



# Prospect of Intelligent Agent System Driven by Large Language Models

## 大模型驱动的智能代理系统展望

范晨悠

华南师范大学人工智能学院

2023-10-26



# Prospect of Intelligent Agent System Driven by Large Language Models

## 大模型驱动的智能代理系统展望

范晨悠, CCF高级会员  
华南师范大学人工智能学院

# 狼人杀角色：

- 上帝、村民、狼人、守卫、女巫和预言家
  - **狼人**夜间可以指定放逐一名玩家
  - **预言家**可以查验身份
  - **女巫**可以解救、放逐
  - **守卫**可以守护
- 所有人白天进行一轮辩论，通过多数投票 (majority voting) 放逐一名玩家
- 好人提供正确的信息，狼人混淆视听



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

Yuzhuang Xu<sup>1</sup>, Shuo Wang<sup>1</sup>, Peng Li<sup>2,\*</sup>, Fuwen Luo<sup>1</sup>

Xiaolong Wang<sup>1</sup>, Weidong Liu<sup>1,3</sup>, Yang Liu<sup>1,2,\*</sup>

<sup>1</sup>Department of Computer Science & Technology, Tsinghua University, Beijing, China

<sup>2</sup>Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China

- Xu et al. 使用8个由LLM模拟出来的角色，通过角色间自动生成的对话，进行游戏。
- LLM 学习到了人类游戏中的**信任、伪装、对抗、和领导力**。
- 游戏过程只需要设计 prompt 不必对模型中的参数进行调整。



2.村民: 我怀疑5号是狼人。  
(好人提出观点)

3.狼人: 我也怀疑5号是狼人。  
(狼人顺势攻击好人)

1.狼人  
-----  
请大家发表观点。  
(隐藏自己狼人身份)



4.预言家  
-----  
我是村民。  
(伪装身份, 保护自己)

6.女巫: 5号昨天被狼人击杀。  
(主动提供信息, 5号可能好人)

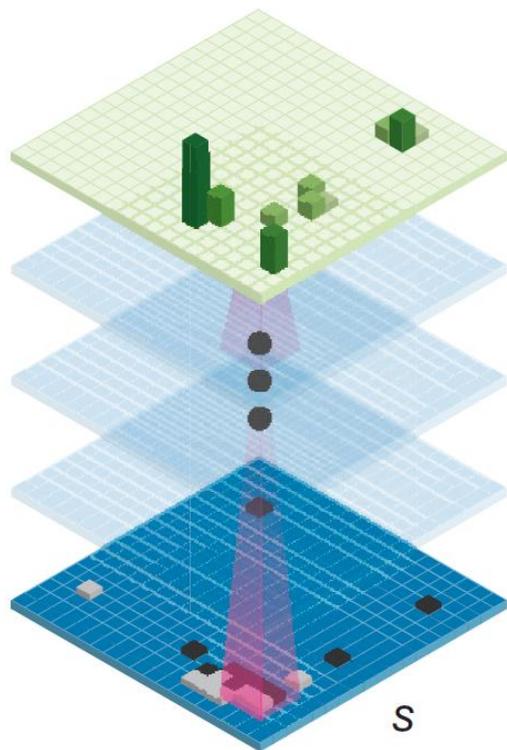
7.守卫: 我会在夜晚守护自己, 避免被狼人杀死。  
(表明好人身份)

5.村民: 我是好人。我怀疑3号是狼人, 他说的话很可疑。  
(反击3号栽赃陷害)

# 多角色的智能代理系统 研究动机

- 如何虚拟出多个智能代理角色参与到复杂任务中？
- 如何实现不同智能代理之间的合作、竞争？
- 如何使用大语言模型LLM 驱动代理进行对话、讨论，从而学习到任务策略，并可以反思和改进自己的策略？
- 如何收集环境或者外部的反馈？如何获取外部的知识？

# 智能代理系统概念



Part-1 规划

Part-2 记忆和存储

Part-3 外部工具

Part-4 具身

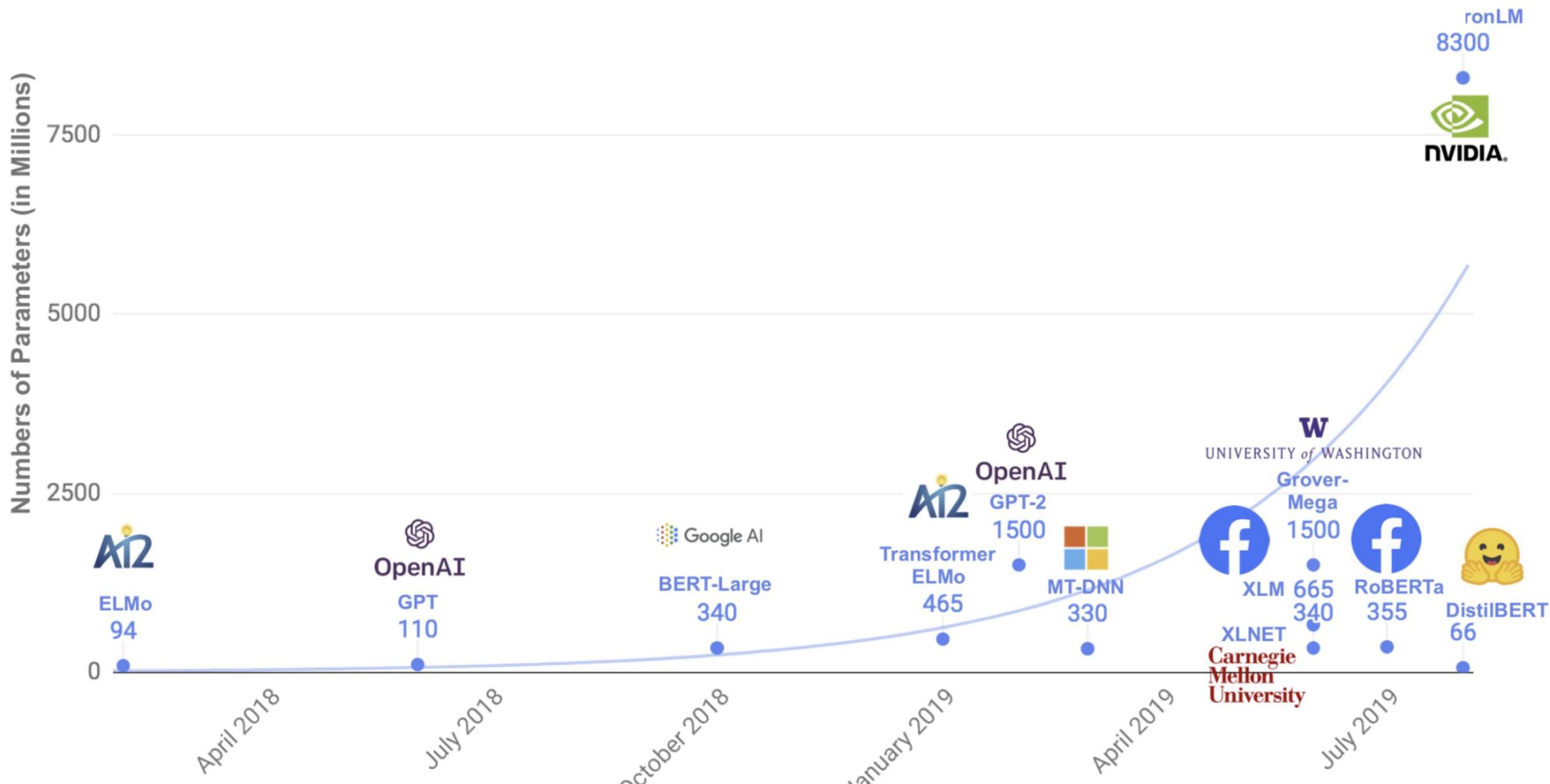
# 引用和致谢

- 本次汇报内容受到 Lil'Log 博客文章启发，并整理来自近期论文
- 原文链接 <https://lilianweng.github.io/posts/2023-06-23-agent/>
- Lilian Weng 目前是 OpenAI AI Safety Team 主管
- 她毕业于印第安纳大学布鲁明顿分校 Indiana University Bloomington

