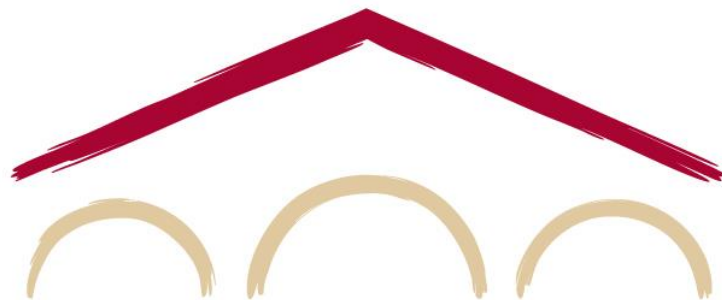


# 自然语言处理 与深度学习

## CS224N/Ling284



Diyi Yang

第10讲：RAG 与语言 Agent

# 概览

上一讲回顾：P E F T - - A d a p t e r s ( 5 分钟 )

1. 问答与 R A G ( 1 5 分钟 )
2. 语言 A g e n t 介绍 ( 5 分钟 )
3. 推理与规划 ( 1 5 分钟 )
4. 记忆 ( 1 5 分钟 )
5. 工具使用 ( 1 0 分钟 )
6. A g e n t 数据和评估 ( 1 5 分钟 )

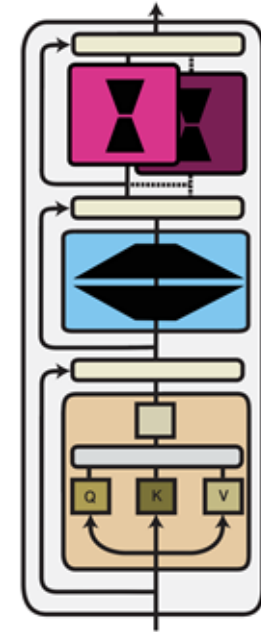
- Redeem credits! Select keywords for your project!
- 本周五 H u g g i n g F a c e 教程

# A functional perspective of adaptation

- Function composition augments a model's functions with **new task-specific functions**:

$$f'_i(\mathbf{x}) = f_{\theta_i}(\mathbf{x}) \odot f_{\phi_i}(\mathbf{x})$$

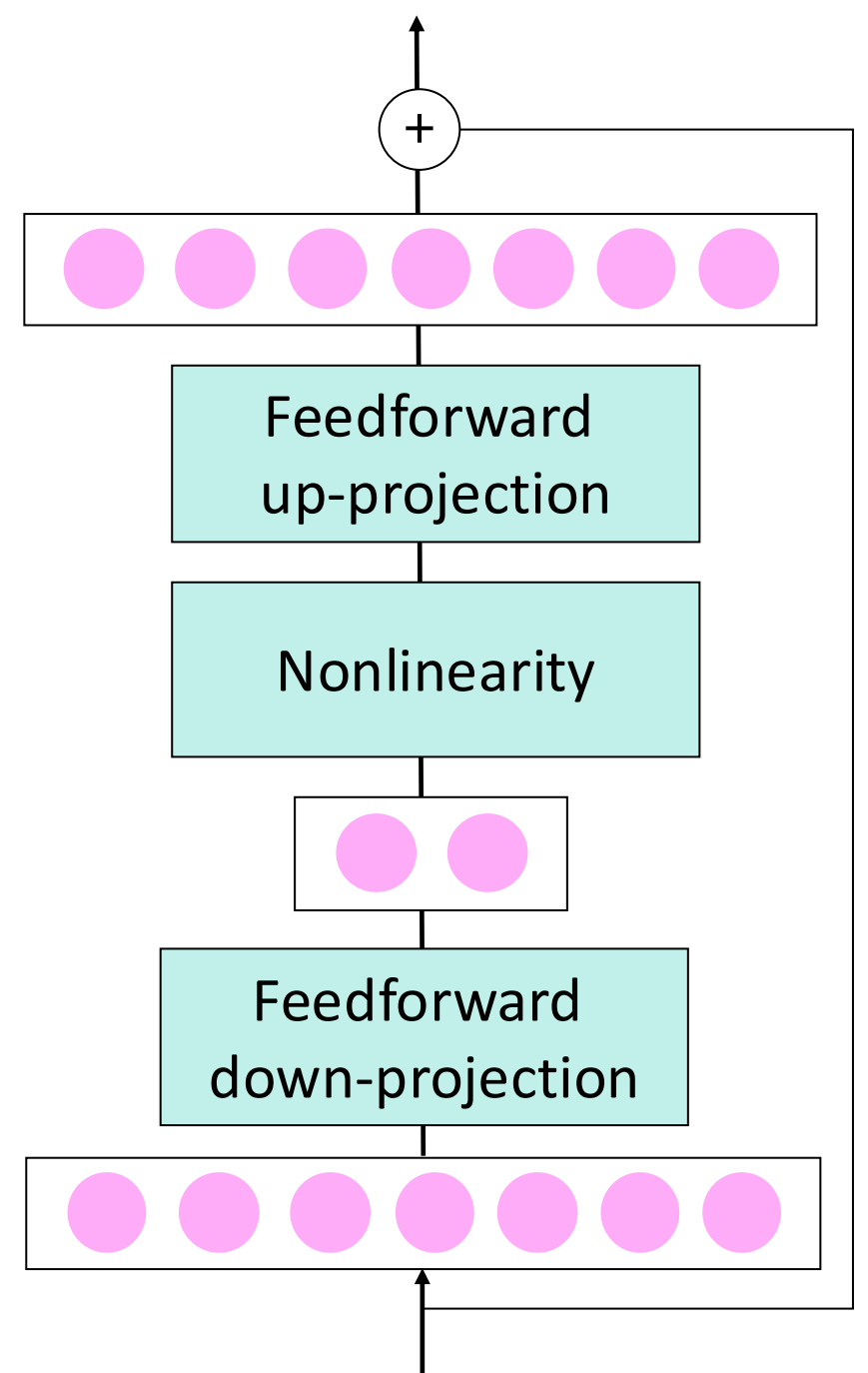
- Most commonly used in multi-task learning where 不同任务模块被组合。



Function  
组合

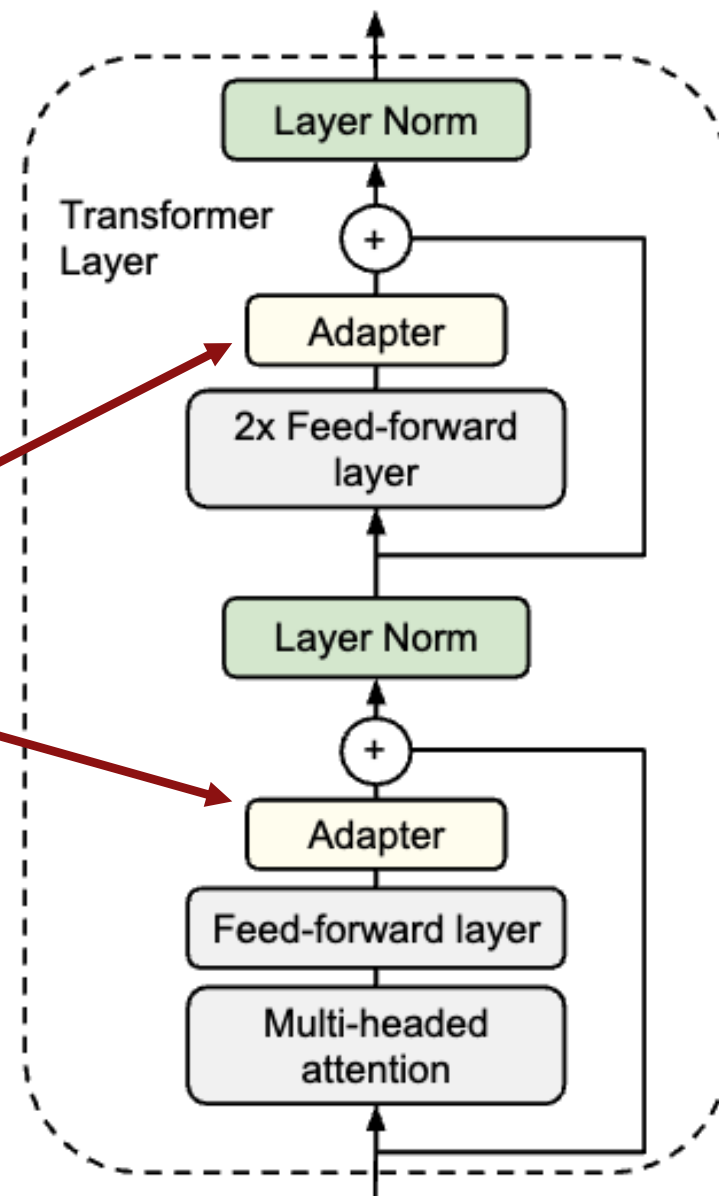
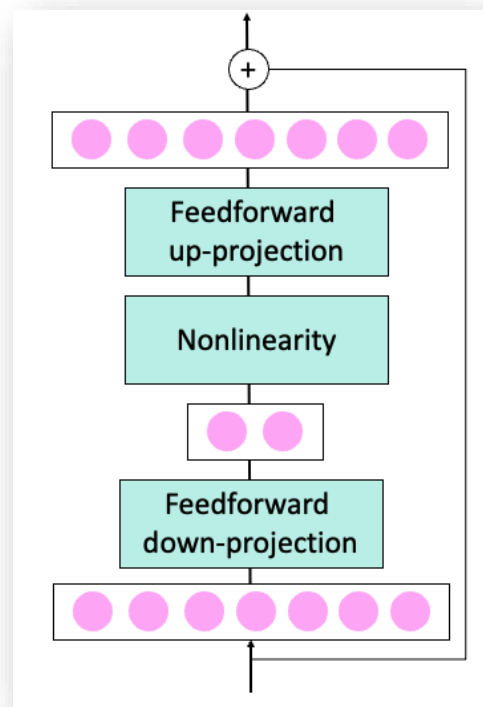
# Adapter (Houlsby et al. 2019)

- Insert a new function  $f_\phi$  预训练模型的层之间 trained model to **adapt to** a downstream task --- known as “adapters”
- An **adapter** in a Transformer layer consists of:
  - A feed-forward down-projection  $W^D \in R^{k \times d}$
  - A feed-forward up-projection  $W^U \in R^{d \times k}$
  - $f_\phi(\mathbf{x}) = W^U(\sigma(W^D \mathbf{x}))$

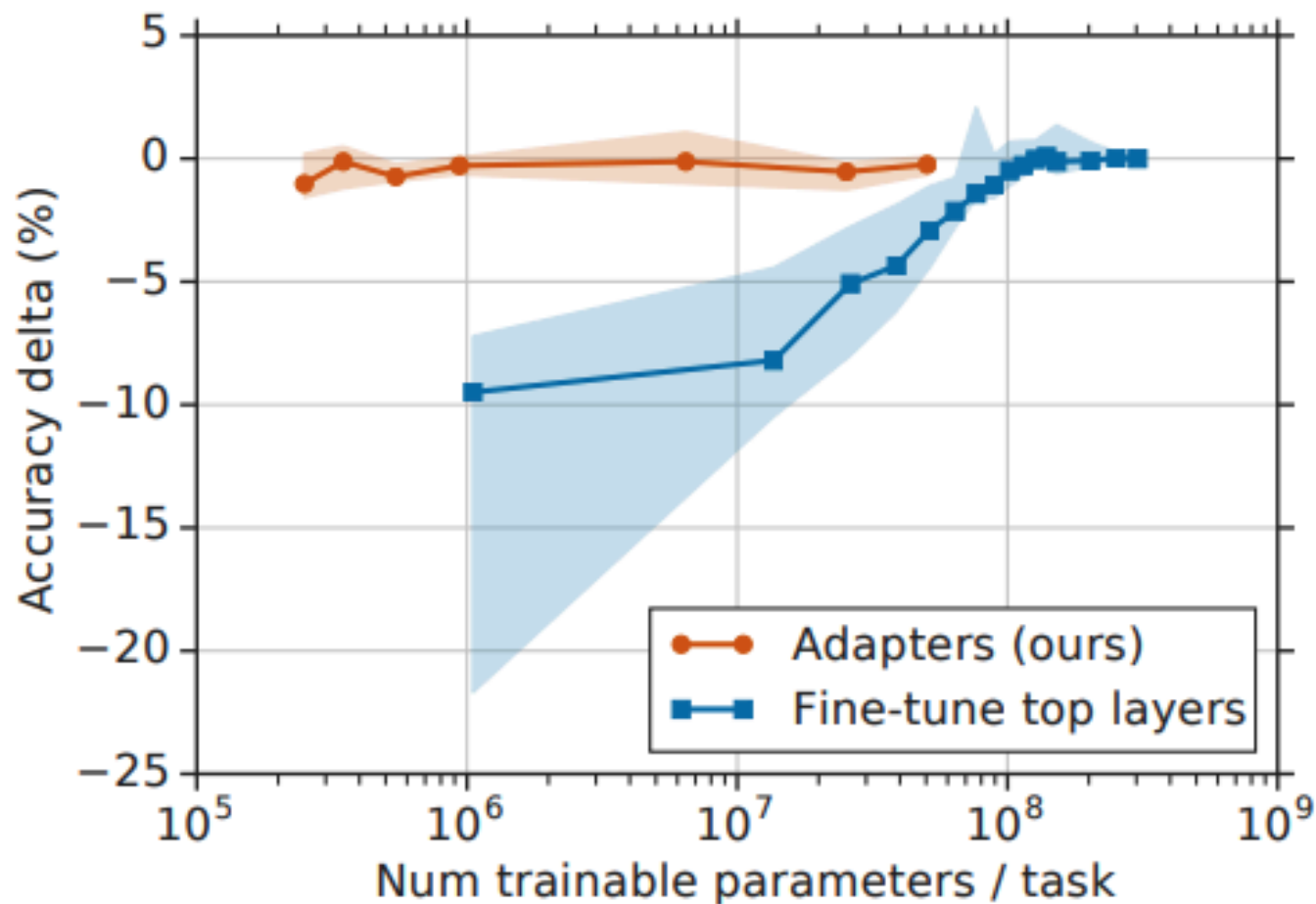


# Adapter (Houlsby et al. 2019)

- The adapter is usually placed after the multi-head attention and/or 在前馈层之后
- Most approaches have used this bottleneck design 线性层中的权重矩阵



## 准确率和参数数量之间的权衡

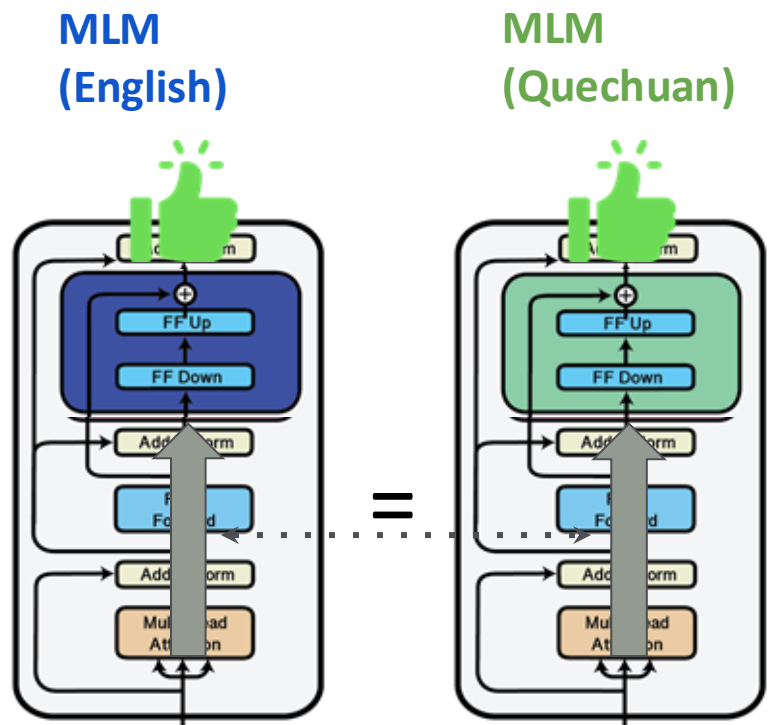


曲线显示第 20、50 和 80 在九个任务上的百分位数来自 GLUE 基准。

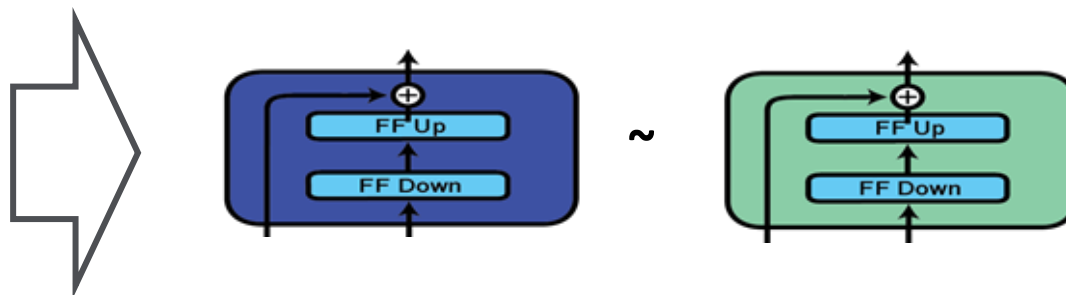
Adapter based tuning attains a similar performance to full finetuning with two orders of 训练 parameters

# 语言 Adapters ? 任务知识

# 语言知识

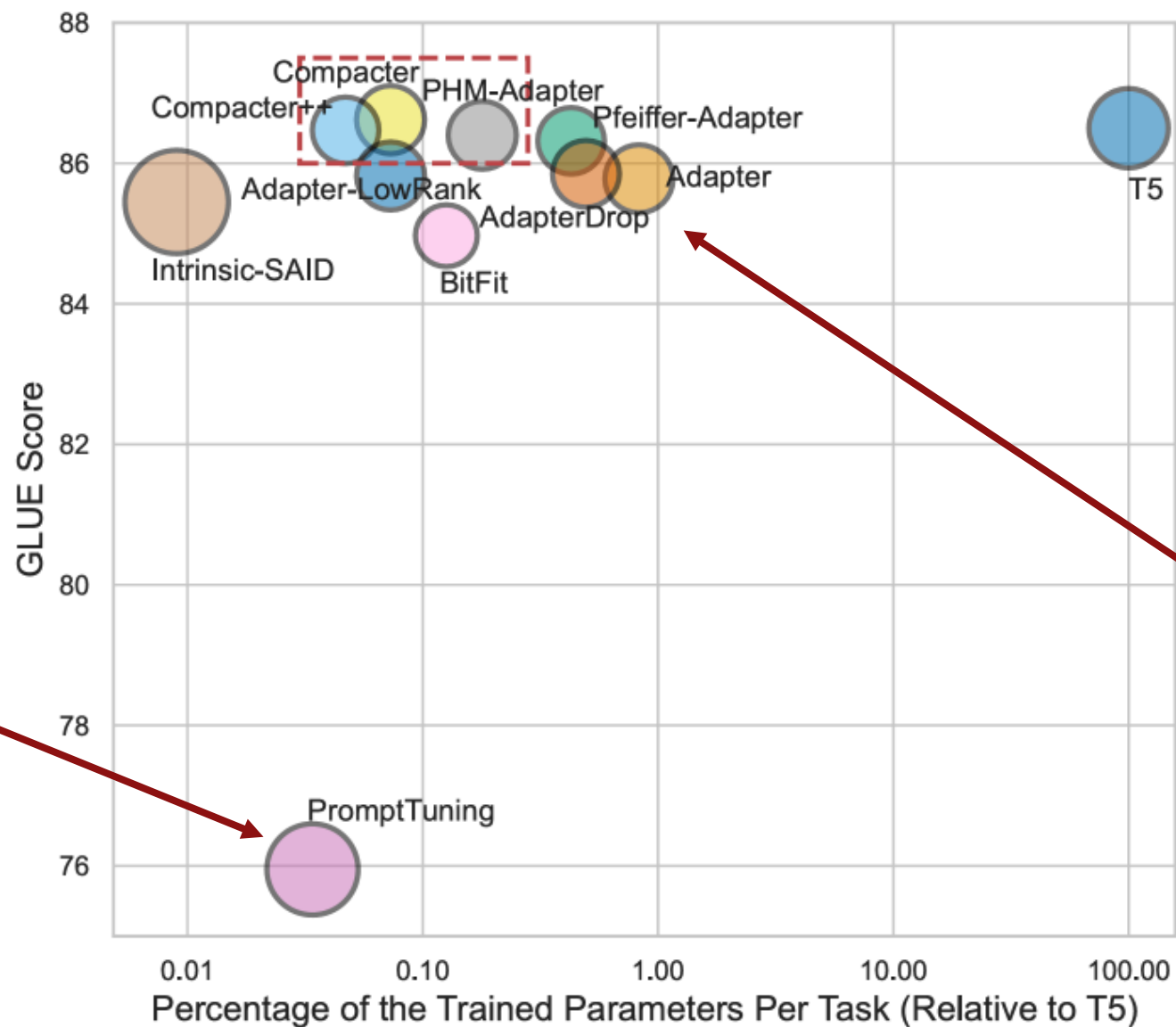


- Adapters 学习变换 that make the underlying model **more suited** to a task or language.
- Using masked language modelling (MLM), we can learn 语言特定的变换 for e.g. **English** and **Quechua**.



