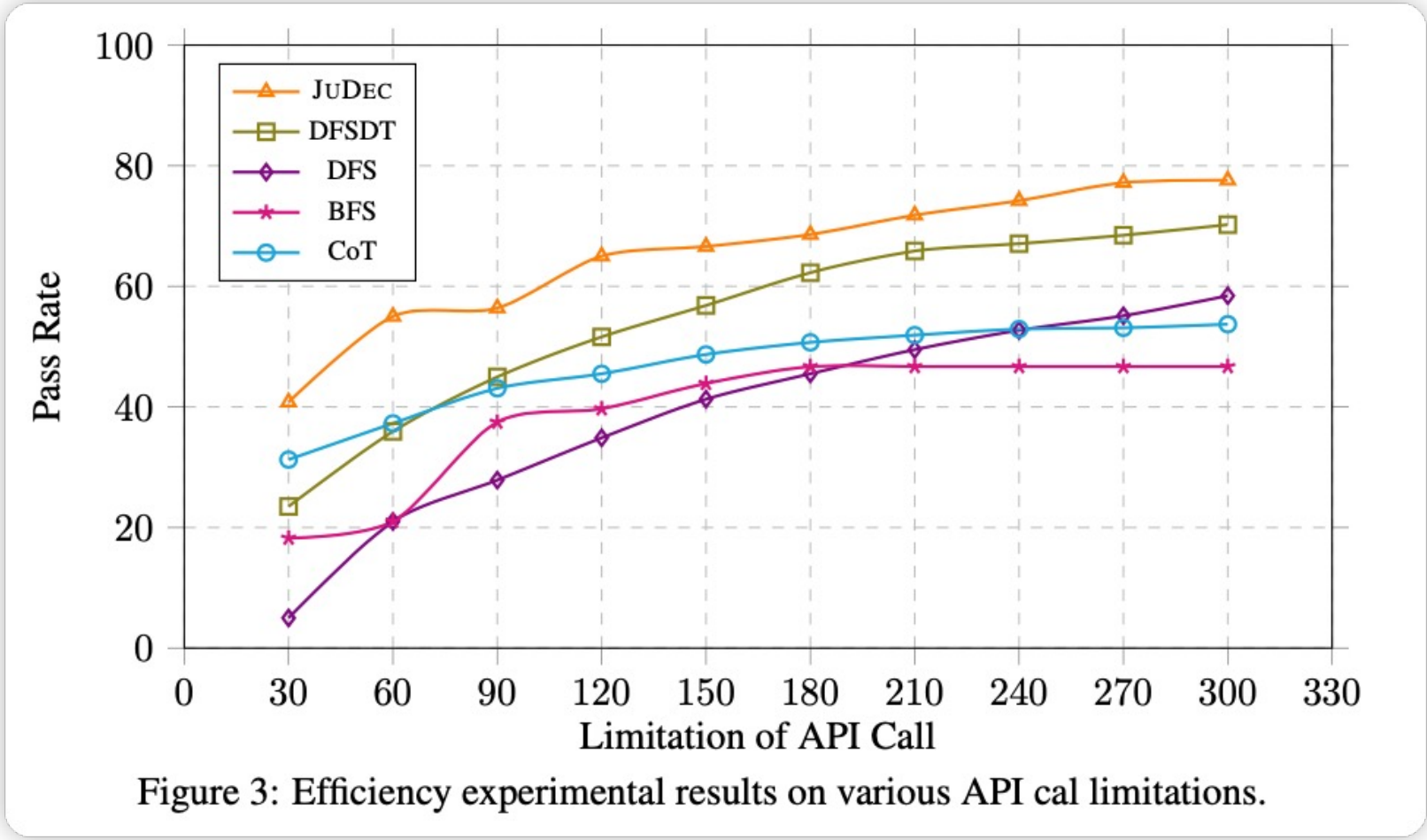



Figure 3: Efficiency experimental results on various API cal limitations.

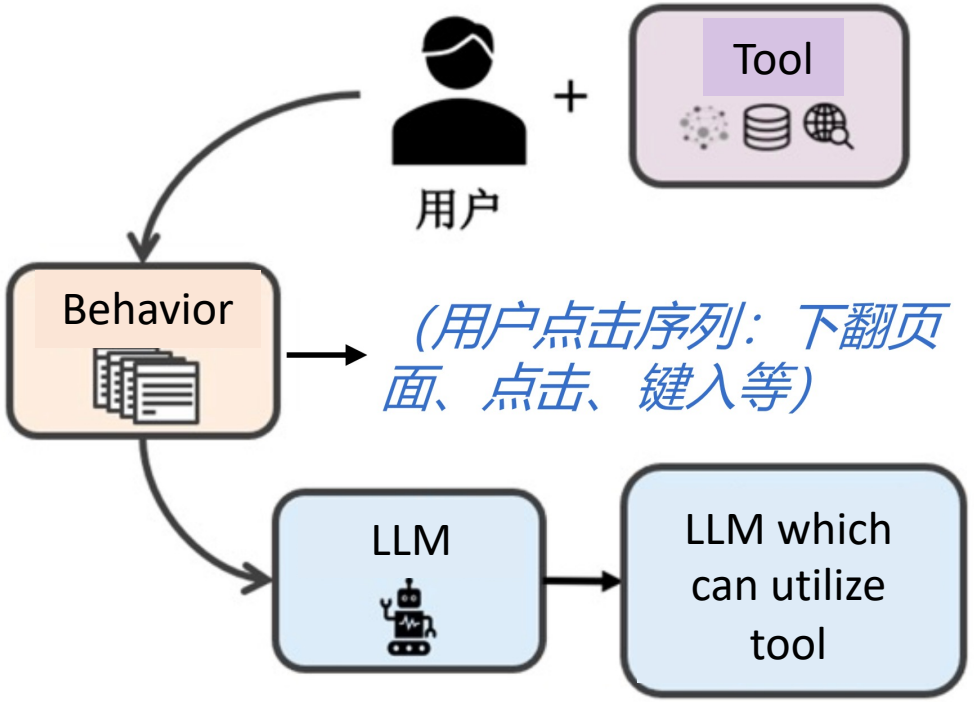


# Tool Use

THUNLP

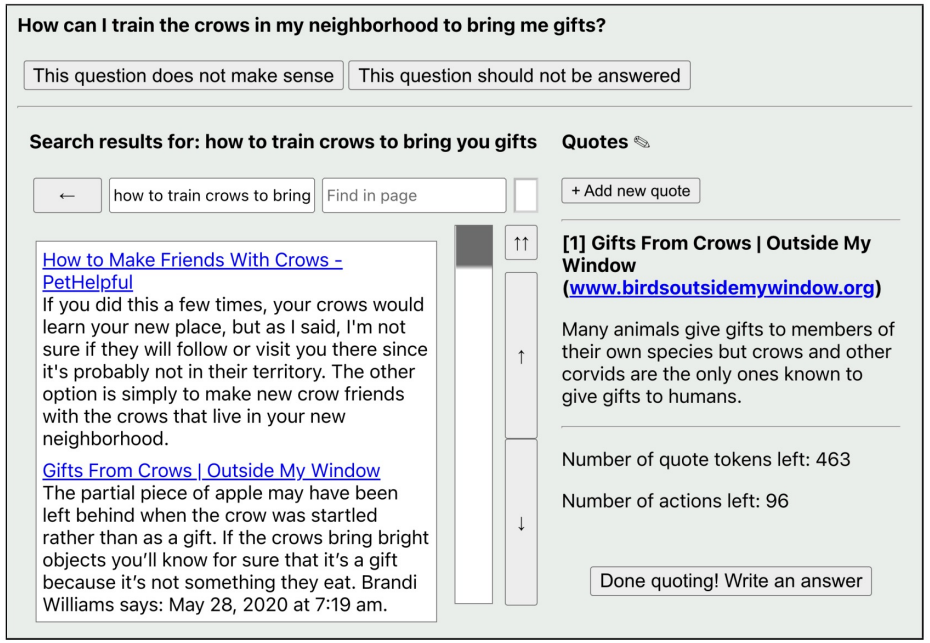
# Learning to Use Tool

- Imitation Learning
  - By recording data on human tool usage behaviors, large models mimic human actions to learn about tools
- The simplest and most direct method of tool learning.



# WebGPT

- Supervised Learning OpenAI
  - Clone human behavior to use search engines
  - Supervised fine-tuning + reinforcement learning
  - Only need 6,000 annotated data