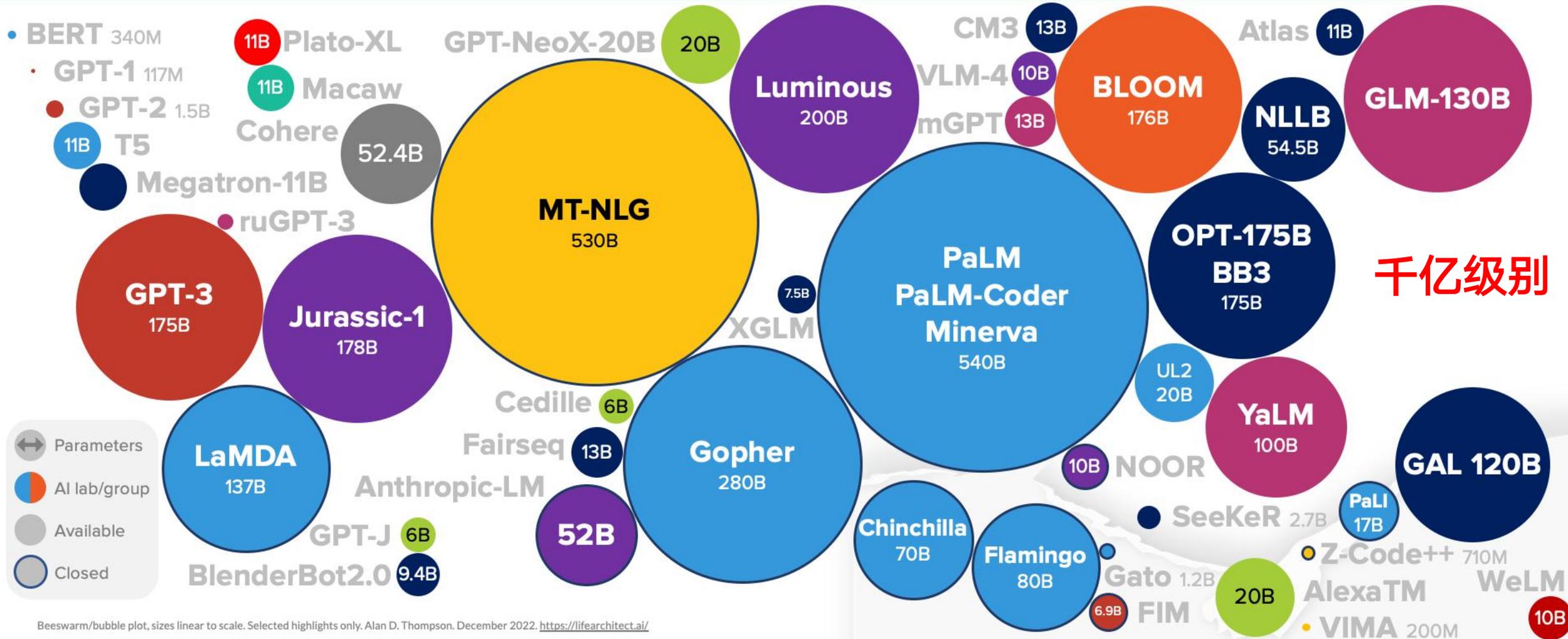
# Reference

- 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 大型语言模型 LLM 参数指数增长



# LANGUAGE MODEL SIZES TO DEC/2022



- LLM 可以用于生成文本、回答常识问题、完成基本推理。
- LLM 还可以完成写作任务、自动摘要、机器翻译、自动编程等各种应用。

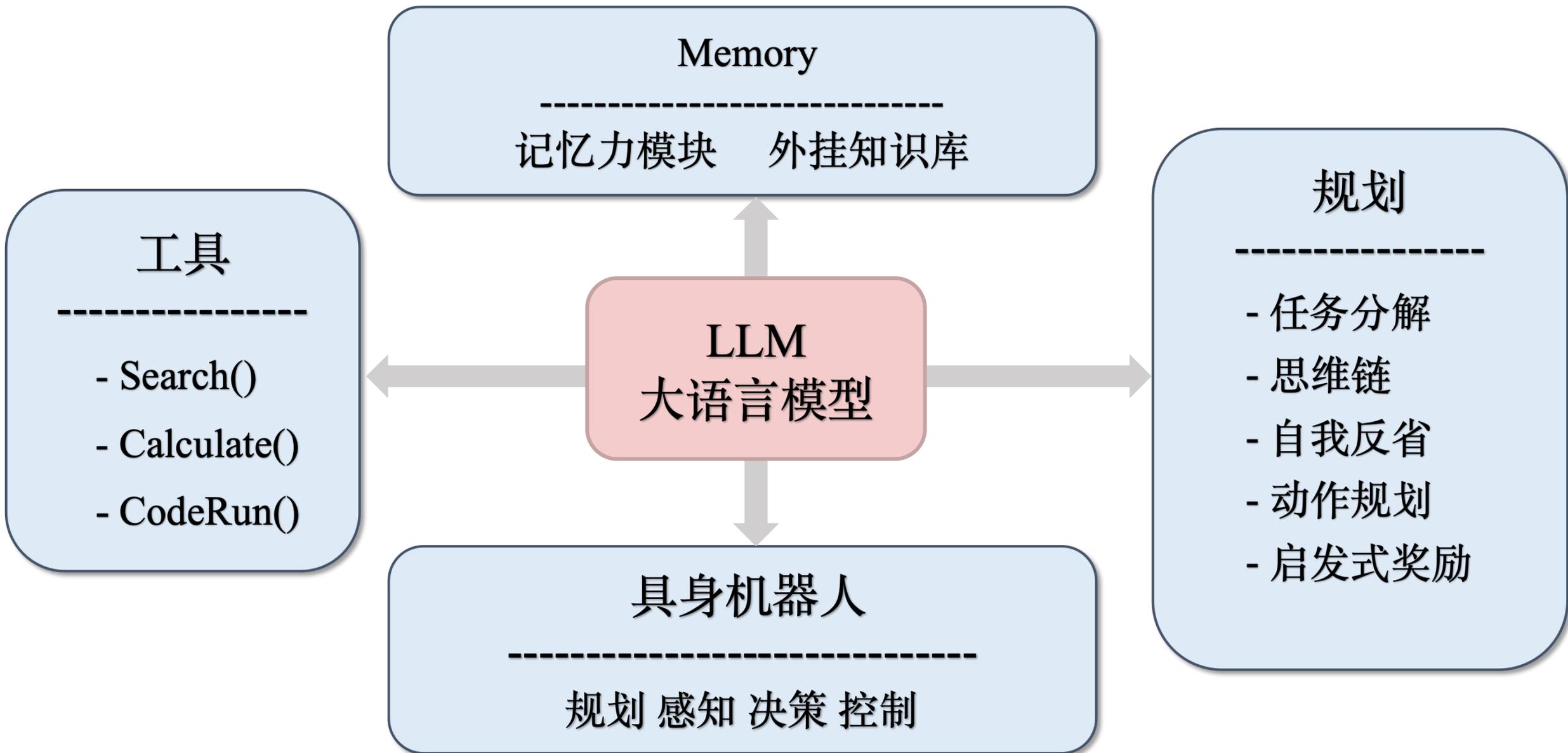


how to merge dictionaries in python?



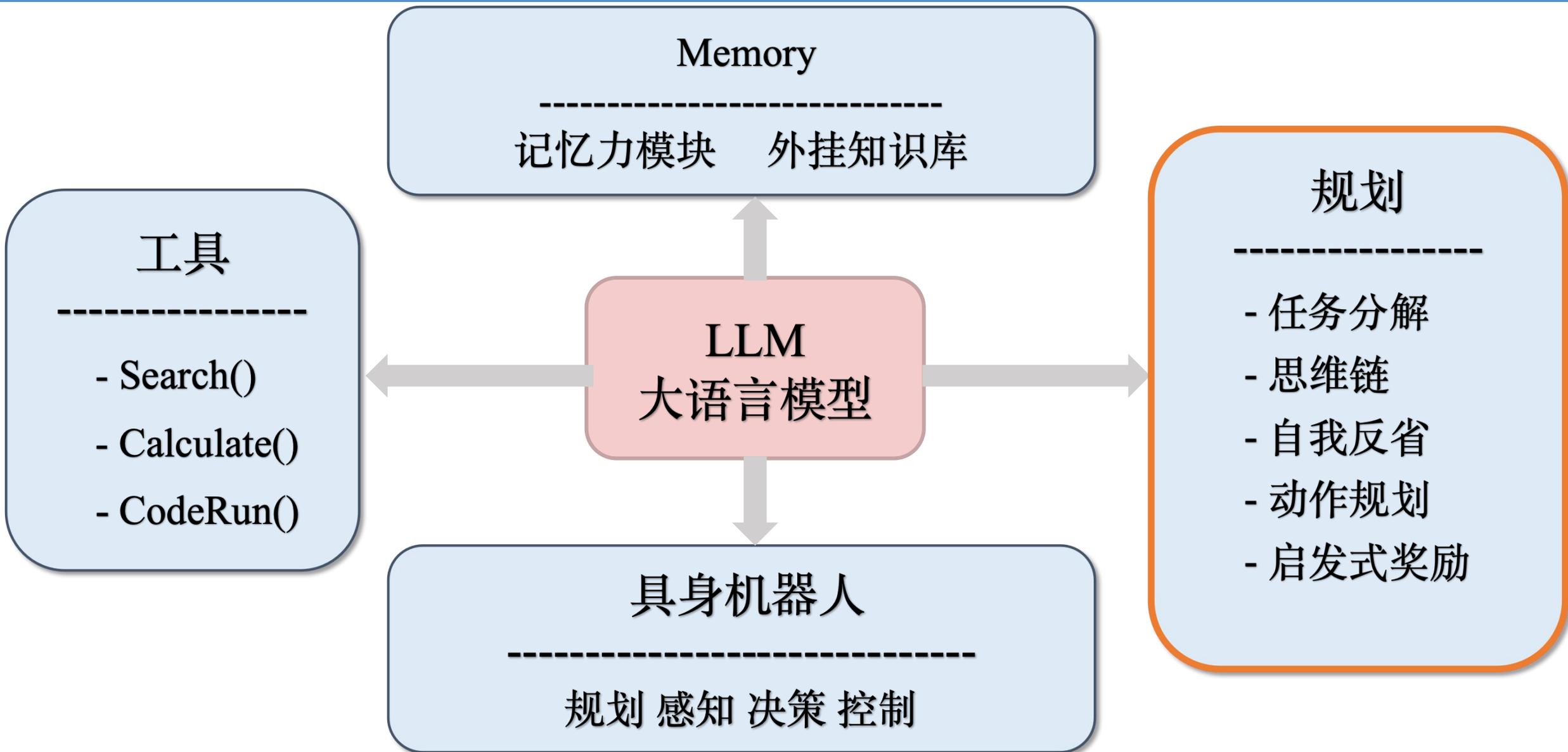
To merge two dictionaries in Python, you can use the ``update()`

# 智能代理系统 AI AGENT 组成



- 1. 规划 (Planning)** : 复杂任务分解为多个子任务, 执行动作 (问答、查询、使用工具), 自我反思以改进未来的动作。
- 2. 记忆 (Memory)** : 利用短期记忆, 进行上下文学习, 如 Prompt; 利用长期记忆, 实现外部文档检索、外部知识提取。
- 3. 工具使用 (Tool use)** : 使用外部API、搜索引擎、求解器、代码执行, 在线学习, 获取模型权重中无法实现的额外能力。
- 4. 具身交互 (Embodiment)** : 使用具身机器人进行物理环境交互, 采取行动, 实现对环境的实际改变。

# 智能代理系统 - 规划



# 1. 规划 Planning by AI-Agent

1. **子目标分解 (Subgoal 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

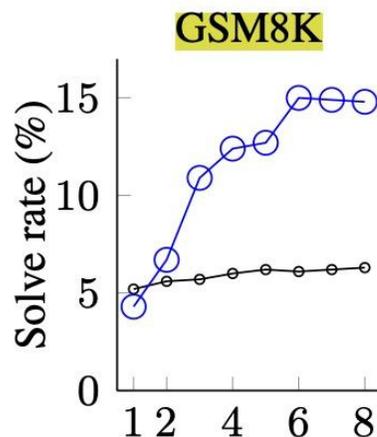
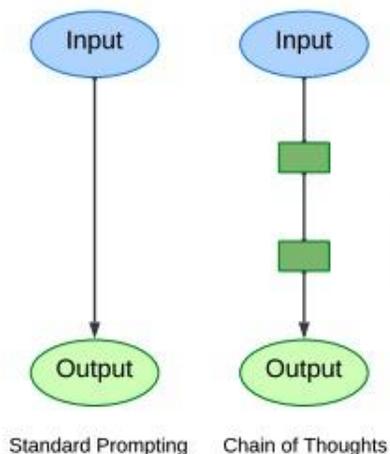
2. **反思与完善 (Reflection and refinement)** : 对过去的行动进行自我批评和反思, 纠正以前的错误来迭代地改进。

- 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

# 思维链，思维树 Chain / Tree-of-Thought

## CoT 指示模型 “逐步思考”

- 在Prompt时，问题的结尾附加“Let's think step by step” (ZeroShot-CoT)
- 或者直接提供带有步骤的示例模板
- 为模型的思维过程提供了解释



