Prospect of Intelligent Agent System Driven by Large Language Models

大模型驱动的智能代理系统展望

Chenyou Fan
South China Normal University
IUCV Alumnus

2023-10-20

Werewolf Game roles:

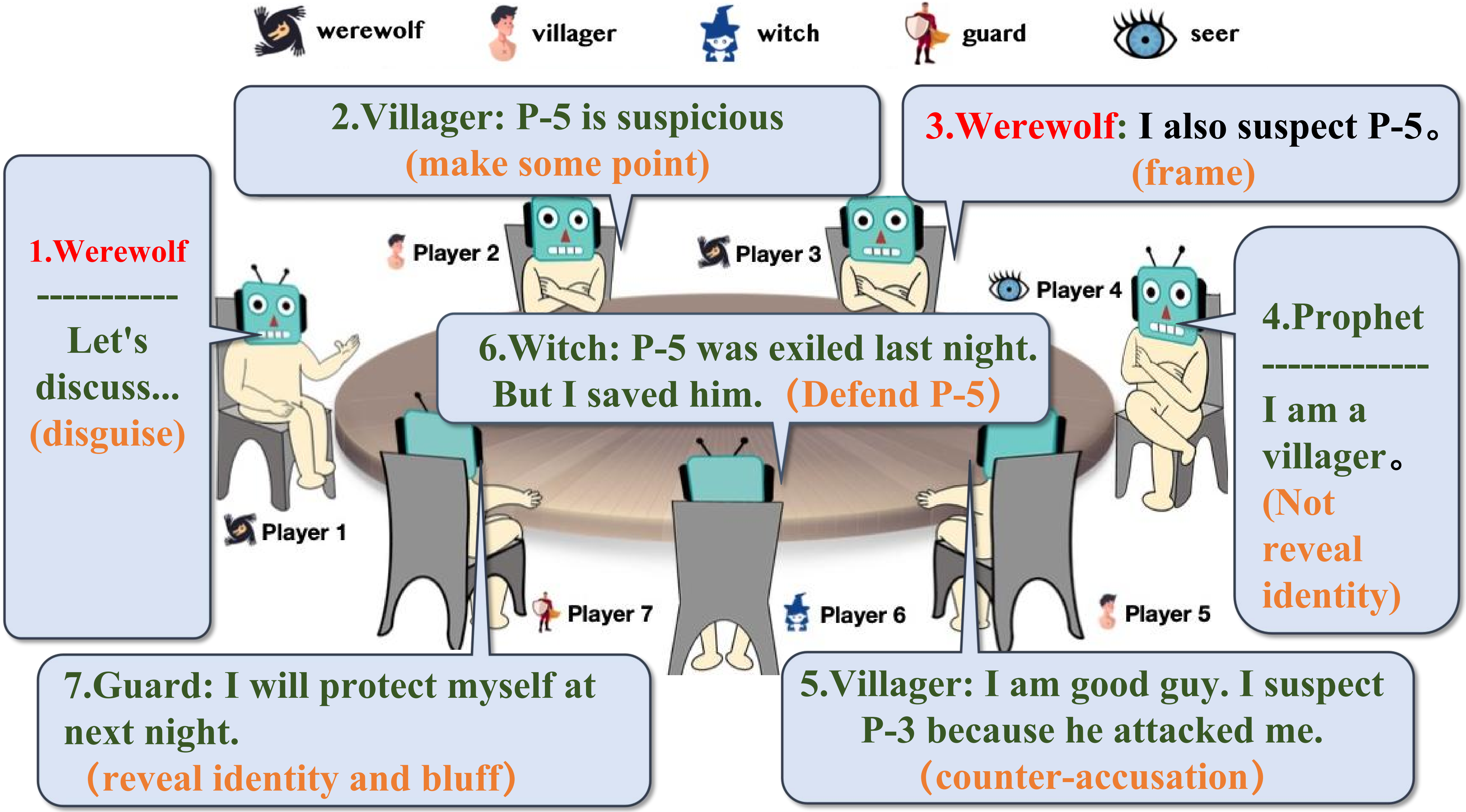
God, Werewolves , Guardian, Witch, Prophet, and Villagers.

- **Each round has two phases: night and daytime.**
- **During the night, Werewolves can choose to kill a player.**
- **The Prophet can verify identities, the Witch can save or exile once, and the Guardian can protect one player.**
- **During the daytime, everyone engages in a round of debate and uses majority voting to exile a player.**
- **Common strategy: the good guys are truthful, while the Werewolves can lie to deceive and frame good guys.**
- **It’s a complicated dialogic game.**

Exploring Large Language Models for Communication Games: An Empirical Study on Werewolf

Yuzhuang Xu[1], Shuo Wang[1], Peng Li[2,*], Fuwen Luo[1]
Xiaolong Wang[1], Weidong Liu[1,3], Yang Liu[1,2,*]
[1]Department of Computer Science & Technology, Tsinghua University, Beijing, China
[2]Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China

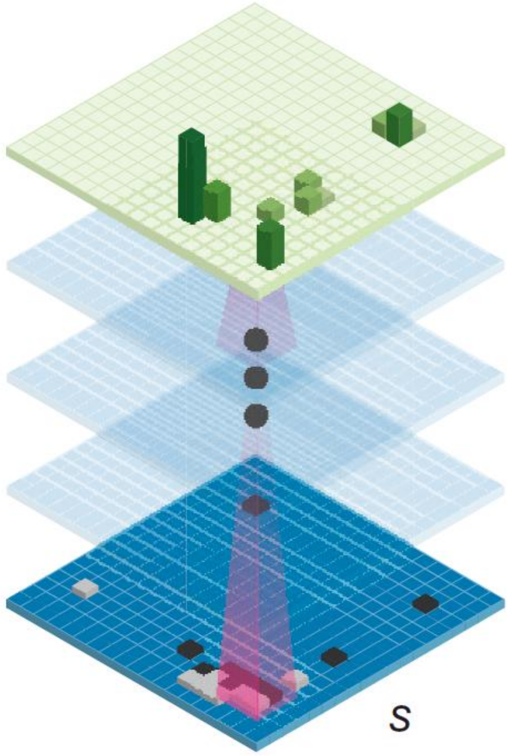
- Xu et al. utilized the LLM to simulate the 8 roles in Werewolf Game.
- The roles can initialize conversations and proceed the game.
- Observe strategic behaviors emerged from LLMs during gameplay such as trust, confrontation, camouflage, and leadership.
- The game is proceeded through automatic prompting, without parameter tuning the LLMs.



Xu et al. Exploring Large Language Models for Communication Games. 2023-09.

Research Motivation of the AI Agent System

- How to simulate **multiple roles** participating in complex tasks?
- How to achieve **cooperation and competition** of agents?
- How to drive agents to **learn task strategies** through dialogue?
- How to induce **reflection and refinement of** their own strategies?
- How to **collect feedback** from the environment or experts?
- How to acquire **external knowledge**?



AI Agent System Overview

Part-1 Planning

Part-2 Memory

Part-3 Tool Use

Part-4 Embody

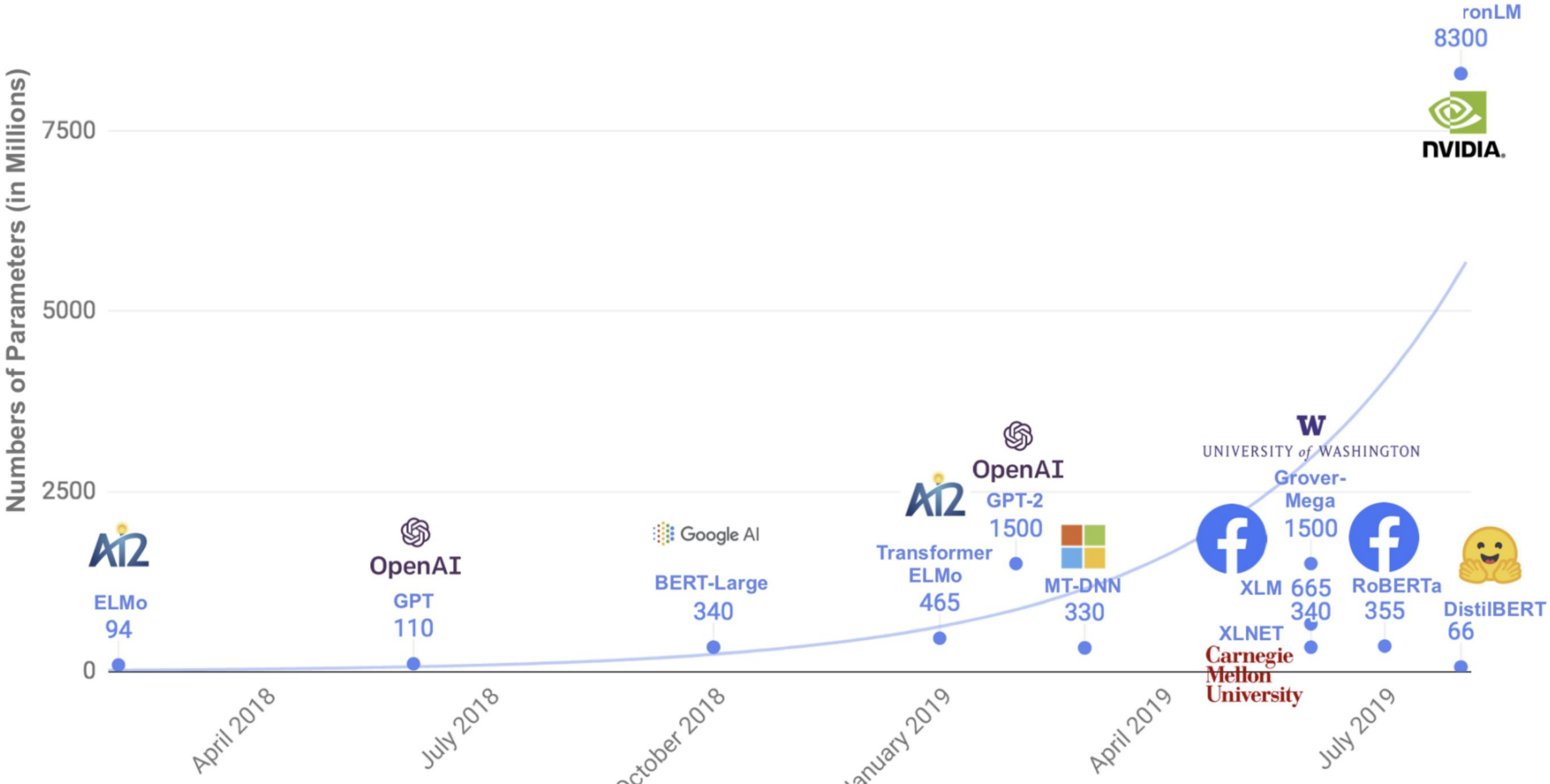
Credit

- This presentation is inspired and re-organized from Lil'Log
- Blog link https://lilianweng.github.io/posts/2023-06-23-agent/
- Lilian Weng is head of AI Safety Team at OpenAI
- She graduated from Indiana University

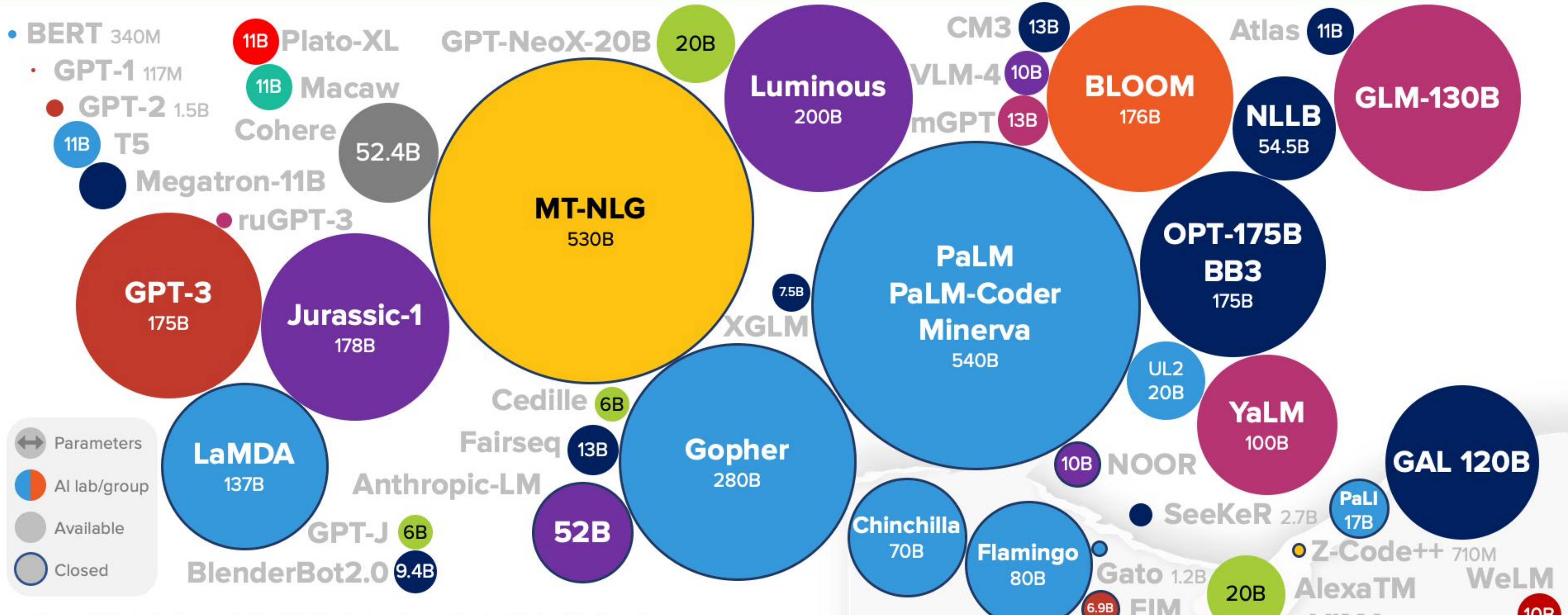
Reference

- LLM Powered Autonomous Agents. https://lilianweng.github.io/posts/2023-06-23-agent/ . Accessed on 2023-09-25.
- Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. NeurIPs 2022.
- Yao et al. *React: Synergizing reasoning and acting in language models*. 2022.
- Shinn et al. *Reflexion: Language Agents with Verbal Reinforcement Learning*. 2023.
- Wang et al. *A Task-Solving Agent through Multi-Persona Self-Collaboration*. 2023.
- ACL 2023 Tutorial: *"Retrieval-based Language Models and Applications."* https://acl2023-retrieval-lm.github.io/ .
- Shi, Weijia, et al. *"Replug: Retrieval-augmented black-box language models."* 2023.
- Shen, Yongliang, et al. *"Hugginggpt: Solving AI tasks with ChatGPT and its friends in huggingface."* 2023.
- Chen, Liting, et al. *"Introspective Tips: Large Language Model for In-Context Decision Making."* 2023.
- Madaan et al. *"Self-refine: Iterative refinement with self-feedback."* 2023.