# 性能比较

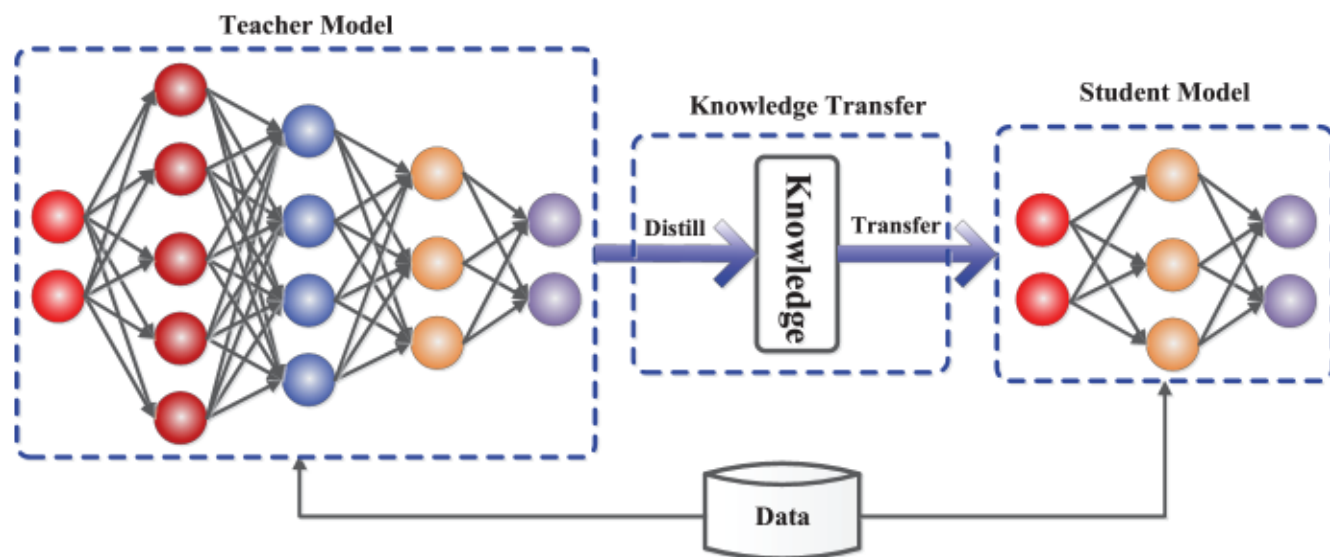
Prompt tuning underperforms the other methods due to limited capacity



Adapter achieves better performance but add more parameters

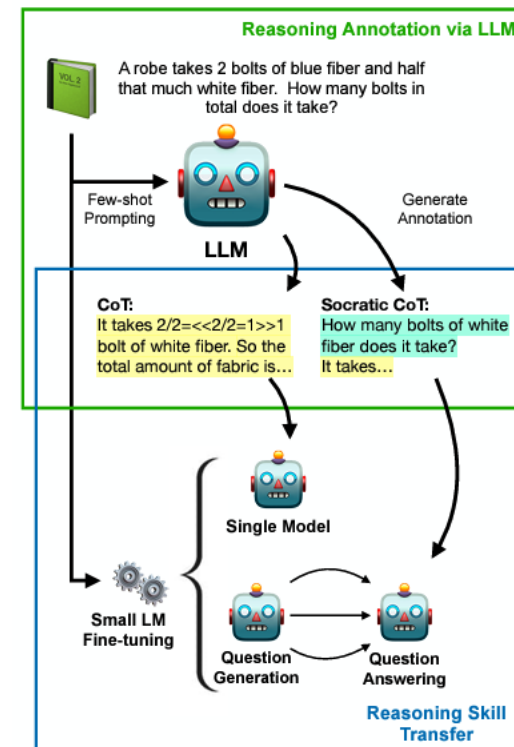
# 其他（高效）适配变体

- **Knowledge distillation** to obtain smaller models



知识蒸馏的通用师生框架 (

[Gou et al.,](#) )



[Shridhar et al., 2023](#)

- 也请查看 : [G i s t T o k e n \(Wu et al, 2024\)](#), [ReFT\(Wu et al, 2024\)](#), etc

# 1 . 问答

## 阅读理解

The first recorded travels by Europeans to China and back date from this time. The most famous traveler of the period was the Venetian Marco Polo, whose account of his trip to "Cambaluc," the capital of the Great Khan, and of life there astounded the people of Europe. The account of his travels, *Il milione* (or, *The Million*, known in English as the *Travels of Marco Polo*), appeared about the year 1299. Some argue over the accuracy of Marco Polo's accounts due to the lack of mentioning the Great Wall of China, tea houses, which would have been a prominent sight since Europeans had yet to adopt a tea culture, as well the practice of foot binding by the women in capital of the Great Khan. Some suggest that Marco Polo acquired much of his knowledge **through contact with Persian traders** since many of the places he named were in Persian.

How did some suspect that Polo learned about China instead of by actually visiting it?

**Answer:** **through contact with Persian traders**

(Rajpurkar et al, 2016): SQuAD: 100,000+ Questions for Machine Comprehension of Text



How many slices of pizza are there?  
Is this a vegetarian pizza?

how many slices of pizza are there? is this a vegetarian pizza?



**ChatGPT**

There are 8 slices of pizza in the image. As for whether it's a vegetarian pizza, it appears to have toppings that could be vegetables, such as pieces that look like diced tomatoes and possibly green peppers. However, without a clearer image or a list of ingredients, it's impossible to be certain if there are no meat products on the pizza. Vegetarian pizzas typically do not include meat, but they can include a variety of vegetable toppings like onions, peppers, mushrooms, olives, and more. If there is any meat or meat-based toppings visible, then it would not be a vegetarian pizza.

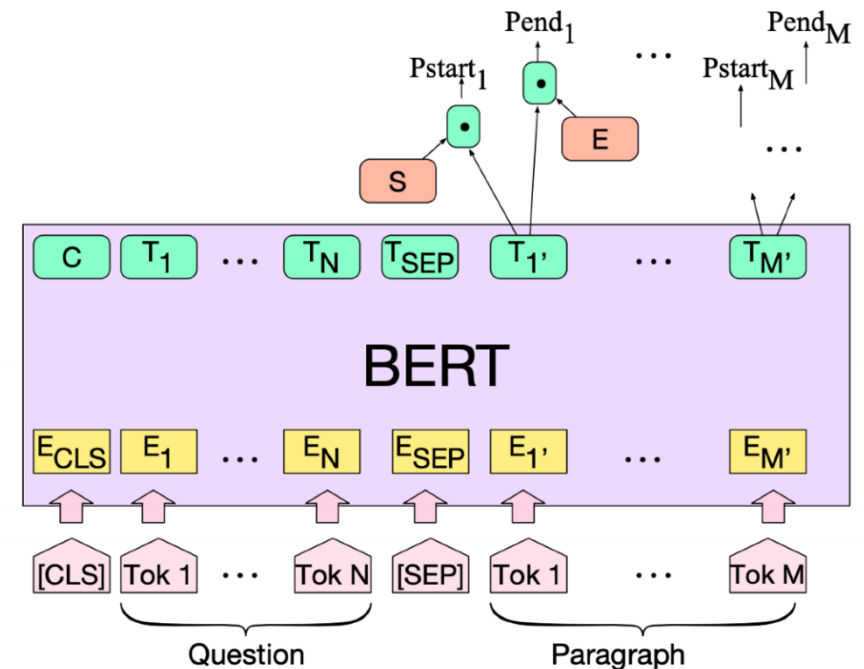
# Conventional methods for reading comprehension

## Feature-based

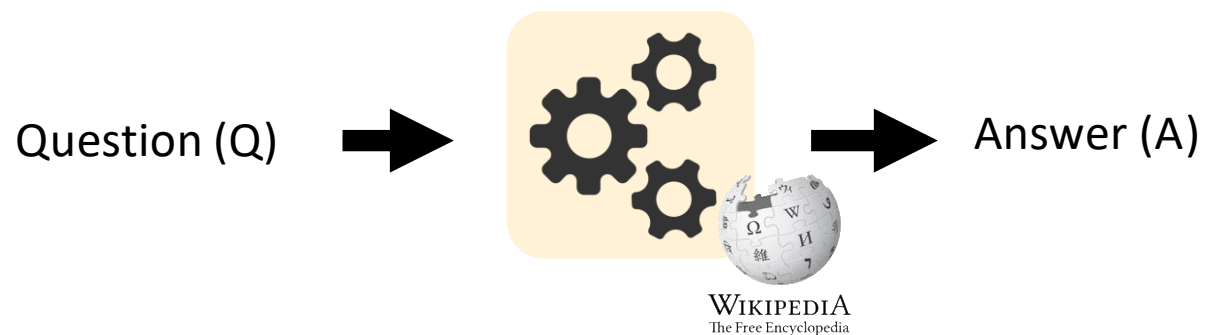
- Generate a list of candidate answers  $(a_1, a_2, \dots, a_M)$
- Define a feature vector  $\phi(p, q, a_i) \in R^d$ : 单词 / 二元组特征 ; 解析树匹配
- 应用多类逻辑回归模型

## Neural approach:

- Problem formulation
  - Input:  $C = (c_1, c_2, \dots, c_N), Q = (q_1, q_2, \dots, q_M)$
  - Output:  $1 \leq \text{start} \leq \text{end} \leq N$
  - $N \sim 100, M \sim 15$



# 开放域问答

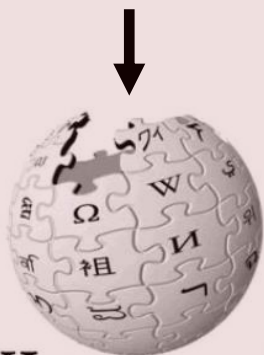


- Different from reading comprehension, we don't assume a given passage.
- Instead, we only have access to a large collection of documents (e.g., Wikipedia). We don't know where the answer is located, and the goal is to return the answer for any open-domain questions.
- 更具挑战性、更实际的问题！

*In contrast to **closed-domain** systems that deal with questions under a specific domain (medicine, technical support).*

# Retrieval augmentation

How many of Warsaw's inhabitants spoke Polish in 1933?



WIKIPEDIA

Document  
检索器



文档  
阅读器



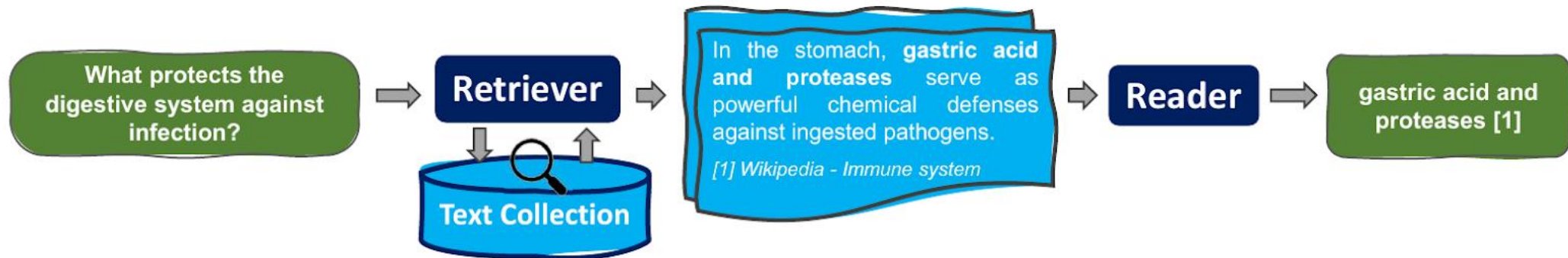
833,500



<https://github.com/facebookresearch/DrQA>

# Using retrieval to overcome LMs' shortcomings

- Instead of asking the LM to memorize everything, can we provide the LM with relevant 且有用的即时内容？
- Retrieval / search is a common mechanism for identifying such relevant information.
  - 动态 : it's easy to update / add documents to your retrieval system
  - **Interpretable**: LM can generate pointers to retrieved documents that support human verification of its generations (citations)



# Retriever-Reader framework

- Input: a large collection of documents  $\mathcal{D} = D_1, D_2, \dots, D_N$  and  $Q$
- Output: an answer string  $A$

Retriever:  $f(\mathcal{D}, Q) \rightarrow P_1, \dots, P_K$        $K$  is pre-defined (e.g., 100)

Reader:  $g(Q, \{P_1, \dots, P_K\}) \rightarrow A$       一个阅读理解问题！

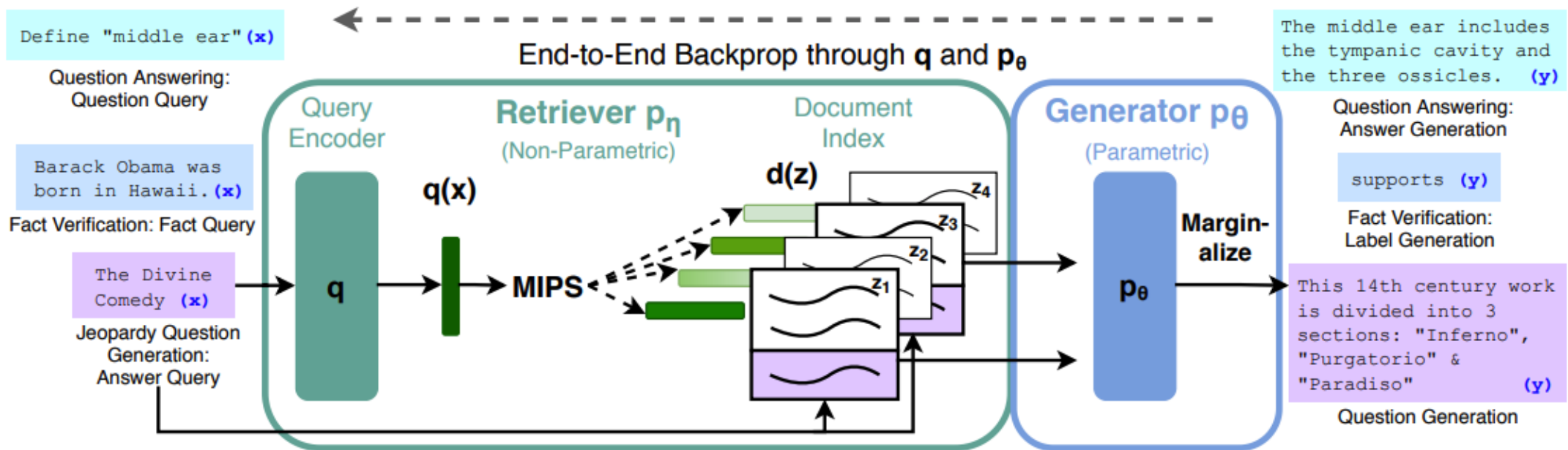
检索器 = 标准 TF - IDF 信息检索稀疏模型 (固定模块)

阅读器 = 神经阅读理解模型

- 可在 SQuAD 等远程监督 QA 数据集上训练
- Or a zero-shot LLM (ChatGPT etc)

# RAG and its open problems

Retrieval Augmented Generation (RAG) is very powerful!



See also works like REALM, DPR, ORQA etc..

[[Lewis et al., 2021](#)]

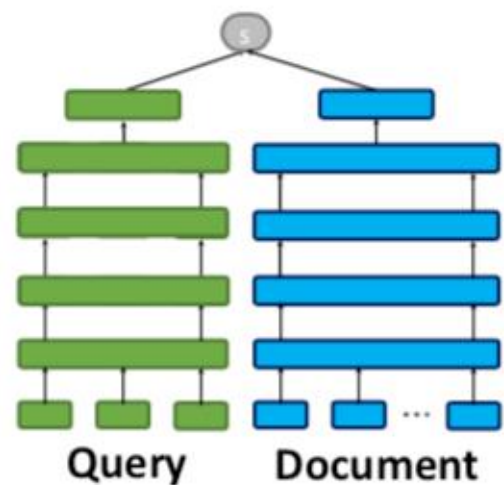
# 不同类型的检索器

如何为检索系统检索相关段落？

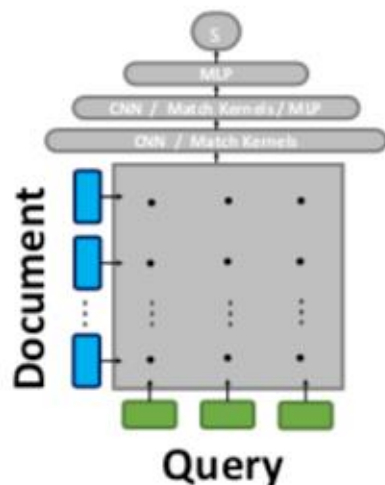
- Word-overlap (BM25)
- Vector retrieval (DPR, Sentence vectors)
- Other, neural systems (ColBERT)

# 快速 v s 慢速检索器

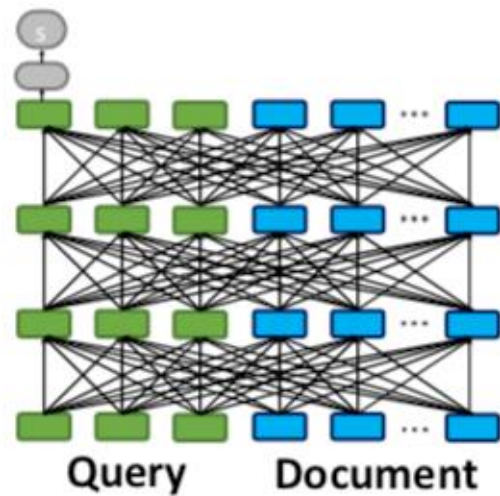
- 最快：在预计算向量上计算相似度
- 最慢：使用语言模型计算相似度
- Other hybrid (pre-computing the 'document index') e.g. in ColBERT



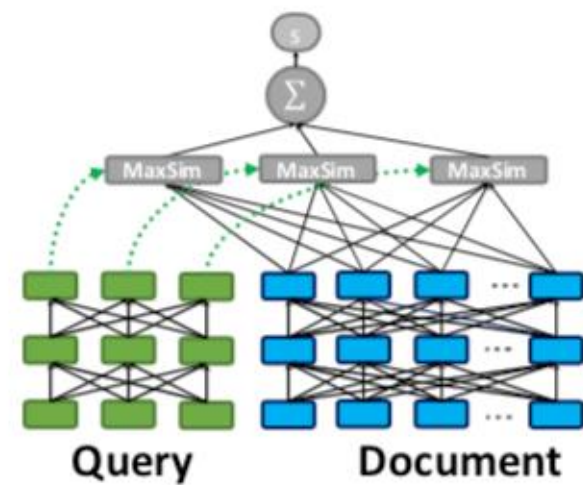
(a) Representation-based Similarity  
(e.g., DSSM, SNRM)



(b) Query-Document Interaction  
(e.g., DRMM, KNRM, Conv-KNRM)