## Chain-of-Thought Prompting

### Model 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

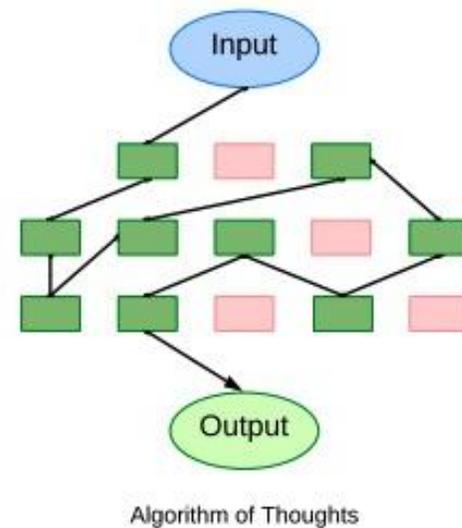
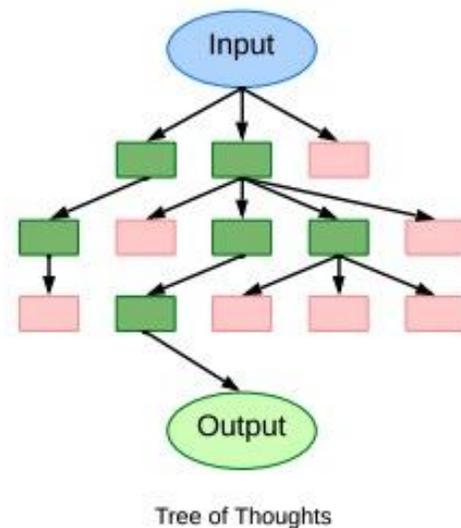
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had  $23 - 20 = 3$ . They bought 6 more apples, so they have  $3 + 6 = 9$ . The answer is 9. ✓

# 思维链，思维树 Chain / Tree-of-Thought

## ToT 可以提供多条思维分支

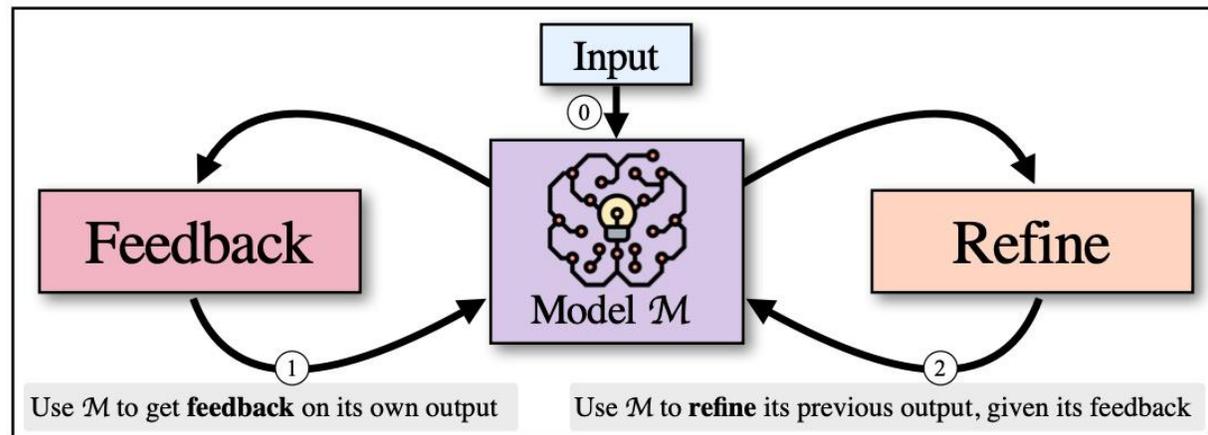
- 在每个状态，通过众多投票 (majority vote)，选择最优的思维链路。

## AoT 调用 BFS / DFS 在整个空间进行搜索



# 反思与完善 - 自我反思 (Reflection and refinement)

- Self-Refine (Madaan et al., 2023) 提出迭代改进模型输出。
- LLM首先生成输出，再用同一个LLM对结果进行反馈，来提供改进建议。