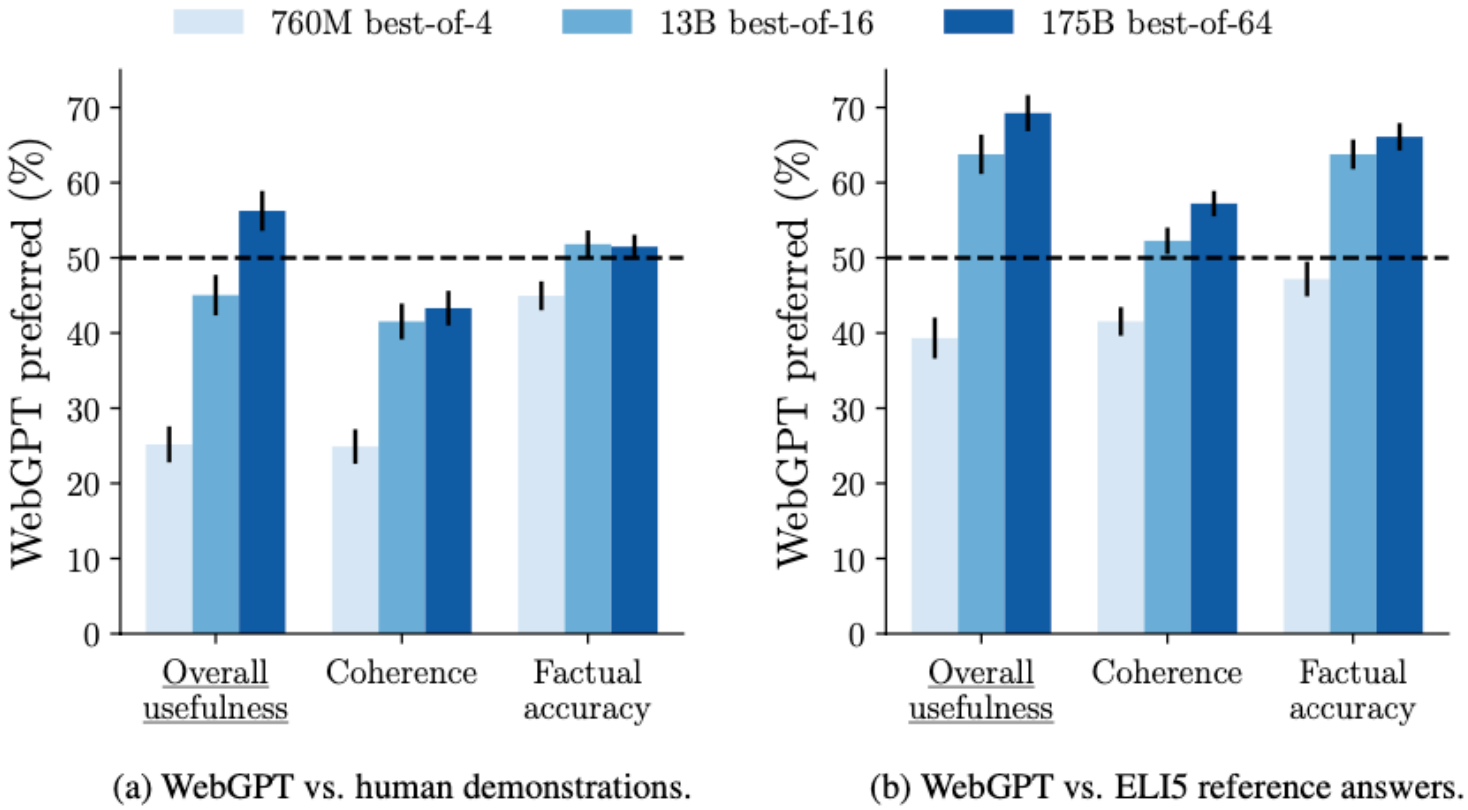



```
♦Question
How can I train the crows in my neighborhood to bring me gifts?

♦Quotes
From Gifts From Crows | Outside My Window (www.birdsoutsidemywindow.org)
> Many animals give gifts to members of their own species but crows and
other corvids are the only ones known to give gifts to humans.

♦Past actions
Search how to train crows to bring you gifts
Click Gifts From Crows | Outside My Window www.birdsoutsidemywindow.org
Quote
Back

♦Title
Search results for: how to train crows to bring you gifts

♦Scrollbar: 0 - 11
♦Text
【0†How to Make Friends With Crows - PetHelpful†pethelpful.com】
If you did this a few times, your crows would learn your new place, but
as I said, I'm not sure if they will follow or visit you there since it's
probably not in their territory. The other option is simply to make new
crow friends with the crows that live in your new neighborhood.

【1†Gifts From Crows | Outside My Window†www.birdsoutsidemywindow.org】
The partial piece of apple may have been left behind when the crow was
startled rather than as a gift. If the crows bring bright objects you'll
know for sure that it's a gift because it's not something they eat.
Brandi Williams says: May 28, 2020 at 7:19 am.

♦Actions left: 96
♦Next action
```

Nakano, Reiichiro, et al. "WebGPT: Browser-assisted question-answering with human feedback." arXiv preprint arXiv:2112.09332 (2021).

# WebGPT

- Supervised Learning OpenAI
  - Excellent performance in long-form QA, even surpassing human experts




(a) WebGPT vs. human demonstrations.

(b) WebGPT vs. ELI5 reference answers.

Nakano, Reiichiro, et al. "WebGPT: Browser-assisted question-answering with human feedback." arXiv preprint arXiv:2112.09332 (2021).

# WebCPM: Chinese WebGPT

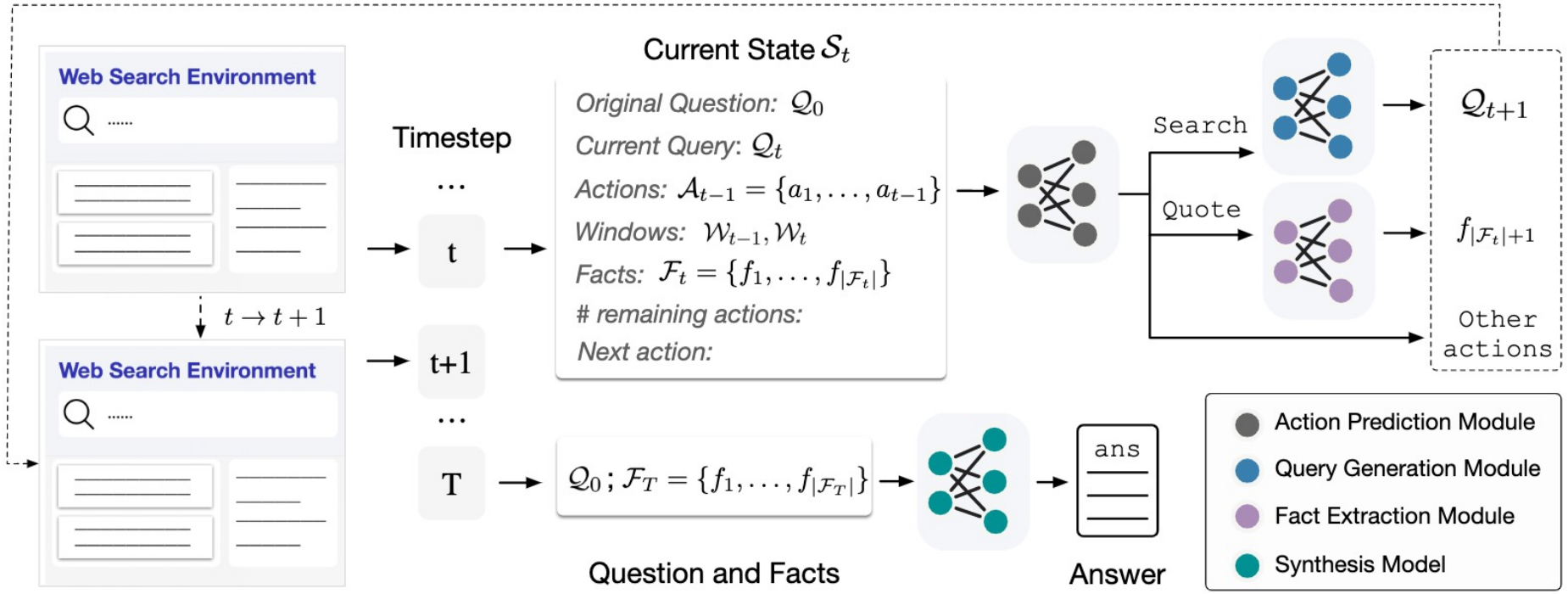
- A case study in Chinese


| Action Name | Functionality |
|---|---|
| Search <query> | Call Bing search with <query> |
| Go Back | Return to the previous window |
| Load Page <1> | Load the details of page <1> |
| Load Page <2> | Load the details of page <2> |
| Load Page <3> | Load the details of page <3> |
| Scroll Up | Scroll up for a pre-set stride |
| Scroll Down | Scroll down for a pre-set stride |
| Quote <content> | Extract <content> from the current page as a supporting fact |
| Merge | Merge two facts into a single fact |
| Finish | End the search process |

Qin, et al. Interactive web search for Chinese long-form question answering. ACL 2023

# WebCPM: Chinese WebGPT

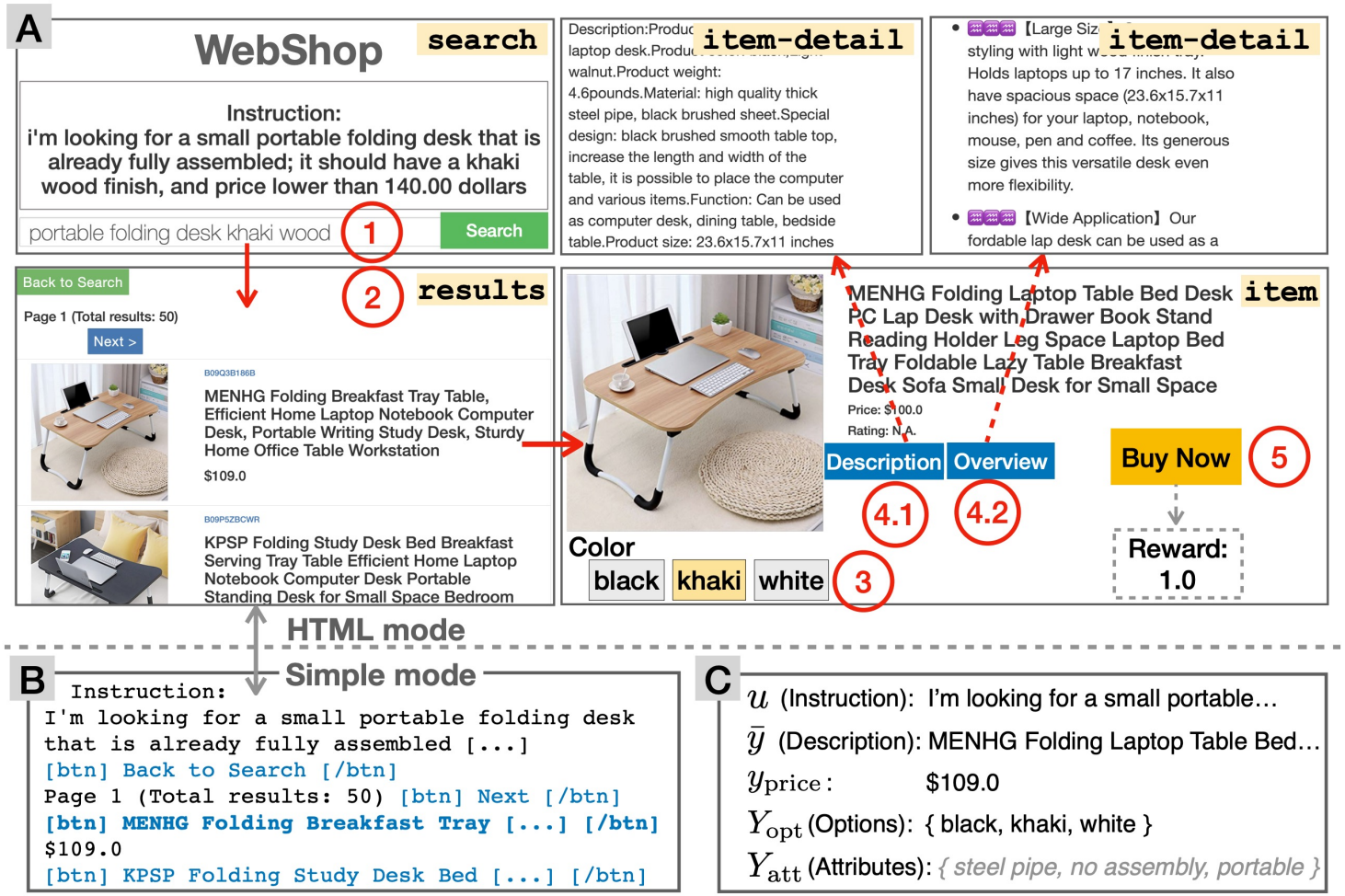
- At each step, the search model executes actions to collect supporting facts, which are sent to the synthesis model for answer generation