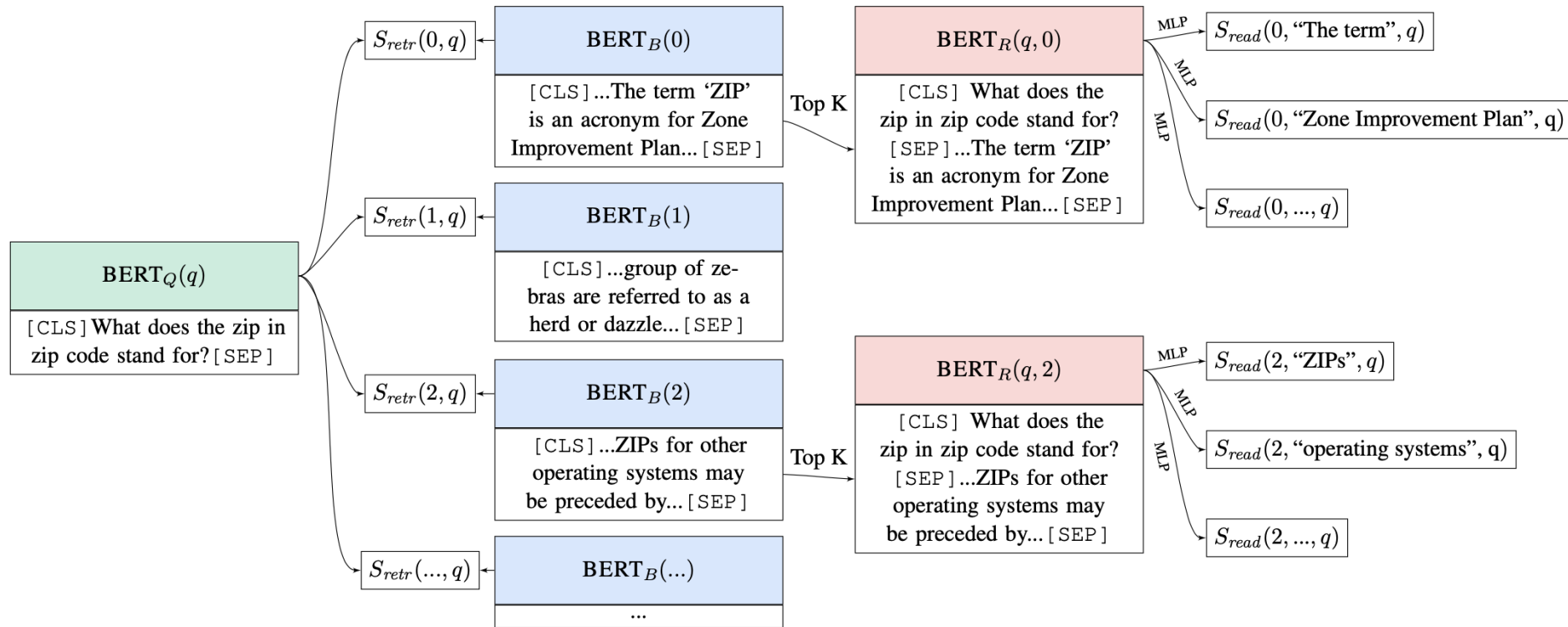


(c) All-to-all Interaction  
(e.g., BERT)



(d) Late Interaction  
(i.e., the proposed ColBERT)

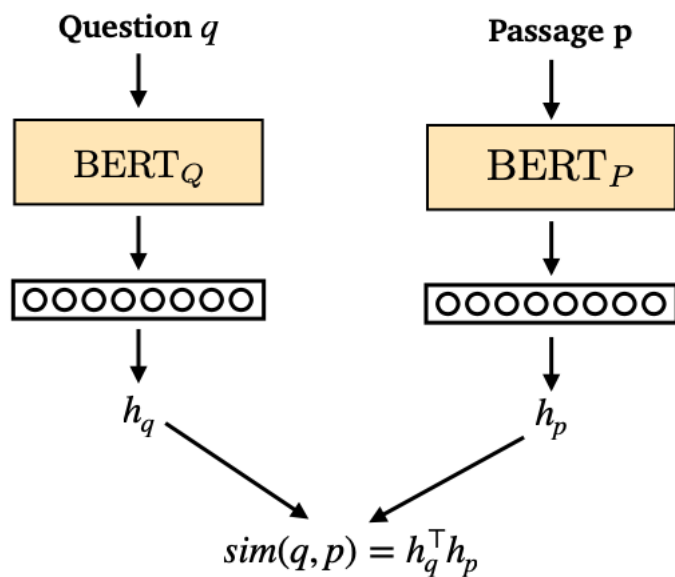
# We can train the retriever! Joint training of reader and retriever



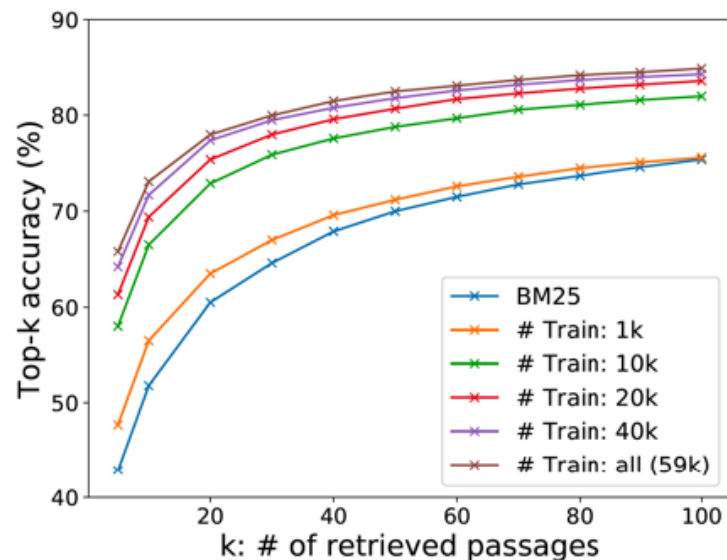
- Each text passage can be encoded as a vector using BERT and the retriever score can be measured as the 问题表示和段落表示之间的点积衡量。
- 然而建模不容易，因为有大量段落（如英文 Wikipedia 2100万）

# 我们可以训练检索器！

- 密集段落检索 ( D P R )
- 也可以仅用问答对训练检索器！



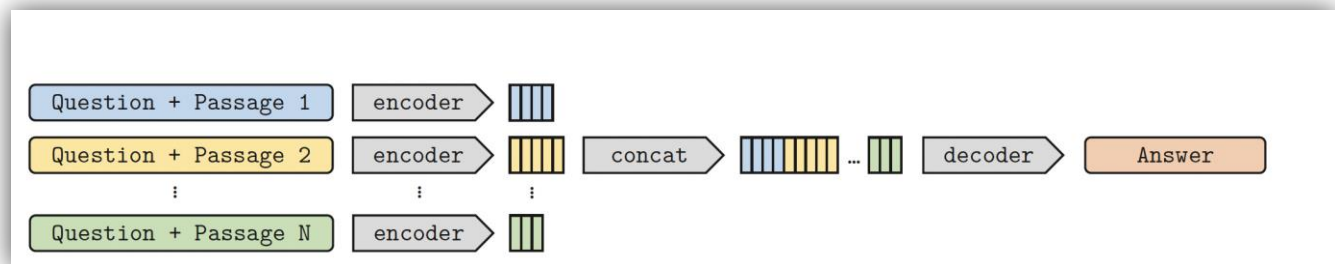
1k Q/A pairs beat BM25!



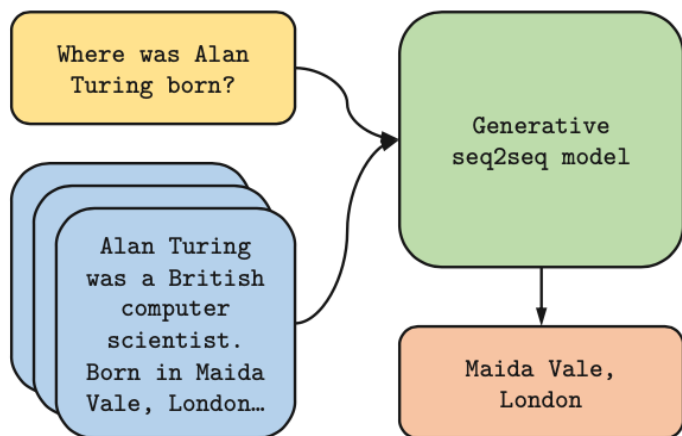
- 可训练检索器 ( 使用 B E R T ) 大幅超越传统 I R 模型

# Deep retrieval + generative models

- Recent work shows that it is beneficial to generate answers instead of to 抽取答案更有益。



Fusion-in-decoder (FID)=DPR+T5



| Model                               | NaturalQuestions | TriviaQA         |
|-------------------------------------|------------------|------------------|
| ORQA (Lee et al., 2019)             | 31.3             | 45.1 -           |
| REALM (Guu et al., 2020)            | 38.2             | - -              |
| DPR (Karpukhin et al., 2020)        | 41.5             | 57.9 -           |
| SpanSeqGen (Min et al., 2020)       | 42.5             | - -              |
| RAG (Lewis et al., 2020)            | 44.5             | 56.1 68.0        |
| T5 (Roberts et al., 2020)           | 36.6             | - 60.5           |
| GPT-3 few shot (Brown et al., 2020) | 29.9             | - 71.2           |
| Fusion-in-Decoder (base)            | 48.2             | 65.0 77.1        |
| Fusion-in-Decoder (large)           | <b>51.4</b>      | <b>67.6 80.1</b> |

Izcard and Grave 2020. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering

# 问题：我们能使用多少文档？

检索器是关键——如果只能用一个文档，必须检索正确。

Why not get lots of documents and pass it to the LM?

## Input Context

Write a high-quality answer for the given question using only the provided search results (some of which might be irrelevant).

Document [1] (Title: Asian Americans in science and technology) Prize in physics for discovery of the subatomic particle  $J/\psi$ . Subrahmanyan Chandrasekhar shared...

**Document [2] (Title: List of Nobel laureates in Physics) The first Nobel Prize in Physics was awarded in 1901 to Wilhelm Conrad Röntgen, of Germany, who received...**

Document [3] (Title: Scientist) and pursued through a unique method, was essentially in place. Ramón y Cajal won the Nobel Prize in 1906 for his remarkable...

Question: who got the first nobel prize in physics

Answer:

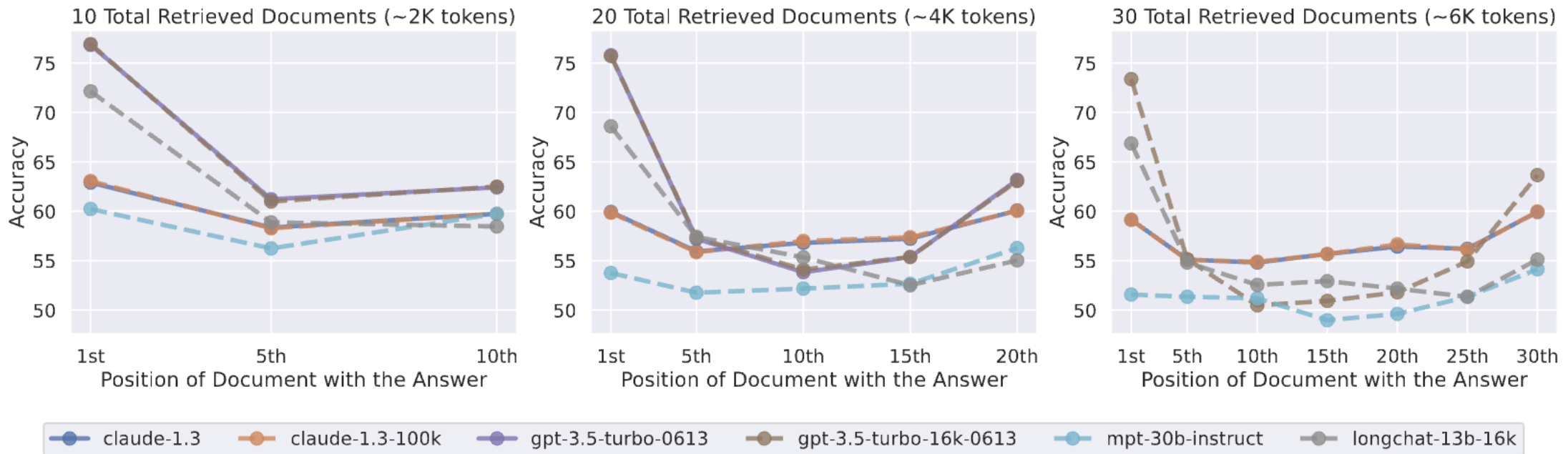
## Desired Answer

Wilhelm Conrad Röntgen

# LM's can't pay attention to the entire context [Liu+ 2023]

长上下文问题——语言模型对上下文关注不够好！

Setup: 1 relevant document, all others irrelevant



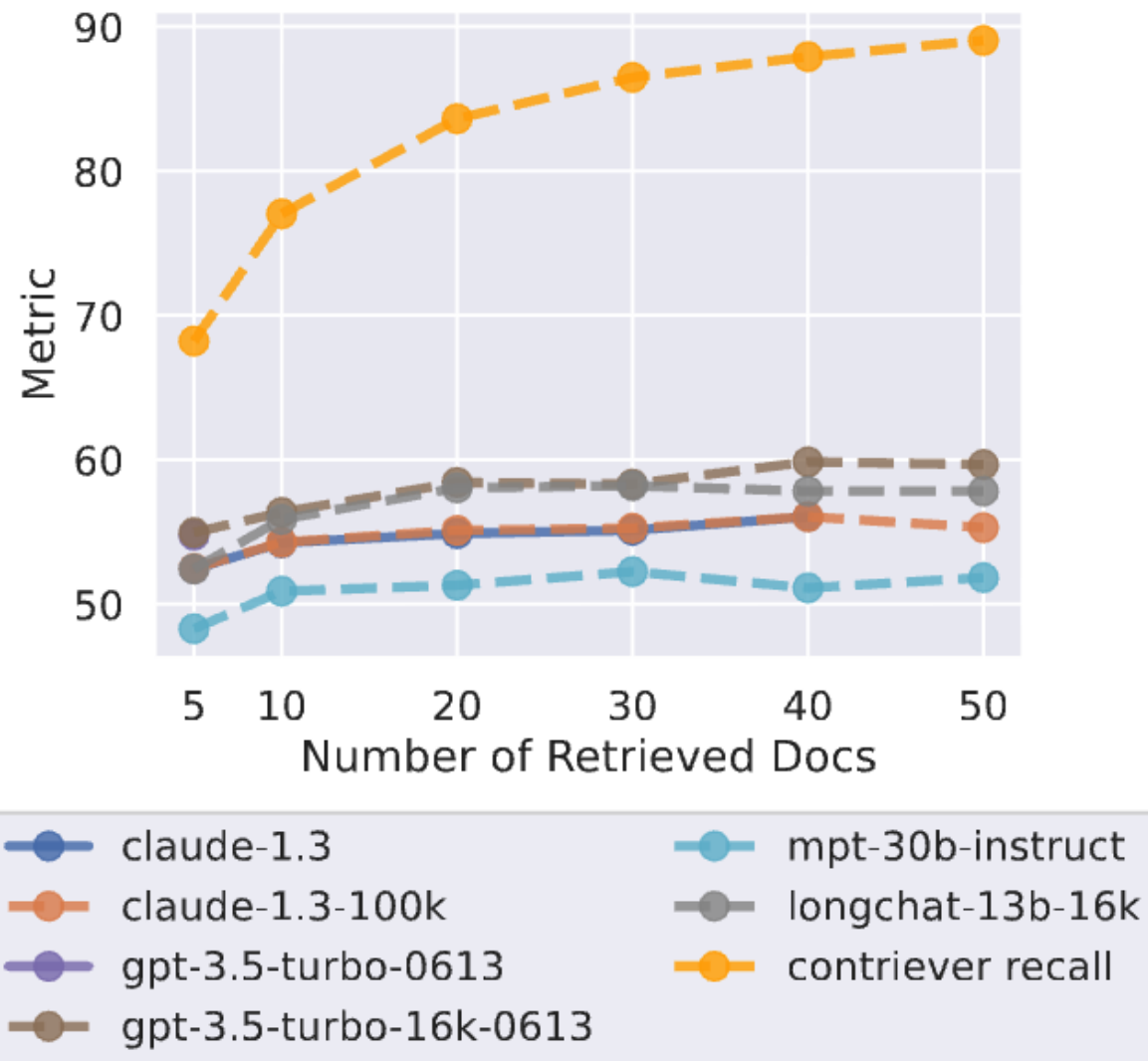
Best Closed-Book performance: GPT-3.5-Turbo, ~56%

Best Oracle (only feed in relevant doc) performance: GPT-3.5-Turbo, ~88.5%

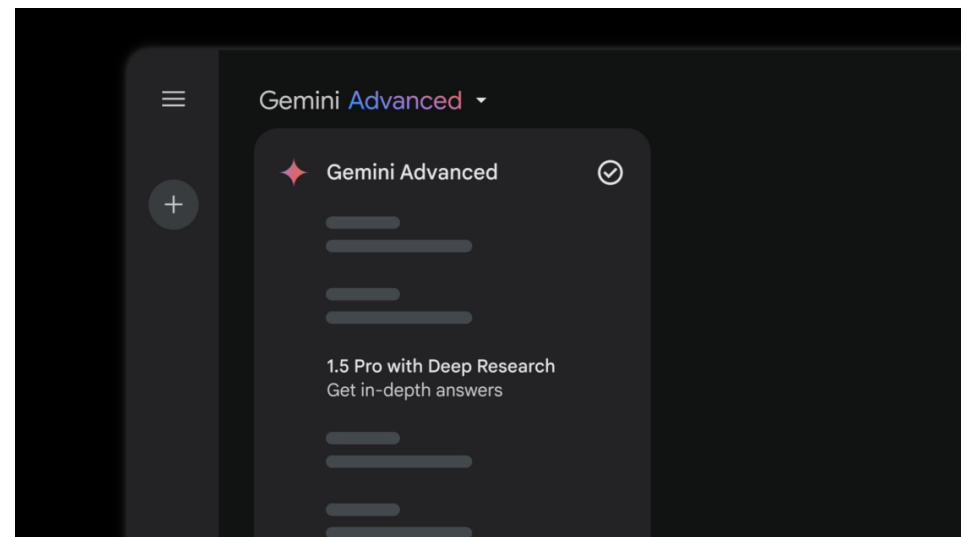
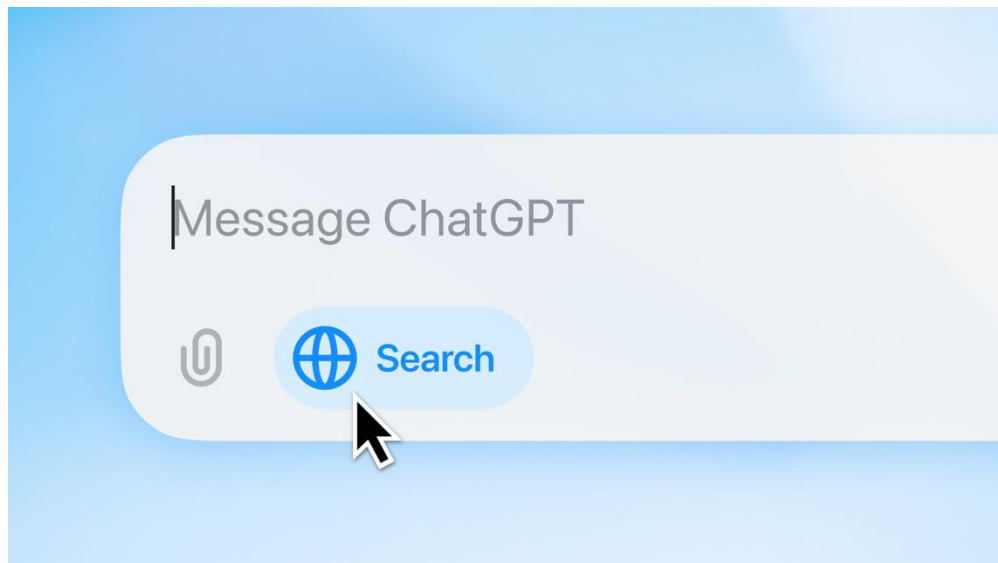
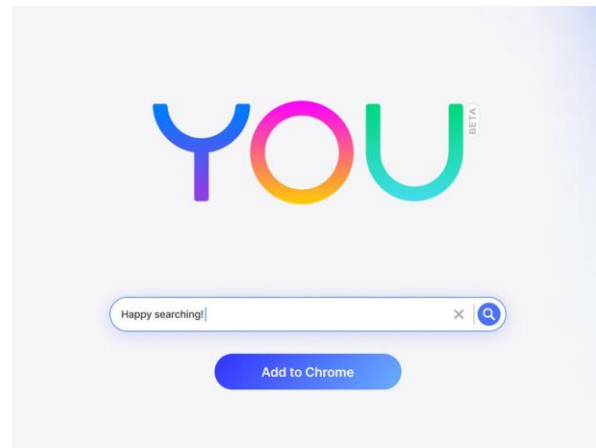
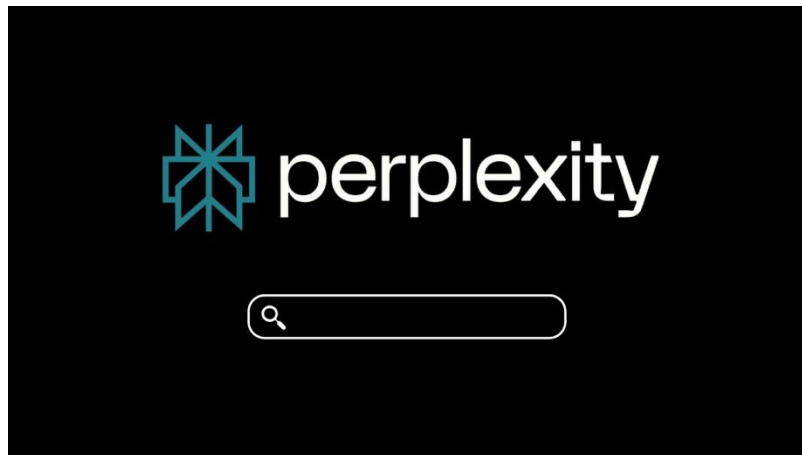
## 实际上：语言模型无法使用太多文档

Retriever performance (yellow)  
rises slowly to 90% recall

RAG performance (other lines)  
saturate very quickly – after 10-  
20 documents.