2017-2023 LLM Parameter Explosion



LANGUAGE MODEL SIZES TO DEC/2022

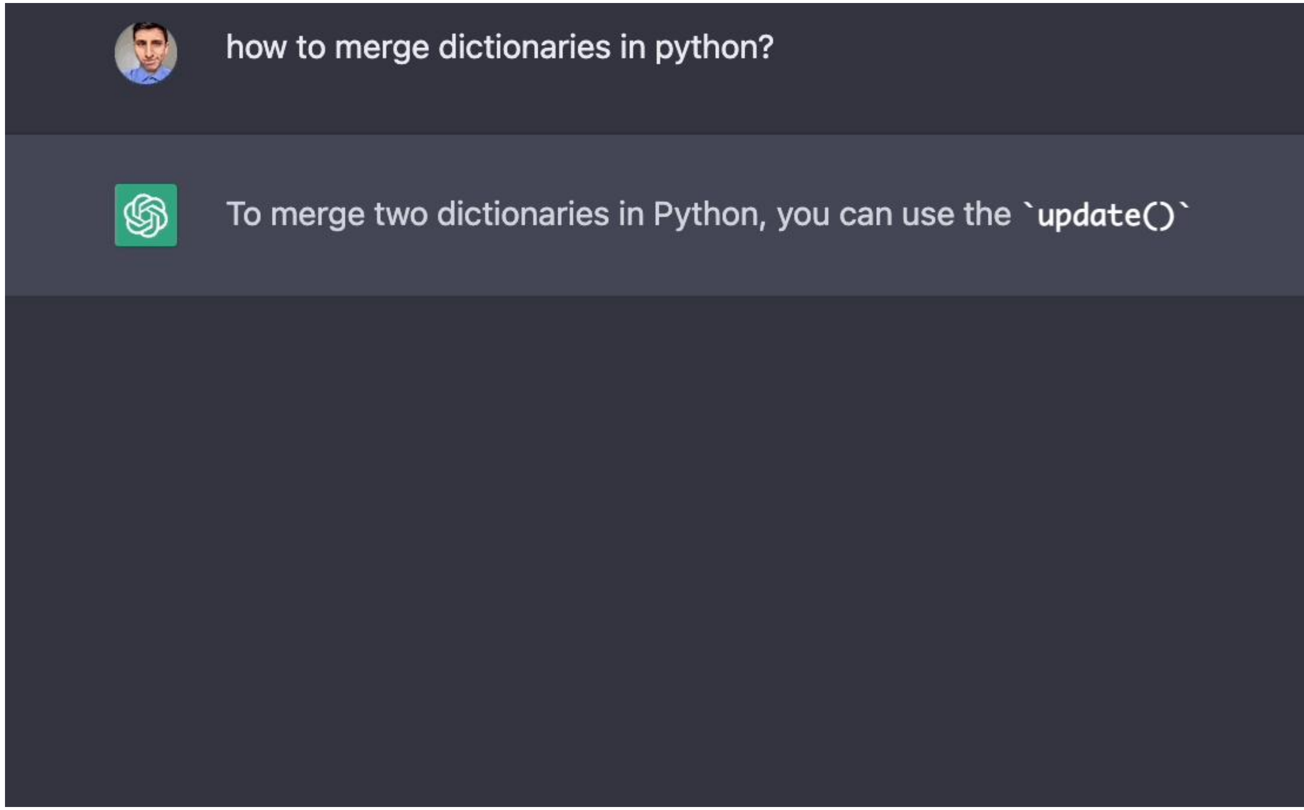


Beeswarm/bubble plot, sizes linear to scale. Selected highlights only. Alan D. Thompson. December 2022. https://lifearchitect.ai/

LifeArchitect.ai/models

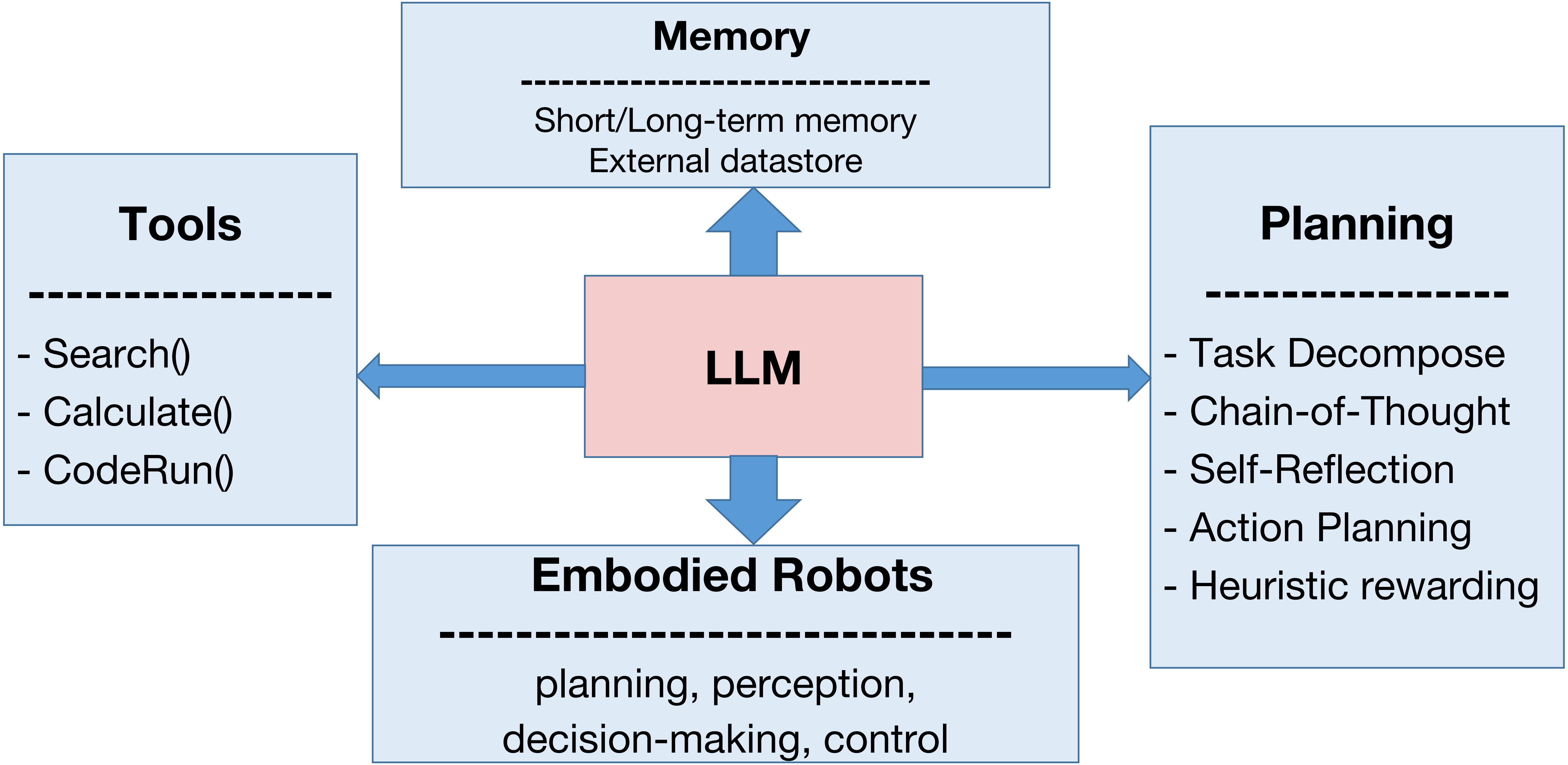
Basic LLM applications

- Text/Code generation
- QA
- Arithmetic reasoning
- Summarization



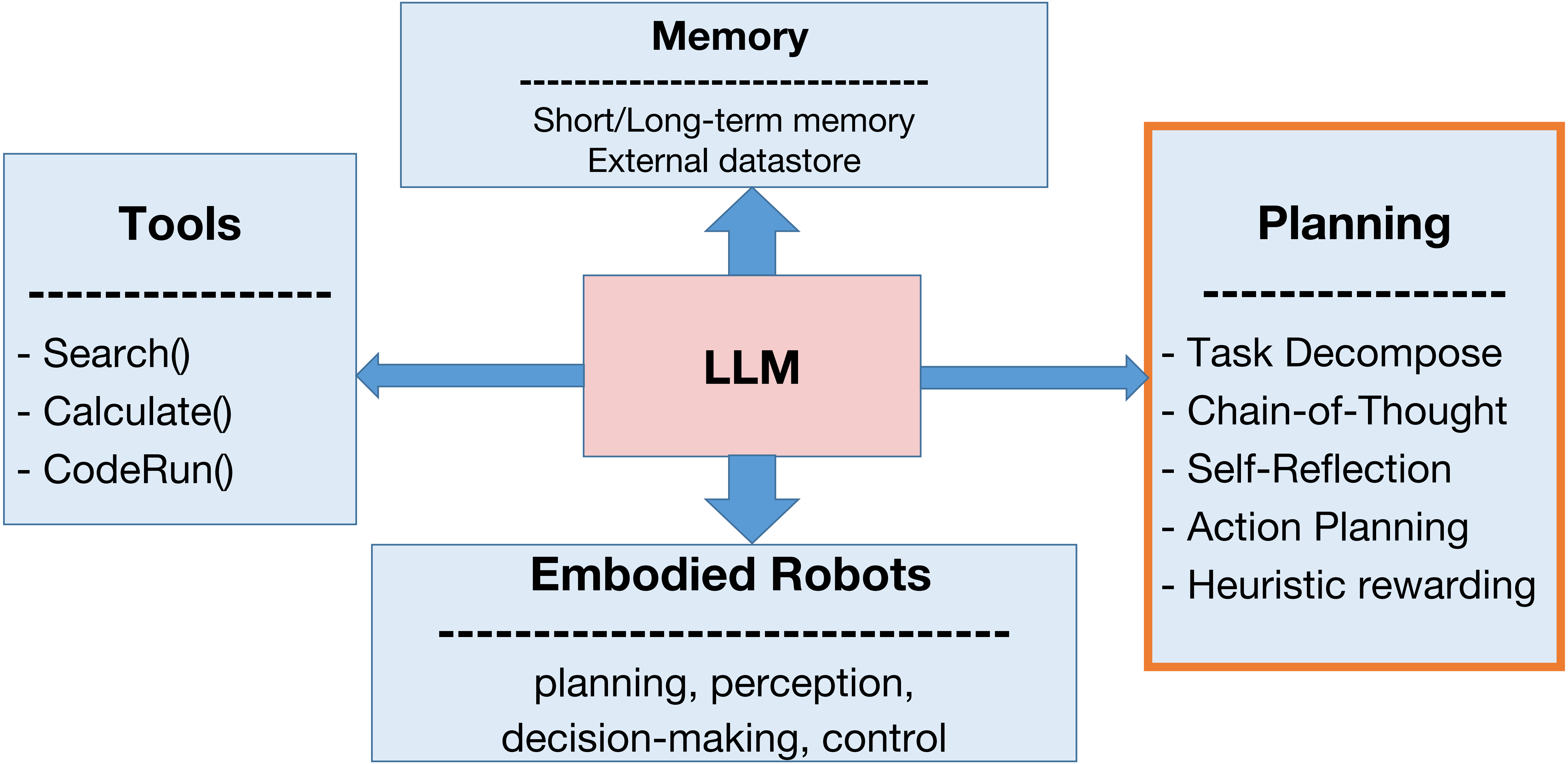
Components of an AI AGENT system

华南师范大学
SOUTH CHINA NORMAL UNIVERSITY



1. **Planning:** The agent breaks down large tasks into sub-tasks, do self-reflection over past actions, and refine future actions.
2. **Memory:** The agent uses short-term mem for in-context learning (e.g., prompt), and long-term mem for retrieving external knowledge.
3. **Tool use:** The agent learns to call external APIs for extra information that is missing from the model weights, e.g., math solver, code execution, query Wiki database.
4. **Embodiment:** Command robots to perform concrete tasks such as cooking and serving.