(d) Code optimization:  $x, y_t$

```
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  $y_{t+1}$

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



# 反思与完善 - 外部反思 (Reflection and refinement)

ReAct (Yao et al., 2023)

- 精调大模型产生 **行动 - 推理** 序列
- **推理(reasoning)** 生成 **行动计划**
- **行动(acting)** 获取外部知识 - 使用 Wikipedia API 进行搜索查找, 并返回答案

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

# 反思与完善 (Reflection and refinement)

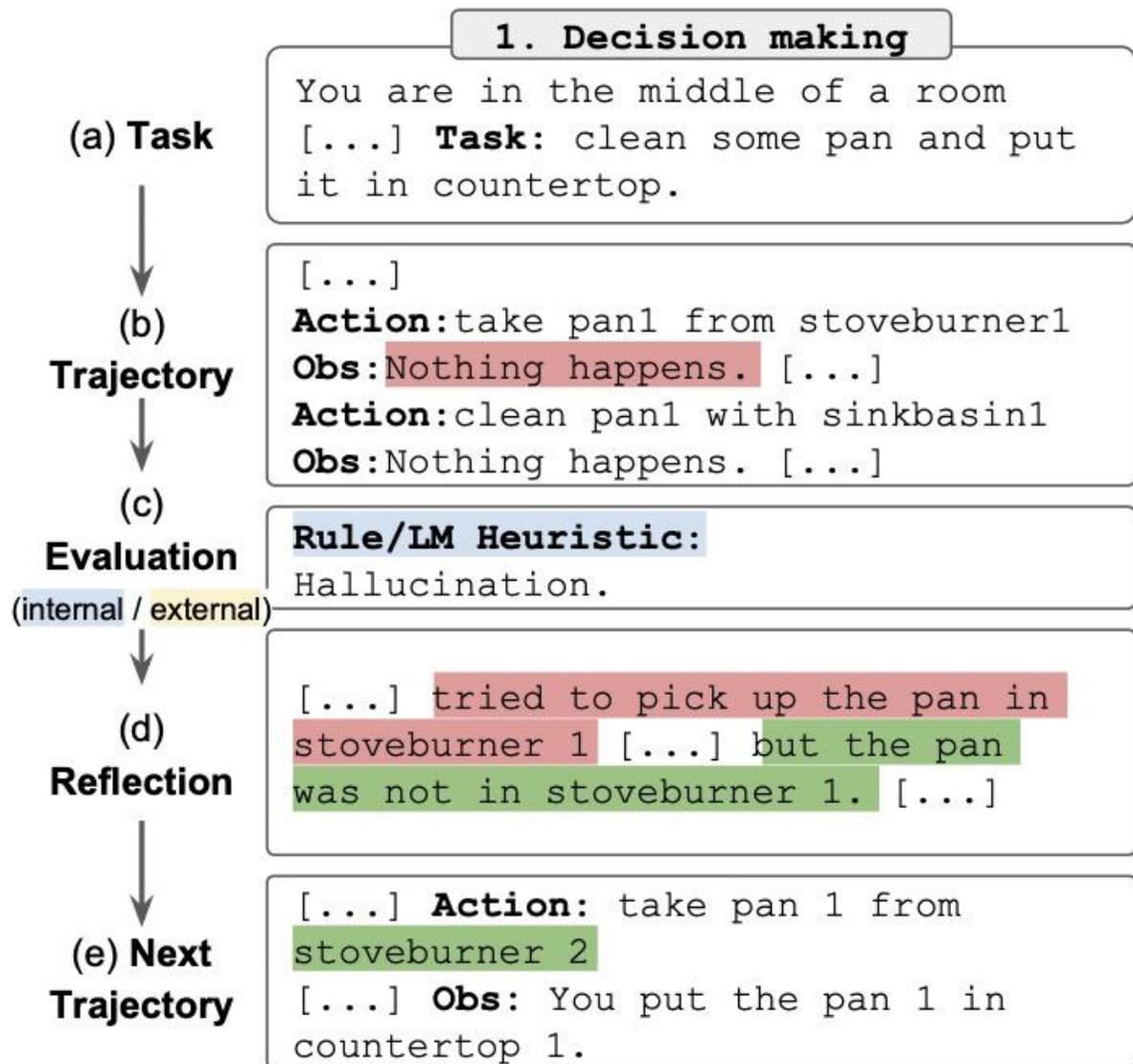
Reflexion (Shinn & Labash 2023) 使用强化学习增强自我反思能力

(a) 使用 ReAct 生成行动计划

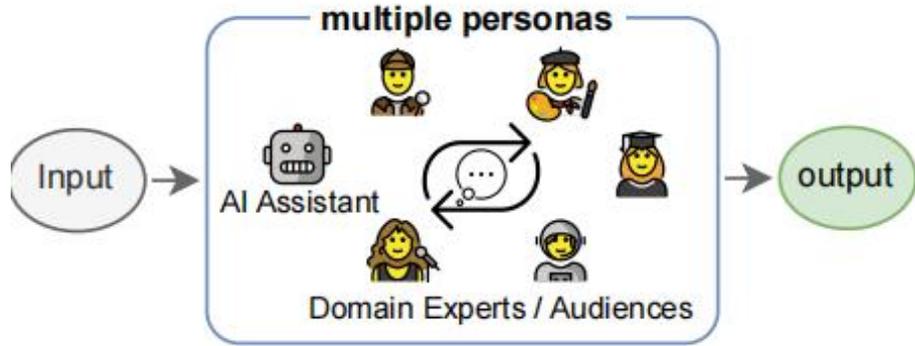
(b) 执行行动并观察结果 (如来自环境反馈)

(c) 对结果生成 0/1 奖励值 (行动是否成功)

(d,e) 使用LLM生成行动的反思, 用作生成后续动作的上下文 (改进后续行动计划)



# 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

Example Task 1: Use numbers and basic arithmetic operations (+ - \* /) to obtain 24.  
Input: 6 12 1 1

**Participants: AI Assistant (you); Math Expert**

Start collaboration!

Math Expert: Let's analyze the task in detail. You need to make sure that you meet of the common divisors of 2, such as 6, 8, 3 and try to construct these first. Also  
AI Assistant (你): **AI 助理角色:**  $12 / (1+1) * 6$   
**数学家角色:** 让我们检查结果... 结果有误  $12 / (1 + 1)) * ($   
Math Expert: Let's check the answer step by step.  $(1+1) = 2, (12 / 2) = 6, 6 * 6 = 36$

similar mistakes.  
AI Assistant (你): **AI 助理角色:**  $12 + (1-1) * 6$   
Math Expert: Thanks for pointing out the mistake. Here is a revised solution  
**数学家角色:** 很接近了, 你可以把  $(1-1)$  换成  $(1+1)$   
avoid using a number that is not part of the input?

AI Assistant (你): You are right, here is a revised solution considering 24 can be  
Math Expert: Let's check it.  $1 + 1 = 2, 6 * 2 = 12, 12 + 12 = 24$ . I believe  
AI Assistant (你): Sure,  $6 * (1+1) + 12 = 24$   
**数学家角色:** 答案正确  
Math Expert: Let's verify the solution.  $1 + 1 = 2, 6 * 2 = 12, 12 + 12 = 24$ . You used

Finish collaboration!

**Final answer:**  $6 * (1 + 1) + 12 = 24$

# 多角色合作竞争 Multi-persona cooperation and competition

## 个人历史信息（短期记忆）：

- 本局说过的话及自省内容，当做上下文信息
- 轮到自己发言时，使用LLM再次自省和推理

## 游戏经验池 Experience Pool（长期记忆）：

- 记录下各个代理每一轮的 <反思，得分>
- 得分反映了该代理玩家输赢的情况
- 通过检索相似的局面（当前反思和经验池中的反思相似度较高），提取得分较高的建议。

### 策略：正面对抗

狼：我投票P5出局

### 策略：伪装

女巫：我是平民

### 策略：信任

平民：我信任P5是好人

### 策略：领导

预言家：让我们一起放逐P1

# 狼人杀中单个角色自省举例

作为代理 {agent\_i} 角色 {role},  
对当前局势选择一个问题提问。

哪一位玩家的行为最可疑？