# LLM 与网络搜索（搜索和引用网络的系统）



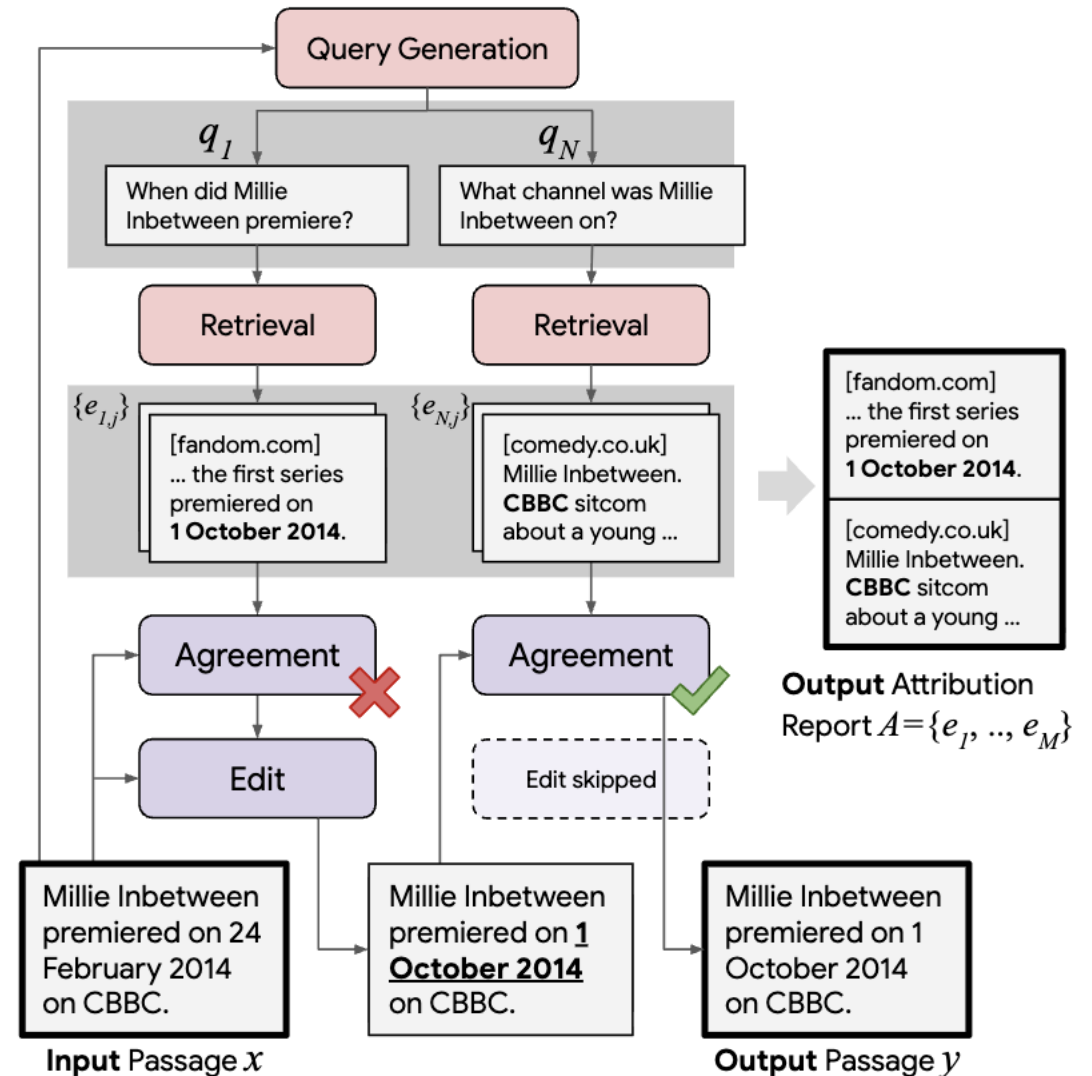
# RARR: research and revise what LLMs say using LLMs [Guo+2023]

Uses QA to check for consistency between sources and responses

Returns citations for each of these items using a LM.

Revises by detecting disagreement between the text and citations, then runs an edit model to revise the text if needed

Evidence snippets are selected to form an attribution report



# LLM 引用与幻觉

Unique benefit of RAG: citing the source

These citations themselves are generated by LLMs

So the citations could *also be hallucinated* 🙄 How often does that happen?

Search: Create an itinerary for exploring Toronto over a span of two days.

Toronto is a vibrant and diverse city with many attractions and activities to suit different tastes and preferences. Here are some suggestions for creating an itinerary for exploring Toronto over two days:

- **Day 1:** Start your day with a visit to the **CN Tower**, the iconic landmark that offers panoramic views of the city and beyond <sup>1</sup>. You can also experience the thrill of walking on the glass floor or the EdgeWalk, the world's highest hands-free walk <sup>1</sup>. Next, head to the nearby **St. Lawrence Market**, one of the world's best food markets, where you can sample a variety of cuisines and local specialties <sup>2</sup>. After lunch, take a stroll along **Queen West**, a trendy neighborhood with eclectic shops, galleries, cafes and street art <sup>3</sup>. In the evening, enjoy a show at one of the many theaters or comedy clubs in the **Entertainment District**, or catch a game at the **Scotiabank Arena** if you're a sports fan.
- **Day 2:** Explore the history and culture of Toronto at the **Royal Ontario Museum**, one of the largest and most comprehensive museums in North America, featuring exhibits on art, natural history, world cultures and more <sup>4</sup>. Then, hop on a ferry to the **Toronto Islands**, a group of islands that offer a relaxing escape from the city, with beaches, parks, trails and amusement rides <sup>3</sup> <sup>5</sup>. You can also rent a bike or kayak to explore the islands at your own pace. For dinner, head to **Chinatown**, one of the largest and most vibrant in North America, where you can find a variety of Asian cuisines and shops <sup>3</sup>.

I hope this helps you plan your trip to Toronto. Have fun! 😊

Learn more:

- 1. [cntower.ca](http://cntower.ca)
- 2. [travel.usnews.com](http://travel.usnews.com)
- 3. [bing.com](http://bing.com)
- 4. [rom.on.ca](http://rom.on.ca)
- 5. [tripadvisor.com](http://tripadvisor.com)

# LLM 引用与幻觉

Unique benefit of RAG: citing the source

These citations themselves are generated by LLMs

So the citations could *also be hallucinated* 🙋 How often does that happen?

Outputs are easy to read / appear useful to rater (1-5 scale)

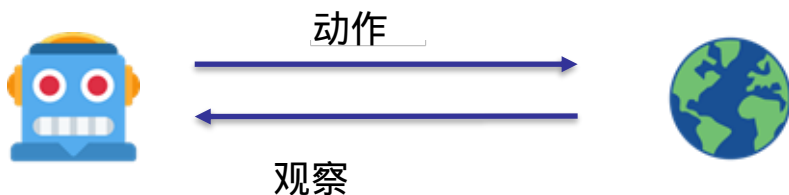
| <u>Perceived Utility (↑)</u> |                          | <u>Fluency (↑)</u> |                          |
|------------------------------|--------------------------|--------------------|--------------------------|
|                              | Average Over All Queries |                    | Average Over All Queries |
| Bing Chat                    | 4.34                     | Bing Chat          | 4.40                     |
| NeevaAI                      | 4.48                     | NeevaAI            | 4.43                     |
| perplexity.ai                | 4.56                     | perplexity.ai      | 4.51                     |
| YouChat                      | 4.62                     | YouChat            | 4.59                     |
| Average                      | 4.50                     | Average            | 4.48                     |

But precision and recall are both low..

| <u>Citation Precision (%; ↑)</u> |                          | <u>Citation Recall (%; ↑)</u> |                          |
|----------------------------------|--------------------------|-------------------------------|--------------------------|
|                                  | Average Over All Queries |                               | Average Over All Queries |
| Bing Chat                        | 89.5                     | Bing Chat                     | 58.7                     |
| NeevaAI                          | 72.0                     | NeevaAI                       | 67.6                     |
| perplexity.ai                    | 72.7                     | perplexity.ai                 | 68.7                     |
| YouChat                          | 63.6                     | YouChat                       | 11.1                     |
| Average                          | 74.5                     | Average                       | 51.5                     |

## 2 . 语言 Agent : 从词语到行动

- LLMs predict textual output for a given input; while agents perform actions based on observations from the environment/world



- Lots of debate on what is an agent and what is not an agent
- Same with the definition of “agentic”
- Here, we focus on key components of LLM empowered systems or language agents

幻灯片致谢 :

语言 Agent : 基础、前景与风险 . Yu Su, Diyi Yang, Shunyu Yao, Tao Yu. EMNLP 2024 Tutorial.

# 语言 Agent : 关键组件

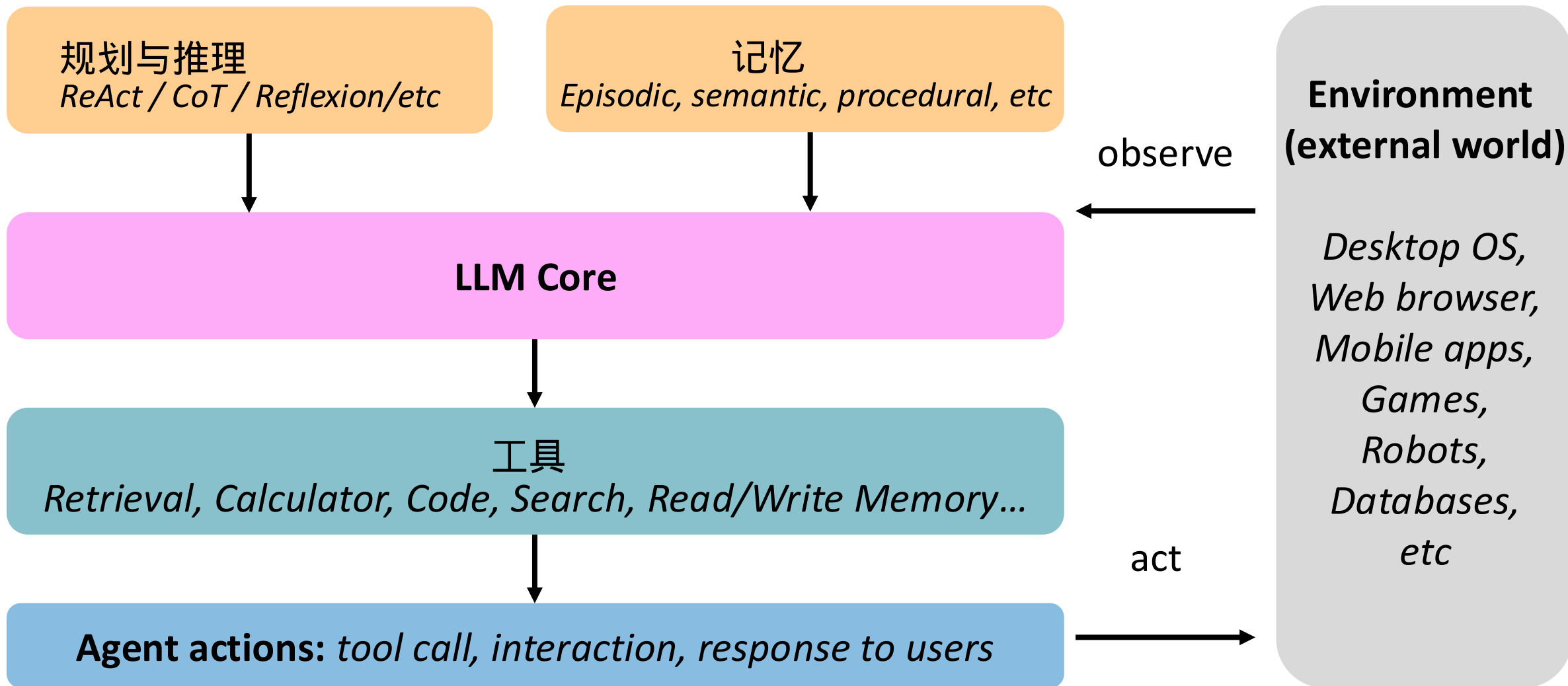
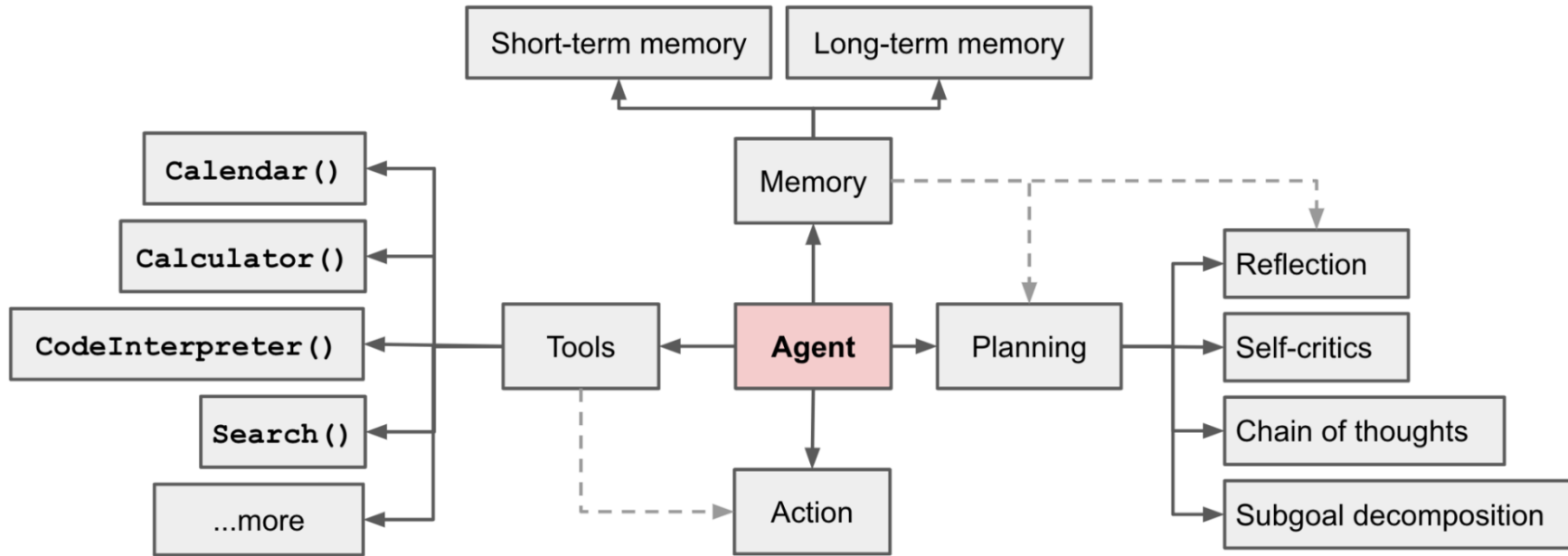
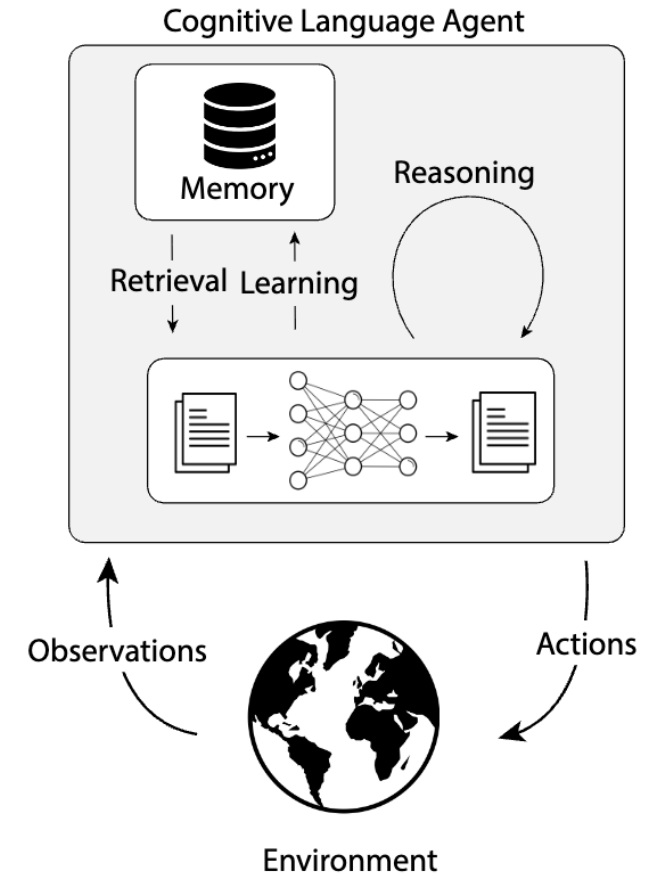


Illustration purpose only; not meant to be the exact process of agent development

# Other types of common components of language agents

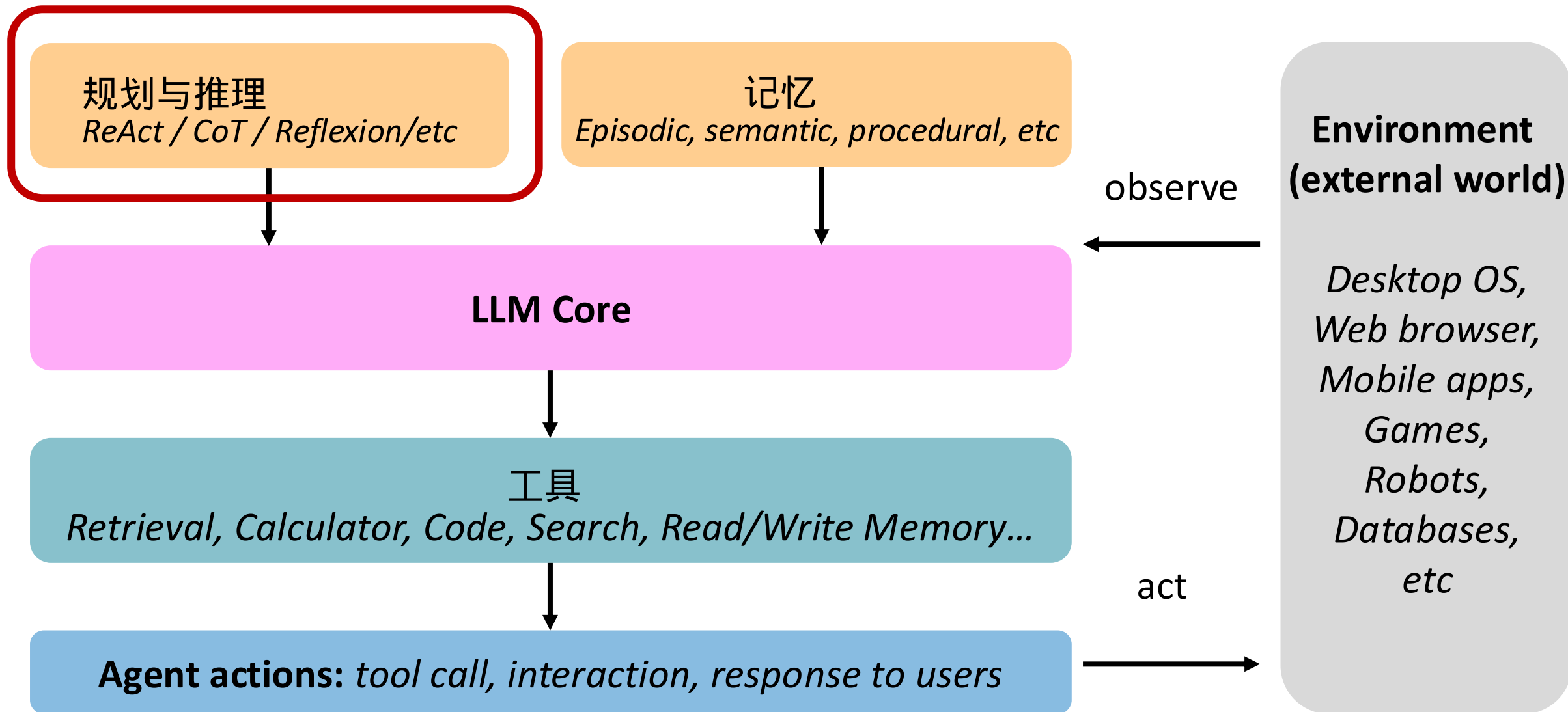


<https://lilianweng.github.io/posts/2023-06-23-agent/>



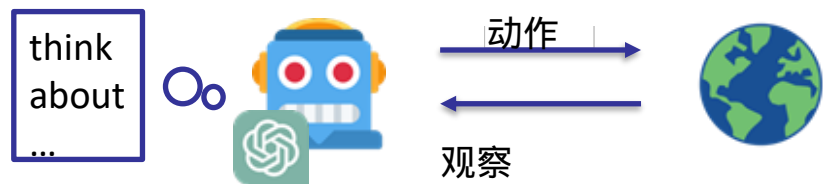
Cognitive Language Agents  
([Sumers and Yao, 2024](#))

### 3 . 语言 Agent : 推理



# Reasoning

- For humans: various mental processes
- For LLMs: intermediate generation that imitate various (but not all) human mental processes
- For agents: internal actions



PERCEPTION

INTUITION  
SYSTEM 1

REASONING  
SYSTEM 1

Math Word Problems (free response)

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.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500?  
Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500.  $9 + 90(2) + 401(3) = 1392$ . The answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about  $0.6 \text{ g/cm}^3$ , which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

# Why is reasoning helpful for agents?

Obs t

你正在做菜，发现盐用完了...