Components of an AI AGENT system



1. 规划 Planning by AI-Agent

1. **子目标与分解（Subgoal and decomposition）**：代理将大型任务分解为更小、更易于处理的子目标，从而实现对复杂任务的高效处理。
   - 思维链 Chain-of-Thought (CoT; Wei et al. 2022) *# 模仿了逐步思考的过程来得出正确结论*
   - 思维树 Tree-of-Thought (ToT; Yao et al. 2023) *# 结合多条推理路径，完善思考过程*
   - 思维算法Algorithm-of-Thought (AoT; Sel et al. 2023) *# 模拟DFS、BFS查找最优思考过程*
2. **反思与完善（Reflection and refinement）**：对过去的行动进行自我批评和反思，纠正以前的错误来迭代地改进。
   - 思考行动 ReAct (Yao et al., 2022) *# 使用 WikiAPI 进行搜索，辅助思考*
   - 内省 Introspective Tips (Chen et al., 2023) *# 从过去的行动轨迹，学习到经验技巧*
   - 多代理合作竞争狼人杀游戏 (Xu et al., 2023) *# 多身份代理合作竞争*

1. Planning by AI-Agent

1. **Subgoal decomposition**: break complicated tasks into sub-tasks which can be solved efficiently.
   - Chain-of-Thought (CoT; Wei et al. 2022)
   - Tree-of-Thought (ToT; Yao et al. 2023)
   - Algorithm-of-Thought (AoT; Sel et al. 2023)

2. **Reflection and refinement**: do self-reflection over past actions, learn from mistakes and refine them for future steps
   - ReAct (Yao et al., 2022) *retrieve Wikipedia to support reasoning*
   - Introspective Tips (Chen et al., 2023) *learn tips from past trajectories*
   - WereWolf Game (Xu et al., 2023) *multi-role cooperation*

Chain / Tree-of-Thought 思维链，思维树

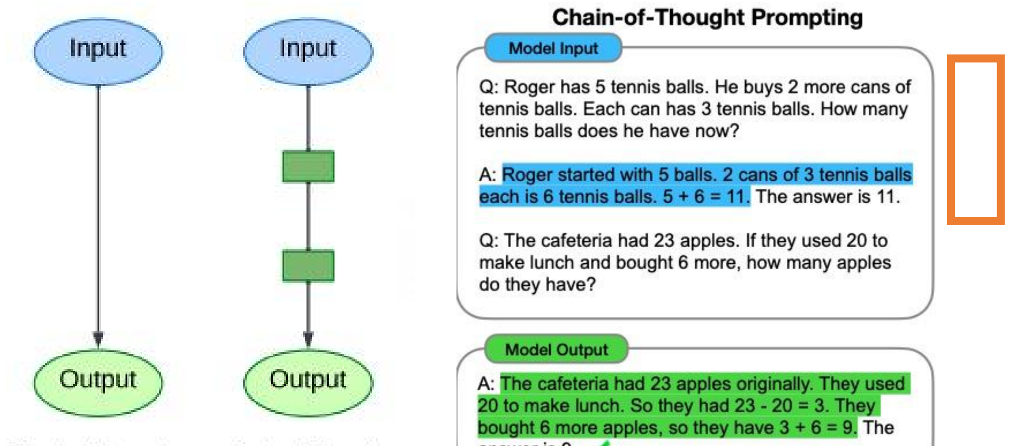
CoT prompts “Think step-by-step”

- In your prompt， always add “Let's think step by step” (ZeroShot-CoT)
- Or, you manually provided a step-by-step instruction template, prepending to your real question (FewShot-CoT)

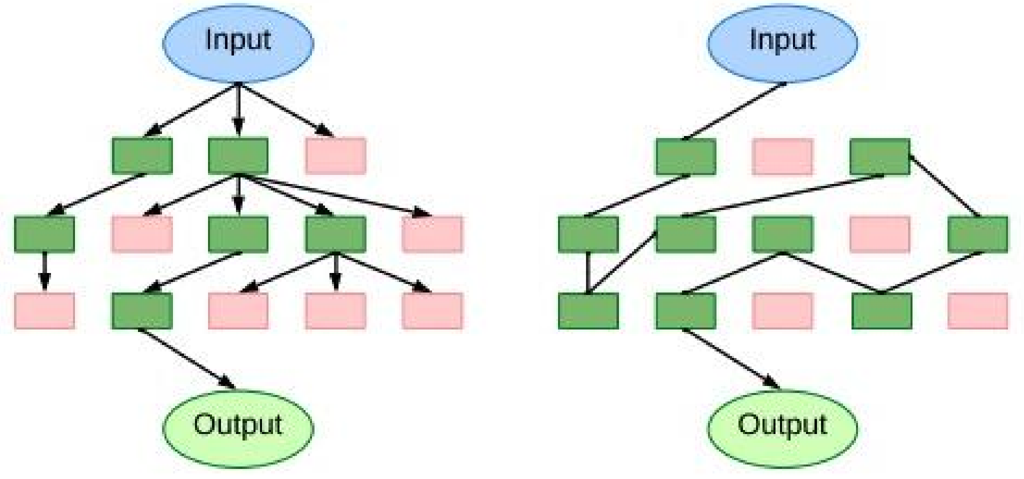
ToT generates multiple thinking routes.

At each intermediate step, use majority voteing to find best CoT.

AoT uses BFS / DFS to search entire state space to find best route.



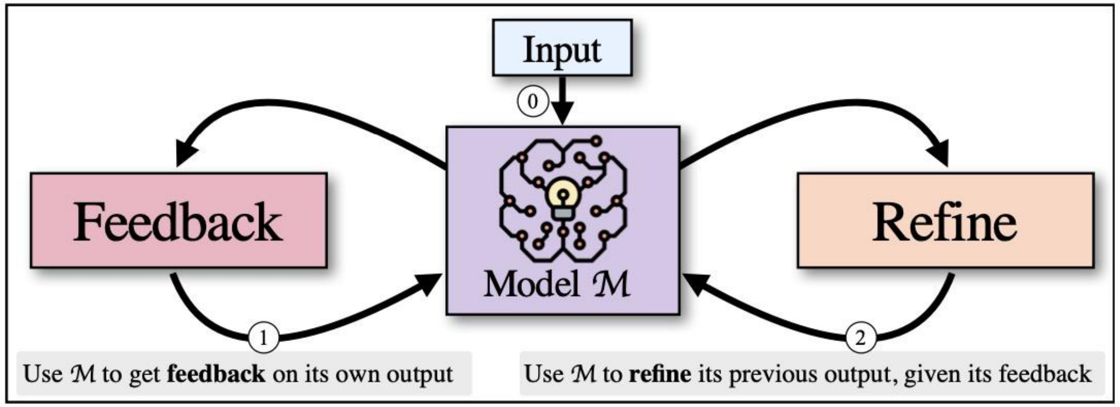
Standard Prompting Chain of Thoughts



Tree of Thoughts Algorithm of Thoughts

Reflection and refinement

- Self-Refine (Madaan et al., 2023) proposes improving initial responses from LLMs through iterative feedback and refinement.



(d) **Code optimization:** x, $\mathbf{y_t}$

```
Generate sum of 1, ..., N
def sum(n):
    res = 0
    for i in range(n+1):
        res += i
    return res
```

(e) FEEDBACK **fb**

```
This code is slow as
it uses brute force.
A better approach is
to use the formula
... (n(n+1))/2.
```

(f) REFINE $\mathbf{y_{t+1}}$

```
Code (refined)

def sum_faster(n):
  return (n*(n+1))//2
```

Reflection and refinement

ReAct (Yao et al., 2023)

- generates reasoning and actions alternately for QA tasks
- reasoning (e.g. chain-of-thought prompting) helps update action plans
- acting (e.g. action plan generation) interfaces external sources such Wiki by using search API

Stage	Description
Question:	Is Avatar an American film made in 2010?
Thought:	I need to search for Avatar and find if it is an American film made in 2010.
Action:	Search Avatar in Wiki database. Return results.
Thought:	The result says that it is an American film made in 2009, so it is not made in 2010.
Action:	Finish and return NO.

Reflection and refinement

Reflexion (Shinn & Labash 2023)

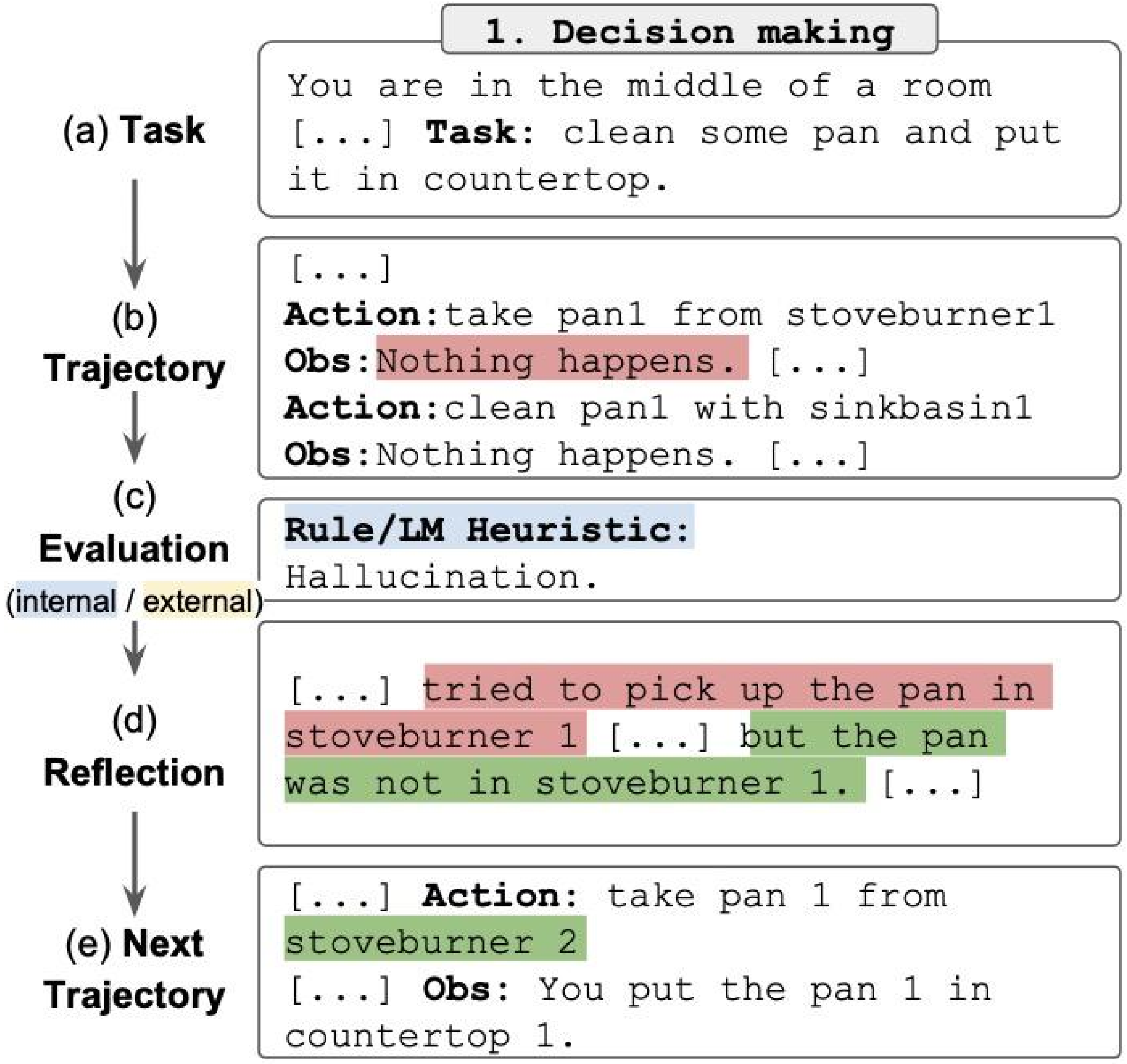
(a) the ReAct generates an action plan

(b) executes the plan and gathers observations

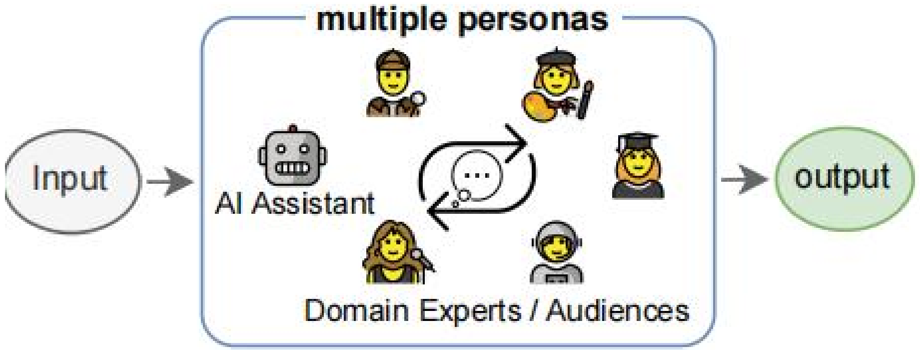
(c) Evaluator generates a binary reward

(d) LLM generates specific feedback given reward

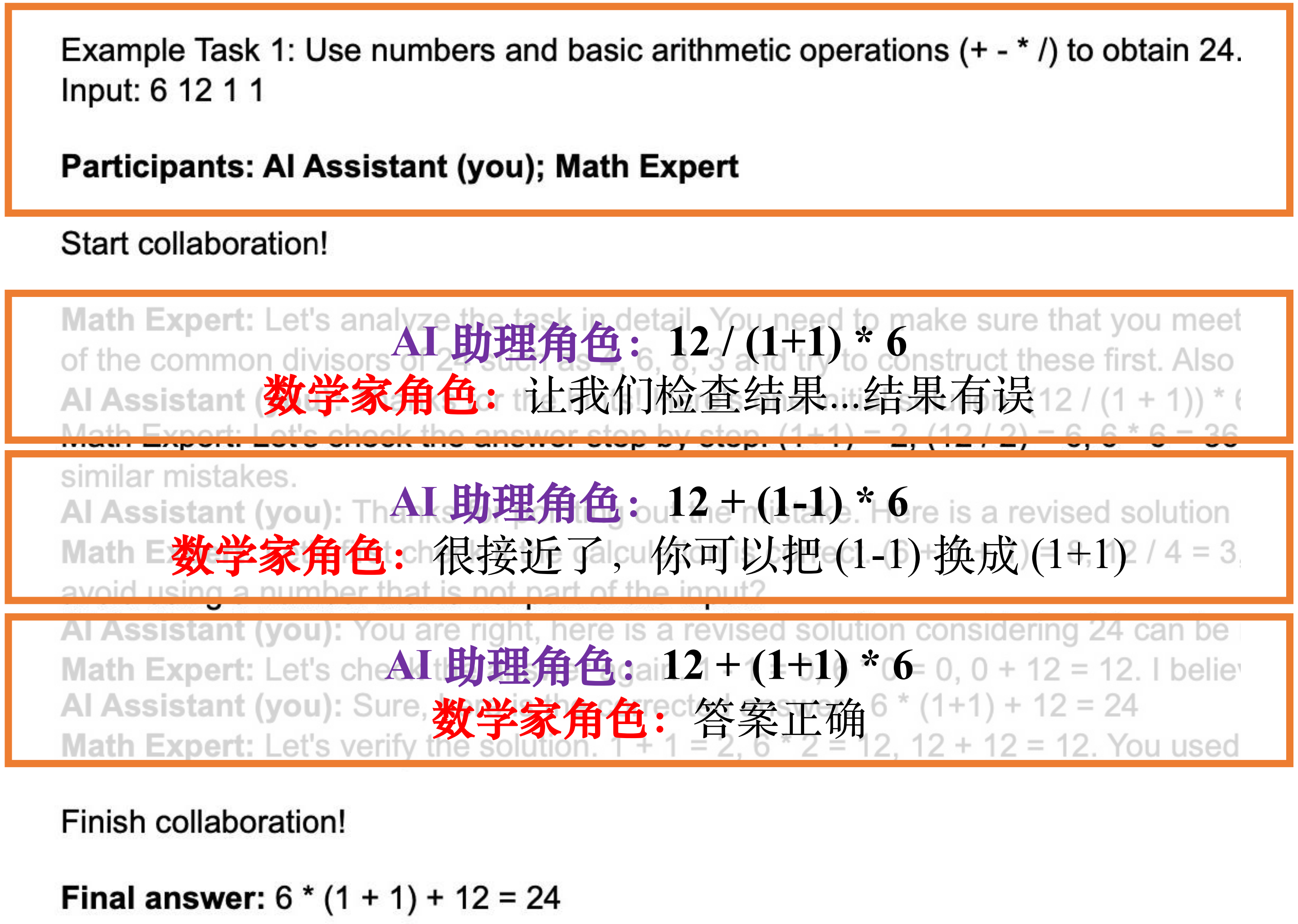
(e) Memory store its own reflective text in an episodic memory buffer to induce better decision-making in subsequent trials