根据上下文回答该问题。

P-2在夜里被狼人击杀。昨天  
白天P-3质疑了P-2。



总结你的观察，生成自省。

我怀疑P-3是狼人，  
因为他的发言攻击了好人P-2。



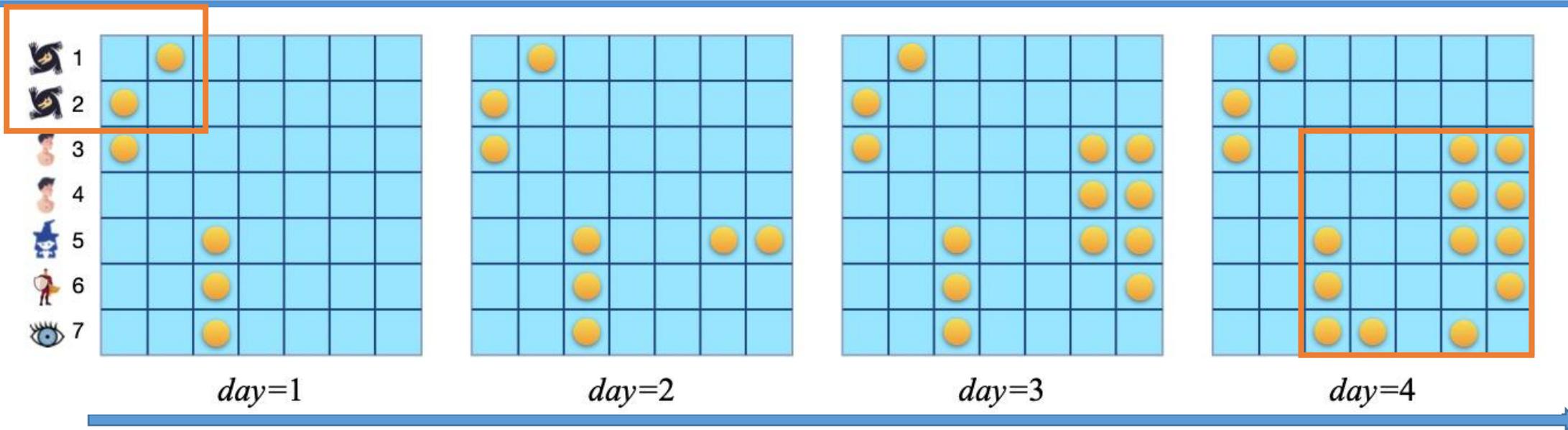
根据过往经验，想要赢得游戏，  
你应该保护好人 P-2

我决定投票放逐 P-3 狼人。



- 玩家A的身份可以确定吗？
- 哪一位玩家的行为最可疑？
- 我应该公布身份吗？
- 谁公开要求投出我的队友？
- 我是女巫，我今天救谁呢？
- 我是先知，我应该查验的可疑玩家？

**预定义的自省问题集合**



## 重新定义“信任 Trust”：可视化信任矩阵

- 当 **代理  $i$**  认为 **代理  $j$**  是其队友的时候，那么建立信任  $(i, j)$
- 信任可以是不对称的，当  $i$  认定  $j$  的时候， $j$  不一定能信任  $i$
- 随着游戏的进行，角色/阵营逐渐明朗，信任矩阵逐步完善

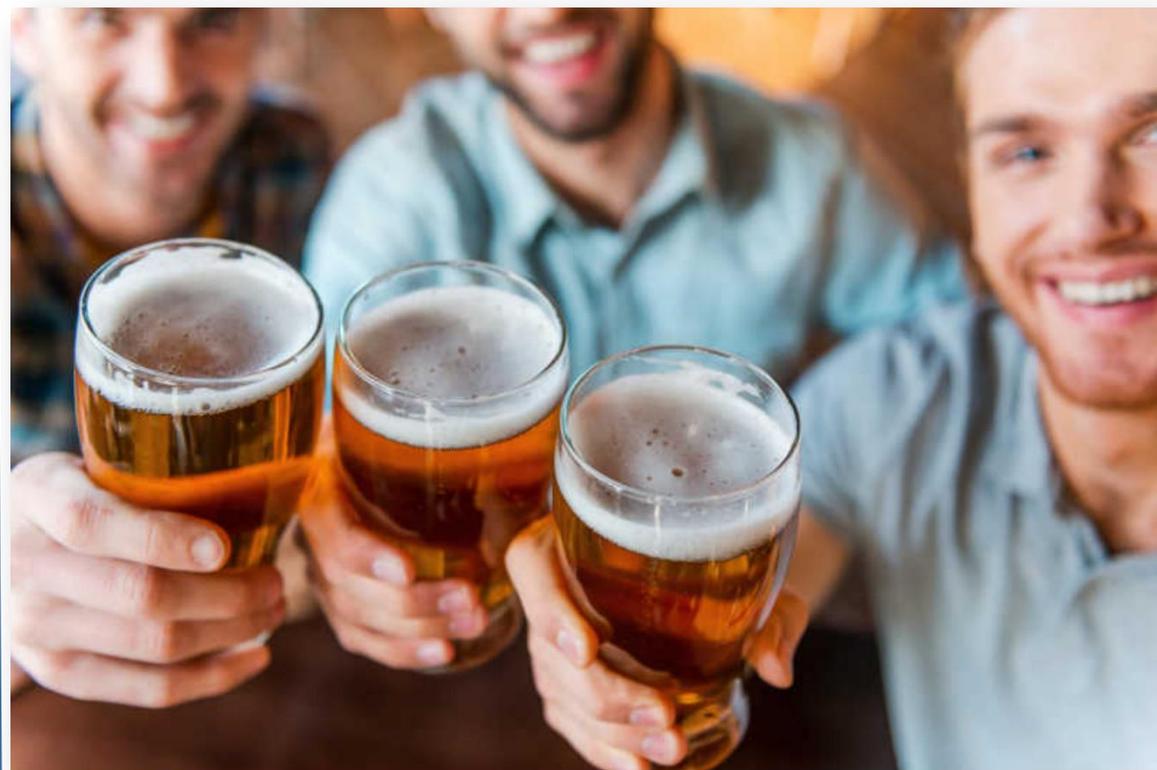
# 多角色推理案例：逻辑学家喝啤酒问题

A, B, C 三个逻辑学家去喝酒。  
服务生问，你们三个都要啤酒吗？

A说：我不知道。

B说：我也不知道。

C说：我们都要。  
请问C怎么知道的。





让我们先来看看需要哪些人参与。然后展开一个多轮协作，直到问题解决。



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.



示例输入作为 CoT



Here are some examples:

Example Task 2: Write a poem that meets the following requirements: (1) the poem has seven lines. (2) the poem needs to be about 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 should be the first letters of the words in the title.  
**Computer Scientist:** A quantum computer is an advanced computing device that uses the power of quantum mechanics to represent information as 0s and 1s, quantum computers use quantum bits or qubits. Qubits can be in a state of superposition, allowing them to perform multiple calculations at the same time. Consider using this information for the poem.

**Ten year old child:** I hope the poem to be fun and easy to understanding. I don't want to see a long poem.

**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.



现在让我们来找出以下任务的参与者。



然后让他们一步一步协作解决一下任务。



Now, identify the participants and collaboratively solve the following task step by step.



参与者：AI 助理，逻辑学家A, B, C



开始合作!

之前的对话中，我无法决定是否所有人都需要啤酒。因为信息缺失，我不知道 B,C 的选择，因此我无法提供确切回答。



逻辑学家A

和A一样，我无法确定其他人的选择。



逻辑学家B

从A,B的回答中，我有理由推断我们三个都需要啤酒。这是因为他们并没有反对。因此轮到我的时候，我就可以做出确定的回答。



逻辑学家C

AI 助理:

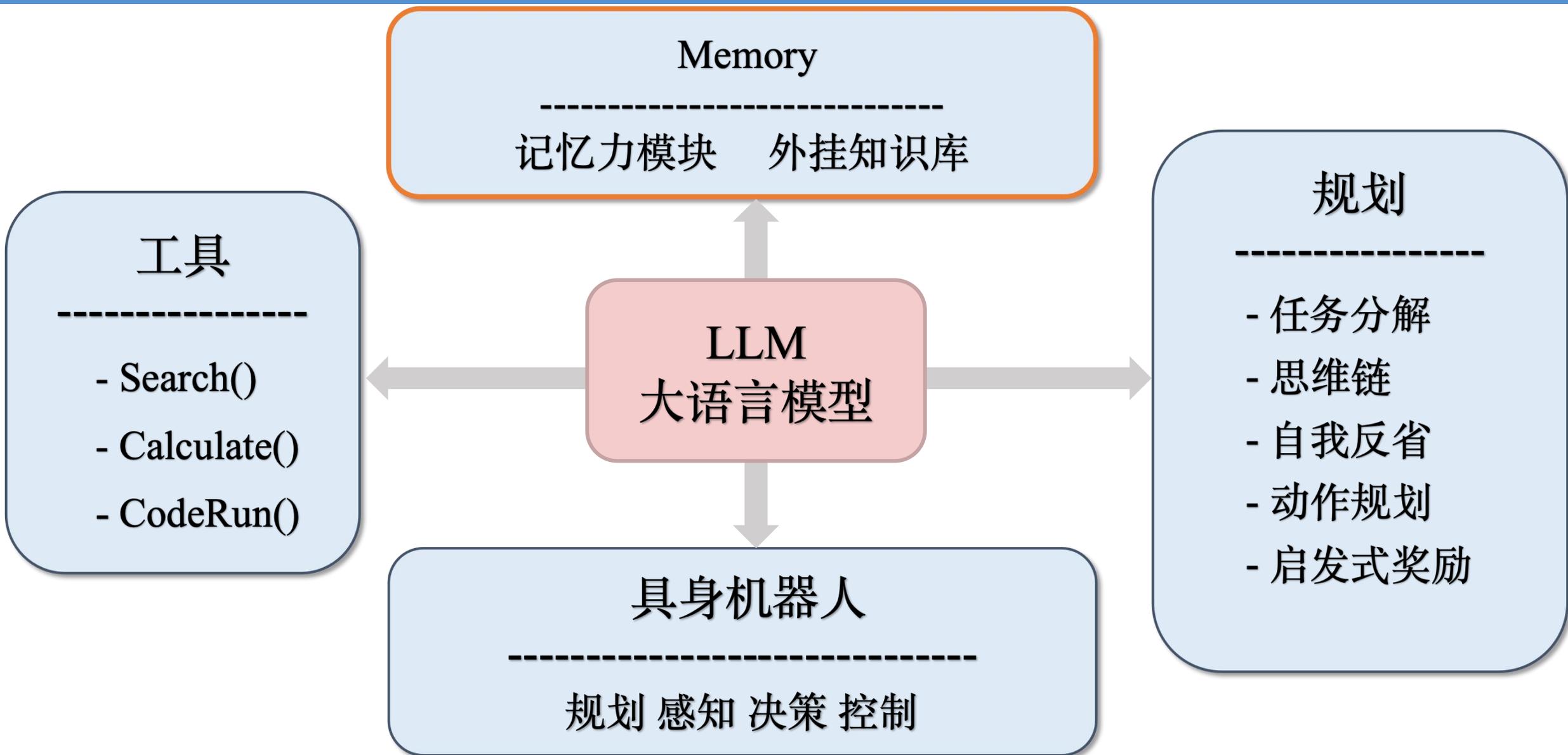


根据上述信息，服务生应该给三个人都提供啤酒。

# 课后习题

- 桌上有**21**枚硬币,其中**10**枚正面朝上。请蒙上眼睛,将硬币分成两组,使其硬币正面朝上的一样多?
- 自己思考、使用 ChatGPT 分角色思考 (数学家? 玩家? 助理? )
- 感兴趣的同学, 请将答案以及 prompt 发送到我的邮箱
- fanchenyou@scnu.edu.cn

## 2. 记忆力模块 Memory



## 2. 记忆力模块： 获取、存储和检索信息

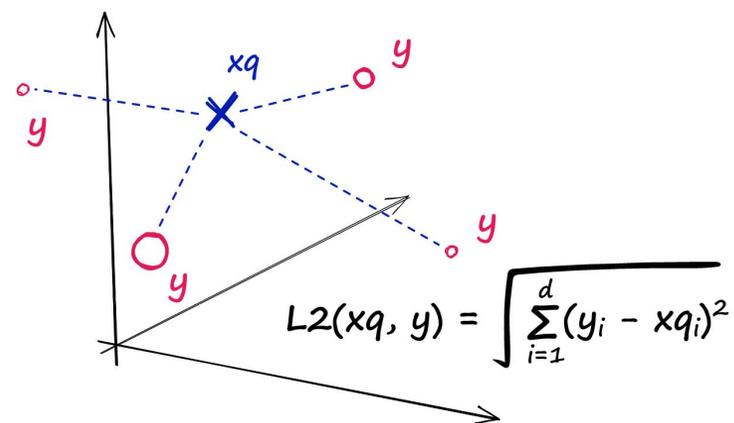
### 1. 短期记忆 Short-Term Memory :

- ✓ 存储处理当前复杂任务所需要的信息，用于学习和推理
- ✓ 基于Transformer架构的LLM模型，存在上下文输入限制，因此限制了能够输入的信息长度