推理

“The dish should be savory, and since salt is out, I should find the soy sauce instead. It is in the cabinet to my right...”

Act t

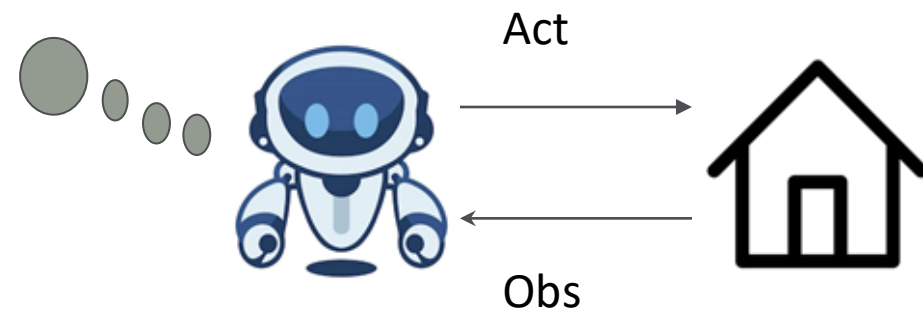
Turn right

Obs t+1

你看到一个柜子和一张桌子...

Act t+1

Open cabinet



推理以行动

1. 泛化
2. Alignment

观察到动作的映射可能很困难

# Reasoning without acting

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

标准

**Answer:** iPod

X

Reason Only

**思考** : 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. 所以答案是 iPhone、iPad 和 iPod Touch。

**答案** : iPhone, iPad, iPod Touch

X

# Reasoning with acting

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

## ReAct (推理 + 行动)

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

**Act 1:** 搜索 [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 .....(long text omitted)

**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:** 搜索 [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:** 搜索 [Front Row (software)]

**Obs 3:** Front Row is a discontinued media center software ...(long text omitted)

**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]



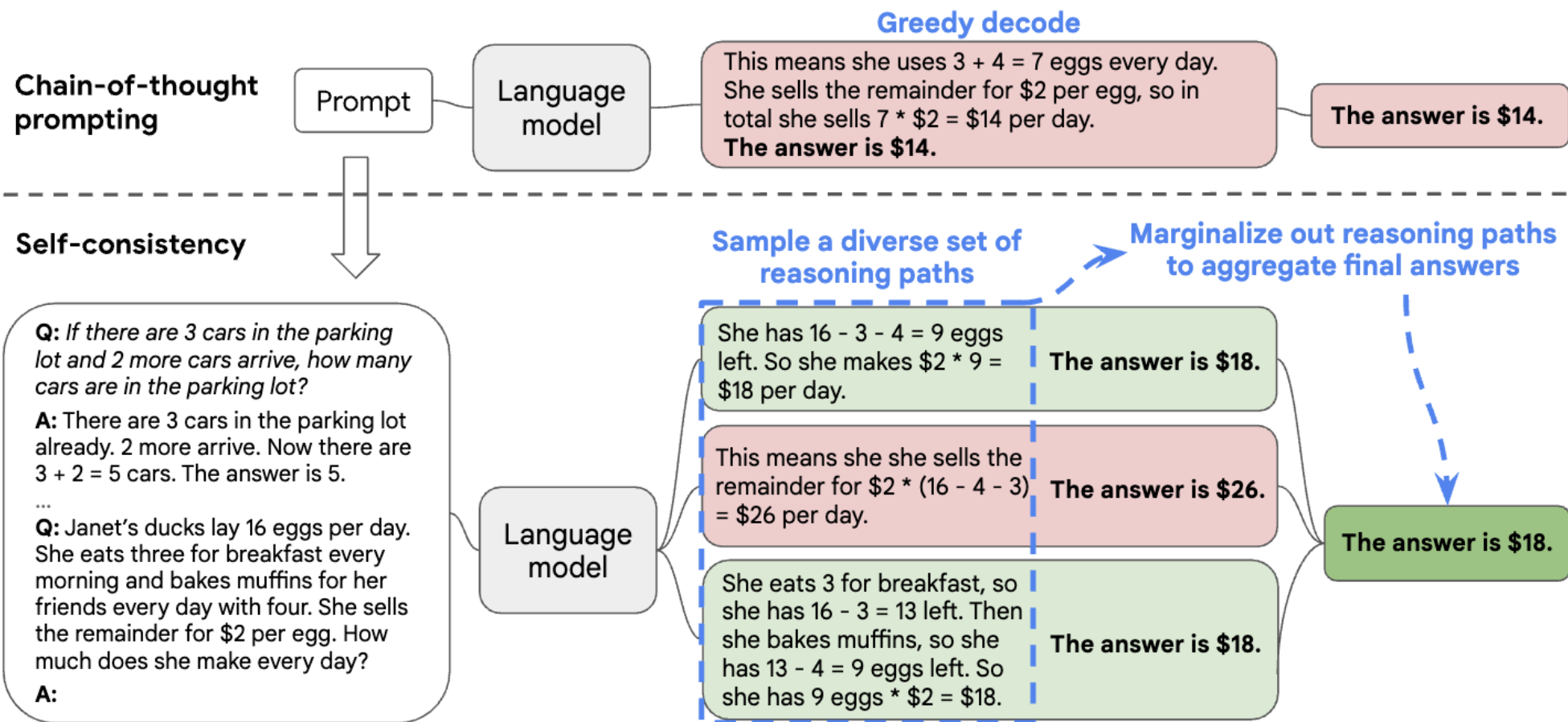
# 比较 ReAct 与 CoT、SC

| Prompt Method <sup>a</sup>         | HotpotQA (EM) | Fever (Acc) |
|------------------------------------|---------------|-------------|
| Standard                           | 28.7          | 57.1        |
| CoT (Wei et al., 2022)             | 29.4          | 56.3        |
| CoT-SC (Wang et al., 2022a)        | 33.4          | 60.4        |
| Act                                | 25.7          | 58.9        |
| ReAct                              | 27.4          | 60.9        |
| CoT-SC → ReAct                     | 34.2          | <b>64.6</b> |
| ReAct → CoT-SC                     | <b>35.1</b>   | 62.0        |
| <b>Supervised SoTA<sup>b</sup></b> | 67.5          | 89.5        |

- ReAct 持续优于 Act
- Hallucination is a serious problem for CoT, ReAct 可以在一定程度上缓解

|         | Type                | Definition   | ReAct | CoT |
|---------|---------------------|--|-------|-----|
| Success | True positive       | Correct reasoning trace and facts  | 94%   | 86% |
|         | False positive      | Hallucinated reasoning trace or facts                                      | 6%    | 14% |
| Failure | Reasoning error     | Wrong reasoning trace (including failing to recover from repetitive steps) | 47%   | 16% |
|         | Search result error | Search return empty or does not contain useful information                 | 23%   | -   |
|         | Hallucination       | Hallucinated reasoning trace or facts                                      | 0%    | 56% |
|         | Label ambiguity     | Right prediction but did not match the label precisely                     | 29%   | 28% |

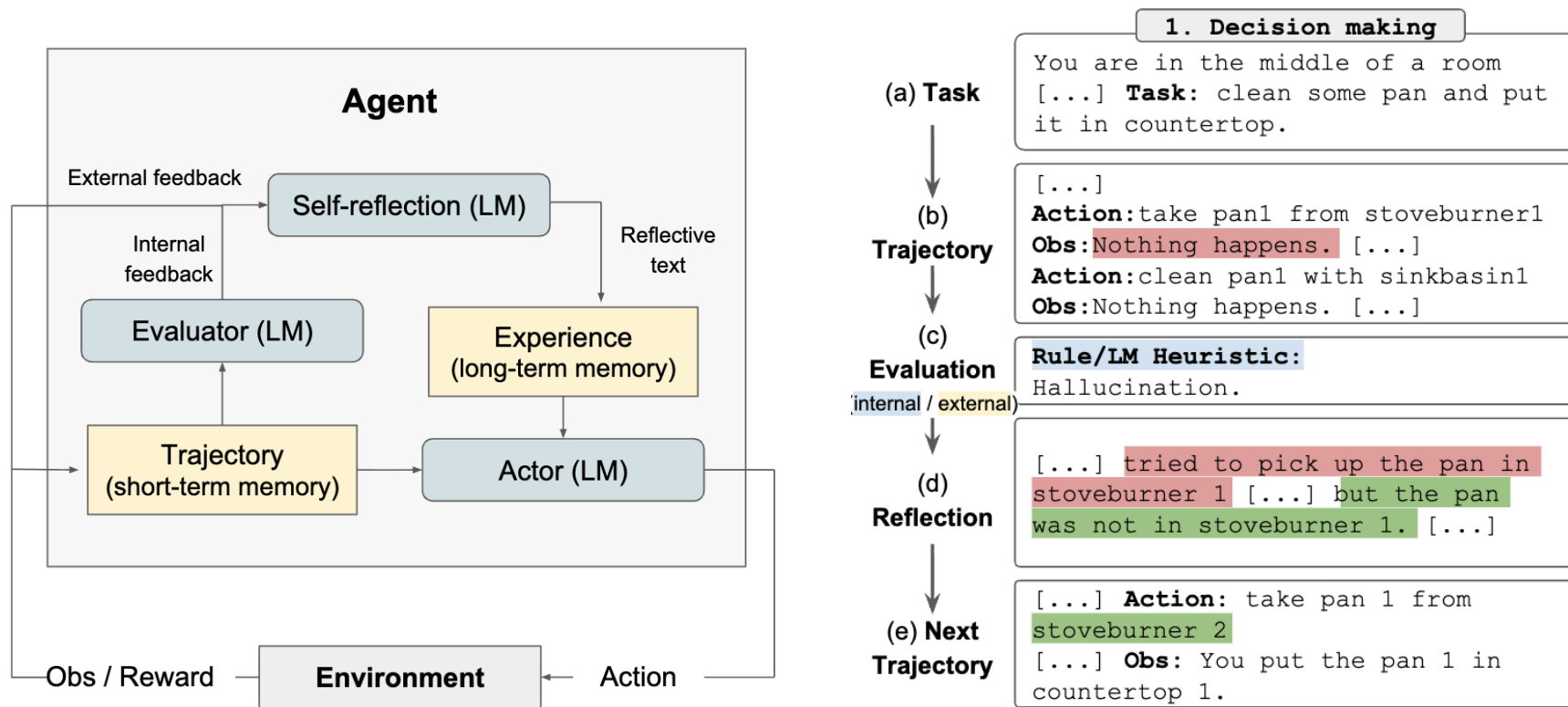
# Self-Consistency [Wang et al., 2023]



- Prompt a language model using chain-of-thought (CoT) prompting;
- 生成多样的推理路径
- 通过选择最一致的答案来聚合。

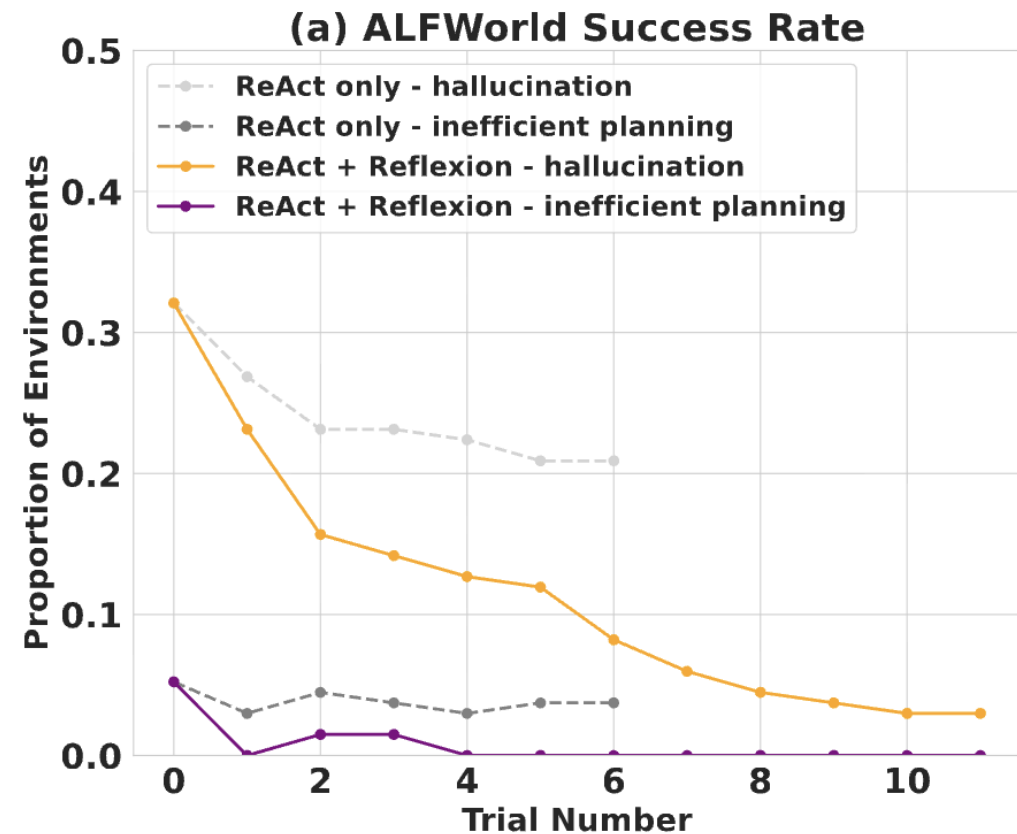
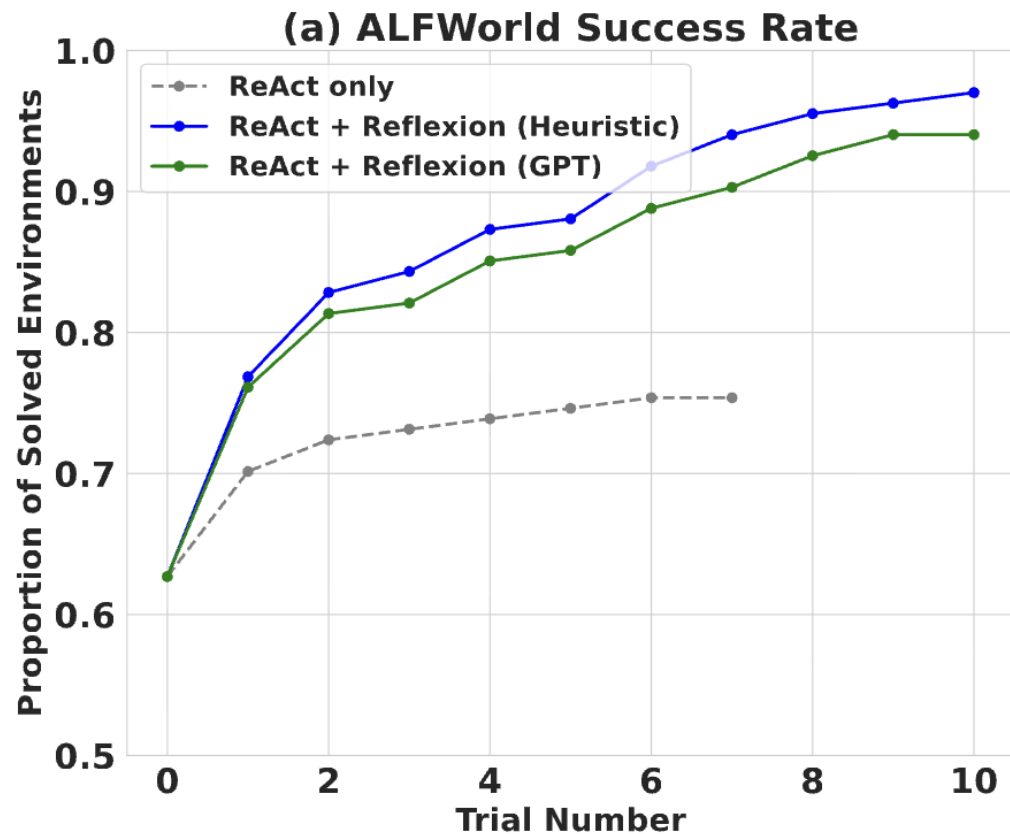
# Reflexion: reinforce agents via natural language feedback

- Reflexion agents verbally reflect on task feedback signals, then maintain their own reflective text to induce better decision-making.



# Reflexion: reinforce agents via natural language feedback

- 相比 ReAct 提升性能
- Reduce error cases by using self-reflection to distill long, failed trajectories into relevant experiences that can be used as "self-hints" in the future



# Multi-agent debate to improve reasoning [[Du et al., 2023](#)]

**User Input:** A treasure hunter found a buried treasure chest filled with gems. There were 175 diamonds, 35 fewer rubies than diamonds, and twice the number of emeralds than the rubies. How many of the gems were there in the chest?

# Multi-agent debate to improve reasoning [Du et al., 2023]

## Round 2

**Agent 1:** Given the information provided in other agents' solutions, we have two answers: One agent found that the number of rubies is 40 and the total number of gems is 225. Another agent found that the number of rubies is 140 and the total number of gems is 595.