多角色自我合作（multi-persona self-collaboration）

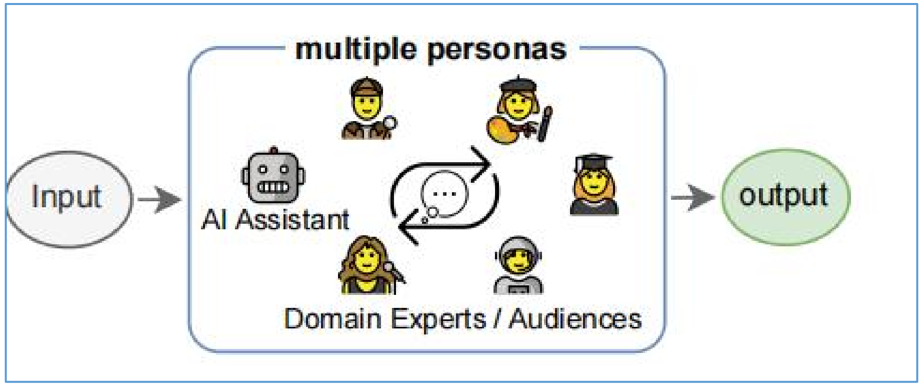


- Wang et al. 提出将单个LLM转变为一个多角色协同认知系统
- 首先，LLM 动态地识别问题中的不同人物角色
- 接着，多角色进行多轮自我协作，直到有效地解决该任务

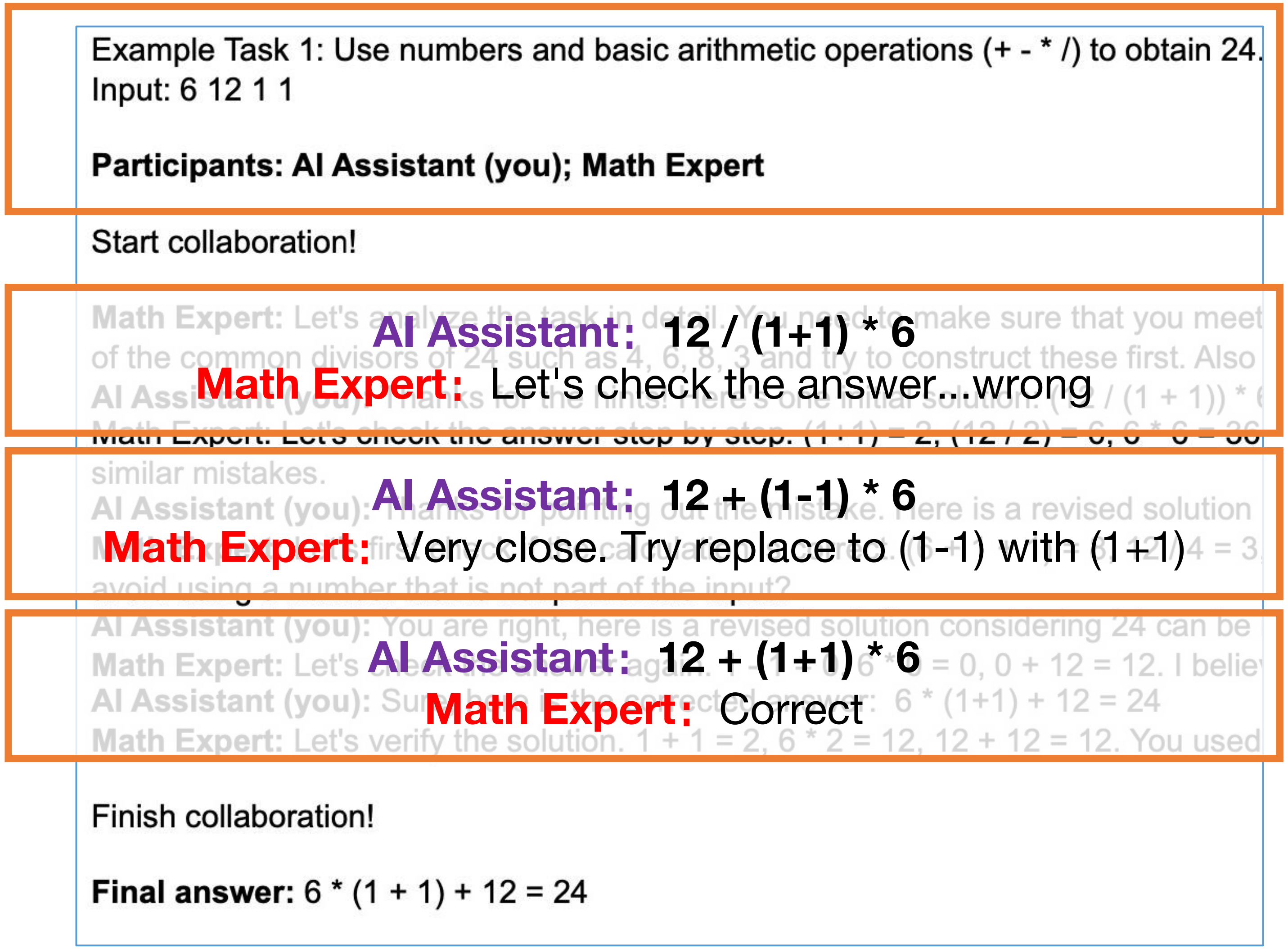


Wang et al. “A Task-Solving Agent through Multi-Persona Self-Collaboration.” 2023.

Multi-persona self-collaboration



- Wang et al. developed a multi-persona **cognitive synergist** based on a single LLM
- The LLM dynamically identifies and simulates different personas given a task
- All roles engage in multi-persona self-collaboration for completing tasks



Wang et al. "A Task-Solving Agent through Multi-Persona Self-Collaboration." 2023.

Multi-persona cooperation and competition

Agent's own context as short-term memory：

- store observs. and reflections of agent i until the speaking time
- take as context for this round of reasoning and speaking

Experience pool as long-term memory：

- we collect and score the pairs of response and reflection from all players at the end of each round
- retrieve from past experiences to get a suggestion which can guide the reasoning at a new round with high scores

Confrontation

Werewolf： P5 is suspicious.

Camouflage

Witch： I am villager.

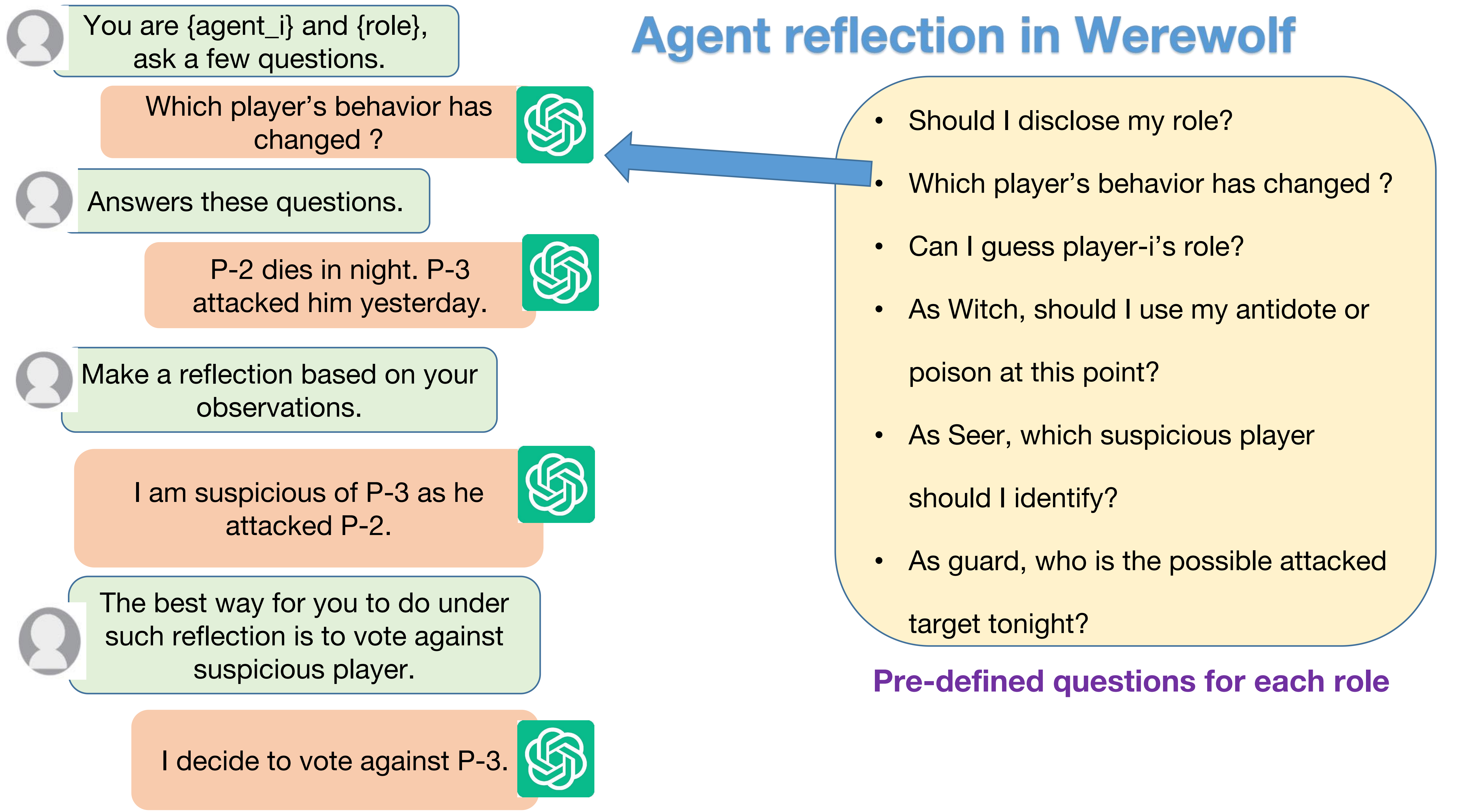
Trust

Villager： I trust P5.

Leadership

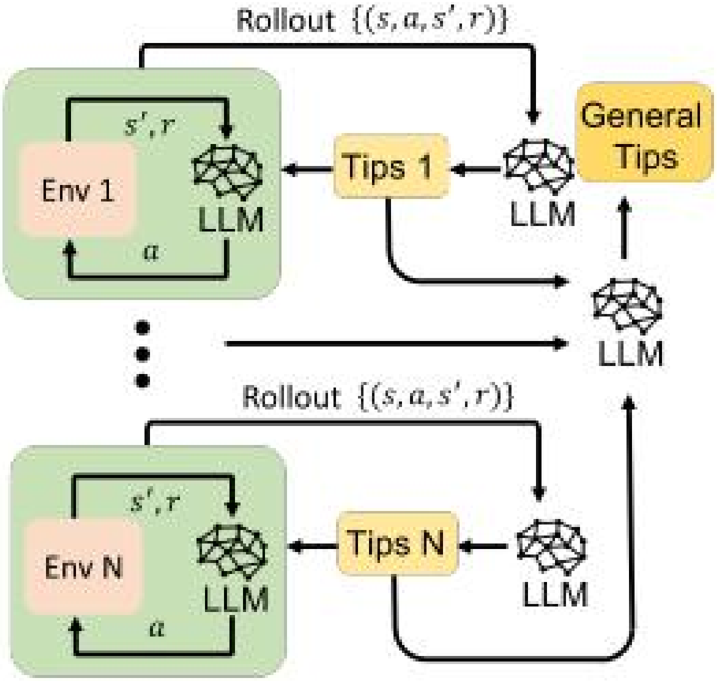
Prophet ： I verify P1 as werewolf. Let's vote againt P1.

Xu et al. Exploring Large Language Models for Communication Games. 2023-09.