Qin, et al. Interactive web search for Chinese long-form question answering. ACL 2023

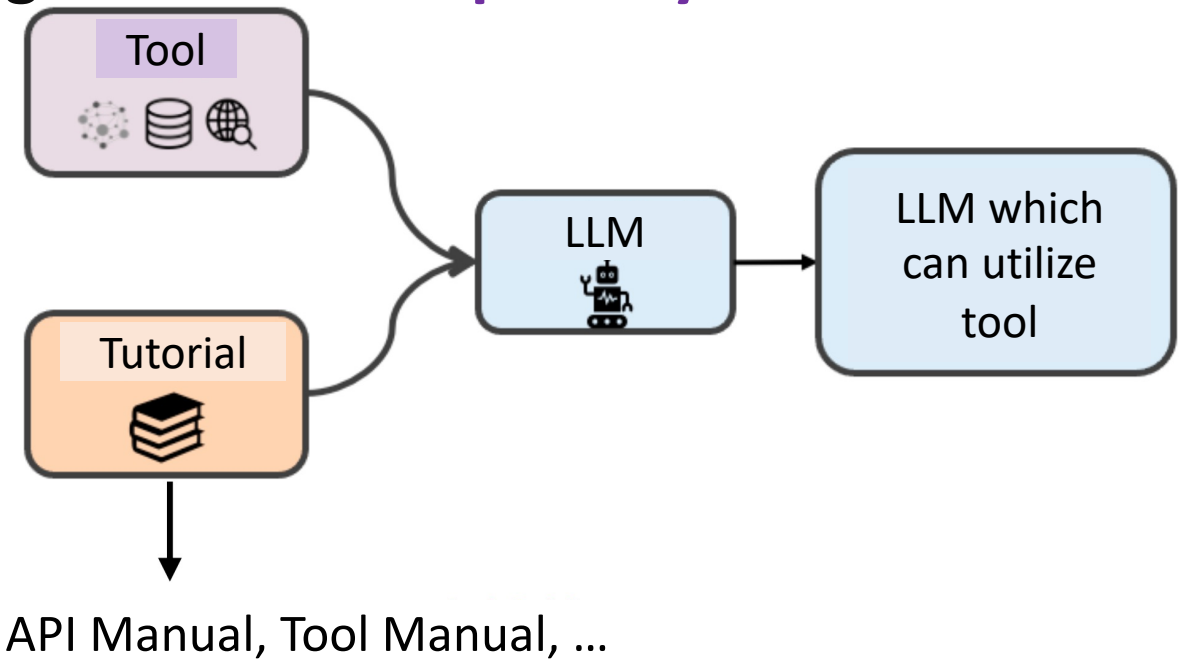
# WebShop

- Learning to perform online shopping



# Learning to Use Tool

- Tutorial Learning
  - By having the model read tool manuals (tutorials), it understands the functions of the tools and how to invoke them
- Almost exclusively, large models from the OpenAI series (such as ChatGPT, GPT-4) possess a high **zero-shot capability** to understand tool manuals.



# Learning to Use Tool

- Describe the functionality; In-context with example(s).

**Zero-shot Prompting:** Here we provide a tool (API) "forecast_weather(city:str, N:int)", which could forecast the weather about a city on a specific date (after N days from today). The returned information covers "temperature", "wind", and "precipitation".
Please write codes using this tool to answer the following question: "What's the average temperature in Beijing next week?"

---

**Few-shot Prompting:** We provide some examples for using a tool. Here is a tool for you to answer question:
Question: "What's the temperature in Shanghai tomorrow?"

```
return forecast_weather("Shanghai", 1)["temperature"]
```

Question: "Will it rain in London in next two days?"

```
for i in range(2):
    if forecast_weather("London", i+1)["precipitation"] > 0:
        return True
return False
```

Question: "What's the average temperature in San Francisco next week?"

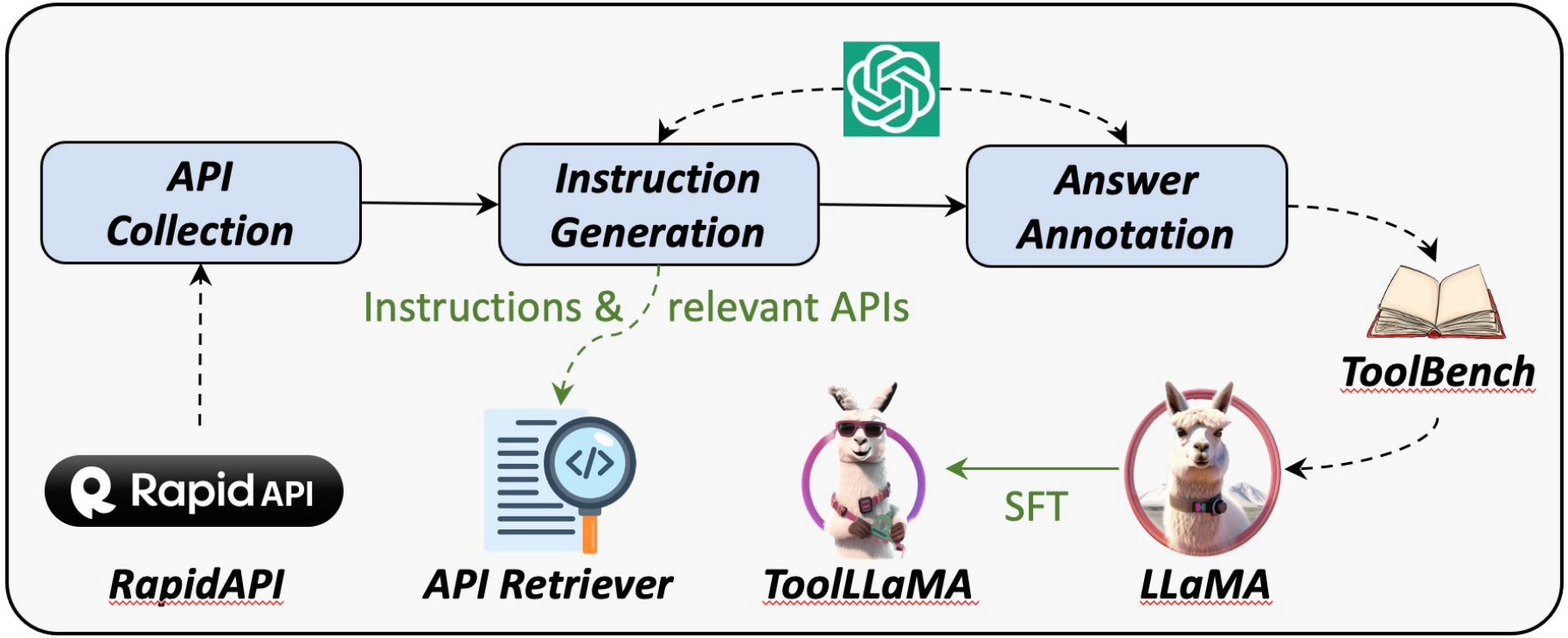
# ToolBench

- Highlights:
  - Over 16,000 real APIs (collected from RapidAPI)
  - Supports single and multi-tool invocation
  - Complex multi-step reasoning tasks

| Resource | ToolBench (this work) | APIBench (Patil et al., 2023) | API-Bank (Li et al., 2023a) | ToolAlpaca (Tang et al., 2023) | T-Bench (Xu et al., 2023b) |
|---|---|---|---|---|---|
| Real-world API? | ✓ | ✗ | ✓ | ✗ | ✓ |
| Real API Response? | ✓ | ✗ | ✓ | ✗ | ✓ |
| Multi-tool Scenario? | ✓ | ✗ | ✗ | ✗ | ✗ |
| API Retrieval? | ✓ | ✓ | ✗ | ✗ | ✗ |
| Multi-step Reasoning? | ✓ | ✗ | ✓ | ✓ | ✓ |
| Number of tools | **3451** | 3 | 53 | 400 | 8 |
| Number of APIs | **16464** | 1645 | 53 | 400 | 232 |
| Number of Instances | 12657 | **17002** | 274 | 3938 | 2746 |
| Number of Real API Calls | **37204** | 0 | 568 | 0 | 0 |
| Avg. Reasoning Traces | 4.1 | 1.0 | 2.1 | 1.0 | **5.9** |