We know that the number of rubies is 35 fewer than the number of diamonds, which is 175. So the number of rubies should be  $175 - 35 = 140$ , as the second agent found. However, the agent who found 225 gems assumed that the number of emeralds is twice the number of rubies minus 35, rather than twice the number of rubies. If we use this assumption, we get: Number of rubies = 140. Number of emeralds =  $2 * 140 - 35 = 245$ . Total number of gems =  $175 + 140 + 245 = 560$ . Our final answer is 560. ❌

**Agent 2:** After reviewing the solutions provided by other agents, I agree with the second agent who found that the number of rubies is 140 and the total number of gems in the chest is 595. Therefore, my answer to the math problem is 595. ✅

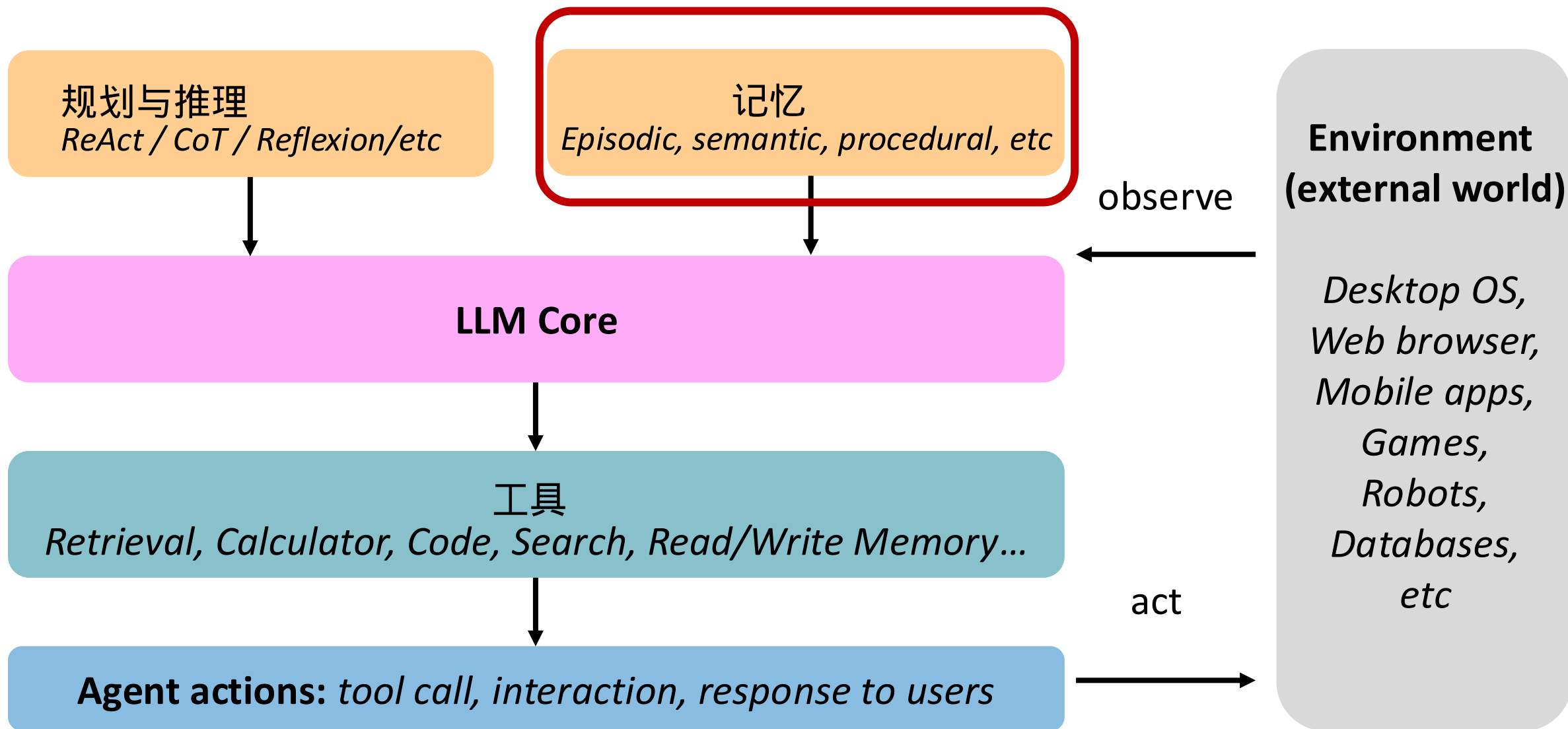
**User Input:** Using the solutions from other agents as additional information, can you give an updated response....

(Copy Agent 2's reply)

(Copy Agent 1's reply)



## 4 . 语言 Agent : 记忆



# 语言 Agent 的不同记忆类型

| Type by content | 定义   | 示例   |
|-----------------|------|--|
| 情景记忆            | 存储经验 | Generative agents<br><a href="#">[Park et al., 2023]</a> |
| 语义记忆            | 存储知识 |  |
| 程序记忆            | 存储技能 | Voyager<br><a href="#">[Wang et al., 2023]</a>           |

Note: here we only categorize based on memory content, which is inspired by [人类长期记忆系统](#)

# 生成式 Agent

## Morning routine



## Catching up



## Packing

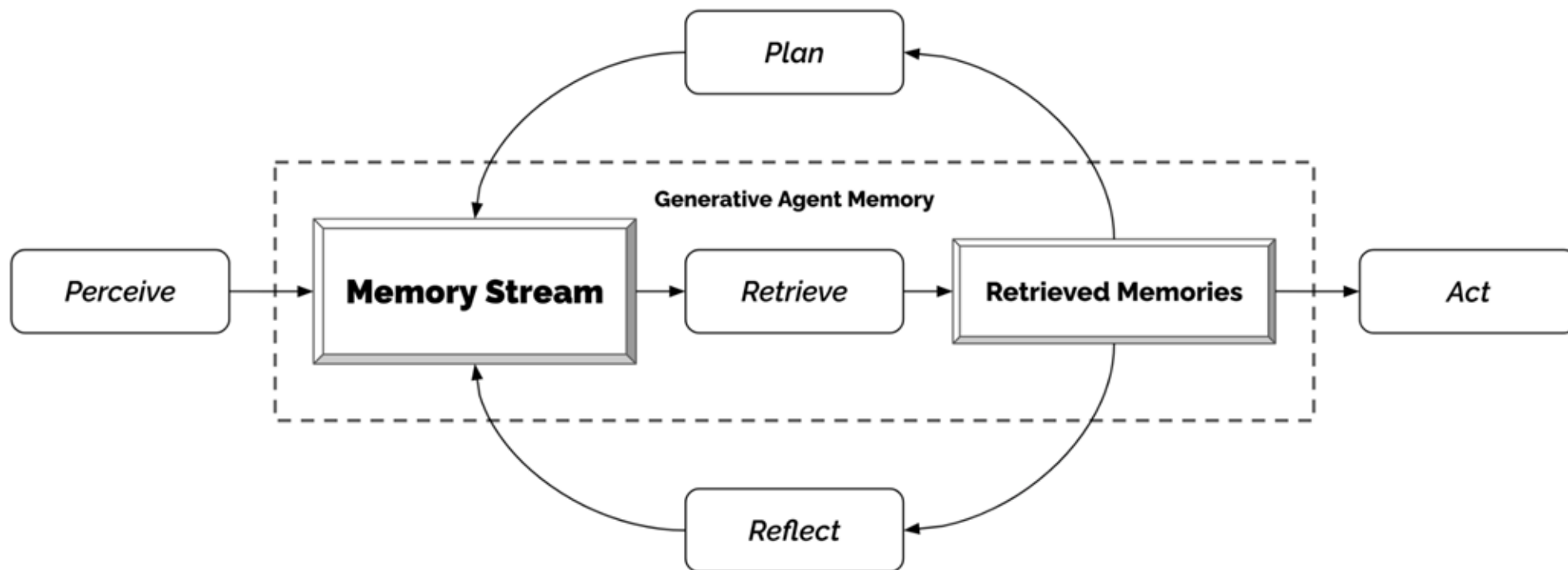


## Beginning workday



# Generative agents: the need for memory

- 上下文窗口不可能容纳所有事件流
- 即使可能，也难以关注或消化相关事件



# 情景记忆

- Write: append-only event streams
- Read: retrieval based on heuristic scores

## Memory Stream

```
2023-02-13 22:48:20: desk is idle
2023-02-13 22:48:20: bed is idle
2023-02-13 22:48:10: closet is idle
2023-02-13 22:48:10: refrigerator is idle
2023-02-13 22:48:10: Isabella Rodriguez is stretching
2023-02-13 22:33:30: shelf is idle
2023-02-13 22:33:30: desk is neat and organized
2023-02-13 22:33:10: Isabella Rodriguez is writing in her journal
2023-02-13 22:18:10: desk is idle
2023-02-13 22:18:10: Isabella Rodriguez is taking a break
2023-02-13 21:49:00: bed is idle
2023-02-13 21:48:50: Isabella Rodriguez is cleaning up the kitchen
2023-02-13 21:48:50: refrigerator is idle
2023-02-13 21:48:50: bed is being used
2023-02-13 21:48:10: shelf is idle
2023-02-13 21:48:10: Isabella Rodriguez is watching a movie
2023-02-13 21:19:10: shelf is organized and tidy
2023-02-13 21:18:10: desk is idle
2023-02-13 21:18:10: Isabella Rodriguez is reading a book
2023-02-13 21:03:40: bed is idle
2023-02-13 21:03:30: refrigerator is idle
2023-02-13 21:03:30: desk is in use with a laptop and some papers on it
```

...

## Q. What are you looking forward to the most right now?

Isabella Rodriguez is excited to be planning a Valentine's Day party at Hobbs Cafe on February 14th from 5pm and is eager to invite everyone to attend the party.

| retrieval | = | recency | importance | relevance |
|-----------|---|---------|------------|-----------|
| 2.34      | = | 0.91    | * 0.63     | * 0.80    |

ordering decorations for the party

|      |   |      |        |        |
|------|---|------|--------|--------|
| 2.21 | = | 0.87 | * 0.63 | * 0.71 |
|------|---|------|--------|--------|

researching ideas for the party

|      |   |      |        |        |
|------|---|------|--------|--------|
| 2.20 | = | 0.85 | * 0.73 | * 0.62 |
|------|---|------|--------|--------|

...

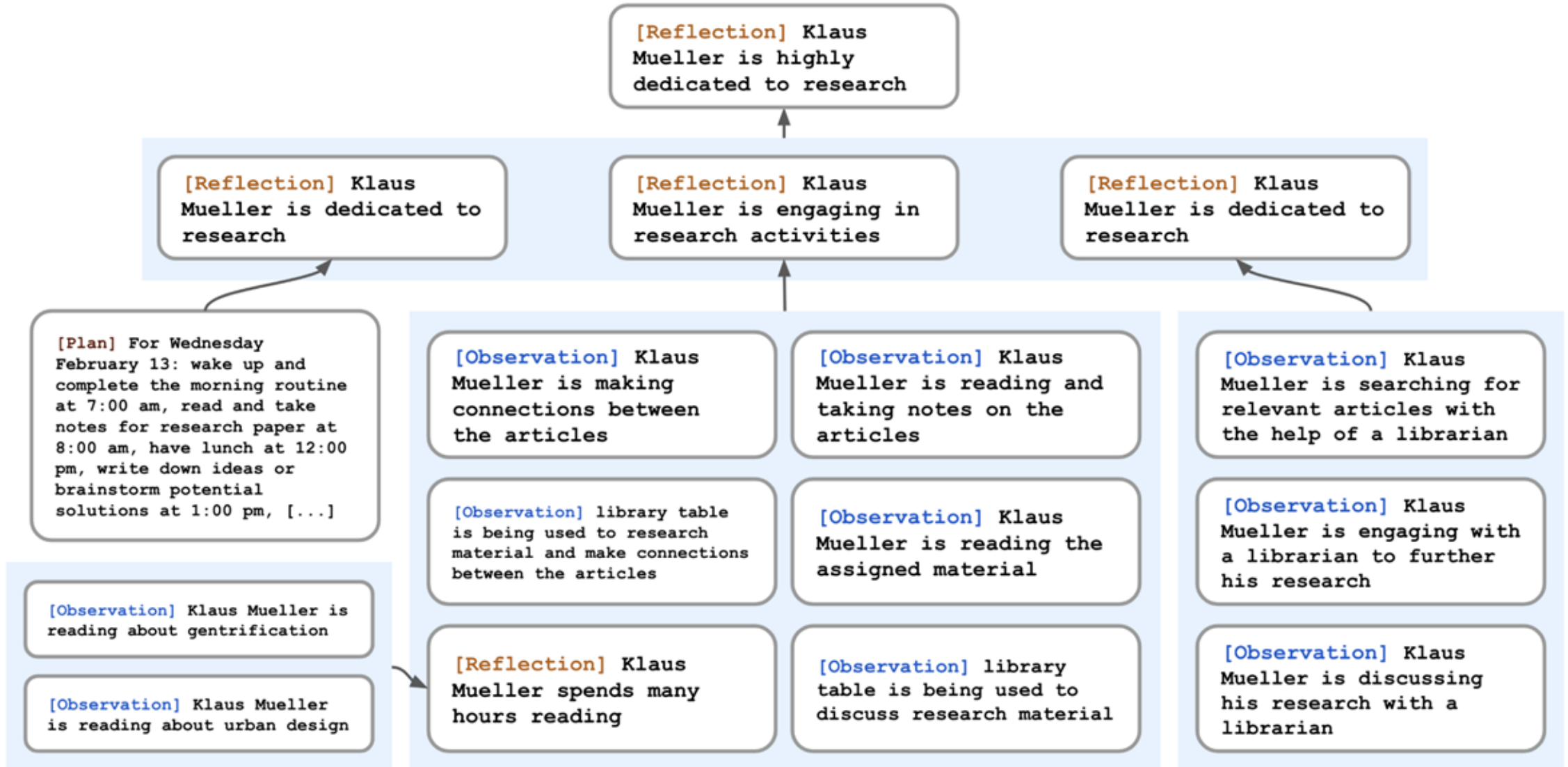
I'm looking forward to the Valentine's Day party that I'm planning at Hobbs Cafe!



Isabella

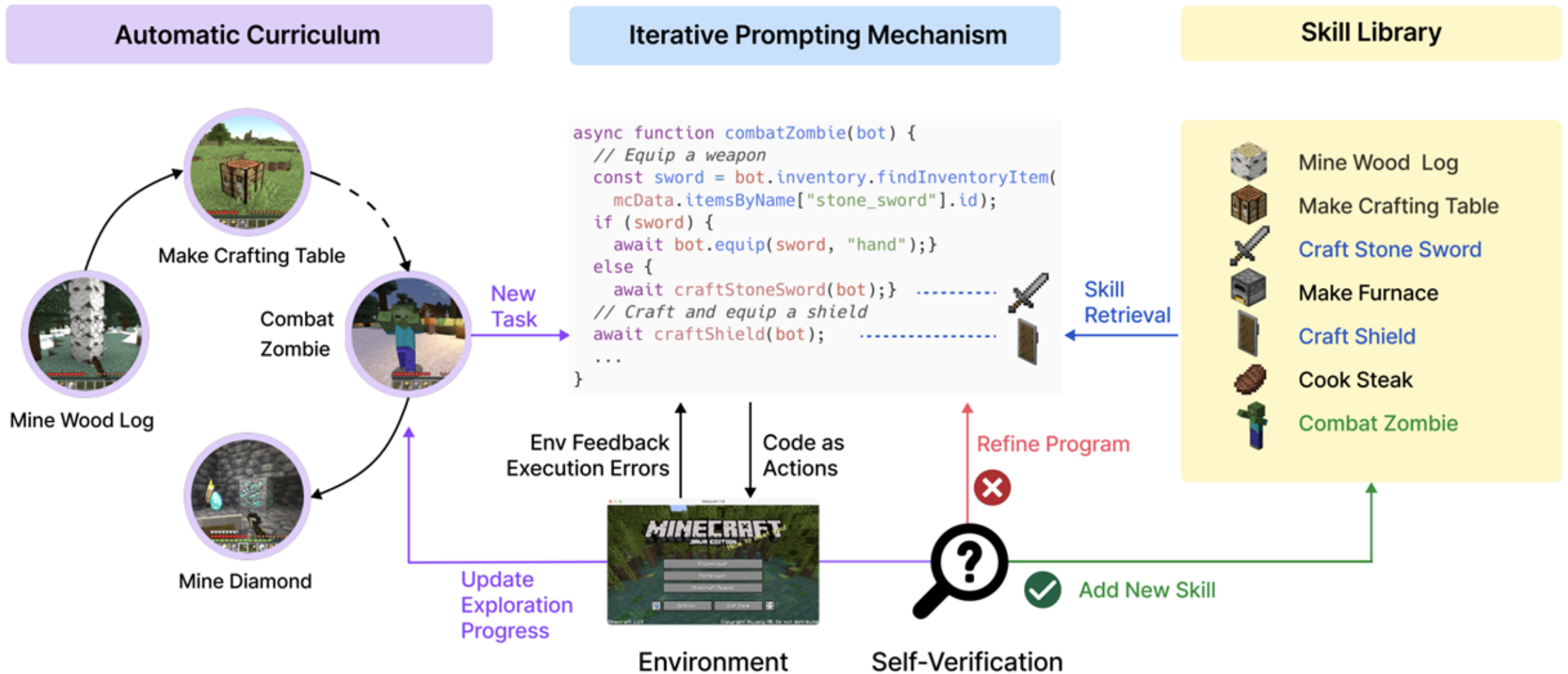
# 语义记忆

- Write: LLM reasoning over events
- Read: retrieval



# 程序记忆

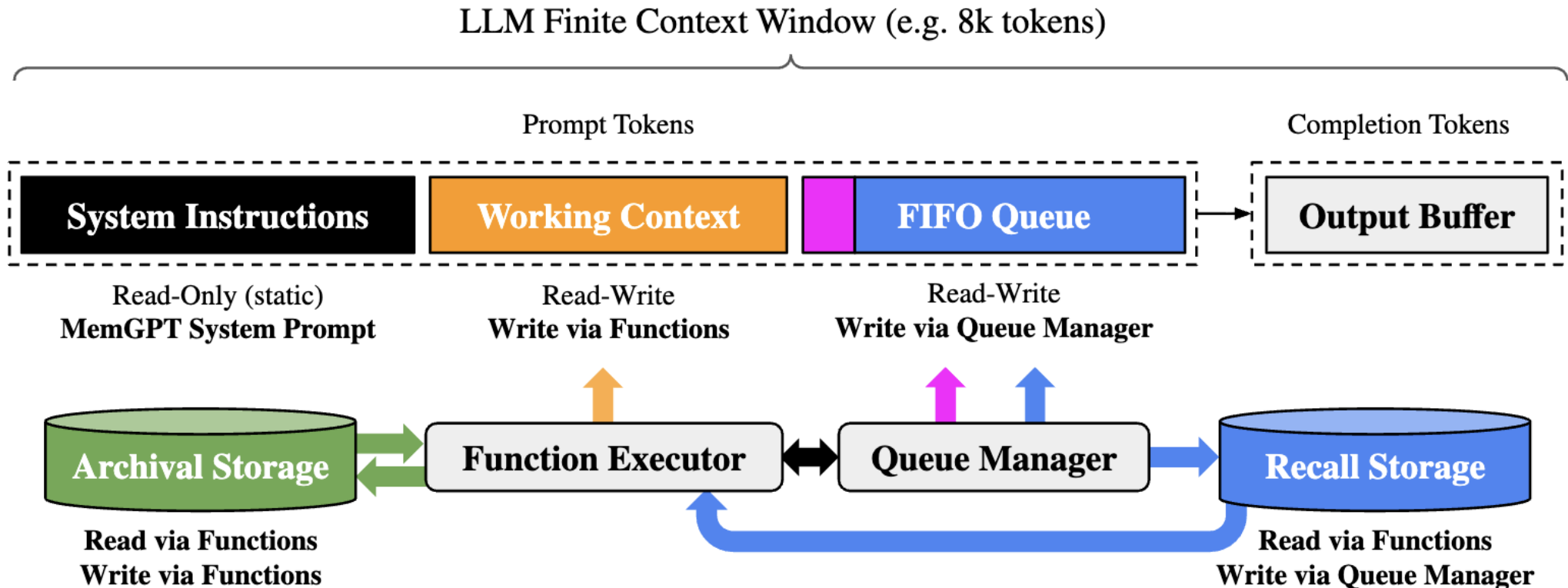
- Write: Code-based skills
- Read: Embedding retrieval



# MemGPT [Packer+2024]

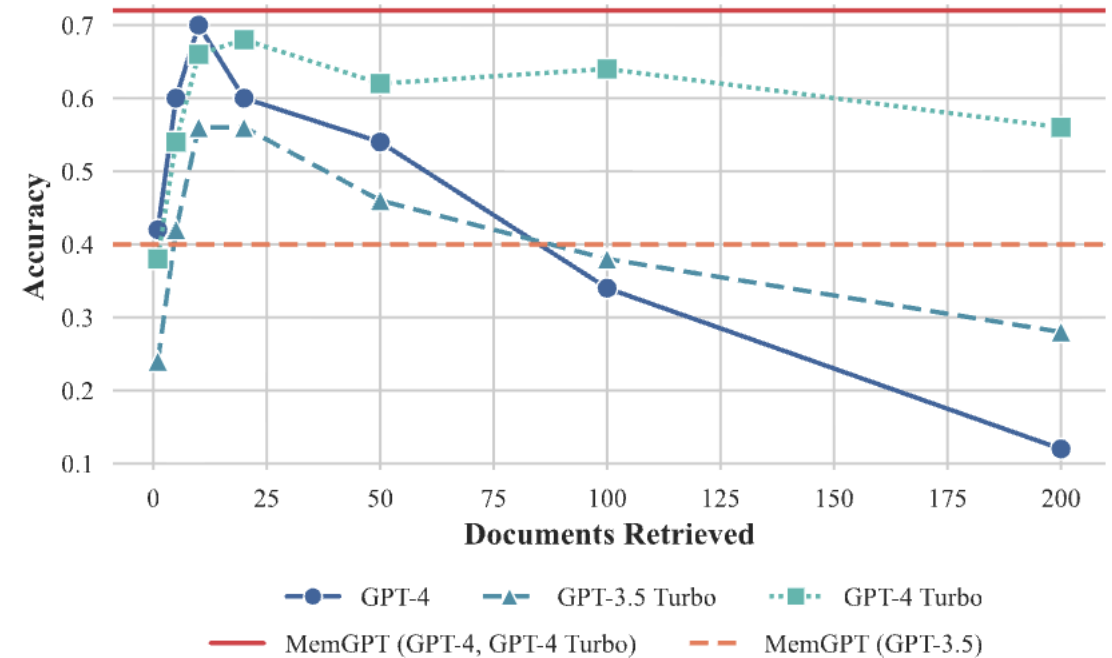
OS-inspired LLM system for virtual context management;

Using function calls, LLM agents can read and write to external data sources, modify their own context, and choose when to return responses to the user.