Agent reflection in Werewolf
You are {agent_i} and {role}, ask a few questions.
Which player's behavior has changed ?
Answers these questions.
P-2 dies in night. P-3 attacked him yesterday.
Make a reflection based on your observations.
I am suspicious of P-3 as he attacked P-2.
The best way for you to do under such reflection is to vote against suspicious player.
I decide to vote against P-3.
• Should I disclose my role?
• Which player's behavior has changed ?
• Can I guess player-i's role?
• As Witch, should I use my antidote or poison at this point?
• As Seer, which suspicious player should I identify?
• As guard, who is the possible attacked target tonight?
Pre-defined questions for each role

Introspective Tips from Expert Demonstrations

- Agents are allowed to participate in several games
- The LLM generates tips as self-reflection during each game
- All final tips that lead to a successful trial are sent to another Expert-LLM for learning to generate general tips
- The Expert-LLM can be used across different settings



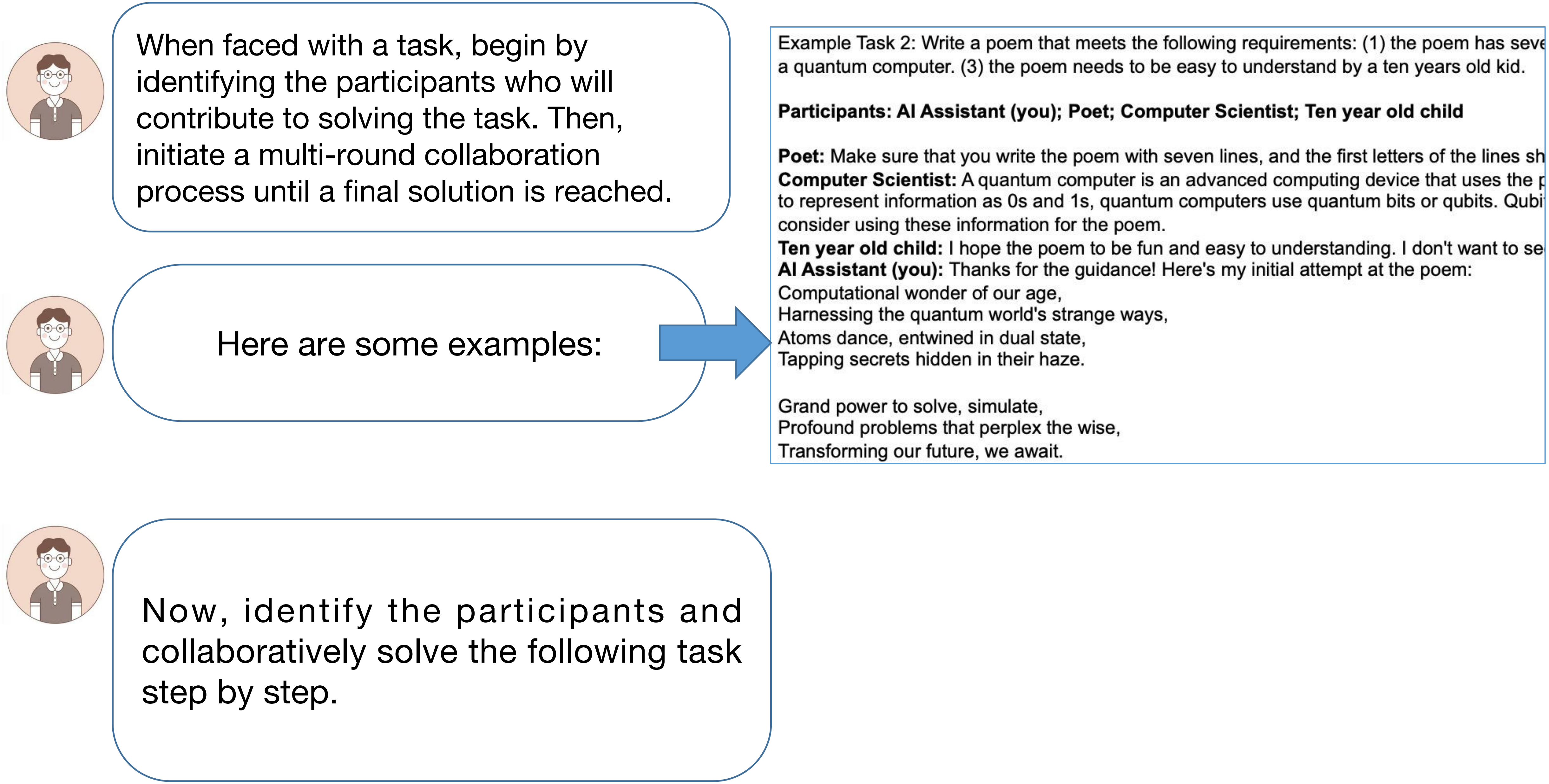
d) Generalized learning: general hint based on N different games

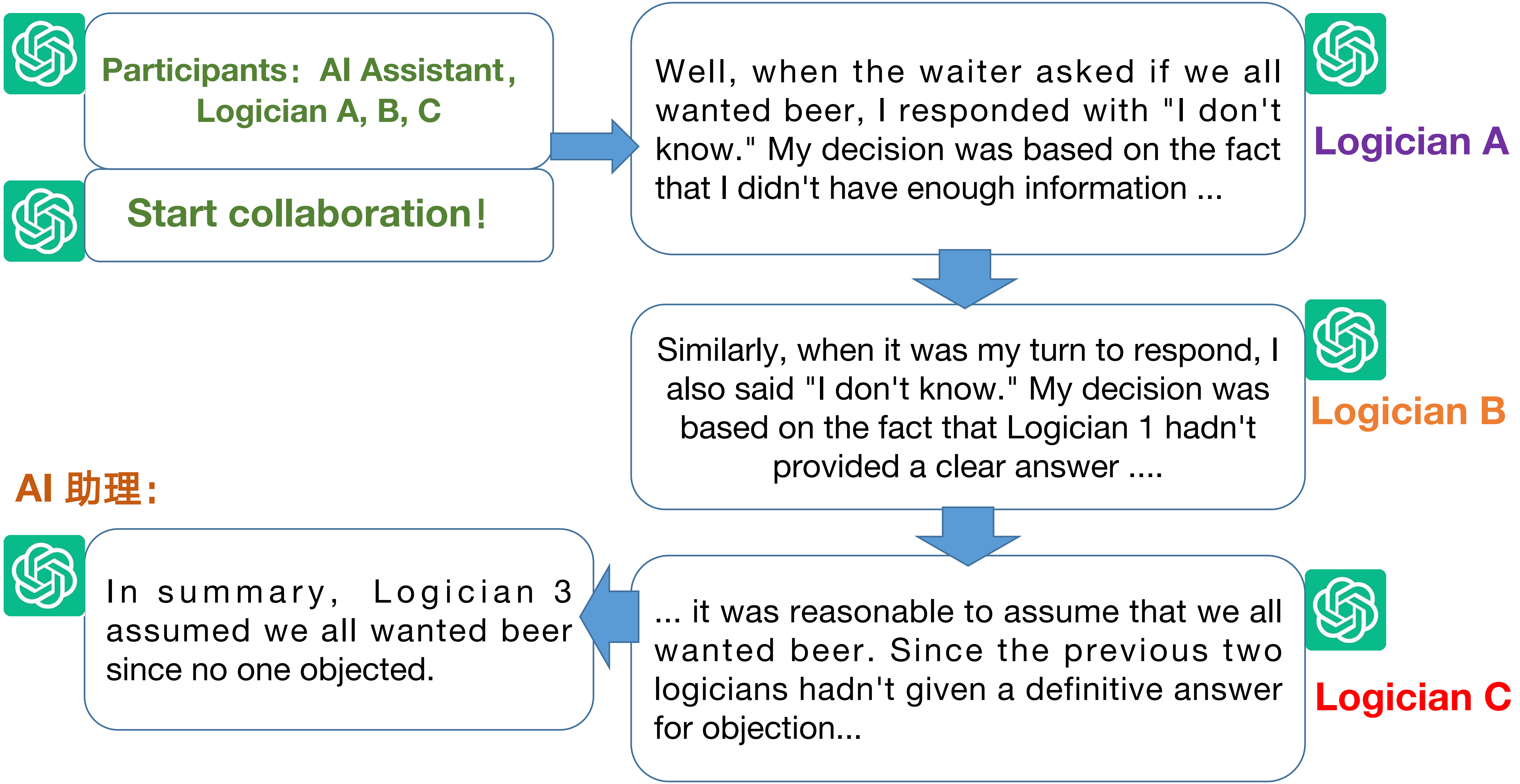
I want you to provide me all possible specific reasons that an agent may fail in a text-based game and summarize all tips for the agent to improve.

Chen, Liting, et al. "Introspective Tips: Large Language Model for In-Context Decision Making." 2023.

Logician Beer Drinking Problem

Three logicians went for a drink.
Waiter asked: 'Do you all want beer?'
A said, 'I don't know'
B said, 'I don't know either'
C said, 'We all need beer.'
Why C is certain about this ?


When faced with a task, begin by identifying the participants who will contribute to solving the task. Then, initiate a multi-round collaboration process until a final solution is reached.
Here are some examples:
Example Task 2: Write a poem that meets the following requirements: (1) the poem has seve
a quantum computer. (3) the poem needs to be easy to understand by a ten years old kid.
Participants: AI Assistant (you); Poet; Computer Scientist; Ten year old child
Poet: Make sure that you write the poem with seven lines, and the first letters of the lines sh
Computer Scientist: A quantum computer is an advanced computing device that uses the p
to represent information as 0s and 1s, quantum computers use quantum bits or qubits. Qubi
consider using these information for the poem.
Ten year old child: I hope the poem to be fun and easy to understanding. I don't want to se
AI Assistant (you): Thanks for the guidance! Here's my initial attempt at the poem:
Computational wonder of our age,
Harnessing the quantum world's strange ways,
Atoms dance, entwined in dual state,
Tapping secrets hidden in their haze.
Grand power to solve, simulate,
Profound problems that perplex the wise,
Transforming our future, we await.
Now, identify the participants and collaboratively solve the following task step by step.



Participants：AI Assistant，Logician A, B, C
Start collaboration!
Well, when the waiter asked if we all wanted beer, I responded with "I don't know." My decision was based on the fact that I didn't have enough information ...
Logician A
Similarly, when it was my turn to respond, I also said "I don't know." My decision was based on the fact that Logician 1 hadn't provided a clear answer ....
Logician B
AI 助理：
In summary, Logician 3 assumed we all wanted beer since no one objected.
... it was reasonable to assume that we all wanted beer. Since the previous two logicians hadn't given a definitive answer for objection...
Logician C

Q:Logician 1, 2, 3 went for a drink, and the waiter asked, 'Do you all want beer?' Logician 1 said, 'I don't know,' Logician 2 said, 'I don't know,' and Logician 3 said, 'We all need beer.' Participants: AI Assistant (you); Logician 1; Logician 2; Logician 3. AI Assistant: Let's begin retrospecting the conversation step by step between logician 1, 2, 3. Tell us why you decided or not decided in the former conversation. Logician 1, you speak first. Then Logician 2 speak. Lastly, Logician 3 speak.

A:Logician 1: Well, when the waiter asked if we all wanted beer, I responded with "I don't know." My decision was based on the fact that I didn't have enough information at that moment to make a definitive choice. Since Logician 2 and Logician 3 hadn't expressed their preferences yet, I couldn't determine if we all wanted beer or not.

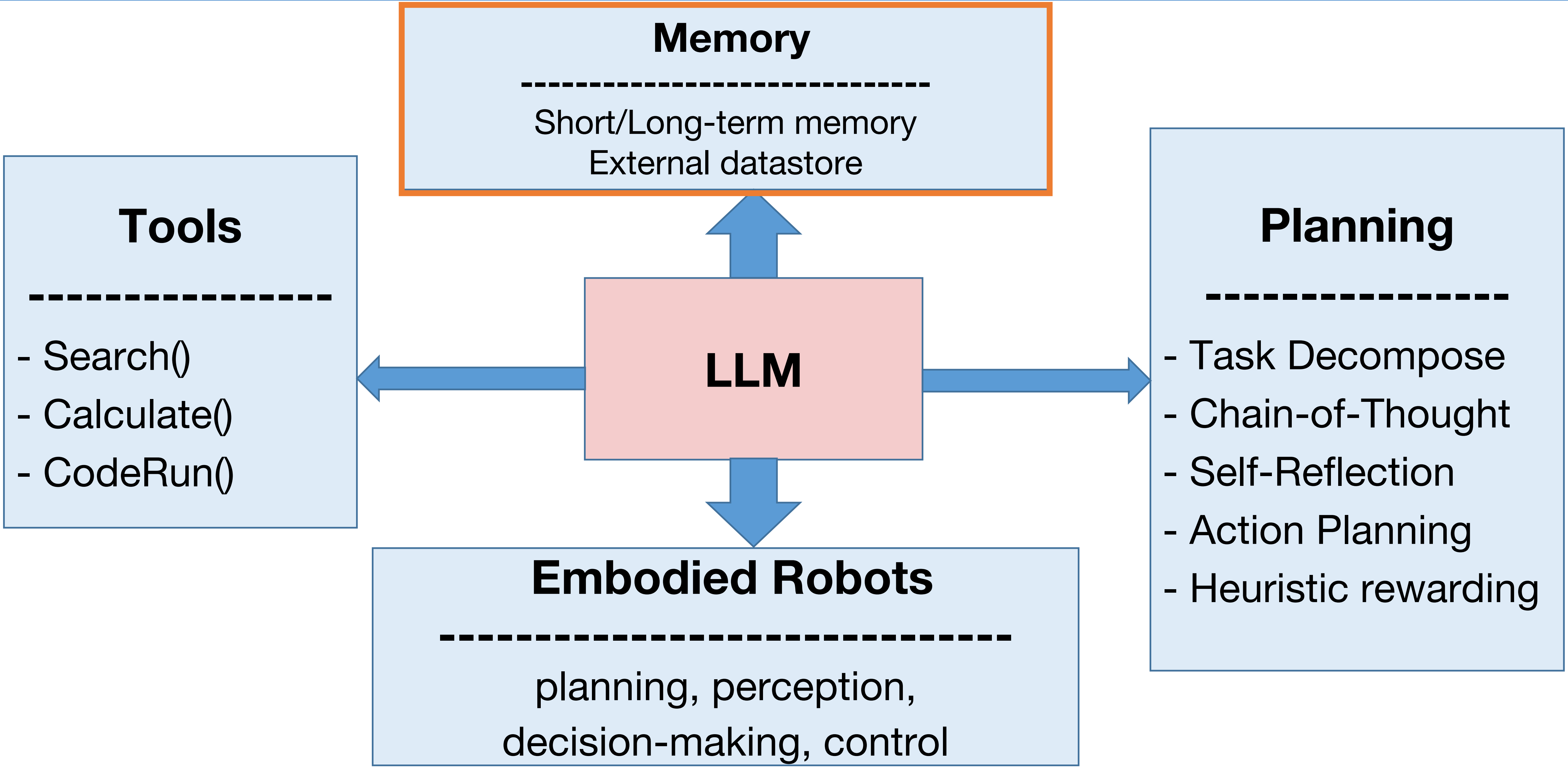
Logician 2: Similarly, when it was my turn to respond, I also said "I don't know." My decision was based on the fact that Logician 1 hadn't provided a clear answer, and I still didn't have enough information to make a definitive choice. I was waiting for Logician 3's response to gather more information.

Logician 3: Lastly, when it was my turn to speak, I said, "We all need beer." My decision was based on the assumption that if Logician 1 and Logician 2 didn't object or express a different preference, then it was reasonable to assume that we all wanted beer. Since the previous two logicians hadn't given a definitive answer, I made the assumption that we all wanted beer.