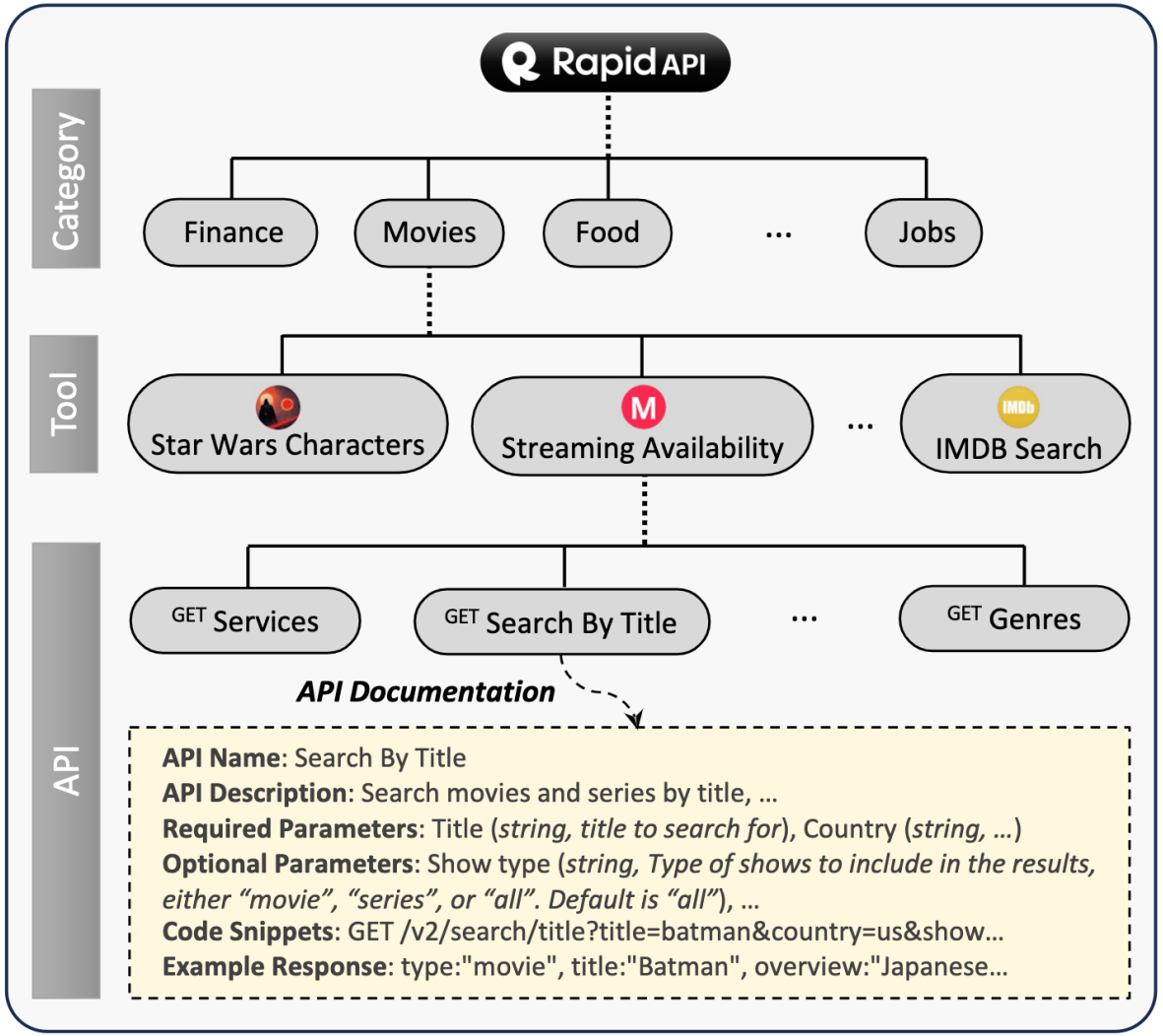
# ToolBench Construction

- API Collection
- Instruction Generation
- Answer Annotation



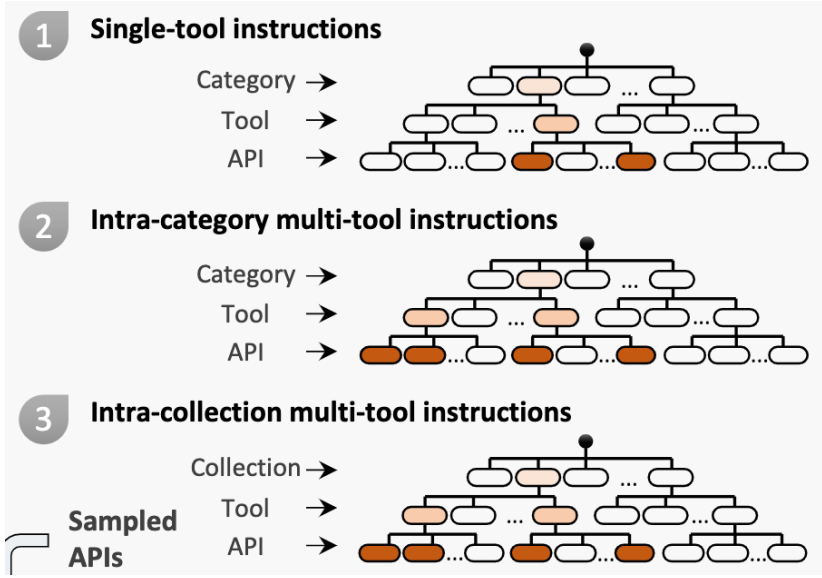
# ToolBench Construction

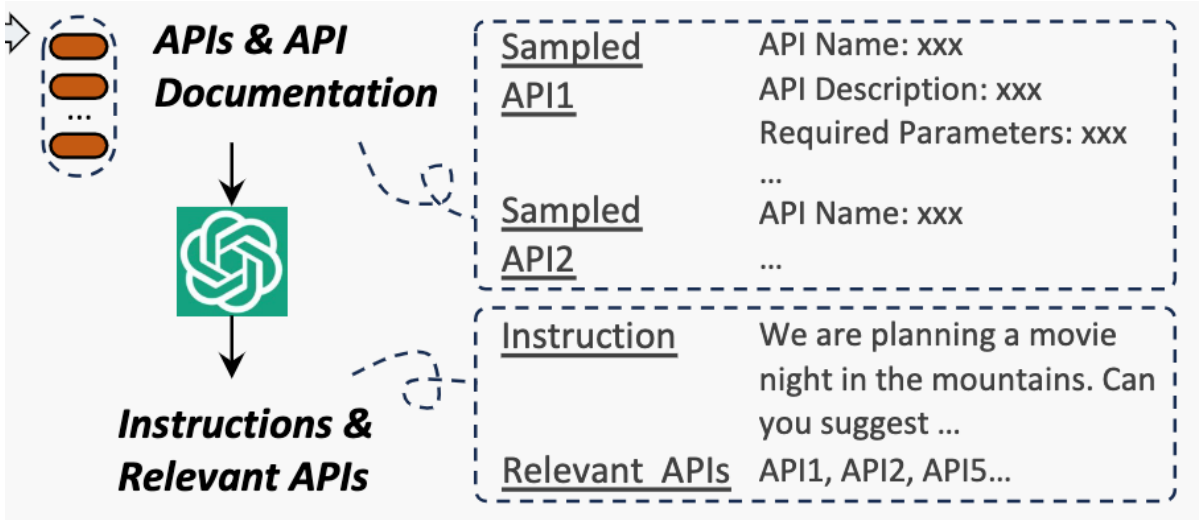
- API Collection
  - RapidAPI Hub: https://rapidapi.com/hub
  - Filter over 16,000 high-quality APIs from more than 50,000 APIs
  - Include 49 categories



# ToolBench Construction

- Instruction Generation
  - Single Tool + Multi-Tool
  - (1) Sample a collection of APIs: $\mathbb{S}_N^{\text{sub}} = \{\text{API}_1, \cdots, \text{API}_N\}$
  - (2) ChatGPT automatically generate instructions that may require calling one or more APIs in the collection: $\underset{\substack{\{\text{API}_1,\cdots,\text{API}_N\}\in\mathbb{S}_{\text{API}},\\ \{\text{seed}_1,\cdots,\text{seed}_3\}\in\mathbb{S}_{\text{seed}}}}{\text{ChatGPT}}(\{[\mathbb{S}_1^{\text{rel}}, \text{Inst}_1], \cdots, [\mathbb{S}_{N'}^{\text{rel}}, \text{Inst}_{N'}]\}|\text{API}_1, \cdots, \text{API}_N, \text{seed}_1, \cdots, \text{seed}_3).$