# MemGPT [Packer+2024]

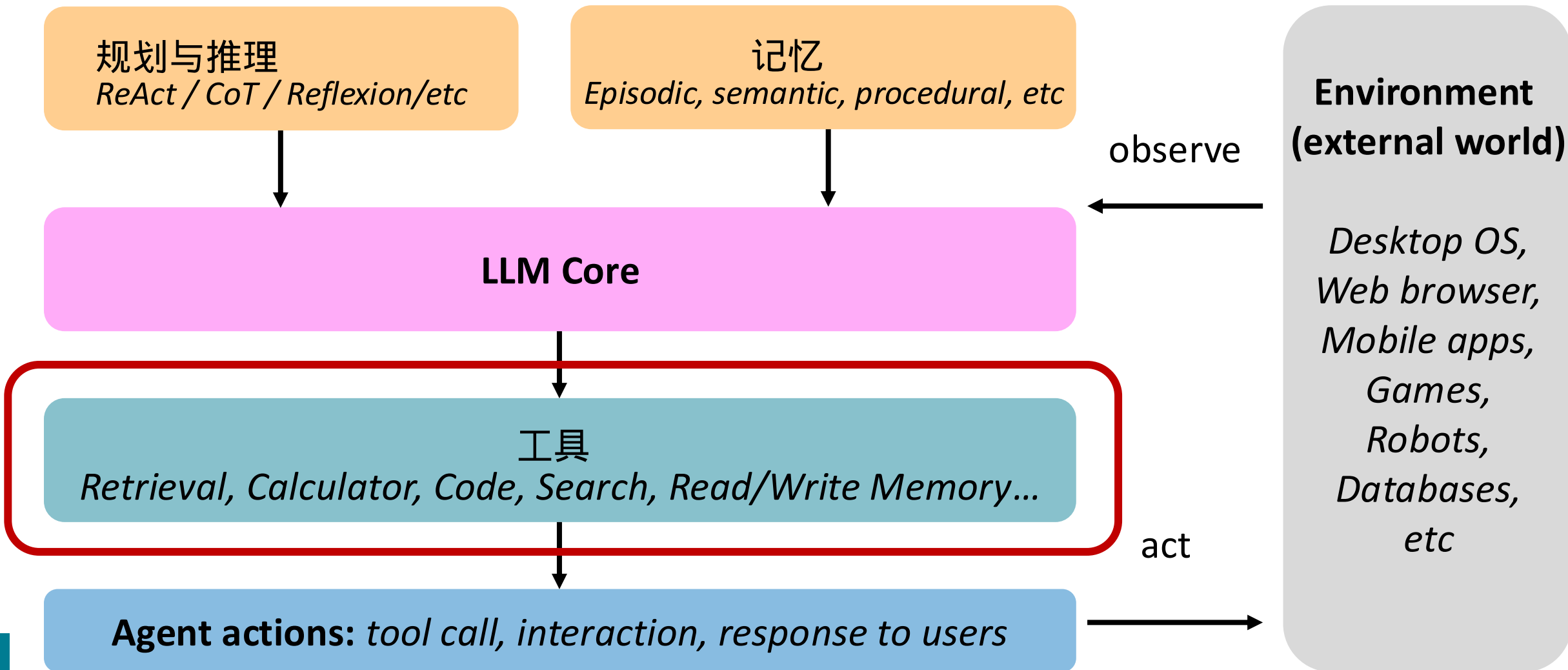
| Model           | Accuracy $\uparrow$ | ROUGE-L (R) $\uparrow$ |
|-----------------|---------------------|------------------------|
| GPT-3.5 Turbo   | 38.7%               | 0.394                  |
| + MemGPT        | 66.9%               | 0.629                  |
| GPT-4           | 32.1%               | 0.296                  |
| + MemGPT        | 92.5%               | 0.814                  |
| GPT-4 Turbo     | 35.3%               | 0.359                  |
| + <b>MemGPT</b> | <b>93.4%</b>        | <b>0.827</b>           |



Deep memory retrieval (DMR) performance

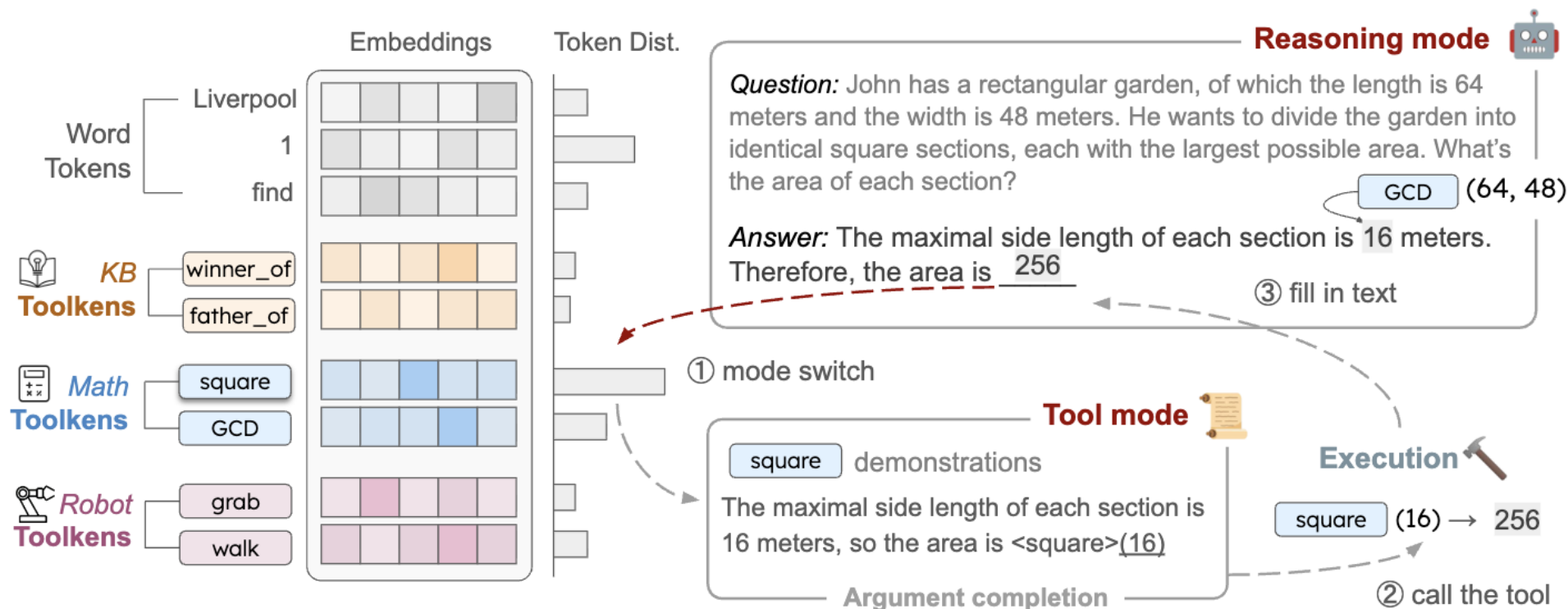
MemGPT's performance is unaffected by increased context length

## 5 . 语言 Agent : 工具使用



# Toolken GPT [Hao et al., 2024]

- ToolkenGPT represents each tool as a token (“toolken”) and learns an embedding for it, 以与生成常规词 Token 相同的方式启用工具调用。一旦 Toolken is triggered, LM is prompted to complete arguments for the tool to execute.



## Toolformer : 语言模型可以自学使用工具

- Toolformer trains LMs to decide which APIs to call, when to call them, what arguments to pass, and how to best 整合结果。
- Use self-supervised learning with only a handful of demonstrations for each API.
- Tools: calculator, Q&A system, search engine, translation system, calendar.

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

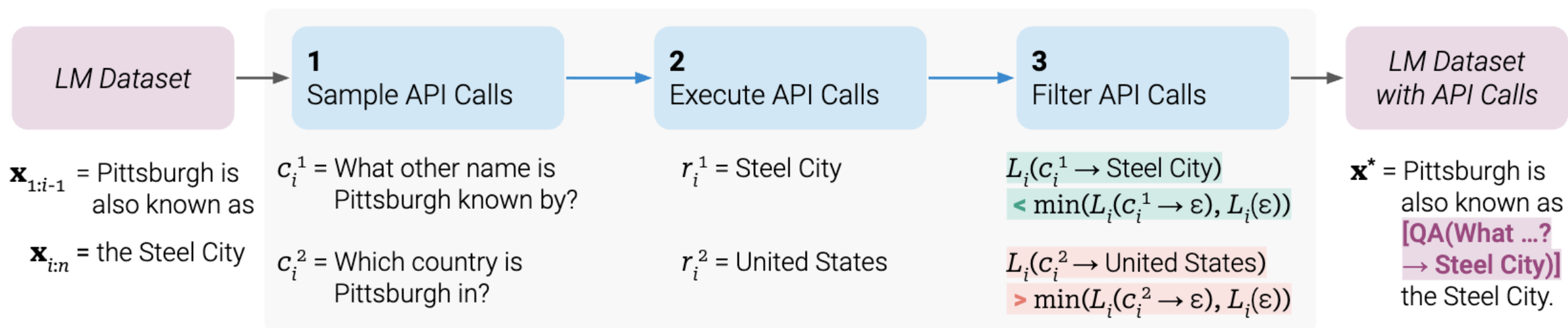
Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

# Toolformer : 语言模型可以自学使用工具

- For an input  $\mathbf{x}$ , sample a position  $i$  and corresponding API call candidates  $c_i^1, c_i^2, \dots, c_i^k$ .
- We then execute these API calls and filter out all calls which do not reduce the loss  $L_i$  over the next tokens.
- All remaining API calls are interleaved with the original text, resulting in a new text  $\mathbf{x}^*$



# Toolformer : 语言模型可以自学使用工具

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of 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.

**Input:** Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

**Output:** Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

**Input: x**

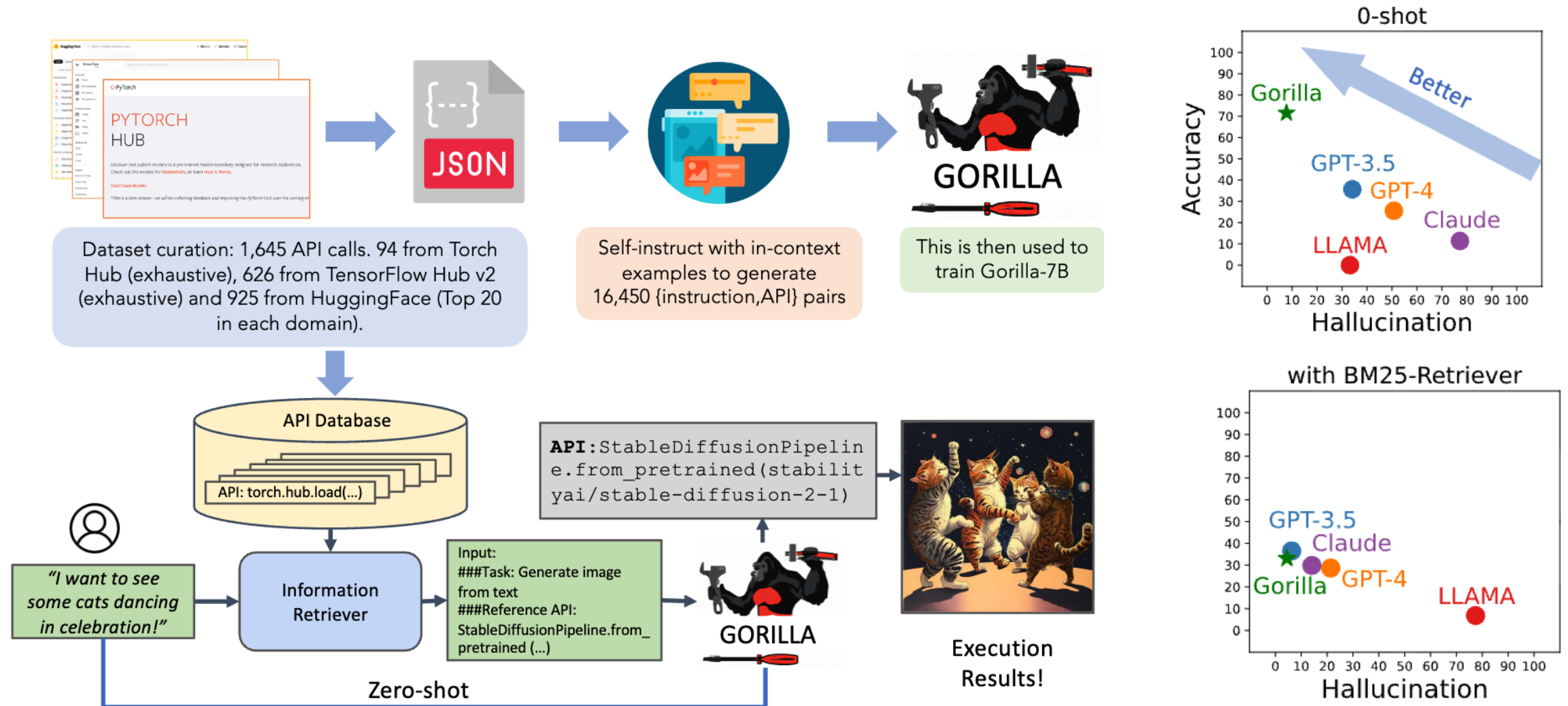
**Output:**

| Model                 | SQuAD       | Google-RE   | T-REx       |
|-----------------------|-------------|-------------|-------------|
| GPT-J                 | 17.8        | 4.9         | 31.9        |
| GPT-J + CC            | 19.2        | 5.6         | 33.2        |
| Toolformer (disabled) | 22.1        | 6.3         | 34.9        |
| Toolformer            | <b>33.8</b> | <b>11.5</b> | <b>53.5</b> |
| OPT (66B)             | 21.6        | 2.9         | 30.1        |
| GPT-3 (175B)          | 26.8        | 7.0         | 39.8        |

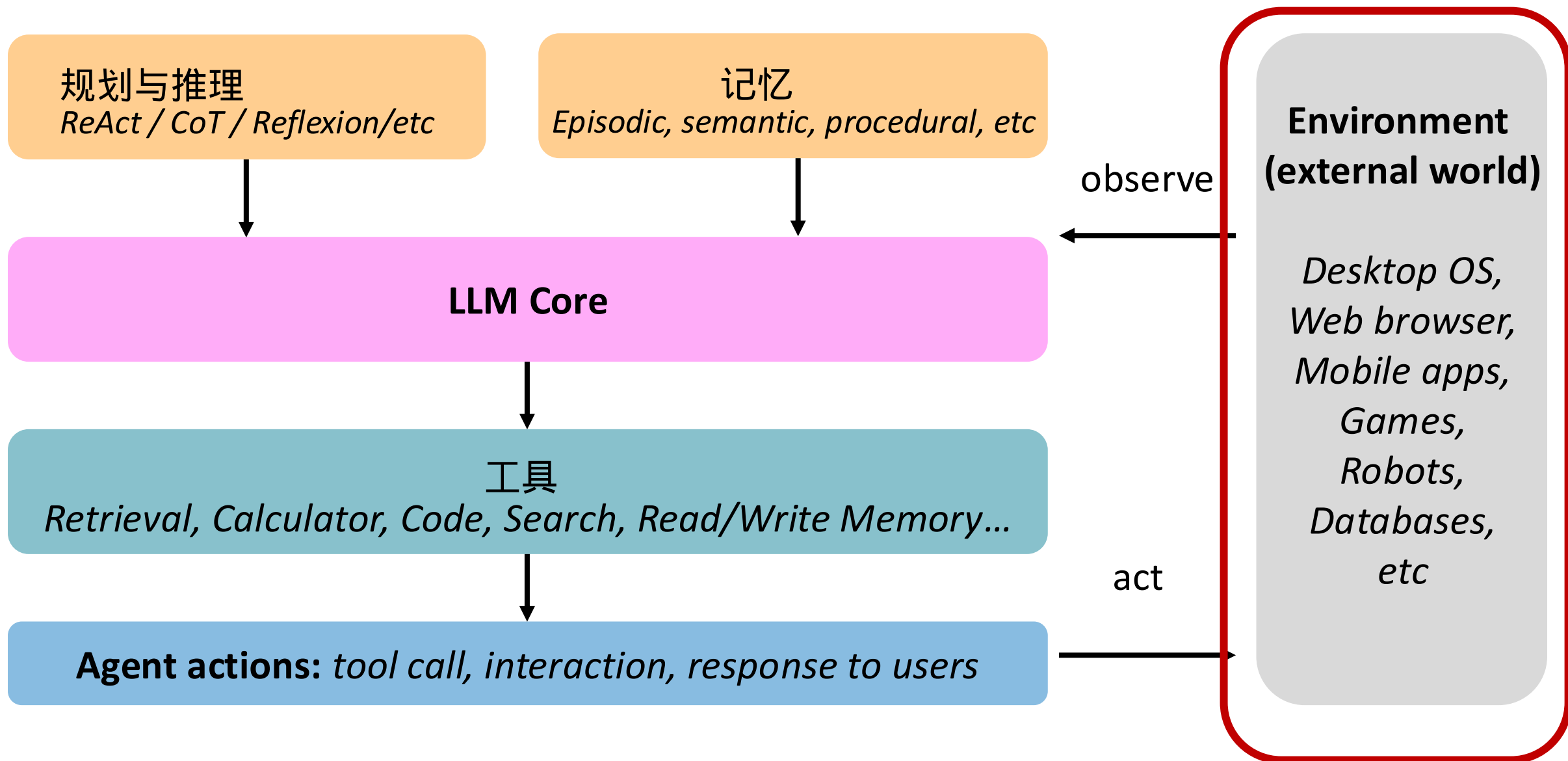
| Model                 | WebQS       | NQ          | TriviaQA    |
|-----------------------|-------------|-------------|-------------|
| GPT-J                 | 18.5        | 12.8        | 43.9        |
| GPT-J + CC            | 18.4        | 12.2        | 45.6        |
| Toolformer (disabled) | 18.9        | 12.6        | 46.7        |
| Toolformer            | <b>26.3</b> | <b>17.7</b> | <b>48.8</b> |
| OPT (66B)             | 18.6        | 11.4        | 45.7        |
| GPT-3 (175B)          | <u>29.0</u> | <u>22.6</u> | <u>65.9</u> |

# Gorilla: LLMs connected with massive APIs (Patil et al., 2023)

- Use self-instruct to generate {instruction, API} pairs and fine-tune LLaMa on it