So, in summary, Logician 1 and Logician 2 were undecided because they didn't have enough information, while Logician 3 assumed that since no one objected, we all wanted beer.

- Logician 1, 2, 3 went for a drink, and the waiter asked, 'Do you all want beer?' Logician 1 said, 'I don't know,' Logician 2 said, 'I don't know,' and Logician 3 said, 'We all need beer.' Participants: AI Assistant (you); Logician 1; Logician 2; Logician 3. AI Assistant: Let's begin retrospecting the conversation above step by step and answer Why Logician 3 said they all need beer? Tell us why you decided or not decided in the former conversation. Logician 1, you speak first. Then Logician 2 speak. Lastly, Logician 3 speak.
- A:Logician 1: Well, when the waiter asked if we all want beer, I said, 'I don't know.' The reason I said that is because I don't have enough information to make a decision yet. It could be that some of us want beer and some don't, or maybe we all want beer. I simply couldn't determine the answer at that time.
- Logician 2: I agree with Logician 1. When the waiter asked if we all want beer, I also responded with 'I don't know.' Like Logician 1 said, we don't have enough information to answer that question definitively. It's possible that some of us want beer and some don't, or perhaps we all want beer. Without any additional details, it's impossible for me to make a decision.
- Logician 3: Well, after hearing what Logician 1 and Logician 2 said, I concluded that we all need beer. Since both Logician 1 and Logician 2 couldn't decide whether they wanted beer or not, it implies that there is no one among us who does not want beer. Therefore, the logical conclusion is that we all need beer.
- AI Assistant: Thank you for your explanations. Based on Logician 3's reasoning, their conclusion is that since neither Logician 1 nor Logician 2 could decide between wanting or not wanting beer, it suggests that everyone in the group wants beer.

Components of an AI AGENT system



2. Memory: store, retain, and retrieve info

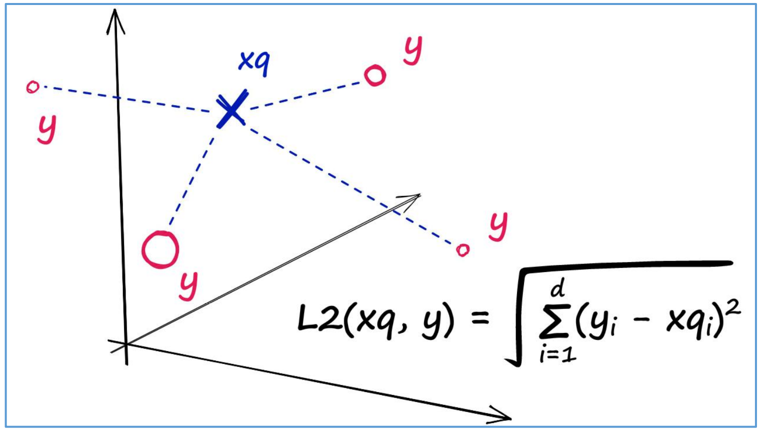
1. Short-Term Memory：

- ✓ facilitates in-context learning such as prompt
- ✓ is restricted by the finite context window length of Transformer-based models
- ✓ e.g., Token limit: ChatGPT <= 4000, GPT-4 <= 32,000

LLM	Token Limit	Estimated word count
GPT4	32,768	25,000
GPT3.5	4,096	3,083
Llama2	2,048	1,563

2. Long-Term Memory:

- ✓ serves as the external vector store that the agent can attend to at query time
- ✓ Knowledge is usually stored as Embedding and Indexed , for fast access and retrieval



A Retrieval-based LM

Typical LMs for fact probing

The capital city of Ontario is **Toronto**

LM

The capital city of Ontario is _____

LM

Test time

A Retrieval-based LM

Retrieval-based LMs

The capital city of Ontario is **Toronto**

LM

Training time

The capital city of Ontario is ____

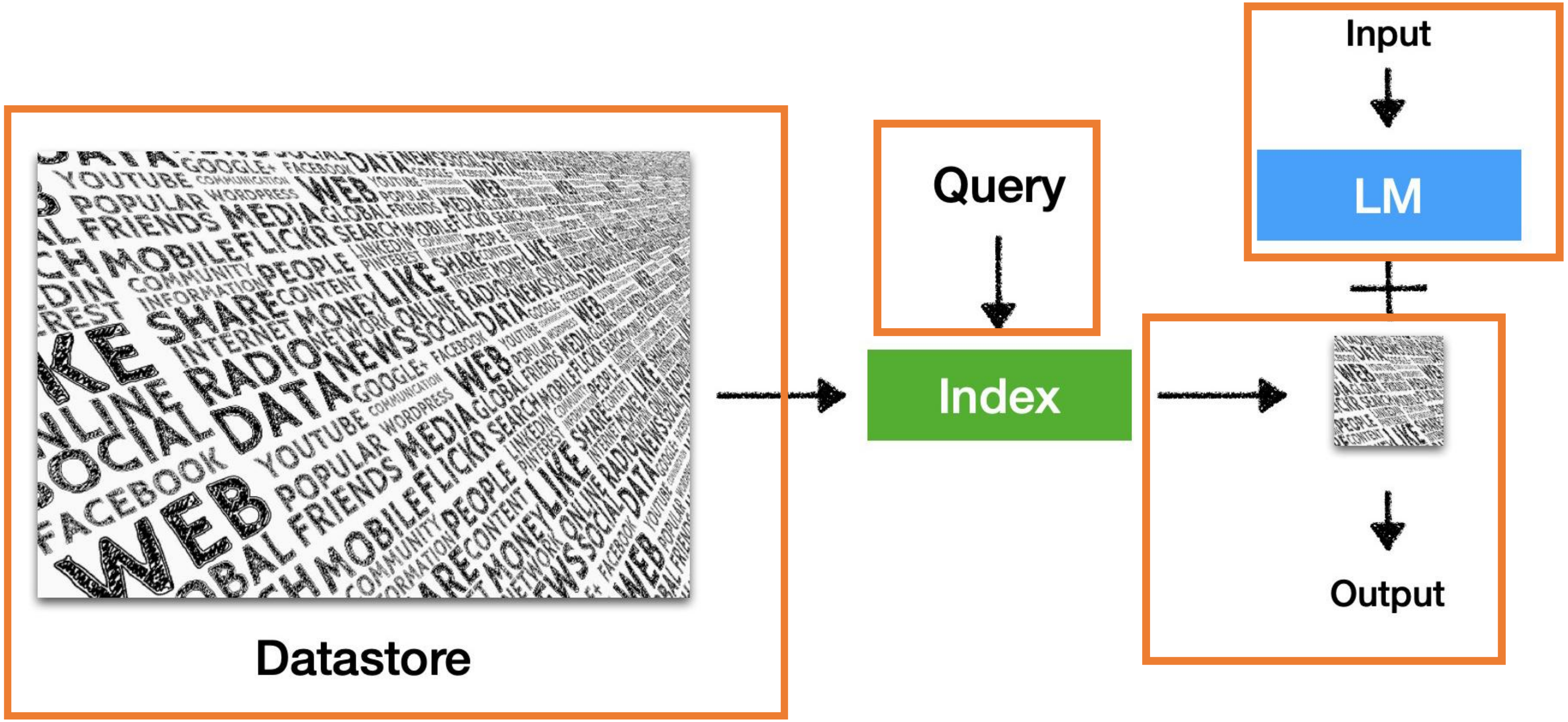
LM

Test time

ACL Tutorial https://acl2023-retrieval-lm.github.io/

A Retrieval-based LM

定义：A language model (LM) that uses external datastore at test time



1. Understand input
2. Generate query
3. **Similarity search**
4. Return answer

Maximum Inner Product Search

- Extract document embedding
- Choose some distance metrics
- Given a query q, use ANN to retrieval similar documents
- Return top-K most similar docs and combine them as external information

sim: a similarity score between two pieces of text

Example $\mathrm{sim}(i,j) = \mathrm{tf}_{i,j} \times \log \frac{N}{\mathrm{df}_i}$

# of total docs

# of docs containing i

# of occurrences of i in j

Example $\mathrm{sim}(i,j) = \mathrm{Encoder}(i) \cdot \mathrm{Encoder}(j)$

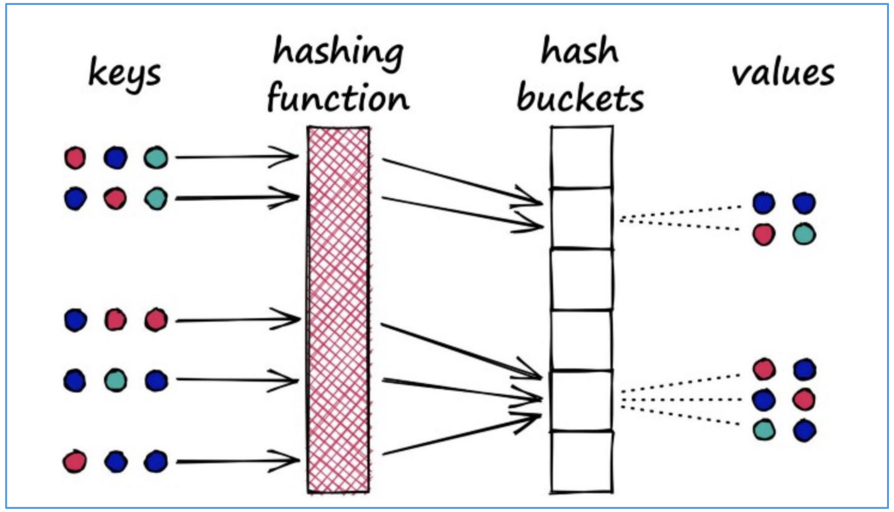
Maps the text into an h-dimensional vector

https://zhuanlan.zhihu.com/p/595249861

Maximum Inner Product Search -- ANN

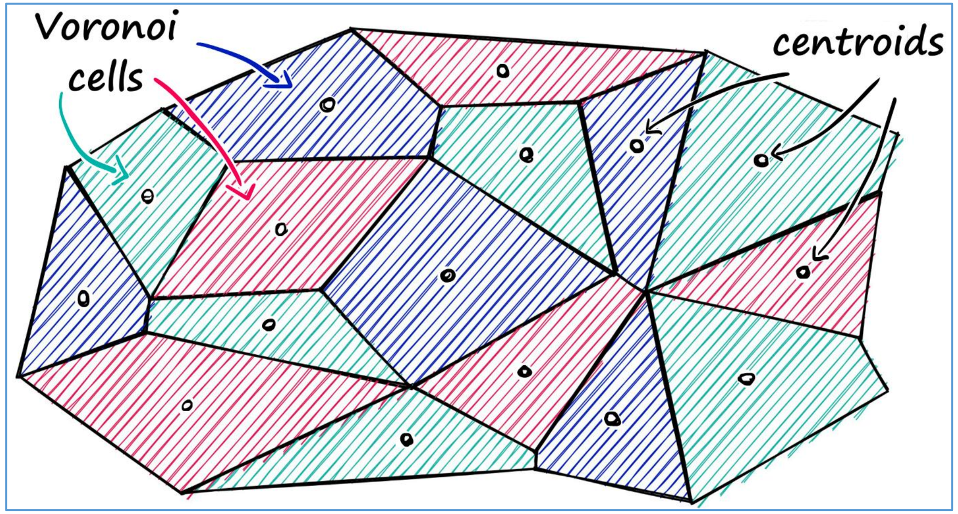
LSH (Locality-Sensitive Hashing)

- similar input items are mapped to the same buckets with high probability



Inverted File Index

- divide space into many cells with their centroids.
- when we introduce a new query vector, we first measure its distance between centroids, then restrict our search scope to that centroid's cell

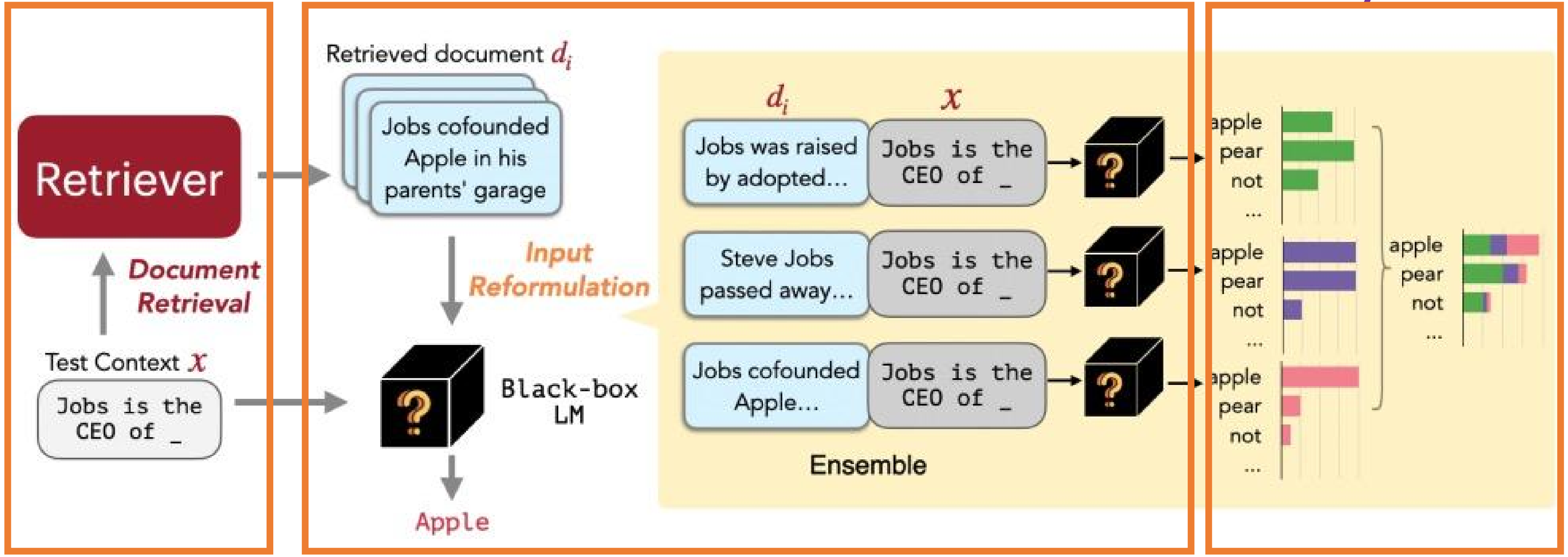


https://www.pinecone.io/learn/series/faiss/

Overview of retrieval process with LLMs

- Retrieve a subset of similar documents
- Combine input with retrieved documents
- Prompt LLM and ensemble final predictions