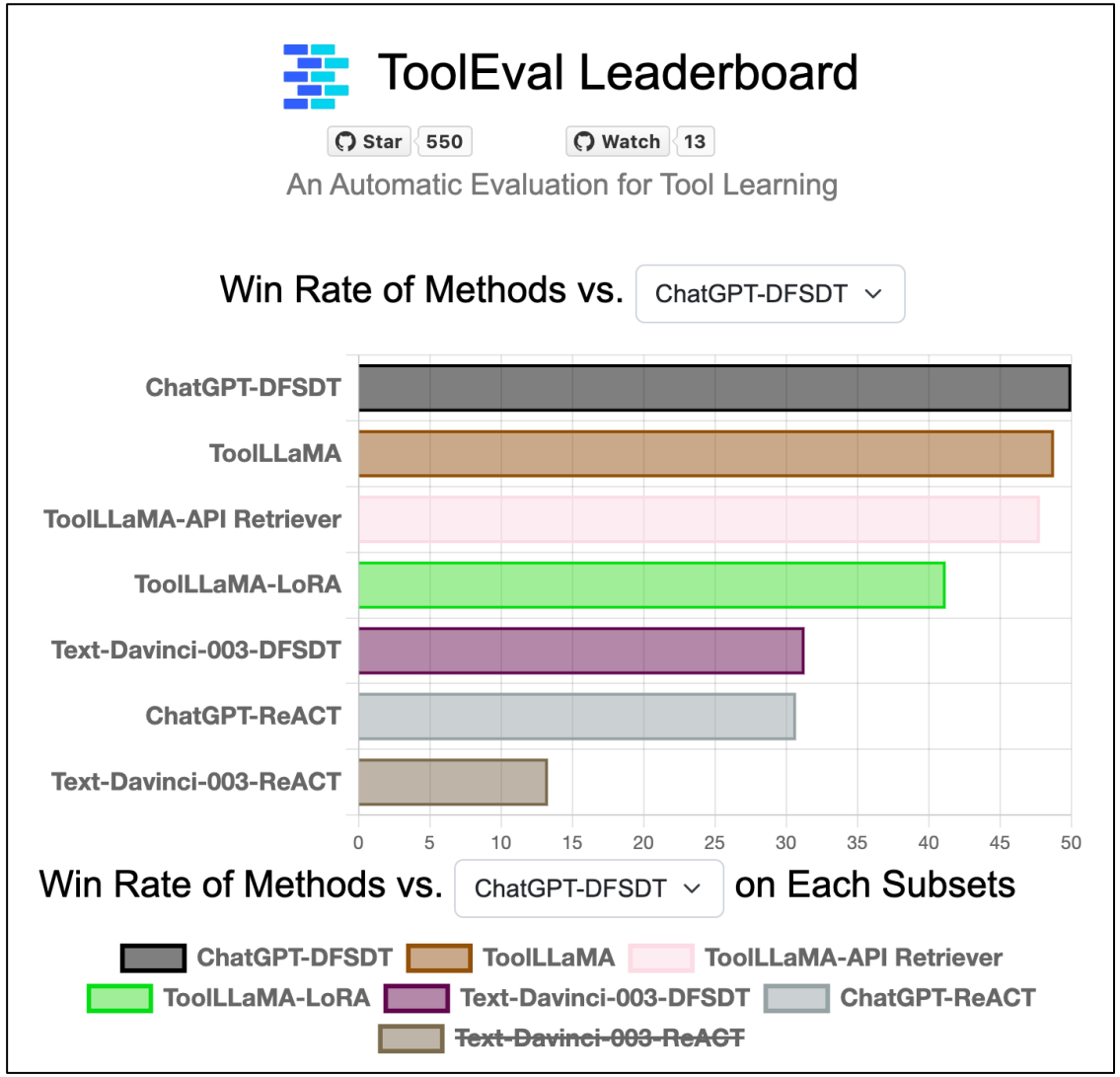
# ToolBench Construction

- Answer Annotation
  - gpt-3.5-turbo-16k: feature of function call
- Issues with ReACT
  - Error Propagation: An error in a single step annotation can render the entire action sequence unusable
  - Limited Exploration: ReACT can only sample one sequence from the infinite action sequence space based on the LM's probabilities
- DFSDT: Dynamically extends the TOT to the tool learning scenario

| Method | Single-tool (I1) | Category (I2) | Collection (I3) | Average |
|---|---|---|---|---|
| ReACT | 43.98 | 23.62 | 20.42 | 29.34 |
| ReACT@N | 50.80 | 36.14 | 32.87 | 39.94 |
| DFSDT | **54.10** | **47.35** | **44.80** | **48.75** |

# ToolEval

- Automatic Evaluation Framework Based on ChatGPT
- Two metrics:
  - Success rate: The proportion of commands successfully completed within a limited number of API calls
  - Preference: Comparison of quality/usefulness between two answers, i.e., which one is better?
- Highly consistent with human experts (~80%).



# ToolLLaMA

- Demonstrate exceptionally high generalizability to OOD commands and APIs, significantly outperforming ChatGPT+ReACT

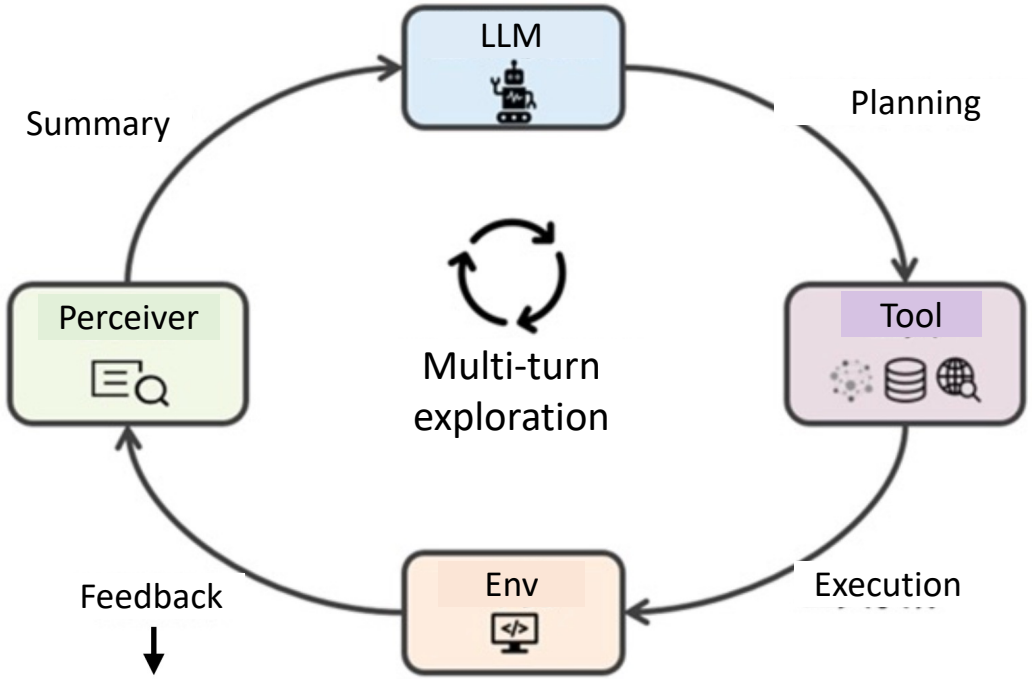
| Model | I1-Inst. | | I1-Tool | | I1-Cat. | | I2-Inst. | | I2-Cat. | | I3-Inst. | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Pass | Win | Pass | Win | Pass | Win | Pass | Win | Pass | Win | Pass | Win | Pass | Win |
| ChatGPT-ReACT | 56.0 | - | 62.0 | - | 66.0 | - | 28.0 | - | 22.0 | - | 30.0 | - | 44.0 | - |
| Vicuna (ReACT & DFSDT) | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - |
| Alpaca (ReACT & DFSDT) | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - | 0.0 | - |
| Text-Davinci-003-DFSDT | 53.0 | 46.0 | 58.0 | 38.0 | 61.0 | 39.0 | 38.0 | 46.0 | 38.0 | 45.0 | 39.0 | 48.0 | 47.8 | 43.7 |
| ChatGPT-DFSDT | **78.0** | **68.0** | **84.0** | **59.0** | **89.0** | **57.0** | **51.0** | **78.0** | **58.0** | 77.0 | **57.0** | **77.0** | **69.6** | **69.3** |
| ToolLLaMA-DFSDT | 68.0 | **68.0** | 80.0 | **59.0** | 75.0 | 56.0 | 47.0 | 75.0 | 56.0 | **80.0** | 40.0 | 72.0 | 61.0 | 68.3 |

- DFSDT >> ReACT

| Method | Single-tool (I1) | Category (I2) | Collection (I3) | Average |
|---|---|---|---|---|
| ReACT | 43.98 | 23.62 | 20.42 | 29.34 |
| ReACT@N | 50.80 | 36.14 | 32.87 | 39.94 |
| DFSDT | **54.10** | **47.35** | **44.80** | **48.75** |

# Learning to Use Tool

- Reinforcement Learning
  - Capable of autonomous exploration and corrects errors based on environmental feedback through reinforcement learning
- There is limited existing research on this topic.




API Calling Success Rate, User Feedback …

# Learning to Use Tool

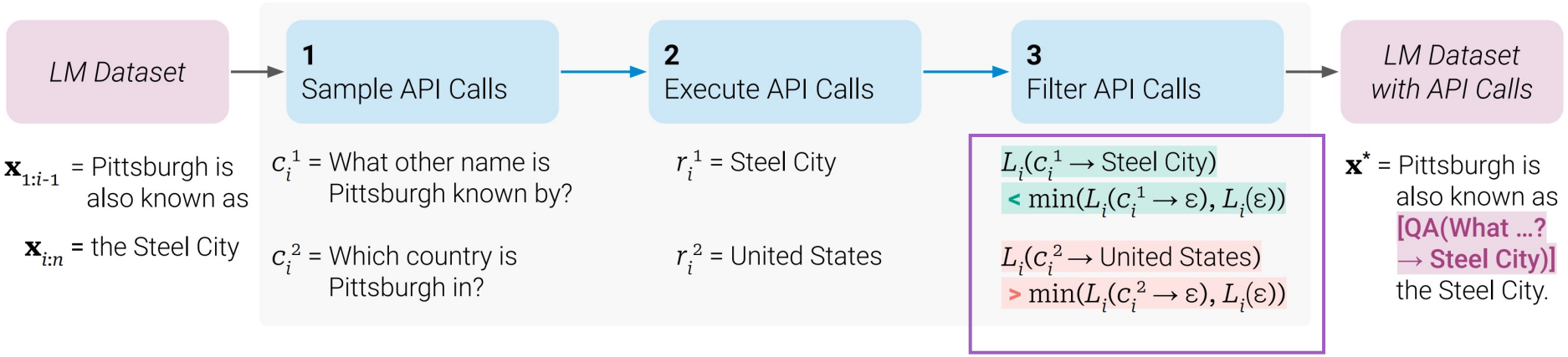
- **Learning from feedback**: often involves reinforcement learning

$$\theta_{\mathcal{C}}^{*} = \arg\max_{\theta_{\mathcal{C}}} \mathop{\mathbb{E}}_{q_i \in Q} \mathop{\mathbb{E}}_{\{a_{i,t}\}_{t=0}^{T_i} \in p_{\theta_{\mathcal{C}}}} \left[ R(\{a_{i,t}\}_{t=0}^{T_i}) \right],$$