- ✓ 例如，输入LLM的prompt token 数量限制，ChatGPT $\leq$ 4000，GPT-4 $\leq$ 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. 长期记忆 Long-Term Memory:

- ✓ 通常是外部的向量数据库，可以长期存放知识
- ✓ 知识以嵌入embedding、索引Index等形式存储
- ✓ 可以快速检索、访问、和查询比较，并取回数据



# 基于检索的大模型 A Retrieval-based LM

## 基于模型参数的事实提取



The capital city of Ontario is **Toronto**



LM



The capital city of Ontario is \_\_\_\_\_



LM

Test time

# 基于检索的大模型 A Retrieval-based LM

## 基于检索的事实提取



The capital city of Ontario is **Toronto**



LM



The capital city of Ontario is \_\_\_\_\_

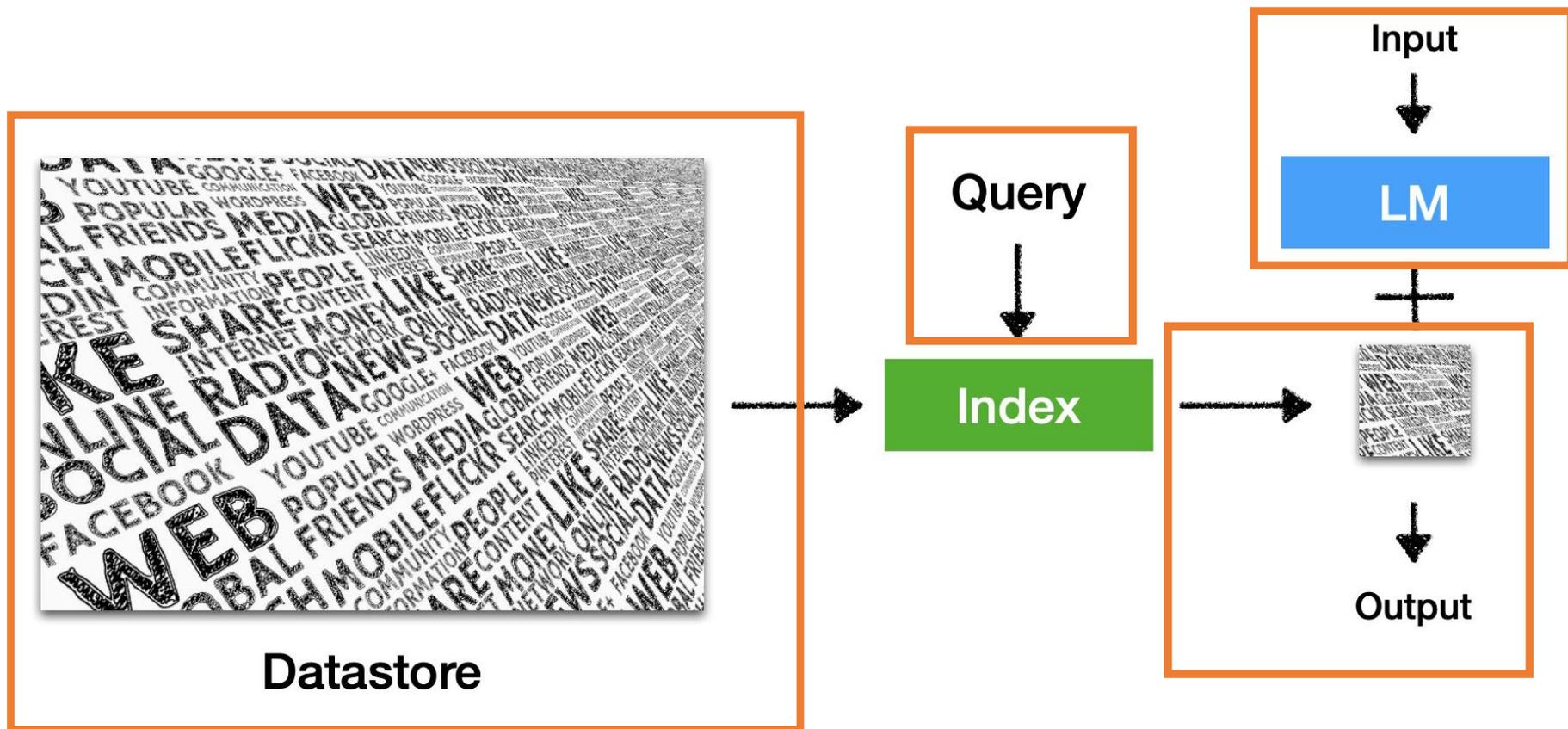


LM

Test time

# 基于检索的大模型 A Retrieval-based LM

定义：推理的时候，使用外部数据库知识的语言模型



# 最大内积检索方法 Maximum Inner Product Search

## MIPS 检索方法

- 将文档知识提取嵌入(embedding), 存储在向量数据库 D
- 在向量空间中选取距离的度量 (distance metric)
- 对给定的查询向量  $q$ , 使用近似最近邻算法 (ANN), 筛选相近的文档
- 一般返回 top-K 个数据库中相似的内容

**sim**: a similarity score between two pieces of text

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

$\text{tf}_{i,j}$ : # of occurrences of  $i$  in  $j$   
 $N$ : # of total docs  
 $\text{df}_i$ : # of docs containing  $i$   
词频-逆文件频率

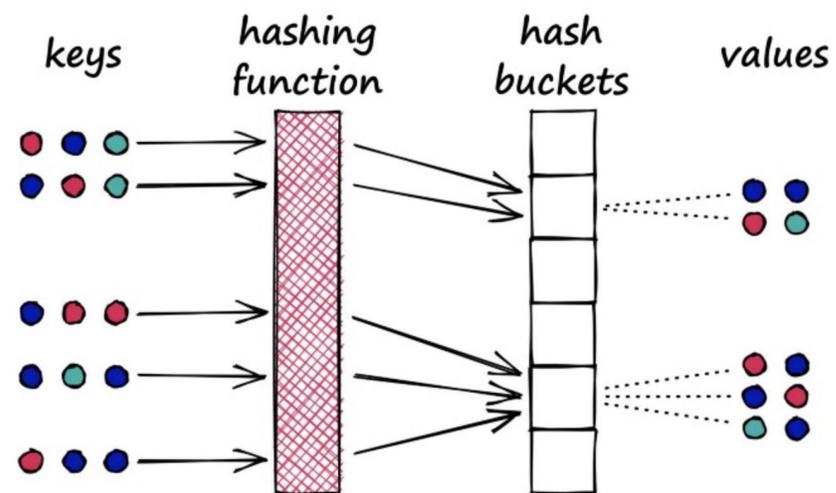
**Example**  $\text{sim}(i, j) = \text{Encoder}(i) \cdot \text{Encoder}(j)$

Maps the text into an  $h$ -dimensional vector

内积相似度

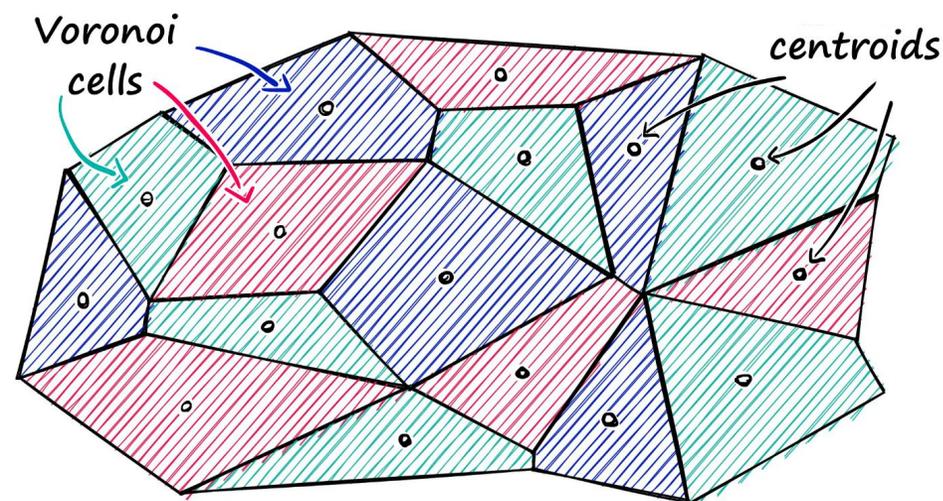
## 局部敏感哈希映射 LSH

- Locality-Sensitive Hashing
- 如果两个文本在原有的数据空间是相似的，那么分别经过哈希函数转换以后的它们也具有很高的相似度



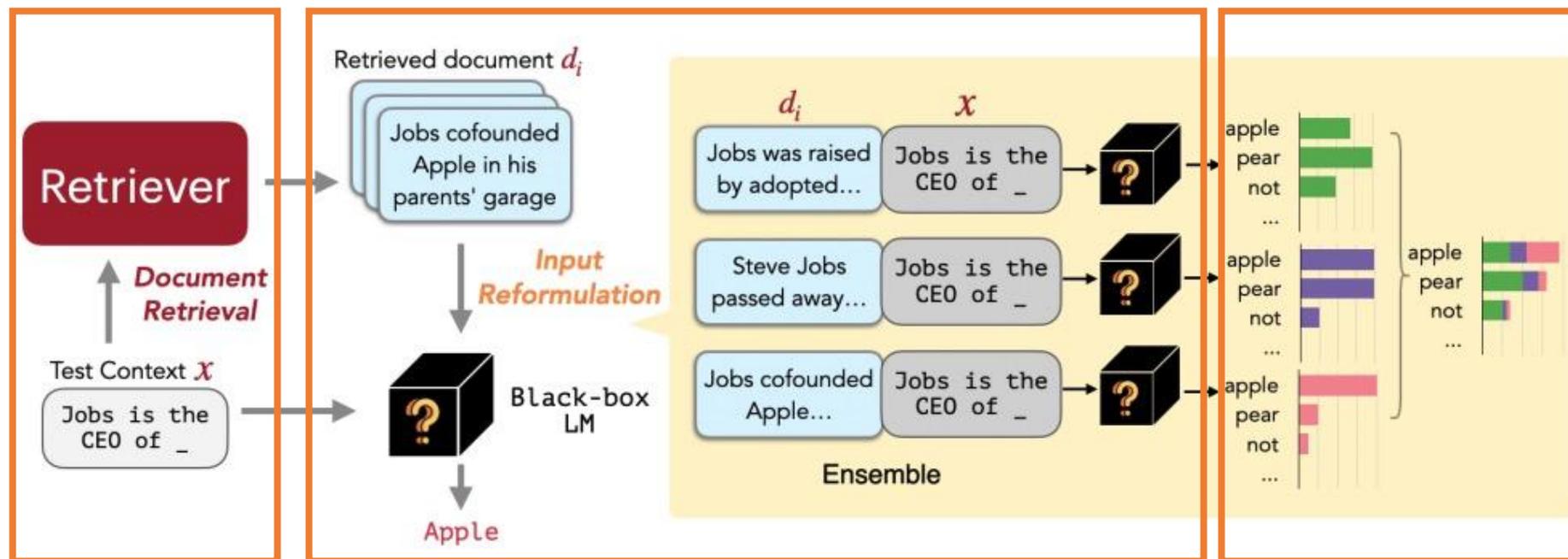
## 逆文件索引 Inverted File Index

- 向量空间划分成多个单元，各保存一个中心点 centroid
- 对query，先找到最近的中心点及对应的单元，再在单元内搜索



# 基于检索的大模型问答流程

- 针对原问题，检索出一个相关文档集合
- 将原问题与检索文档合并，输入LLM进行预测
- 将所有预测结果做ensemble，得到最终预测
- 训练Retriever，更新文档的 embedding 和 index
- 最小化 KL(检索文档的似然度 | LLM对检索文档相关性打分)



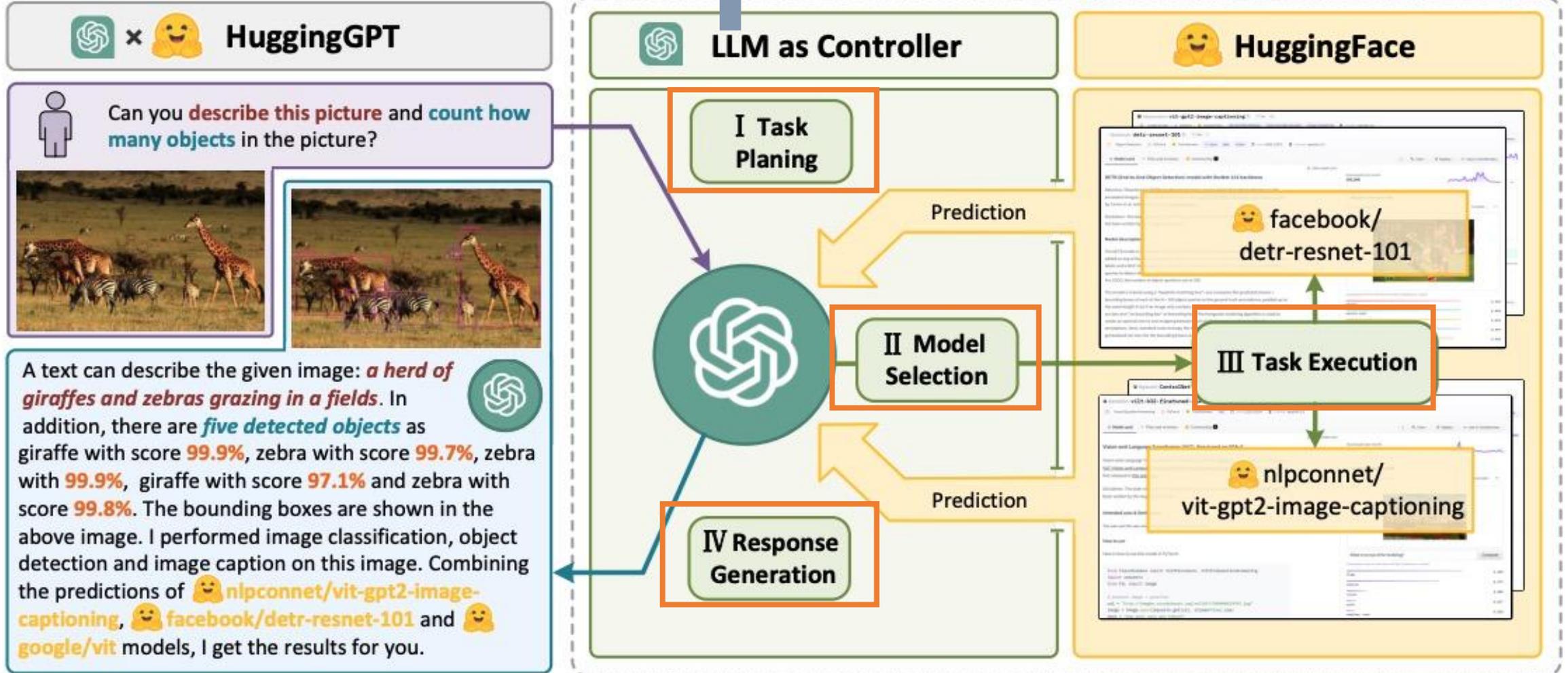
## 3. 使用工具 Tool Use

LLM 外部工具，如API、插件等，可以显著扩展模型的能力，实现专有的任务，如数学求解、程序编译运行等。

- **TALM (Tool Augmented Language Models; Parisi et al. 2022, Schick et al. 2023)** 通过微调语言模型来学习使用外部工具API。LLM阅读API文档决定该工具是否能够改善模型输出，从而进行调用。
- **HuggingGPT (Shen et al. 2023)** 利用ChatGPT作为任务规划器，根据模型描述选择HuggingFace平台上可用的模型，并根据执行结果总结回应。
  - 任务规划 (Task planning) : LLM作为核心大脑，将用户请求解析为多个任务
  - 模型选择 (Model selection) : LLM 根据任务类型和描述，从模型列表中选择最合适的专家模型。模型来自 HuggingFace 海量模型库。
  - 任务执行 (Task execution) : 专家模型在特定任务上执行并记录结果。
  - 响应生成 (Response generation) : LLM 接收执行结果，并提供给用户。

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

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