- Train retriever to update embedding and doc indices
- Minimize KL(embedding similarity | LLM scoring on relevance)



Shi, Weijia, et al. "*Replug: Retrieval-augmented black-box language models.*" 2023

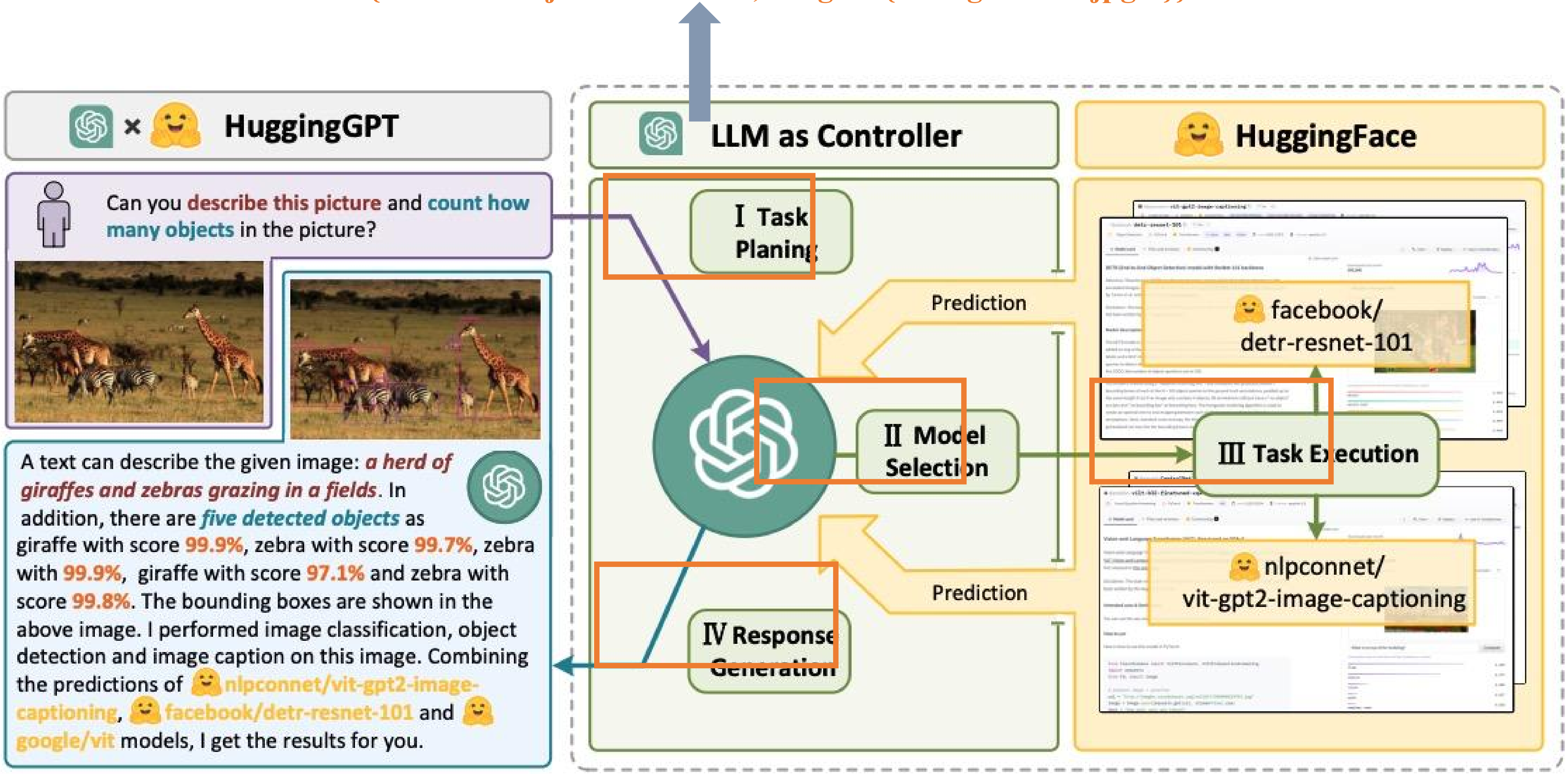
3. Tool Use

The agent learns to call external APIs for extra information that is missing from the model weights, e.g., get news, code execution, math solver, etc.

- **TALM（Tool Augmented Language Models; Parisi et al. 2022， Schick et al. 2023）** learns to read external API docs and choose to use proper API
- **HuggingGPT（Shen et al. 2023）** chooses proper model in HuggingFace, loads model, executes the model inference, and generates responses.

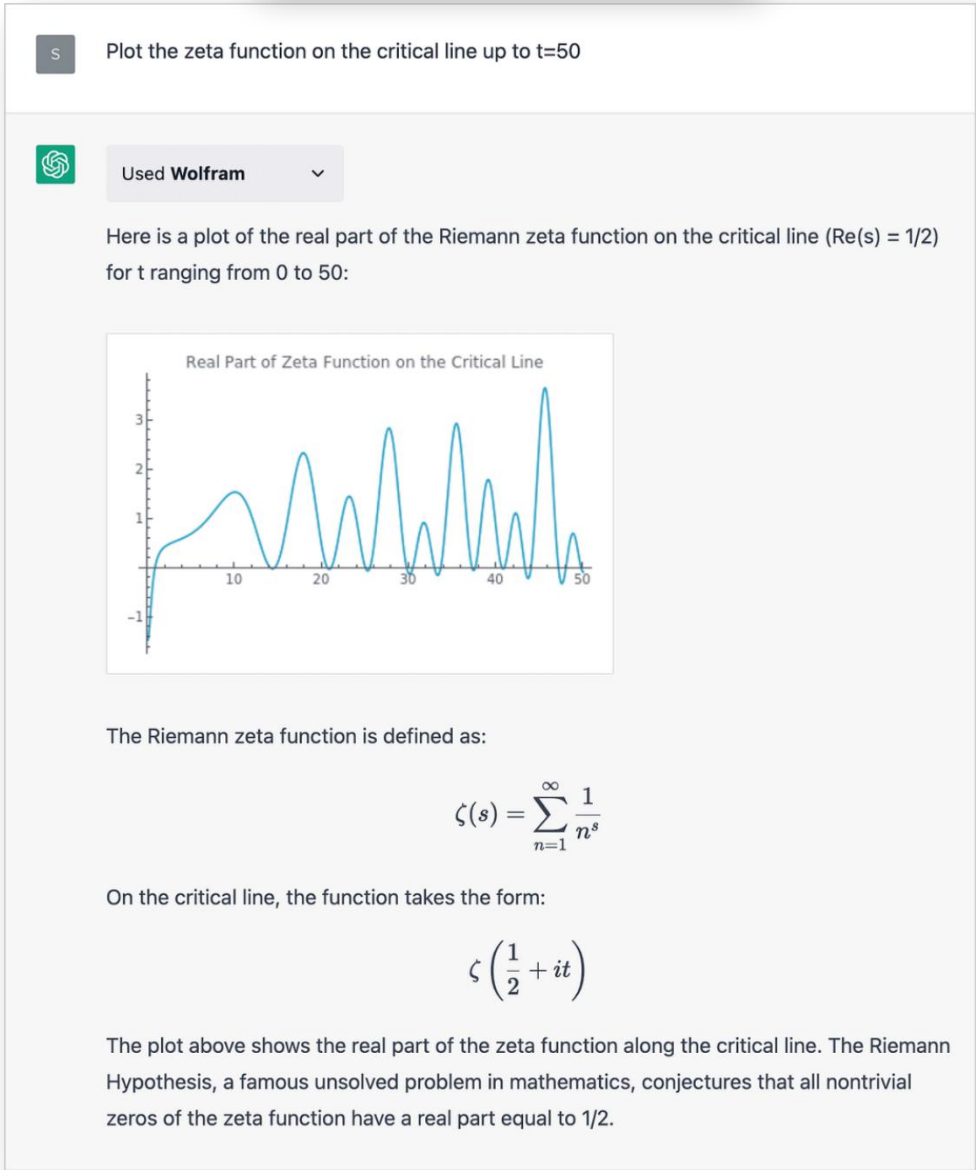
Q: Can you tell me how many objects in e1.jpg?

A: {"task": "object-detection", "args": {"image": "e1.jpg" }}




GitHub Copilot
Visual Studio Code


def common_prefix(a, b) :
"""Return the common prefix of two lists."""
if len(a) < len(b) :
return common_prefix(b,a)
for i in range(len(a)) :
if a[i] != b[i] :
return a[:i]
return a


S
Plot the zeta function on the critical line up to t=50
Used Wolfram
Here is a plot of the real part of the Riemann zeta function on the critical line (Re(s) = 1/2) for t ranging from 0 to 50:
Real Part of Zeta Function on the Critical Line
3
2
1
-1
10
20
30
40
50
The Riemann zeta function is defined as:
$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$
On the critical line, the function takes the form:
$\zeta\left(\frac{1}{2} + it\right)$
The plot above shows the real part of the zeta function along the critical line. The Riemann Hypothesis, a famous unsolved problem in mathematics, conjectures that all nontrivial zeros of the zeta function have a real part equal to 1/2.


OpenAI Codex Model
GitHub
Private Code
GitHub Copilot
Service
Provide Editor context
Provide Suggestions
Improve Suggestions
Public code and text
on the internet
fetch_pic.js
push_to
const fetchNASAPictureOfTh
return fetch('https://ap
method: 'GET',
headers: {
'Content-Type': 'app
},
})
.then(response => resp
.then(json => {
return json;
});
}
Copilot

4. 具身代理 Embodied Agent

LLMs accomplish realistic tasks through robotic planning, controling and feedbacking from interaction of the environment。

- **Embodied Reasoning through Planning (Huang et al. 2022)** leverages environment feedback to facilitate LLMs to form an inner monologue for robotic control .
- **Grounding Language in Robotic Affordances（Ahn et al. 2022）** proposed to prompt the LLM to generate low-level action candidates given a high-level command. Then it grounds LLMs through value functions – affordance functions that capture the likelihood that a particular skill will be able to succeed in the current state.