- Reinforcement Learning (RL) for Tool Use
  - Action space is defined based on tools
  - Agent learns to select the appropriate tool
  - Perform the correct actions that maximize the reward signal

# Toolformer

- Self-supervised Tool Learning
  - Pre-defined tool APIs
  - Encourage models to call and execute tool APIs
  - Design self-supervised loss to see if the tool execution can help language modeling




If the tool execution reduces LM loss, save the instances as training data



# Application

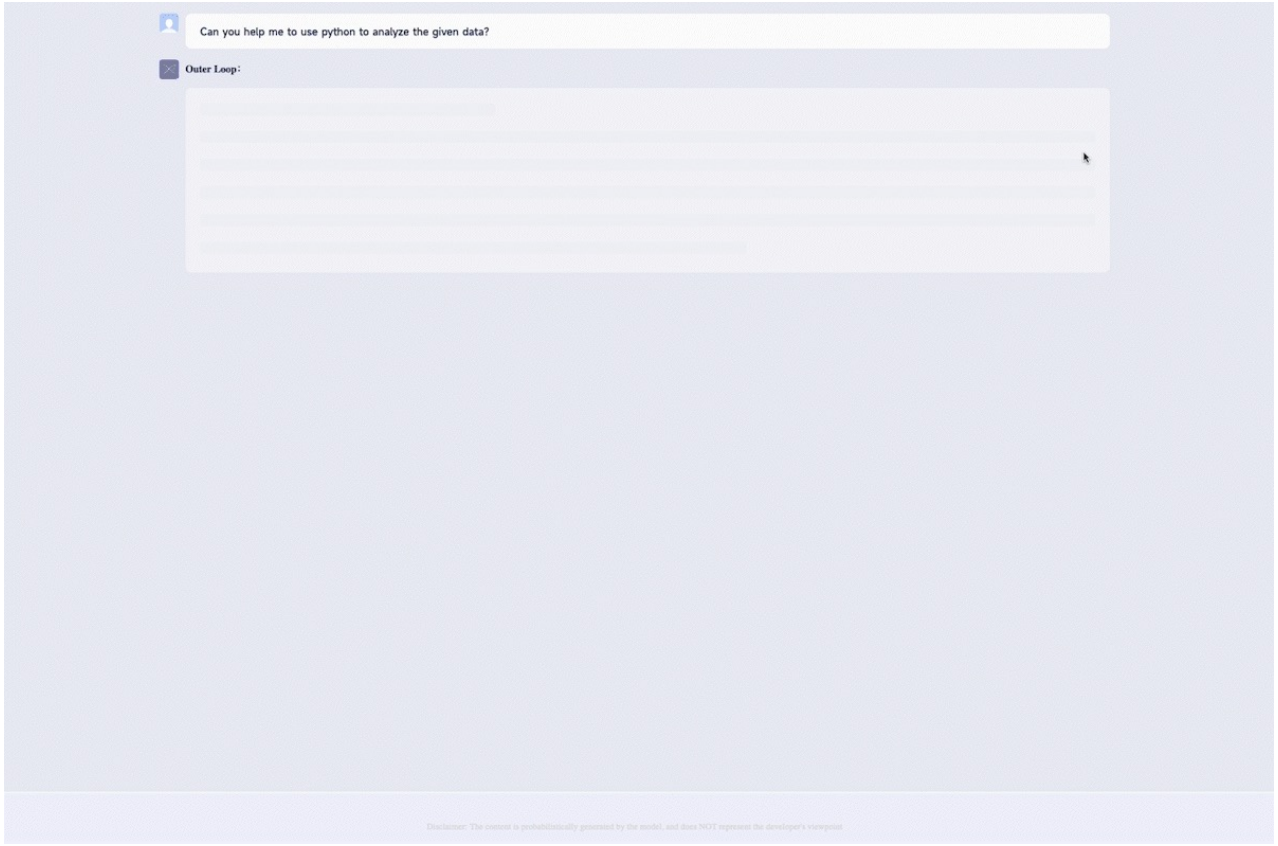
THUNLP

# XAgent

- Dual-loop Mechanism for Planning and Execution
- ToolServer: Tool Execution Docker
- The Universal Language: Function Calling:

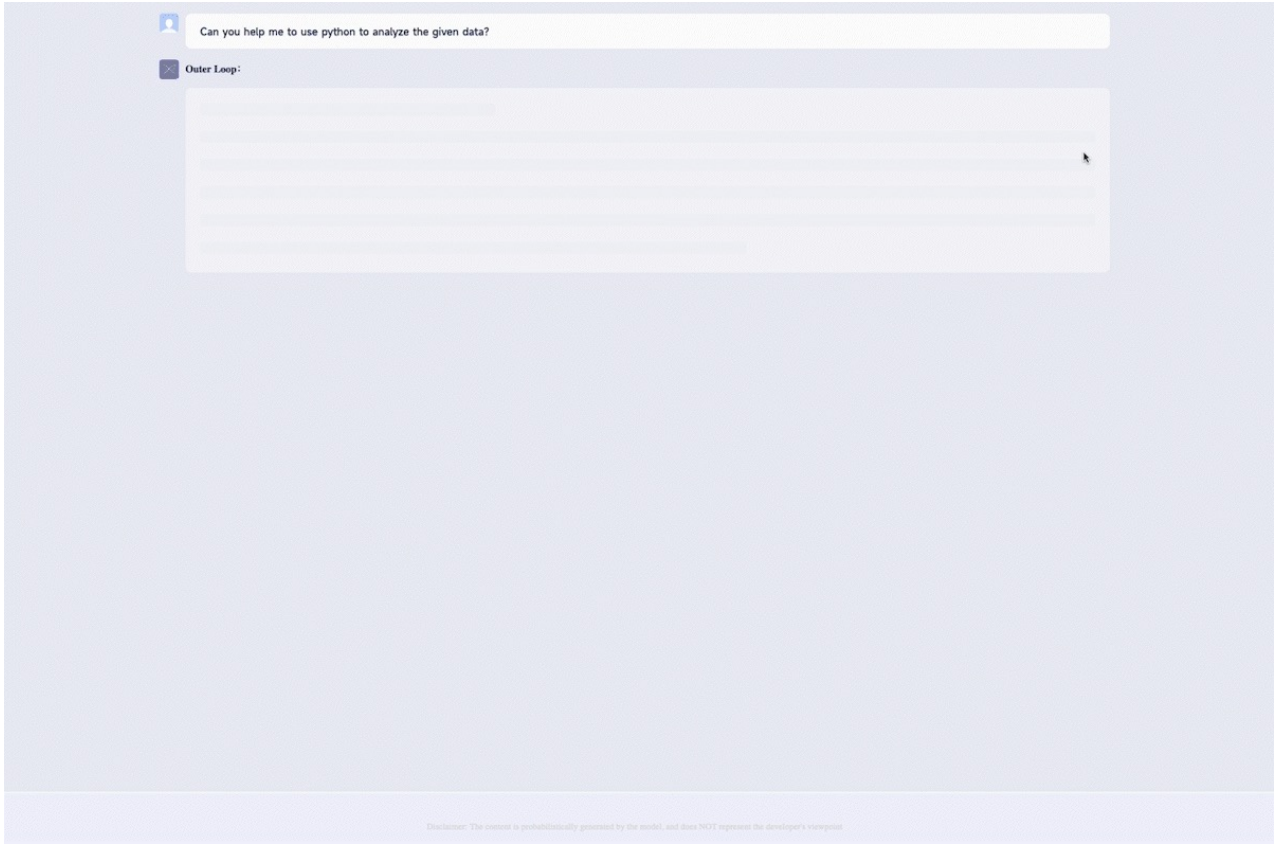
# Example: Data Analysis

- Outer-loop splits the task into four sub-tasks
  - Data inspection and comprehension
  - Verification of the system's Python environment for relevant data analysis libraries
  - Crafting data analysis code for data processing and analysis
  - Compiling an analytical report based on the Python code's execution results.