```
1
2 def common_prefix(a, b):
3     """Return the common prefix of two lists."""
4     if len(a) < len(b):
5         return common_prefix(b,a)
6     for i in range(len(a)):
7         if a[i] != b[i]:
8             return a[:i]
9     return a
10
11
12
```

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:

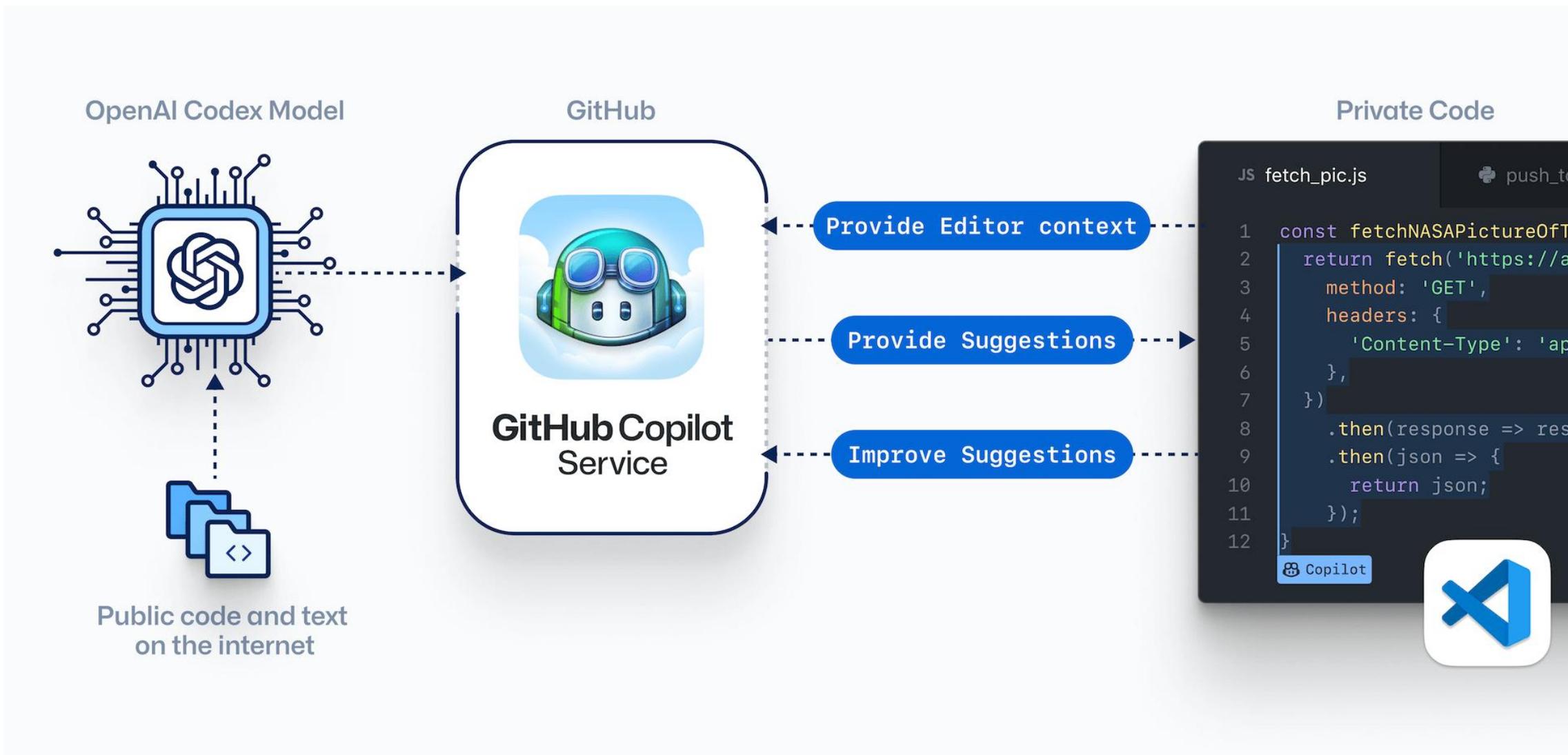
The Riemann zeta function is defined as:

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$$

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.

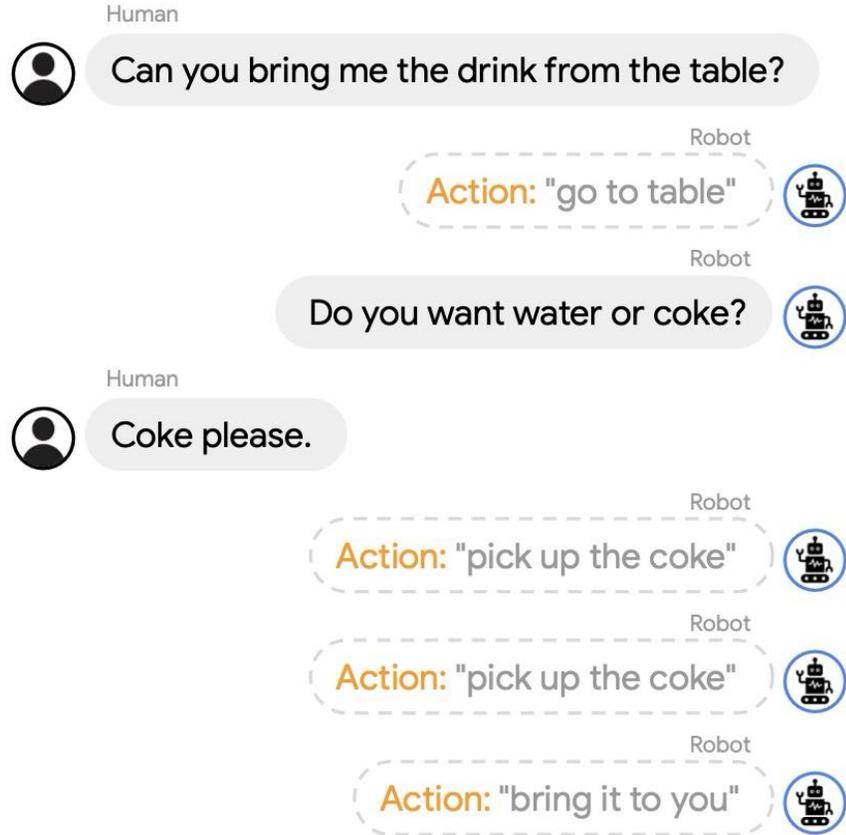


## 4. 具身代理 Embodied Agent

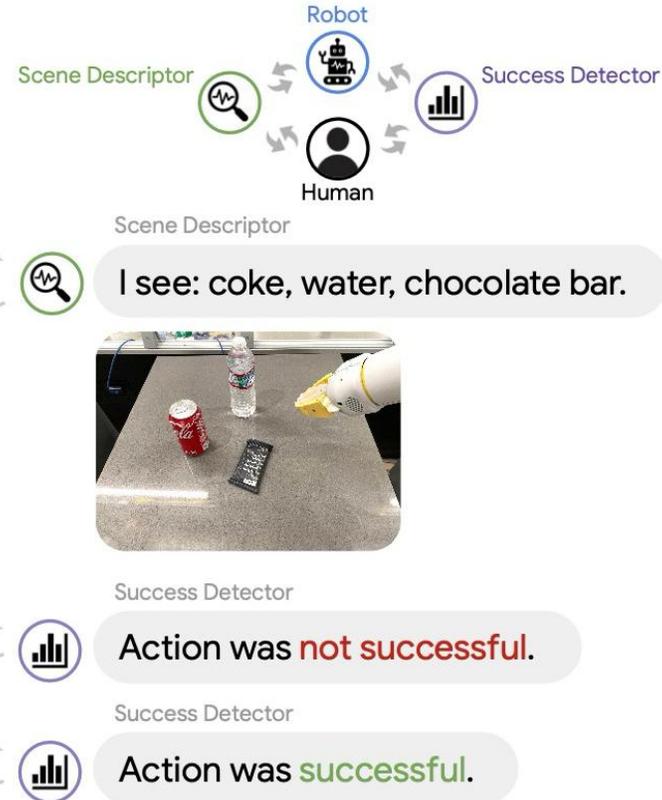
LLM 通过对机器人的规划、操控来实现与环境交互，完成现实任务。

- **Embodied Reasoning through Planning (Huang et al. 2022)** 提出使用LLM来模拟内心独白，提供任务的反馈信息，如感知、控制的结果，以及和人类反馈交互，形成闭环。
- **Grounding Language in Robotic Affordances (Ahn et al. 2022)** 交替进行LLM规划和基于价值函数的可行性打分
  - Step-1: 通过prompt LLM将高层次的指令分解为具体的低层次行动
  - Step-2: 使用一个价值函数来衡量，候选动作在当前状态下的可行性
  - Step-3: 执行上述选中的动作，加入到已有指令，迭代进行上述步骤，直到任务完成。

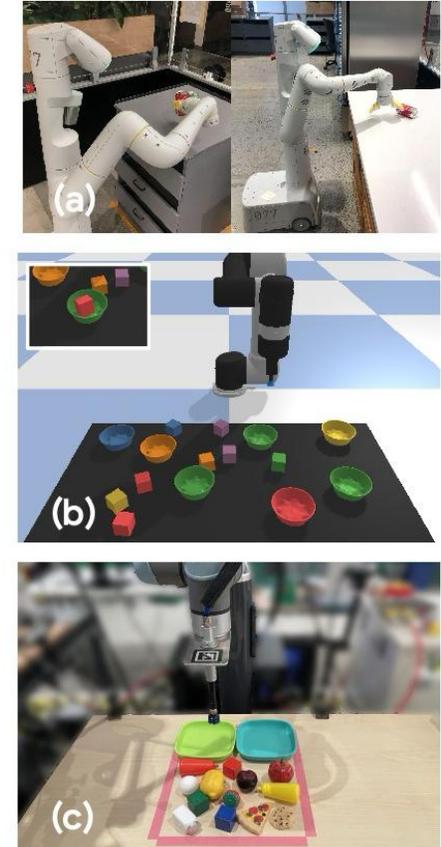
## Robot Planning & Interaction



## Grounded Closed-Loop Feedback



## Robot Environments



**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.

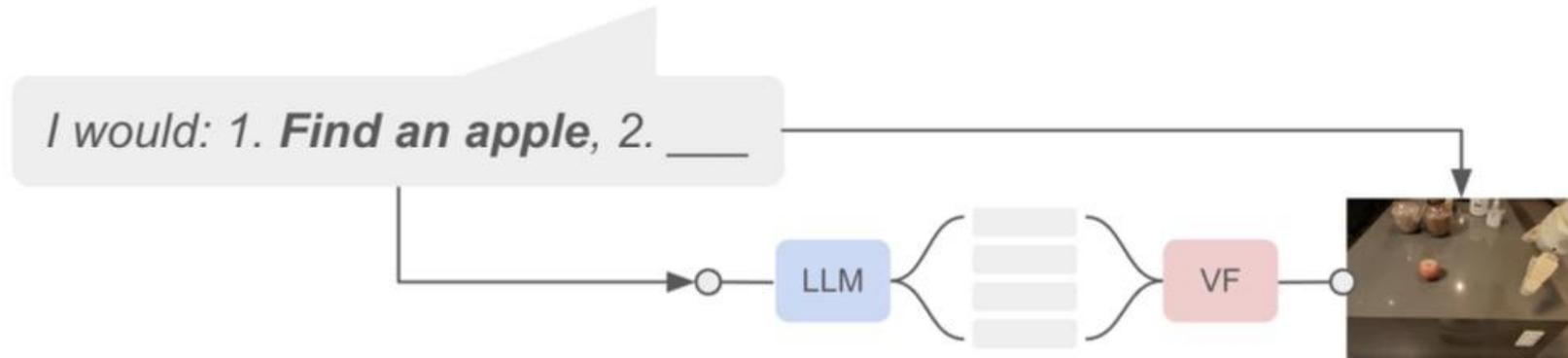
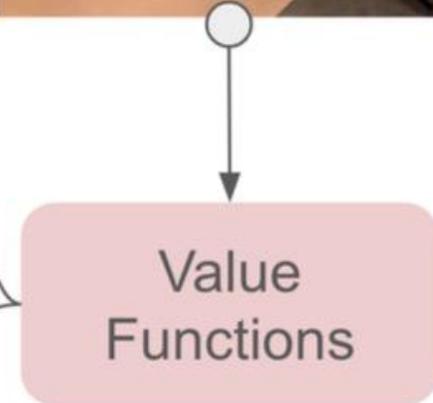
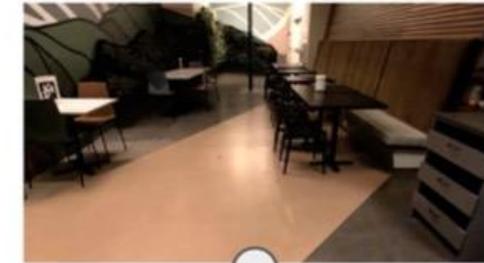
### Instruction Relevance with LLMs



### Combined

-6	<b>Find an apple</b>	0.6
-30	Find a coke	0.6
-30	Find a sponge	0.6
-4	Pick up the apple	0.2
-30	Pick up the coke	0.2
...	...	...
-5	Place the apple	0.1
-30	Place the coke	0.1