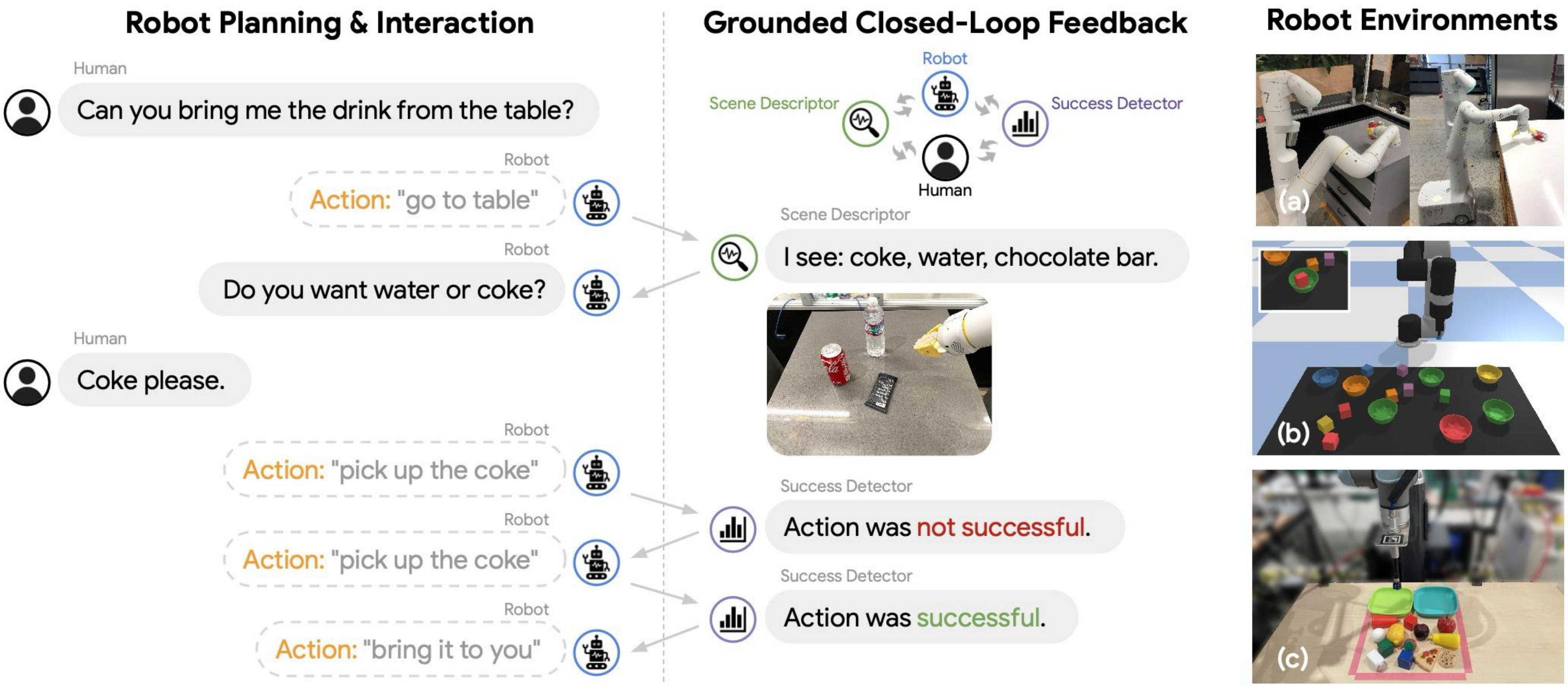
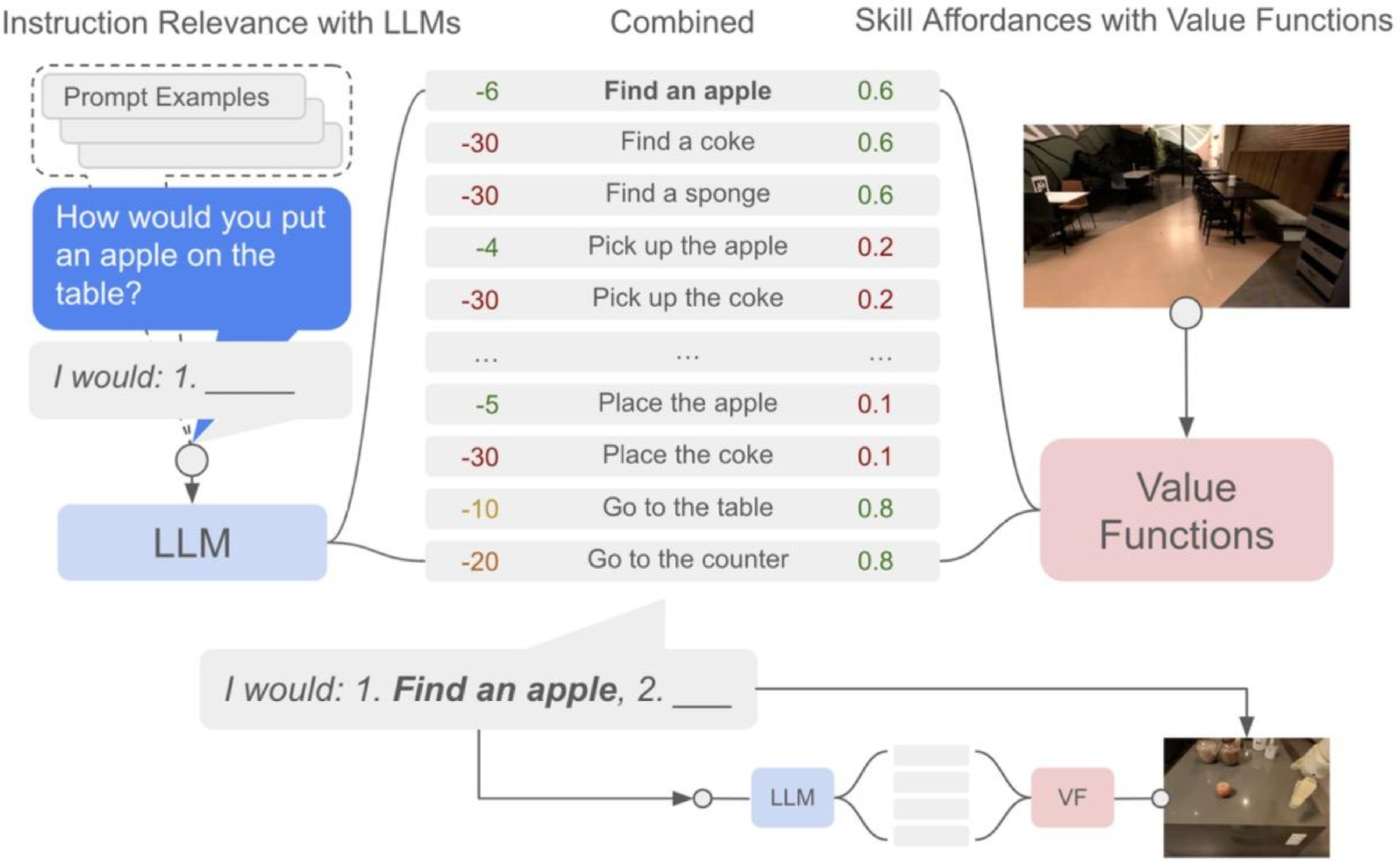
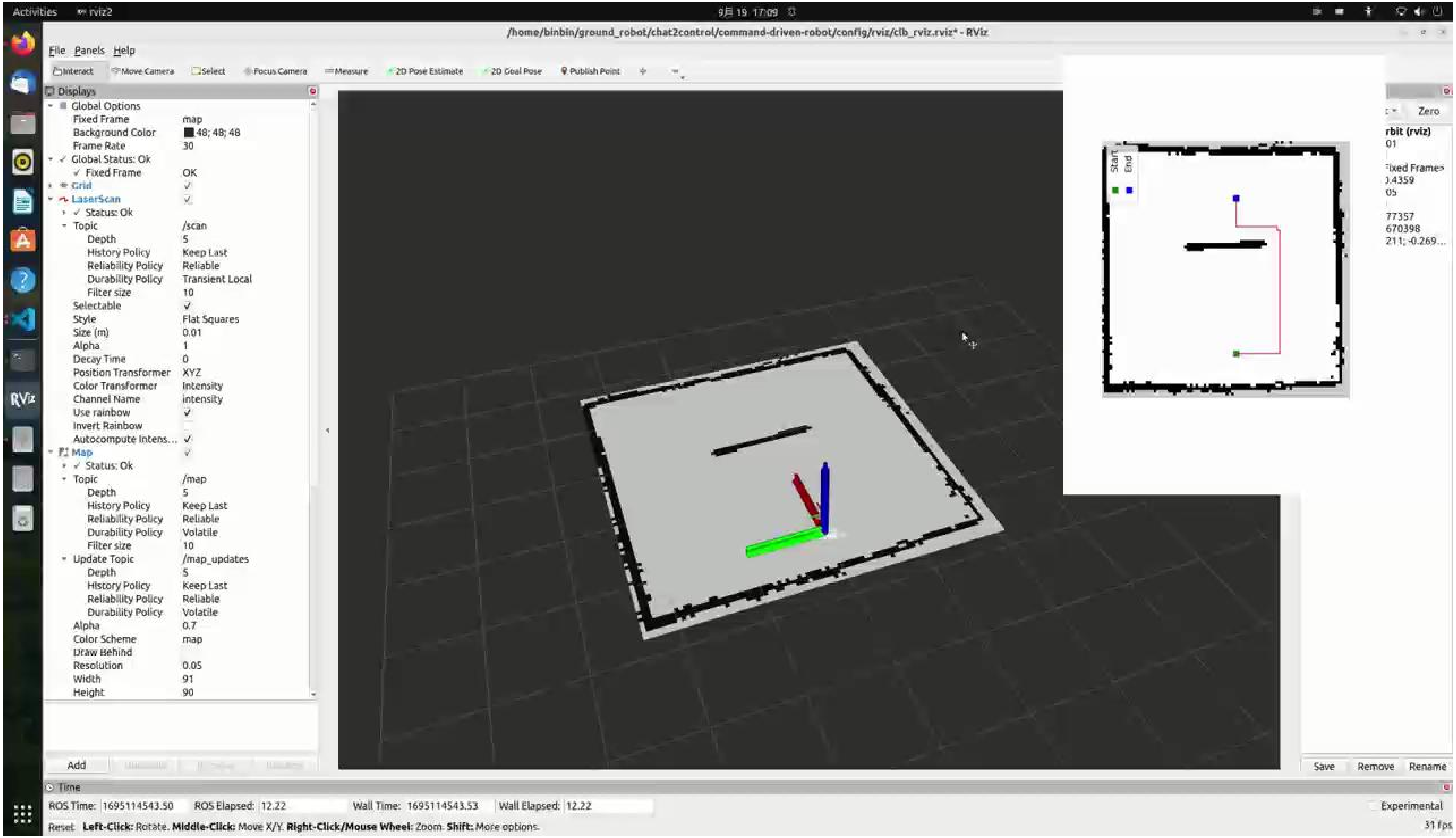


Figure 1: Inner Monologue enables grounded closed-loop feedback for robot planning with large language models by leveraging a collection of perception models (e.g., scene descriptors and success detectors) in tandem with pretrained language-conditioned robot skills. Experiments show our system can reason and replan to accomplish complex long-horizon tasks for (a) mobile manipulation and (b,c) tabletop manipulation in both simulated and real settings.

Huang et al. "Inner monologue: Embodied reasoning through planning with language models." 2022.



Ahn, Michael, et al. "Do as i can, not as i say: Grounding language in robotic affordances." 2022.


Activities
rviz2
9月 19 17:09
/home/binbin/ground_robot/chat2control/command-driven-robot/config/rviz/clb_rviz.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Goal Pose
Publish Point
Displays
Global Options
Fixed Frame map
Background Color 48; 48; 48
Frame Rate 30
Global Status: Ok
Fixed Frame OK
Grid
LaserScan
Status: Ok
Topic /scan
Depth 5
History Policy Keep Last
Reliability Policy Reliable
Durability Policy Transient Local
Filter size 10
Selectable
Style Flat Squares
Size (m) 0.01
Alpha 1
Decay Time 0
Position Transformer XYZ
Color Transformer Intensity
Channel Name intensity
Use rainbow
Invert Rainbow
Autocompute Intens...
Map
Status: Ok
Topic /map
Depth 5
History Policy Keep Last
Reliability Policy Reliable
Durability Policy Volatile
Filter size 10
Update Topic /map_updates
Depth 5
History Policy Keep Last
Reliability Policy Reliable
Durability Policy Volatile
Alpha 0.7
Color Scheme map
Draw Behind
Resolution 0.05
Width 91
Height 90
Add
Start
End
Zero
rbit (rviz)
Save
Remove
Rename
Time
ROS Time: 1695114543.50
ROS Elapsed: 12.22
Wall Time: 1695114543.53
Wall Elapsed: 12.22
Experimental
Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel: Zoom. Shift: More options.
31 fps

Future trends

- **Autonomous:** By human instructions, the AI Agent can execute tasks automatically, such as doing scientific experiments, calling vehicles for transportation, browsing website for information, reading API docs, etc.
- **Generative:** Generate human experiment, memory and experience to realize high-level thinking processes and social interaction with other agents.
- **Embodied:** Embodied AI Agents can interact physical world. Robotic servant can listen for human instructions and perform cooking, cleaning and serving, and improve from human feedback.

AI AGENT Challenges

Finite context length: This limits the inclusion of historical information, detailed instructions, API call context, thus limiting the effectiveness of CoT, Reflection, etc.

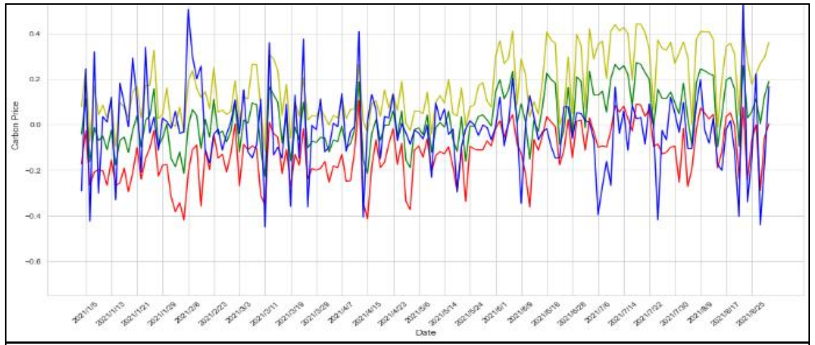
Long-term planning: Still hard to plan a complicated task with large number of steps. Ineffective to improve planning through trial-and-error learning.

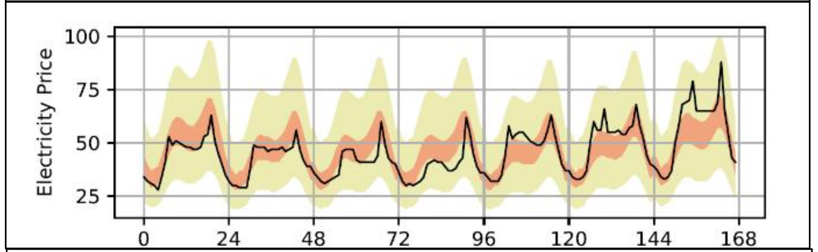
Natural language interface: AI agent relies on natural language as an interface between LLMs and external components such as memory and tools. This could be ineffective due to amibiguity.

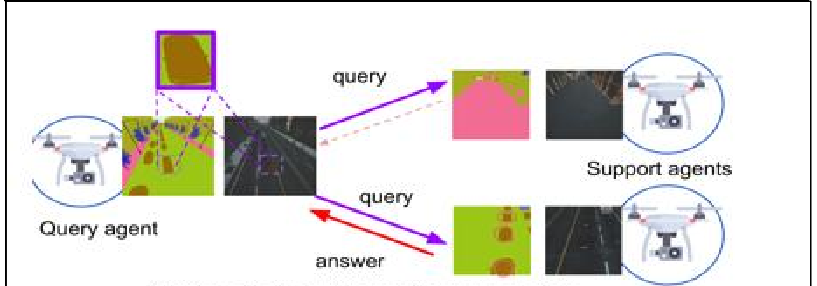
Reliability and safety: Hallucination and rebellious behavior (e.g. refuse to follow an instruction) could happen sometimes in LLMs.

致谢

- 感谢华南师范大学CCF分会邀请
- 感谢计算机系、人工智能学院
- 感谢研究生董瑞琪、刘向阳、庞天琦同学协助
- 欢迎对人工智能感兴趣的同学关注我们的CV&NLP, 时序预测, 分布式学习, 强化学习等研究方向







(b) few-shot multi-agent segmentation

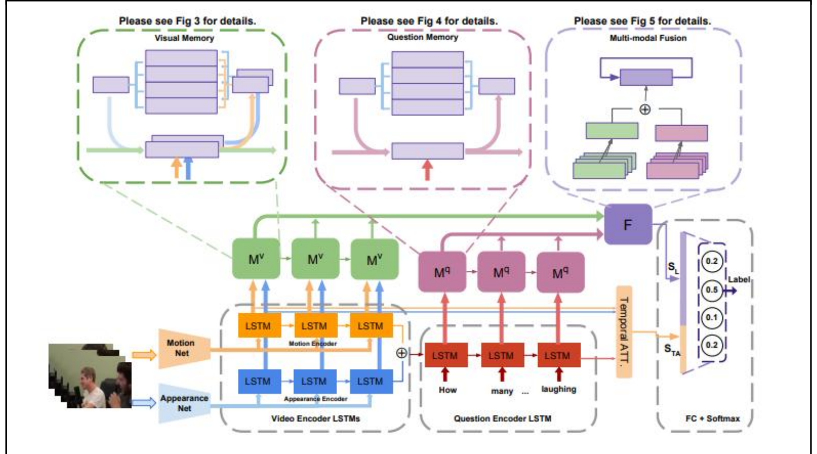


Figure 2. Our proposed VideoQA pipeline with highlighted visual memory, question memory, and multimodal fusion layer.

THANK YOU!