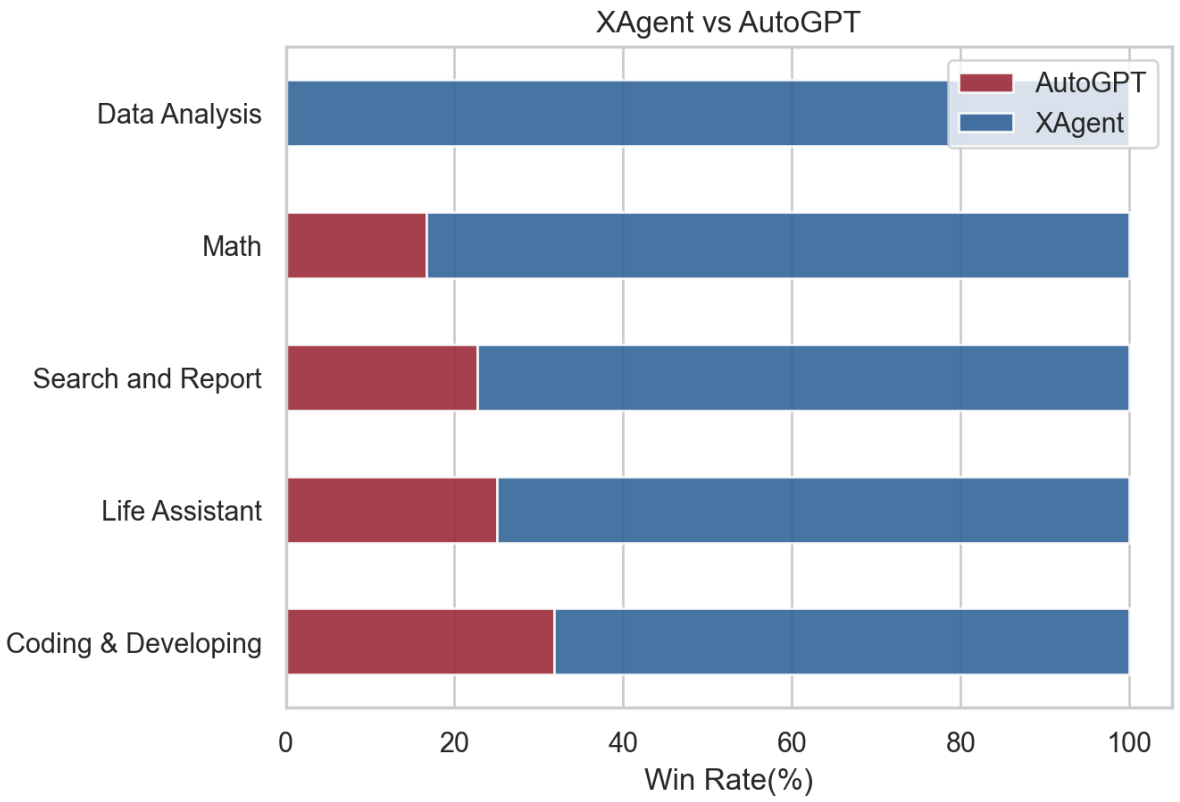


# Case Study: Data Analysis

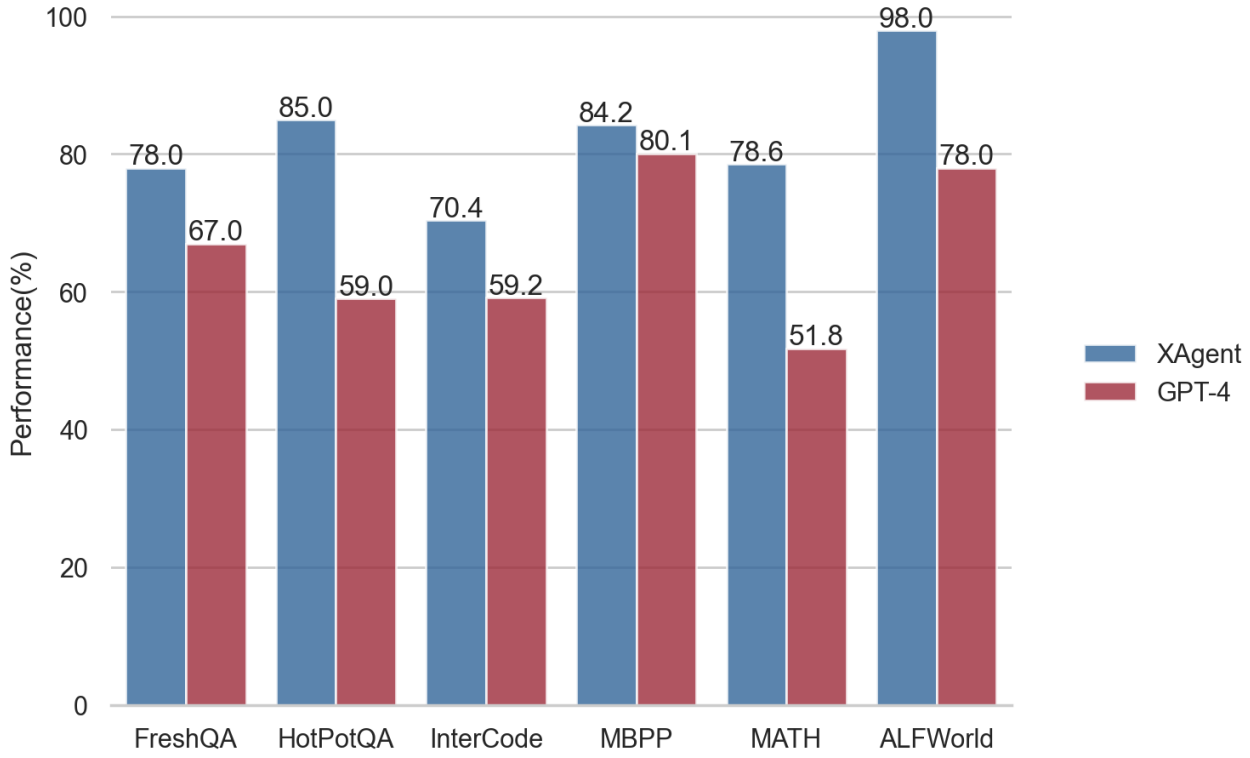
- Inter-loop
  - Employ various data analysis libraries such as pandas, sci-kit learn, seaborn, matplotlib, alongside skills in file handling, shell commands, and Python notebooks



# Performance



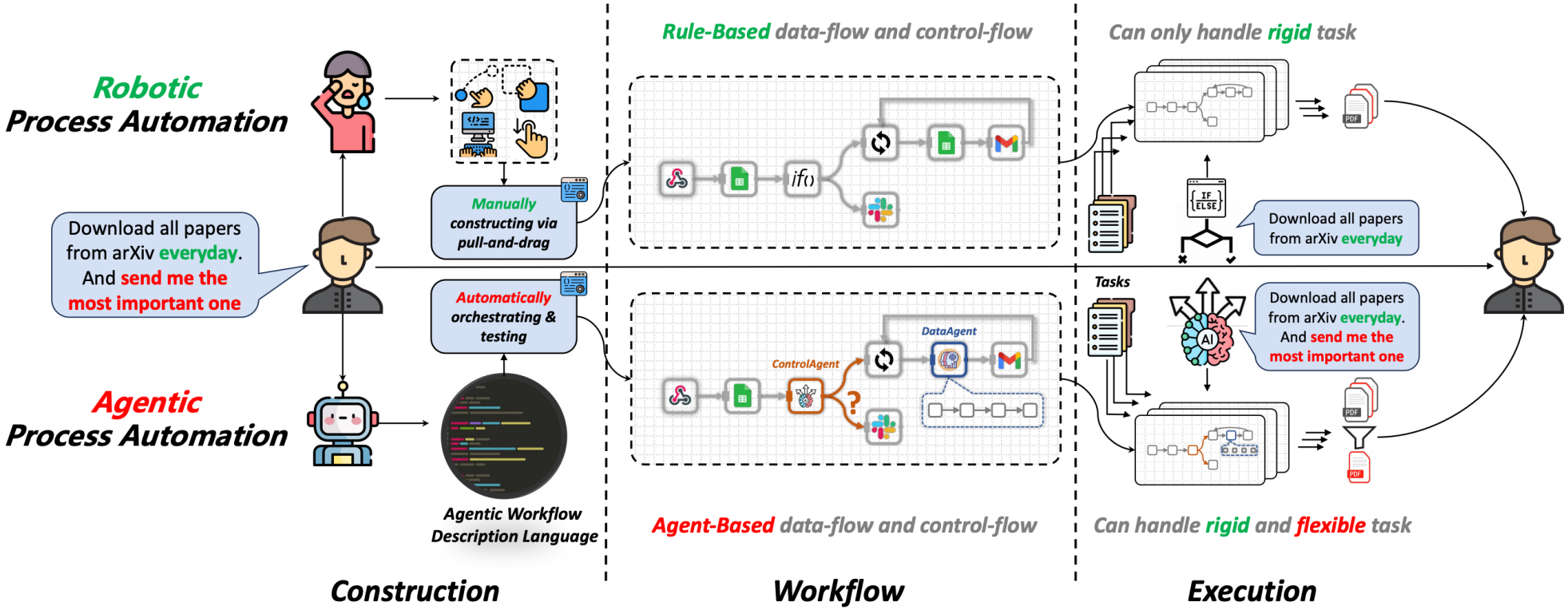

XAgent v.s. AutoGPT on our curated instructions




XAgent v.s. GPT-4 on existing AI benchmarks

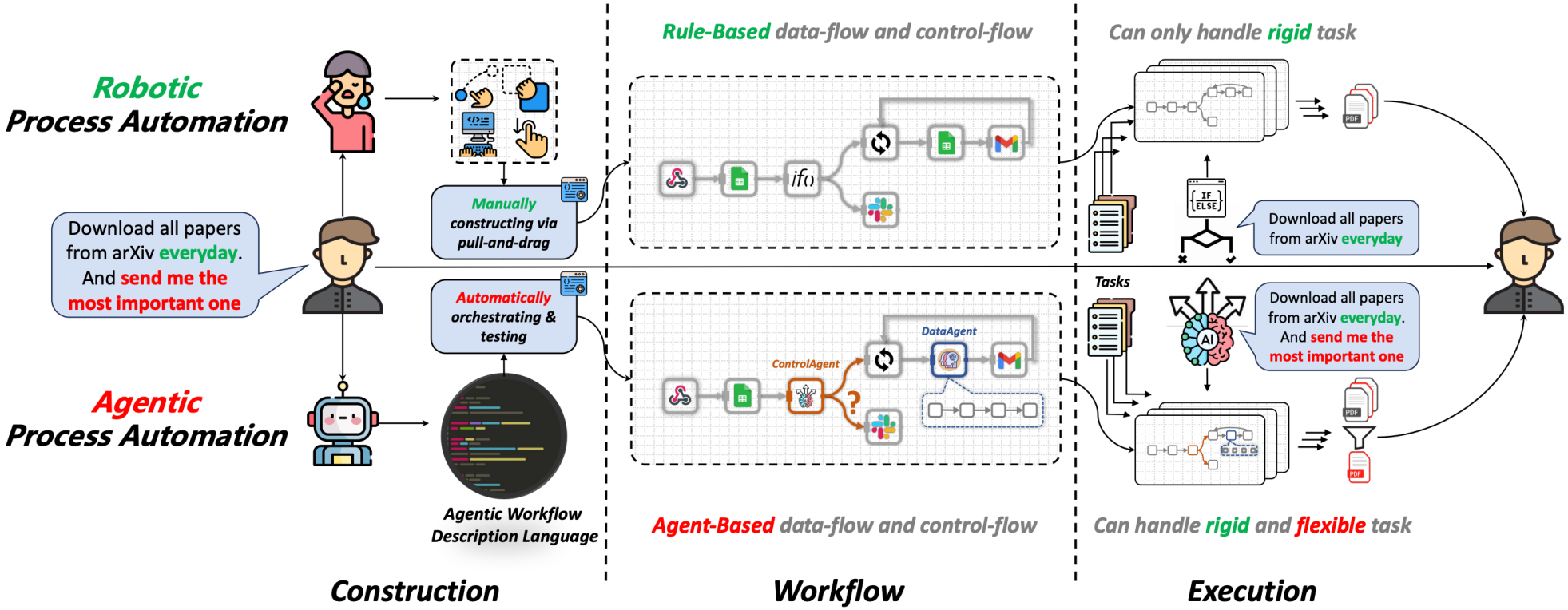
# ProAgent

- Robotic Process Automation (RPA)
  - Involve manually programming rules to coordinate multiple software applications into a solidified workflow. It achieves efficient execution by interacting with software in a manner that simulates human interaction.