-10	Go to the table	0.8
-20	Go to the counter	0.8

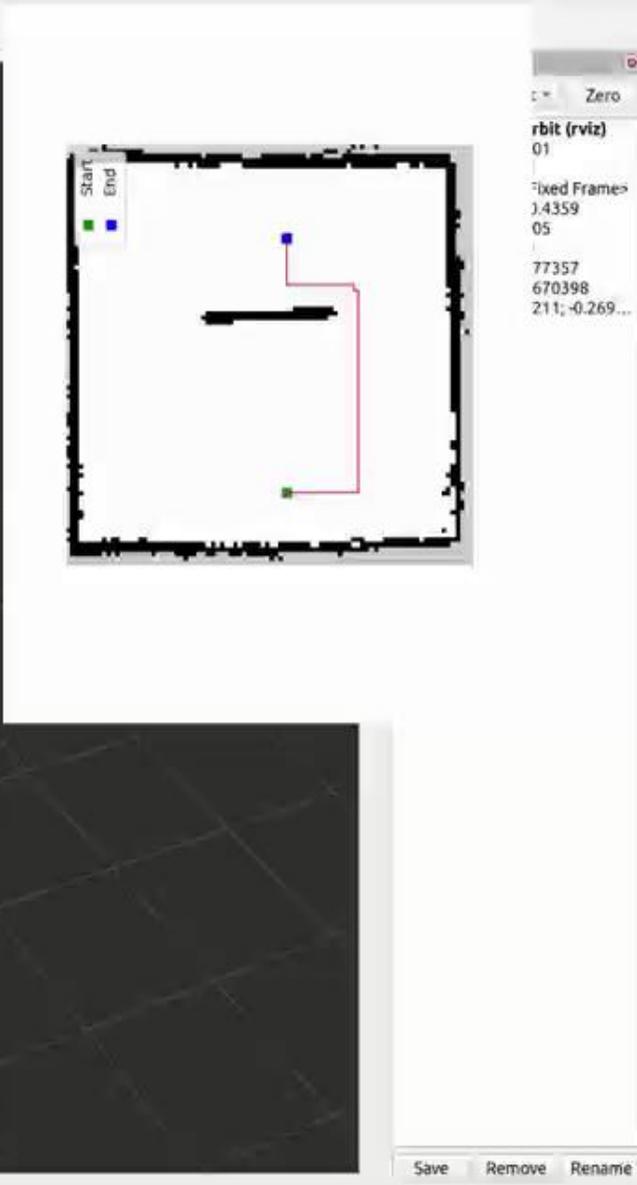
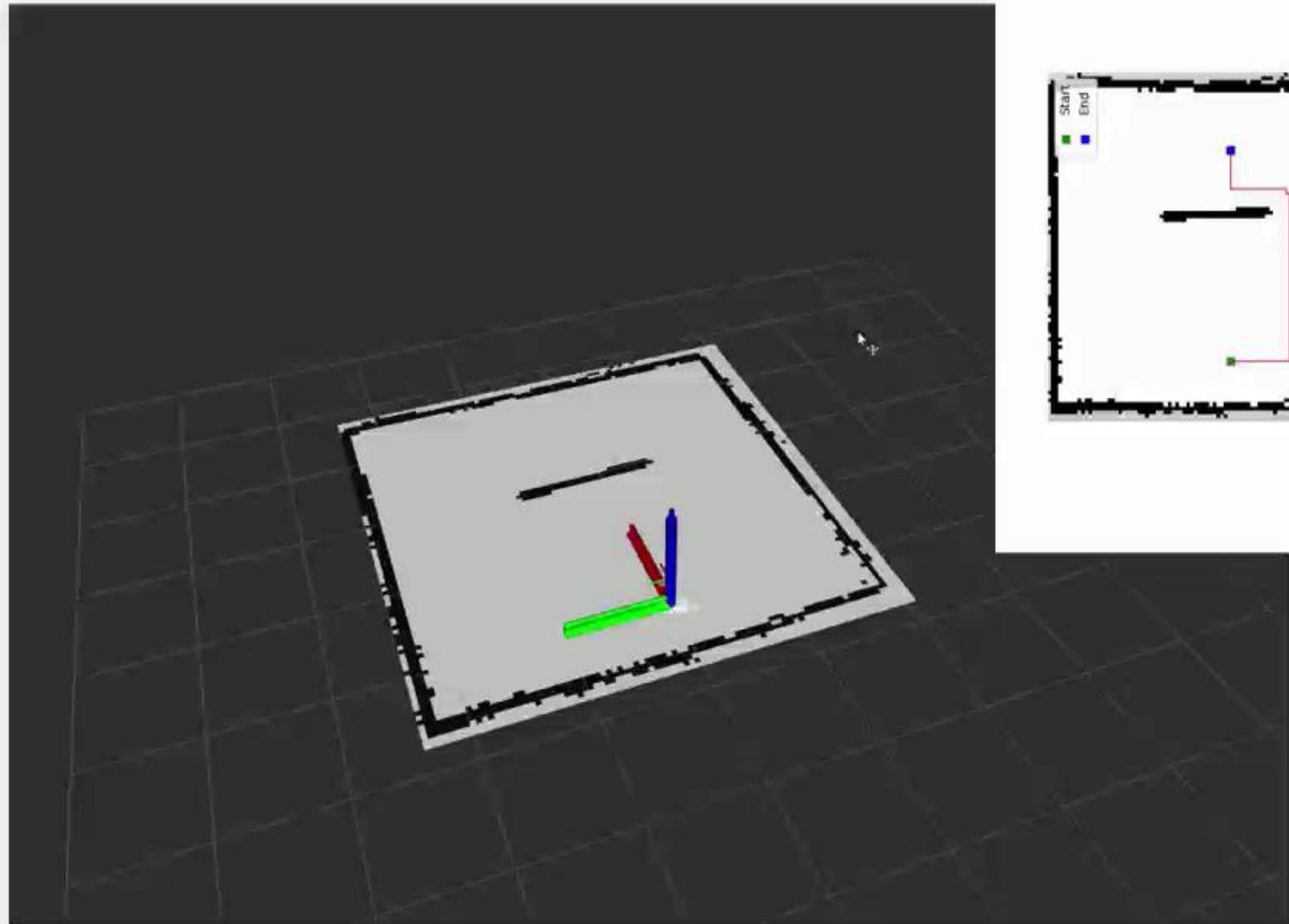
### Skill Affordances with Value Functions



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



Zero

rbit (rviz)  
01

Fixed Frames  
1,4359  
05

77357  
670398  
211; -0.269...

Add Remove

Save Remove Rename

Time

ROS Time: 1695114543.50    ROS Elapsed: 12.22    Wall Time: 1695114543.53    Wall Elapsed: 12.22

Reset: Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel: Zoom. Shift: More options.

Experimental

31 fps

- **Autonomous (自主)**: 只需要人类通过对话提示需求, 就可以自动执行给定任务, 例如设计规划科学实验、呼叫车辆完成运输、浏览网页获取信息、阅读总结文档、执行代码等
- **Generative (生成式)**: 通过生成人类的经验、经历、记忆, 来实现高级的思维和自省, 规划自身的行为。可以和其他代理进行对话、社交、协作。
- **Embodied (具身)**: 具身代理可以与真实的世界进行物理交互。例如, 给出炒菜的指令, 家政助理机器人将任务规划和分解, 并具体实施取材 (感知)、准备 (控制)、烹饪 (规划和决策) 和服务 (反馈)。

- **有限的上下文长度：** 受限于硬件和模型容量，上下文长度限制了将更多历史信息、指令模板、详细指令提供给LLM，因此限制了自省和思维链。
- **长期规划和任务分解中的挑战：** 对复杂任务进行长期规划并有效探索解决方案空间仍然困难。通过试错进行计划调整仍然效率不高。
- **自然语言接口的可靠性：** 当前的代理系统依赖于自然语言作为LLMs与内存和工具等外部组件之间的接口，会出现自然语言的描述不准确的问题。
- **决策的安全性和可靠性：** 智能代理有时存在幻觉或者叛逆行为，表现为拒绝执行某种指令或者坚持执行错误的决策。

# 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们关注我们的  
CV&NLP, 时序预测, 分布式学习, 强化学习等  
研究方向

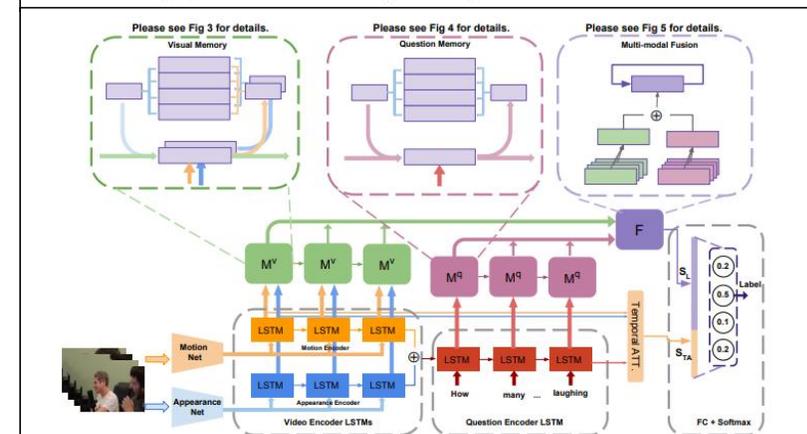
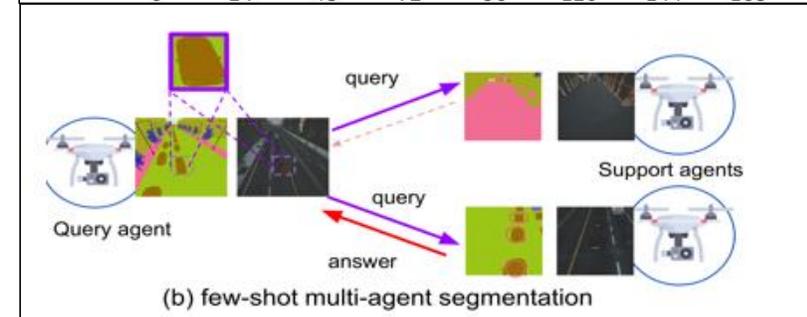
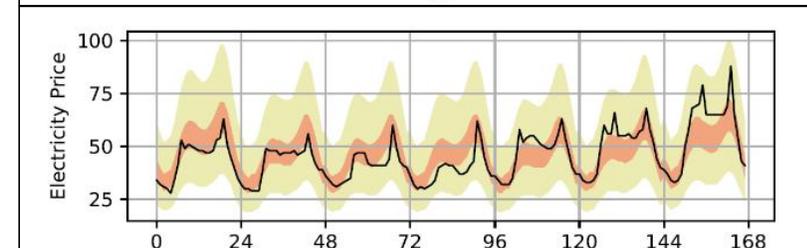
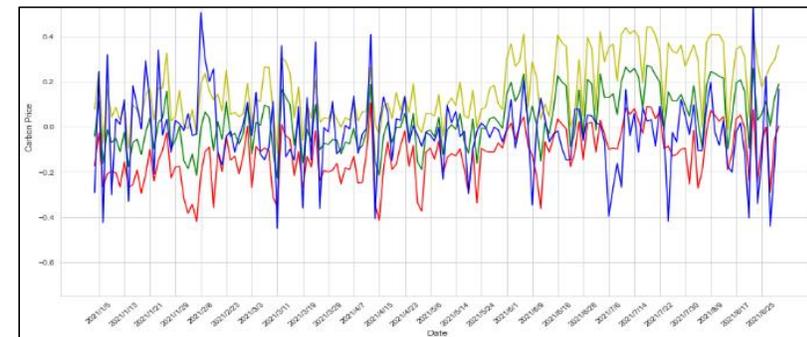


Figure 2. Our proposed VideoQA pipeline with highlighted visual memory, question memory, and multimodal fusion layer.





THANK YOU!