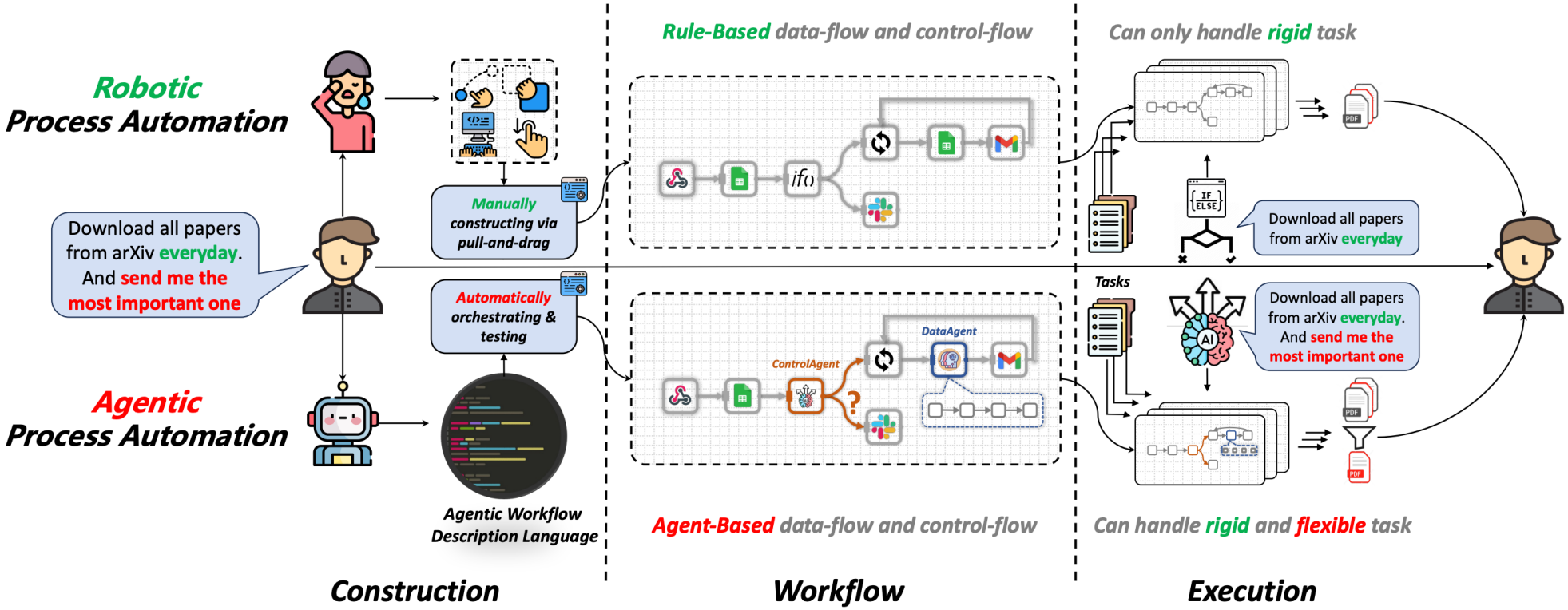
# ProAgent

- Limitation of RPA
  - Constructing RPA workflows requires **substantial human labor**
  - Complex tasks are very flexible, involving **dynamic decision-making**, and are difficult to solidify into rules for representation



# ProAgent

- Agentic Process Automation based on LLM-based Agent
  - The agent **autonomously completes the construction of workflows** with human needs
  - **Dynamically recognizing decision-making** during the build and **actively taking over to complete complex decisions** during execution.



# Example

**Task**

When I send a worksheet of business lines through Web, deal with them according to which type of each business line belong to.

1. To-Customer: Send a message to Slack to report the profits of business lines.
2. To-Business: Write a report which should analyze the data to give some suggestions and then send it to the Gmail of the corresponding managers.

# Example

When I send a worksheet of business lines through web, split them by To-Business and To-Customer.
1. Send slack of profits of To-Business lines
2. Write and send Gmail of To-Customer lines

**Action Define**
Thought: I will first define a trigger and see the input schema, then add a Google Sheet node to read
WebhookTrigger(Comments: check the trigger input schema)

```
def WebhookTrigger(input_data):
  """comments:
  1.check the trigger input schema
  """
  return

def mainWorkflow(trigger_input):
  raise NotImplementedError
```

{"decision": ["toC", "toC", "toB"]}
Execution Success

Webhook

**Action Implement**
Thought: Sence To-Customer or To-Business can be classified based on rules, I wll use ControlAgent.
DecideBusinessLine("decide Whether the business line is toC or toB", ["toC", "toB"] )

```
def WebhookTrigger(input_data):
  ...
def GoogleSheetRead(input_data):
  ...
def DecideBusinessLine(line_data):
  """comments: ...
  """
  task = "decide Whether the business line is toC or toB"
  options = ["toC", "toB"]
  return ControlAgent(task, options, line_data)

def mainWorkflow(trigger_input):
  ...
```

Not Implemented Error

Google Sheet
Webhook
ControlAgent

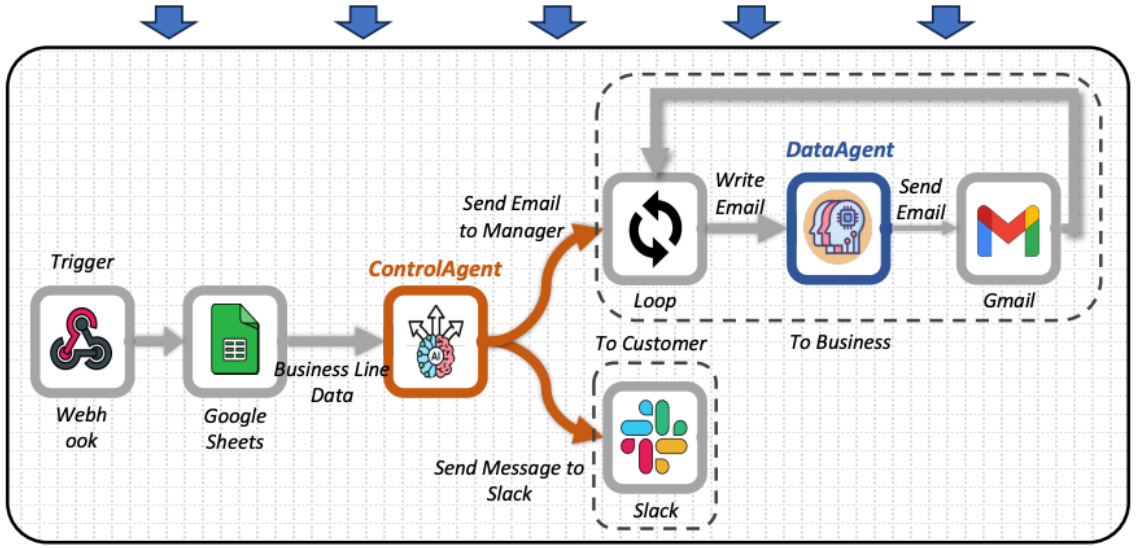
**Action Implement**
Thought: All business line has to write a own email, I can only put an DataAgent.
WriteEmail("Write a email of the business line of profit, together with your suggesiton")

```
def WebhookTrigger(input_data):
  ...
def GoogleSheetRead(input_data):
  ...
def DecideBusinessLine(line_data):
  ...
def WriteEmail(input_data):
  task = "Write a email of the business line of profit,
together with your suggesiton"
  return DataAgent(task, input_data)

def mainWorkflow(trigger_input):
  ...
```

{"email_content": "I'm glad to inform you that . . ."}
Execution Success

DataAgent
Google Sheet
Webhook
ControlAgent
Slack

**Task Submit**
Workflow Description: First get sheet id from webhook, then open the sheet and read info. Automatically select business line and send slack or (newly generated) email.



# Reading Material

## Tool Learning

- **Must-read Papers**

· Tool Learning with Foundation Models. [link]

· Augmented Language Models: a Survey. [link]

· Foundation Models for Decision Making: Problems, Methods, and Opportunities. [link]

- **Further Reading**

· Toolformer: Language Models Can Teach Themselves to Use Tools. [link]

· WebGPT: Browser-assisted question-answering with human feedback. [link]

· ReAct: Synergizing Reasoning and Acting in Language Models. [link]

· Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. [link]

· Inner Monologue: Embodied Reasoning through Planning with Language Models. [link]

For reading material recommendation of this course, please refer to our github.


清華大學
TSINGHUA UNIVERSITY
1911


Tsinghua University
NLP
Natural Language Processing

# Q&A

THUNLP