## 6. 不同 Agent 应用、数据和评估



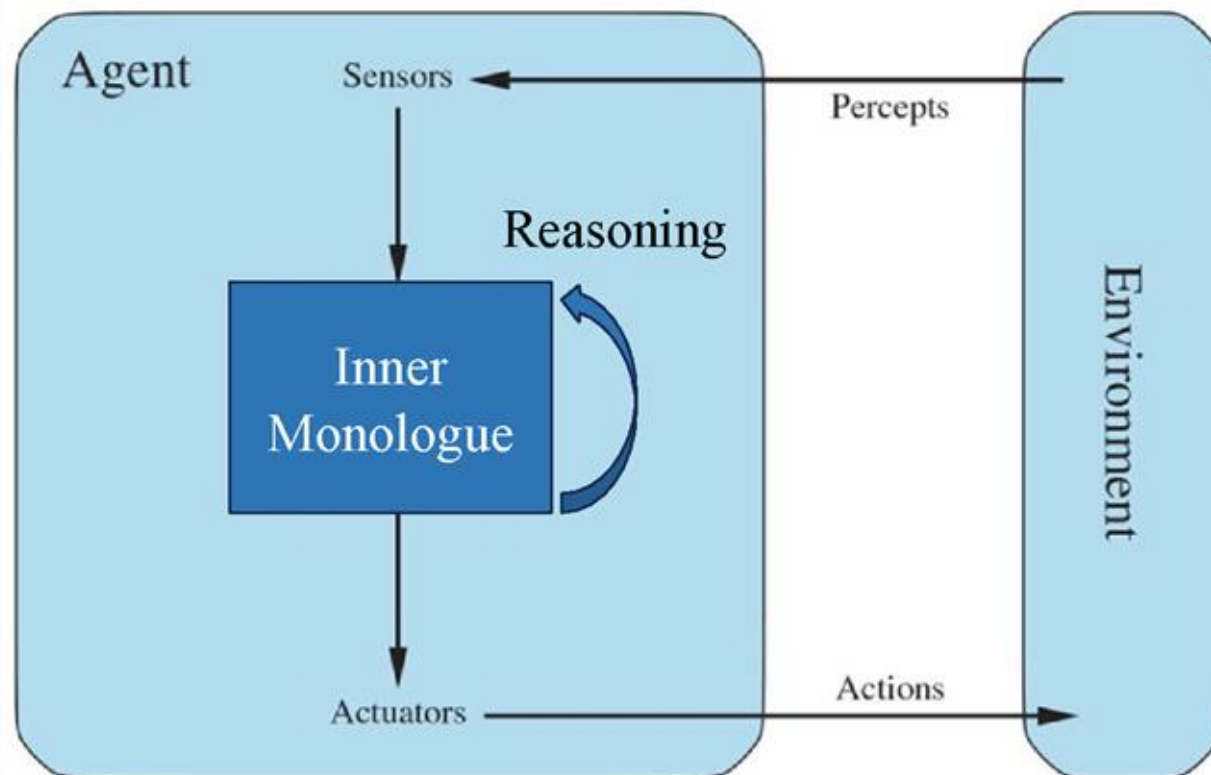
# Agent applications based on environments and domains

## 数字世界

- 编程 Agent
- 游戏 Agent
- 移动端 Agent
- Web / 应用 Agent
- Computer agents

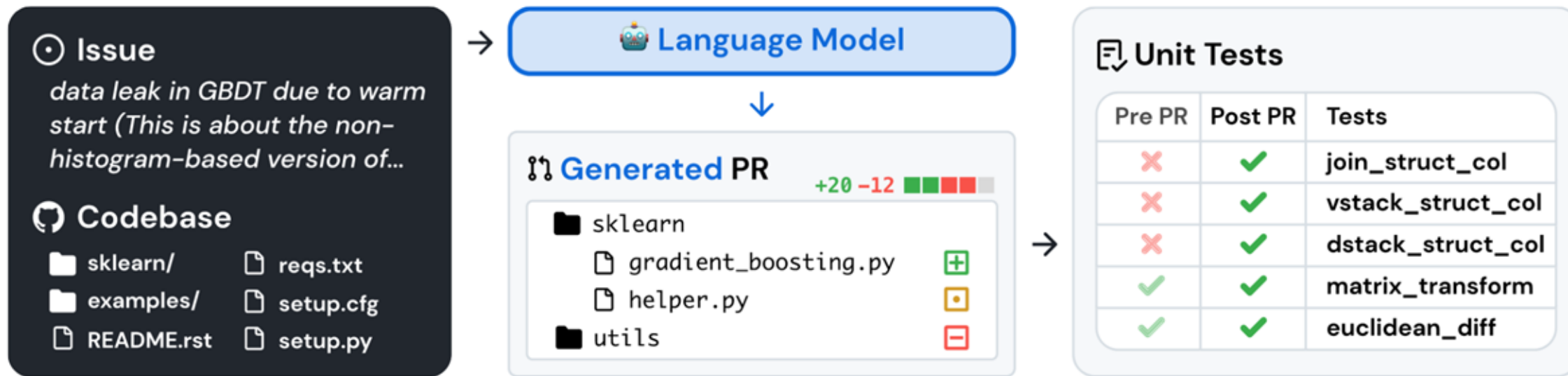
## 物理世界

- Robotics



# Coding agents

- 环境：项目代码仓库、文件系统、I D E ...
- 观察空间：代码文件、执行输出、文档、错误等
- 动作空间：代码编辑、文件搜索 / 查看、测试更新...



SWE-bench ([Jimenez & Yang, 2023](#))

# Web / 应用 Agent

- 环境：Web 浏览器 / 应用
- 观察空间：截图、DOM 树、HTML、历史动作...
- 动作空间：浏览器 / 应用控制（如点击、输入、滚动等）

(a) Find one-way flights from New York to Toronto.

(b) Book a roundtrip on July 1 from Mumbai to London and vice versa on July 5 for two adults.

(c) Find a flight from Chicago to London on 20 April and return on 23 April.

(d) Find Elon Musk's profile and follow, start notifications and like the latest tweet.

(e) Browse comedy films streaming on Netflix that was released from 1992 to 2007.

(f) Open page to schedule an appointment for car knowledge test.

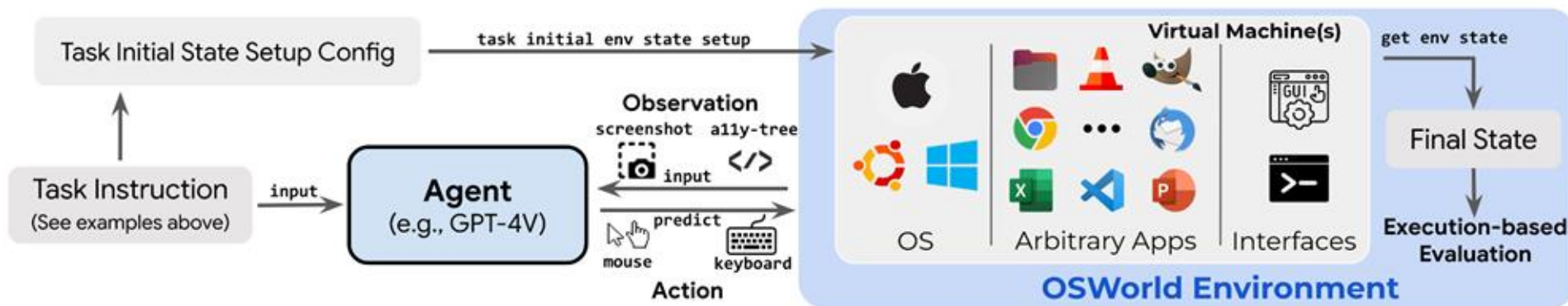
[Mind2Web: Towards a Generalist Agent for the Web.](#) (Deng et al., 2023)

[WebArena: A Realistic Web Environment for Building Autonomous Agents.](#) (Zhou et al., 2023)

# 计算机使用 Agent

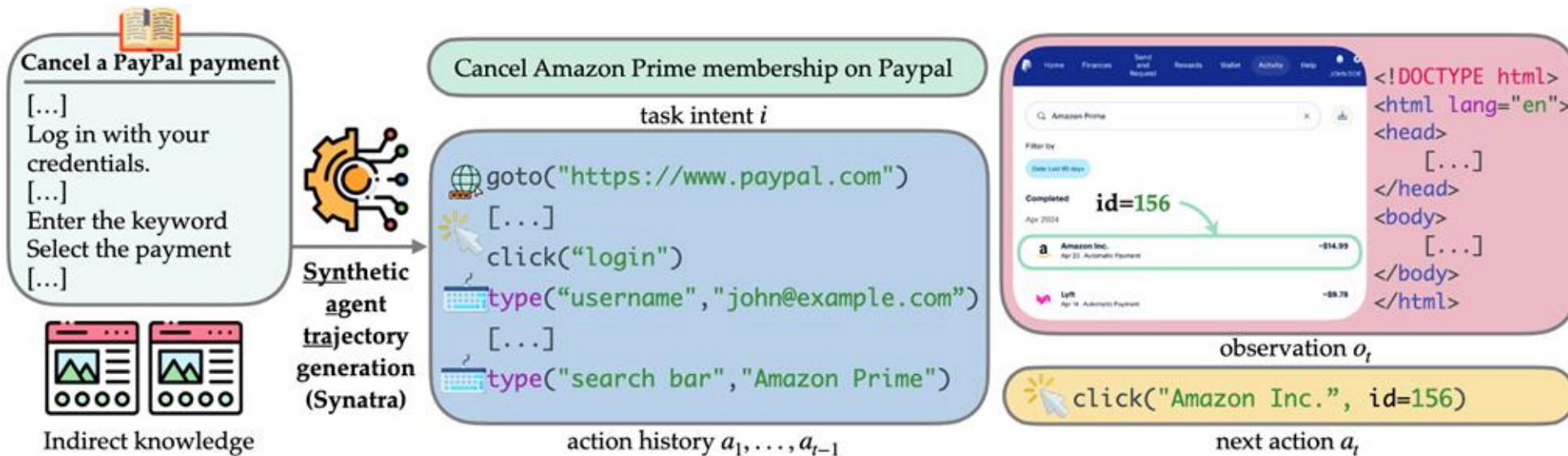
- 环境：桌面操作系统
- 观察空间：桌面截图、无障碍树、历史动作...
- 动作空间：键盘 / 鼠标控制（如点击、输入、拖拽等）

Task instruction I: Update the bookkeeping sheet with my recent transactions over the past few days in the provided folder.



# Agent data --- scaling agent data

- 人类演示
  - Expensive and complex infrastructure setup
  - Expert time & cost
  - Task coverage
- Synthesis/simulation
  - Converting online tutorials into direct training demonstrations [[Synatra](#)]



# Agent data --- scaling agent data

- 人类演示

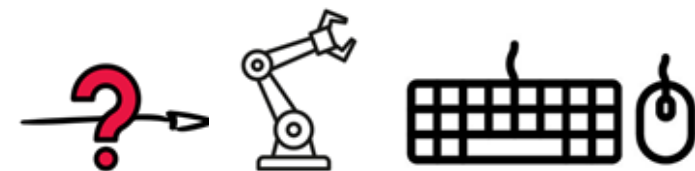
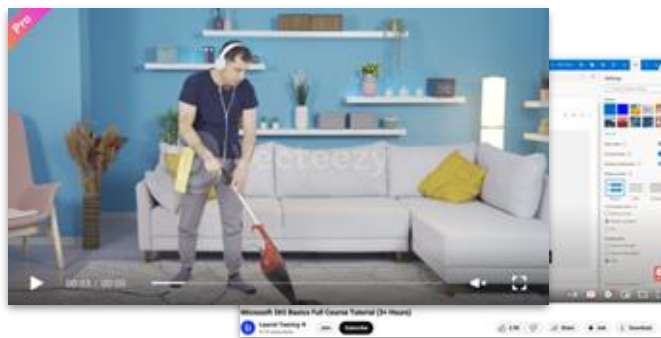
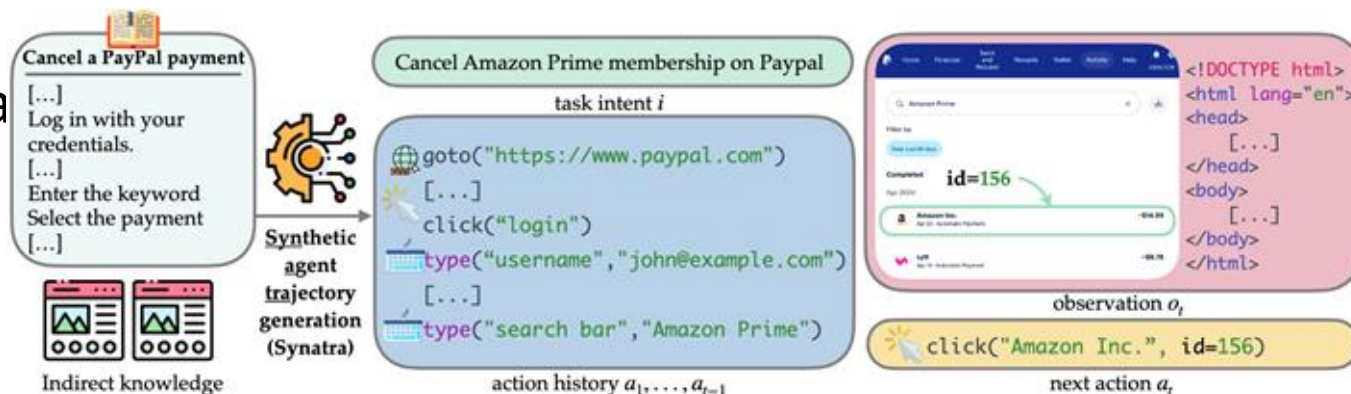
- Expensive and complex infrastructure
- Expert time & cost
- Task coverage

- Synthesis/simulation

- Converting online tutorials into direct training demonstrations [[Synatra](#)]

- 互联网规模的数据

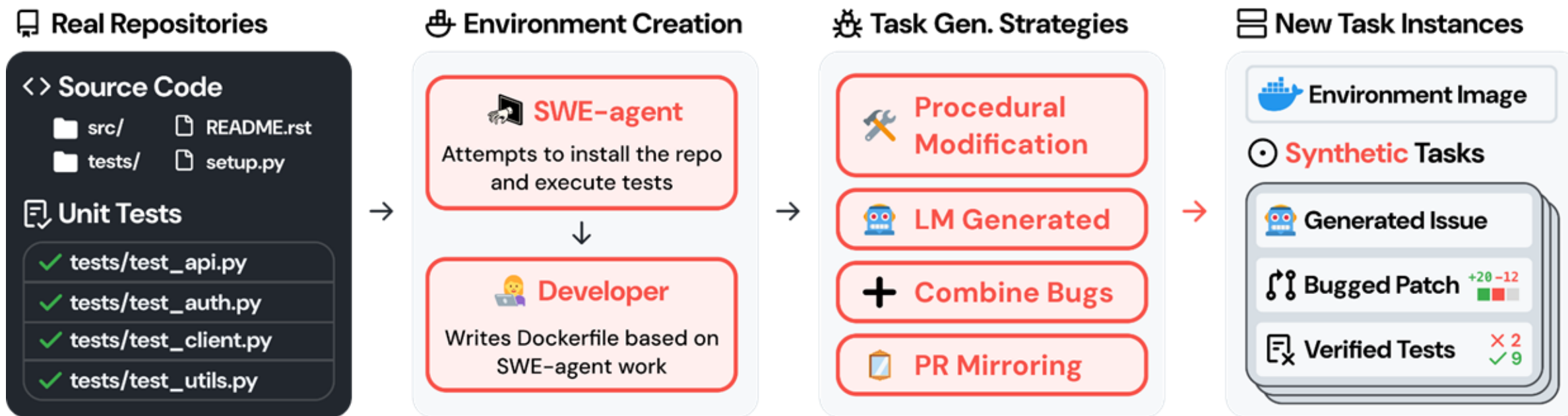
- Numerous videos/data exist online showing how humans perform (agent) tasks, but without grounded trajectories



# SWE - smith 案例研究：为编程 Agent 扩展数据

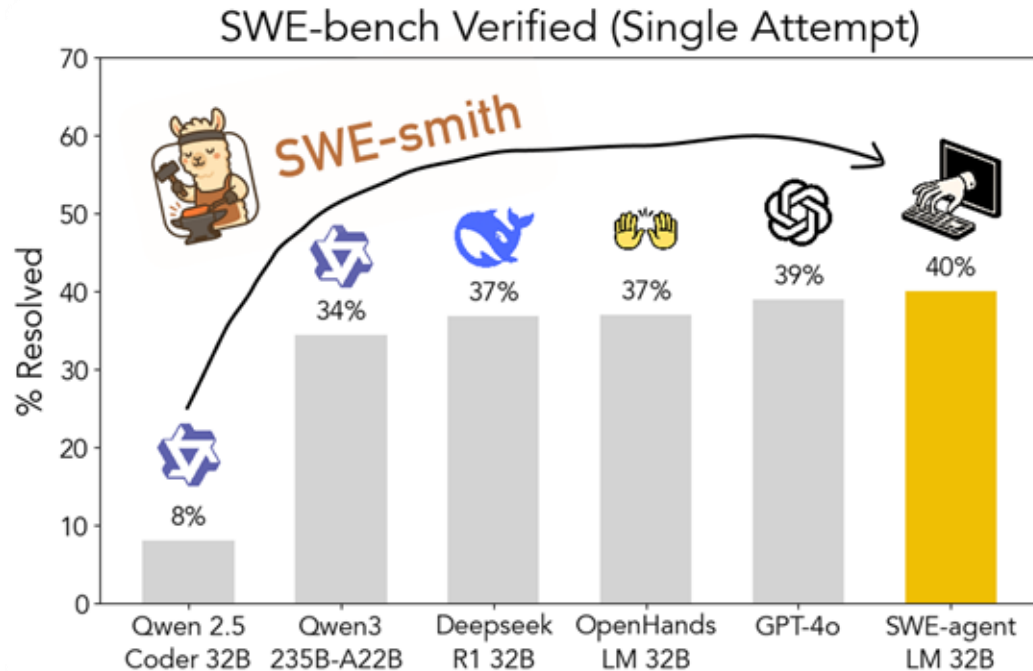


A toolkit for (1) creating execution environments, and (2) synthesizing 100s to 1000s of task instances for any (Python) GitHub repository [[Yang et al., 2025](#)]



# Training details and results

- **Training Technique:** Rejection Sampling Fine Tuning
- **Models:** Claude 3.5/3.7, GPT 4o (Experts); Qwen 2.5 Coder Instruct (Student)
- **Agent 系统** SWE-agent



| Model  | System        | Train Size | Lite        | Verified    |
|--|---------------|------------|-------------|-------------|
| <i>Closed Weight Models</i>                            |               |            |             |             |
| GPT-4o ( <a href="#">OpenAI, 2024a</a> )               | Agentless     | -          | 32.0        | 38.8        |
|  | OpenHands     | -          | 22.0        | -           |
|  | SWE-agent     | -          | 18.3        | 23.0        |
| Claude 3.5 Sonnet ( <a href="#">Anthropic, 2024</a> )  | Agentless     | -          | 40.7        | 50.8        |
|  | AutoCodeRover | -          | -           | 46.2        |
|  | OpenHands     | -          | 41.7        | 53.0        |
|  | SWE-agent     | -          | 23.0        | 33.6        |
| Claude 3.7 Sonnet ( <a href="#">Anthropic, 2025</a> )  | SWE-agent     | -          | <b>48.0</b> | <b>58.2</b> |
| Llama3-SWE-RL-70B ( <a href="#">Wei et al., 2025</a> ) | Agentless     | 11M        | -           | 41.0        |
| <i>Open Weight Models</i>                              |               |            |             |             |
| Lingma-SWE-GPT-72B ( <a href="#">Ma et al., 2024</a> ) | SWE-SynInfer  | -          | -           | 28.8        |
| Qwen3-235B-A22B ( <a href="#">Qwen et al., 2025</a> )  | OpenHands     | -          | -           | 34.4        |
| R2E-Gym-32B ( <a href="#">Jain et al., 2025</a> )      | OpenHands     | 3.3k       | -           | 34.4        |
| SWE-fixer-72B ( <a href="#">Xie et al., 2025a</a> )    | SWE-Fixer     | 110k       | 24.7        | 32.8        |
| SWE-gym-32B ( <a href="#">Pan et al., 2024</a> )       | OpenHands     | 491        | 15.3        | 20.6        |
| SWE-agent-LM-7B  | SWE-agent     | 2k         | 11.7        | 15.2        |
| SWE-agent-LM-32B                                       | SWE-agent     | 5k         | <b>30.7</b> | <b>40.2</b> |

# Agent evaluation

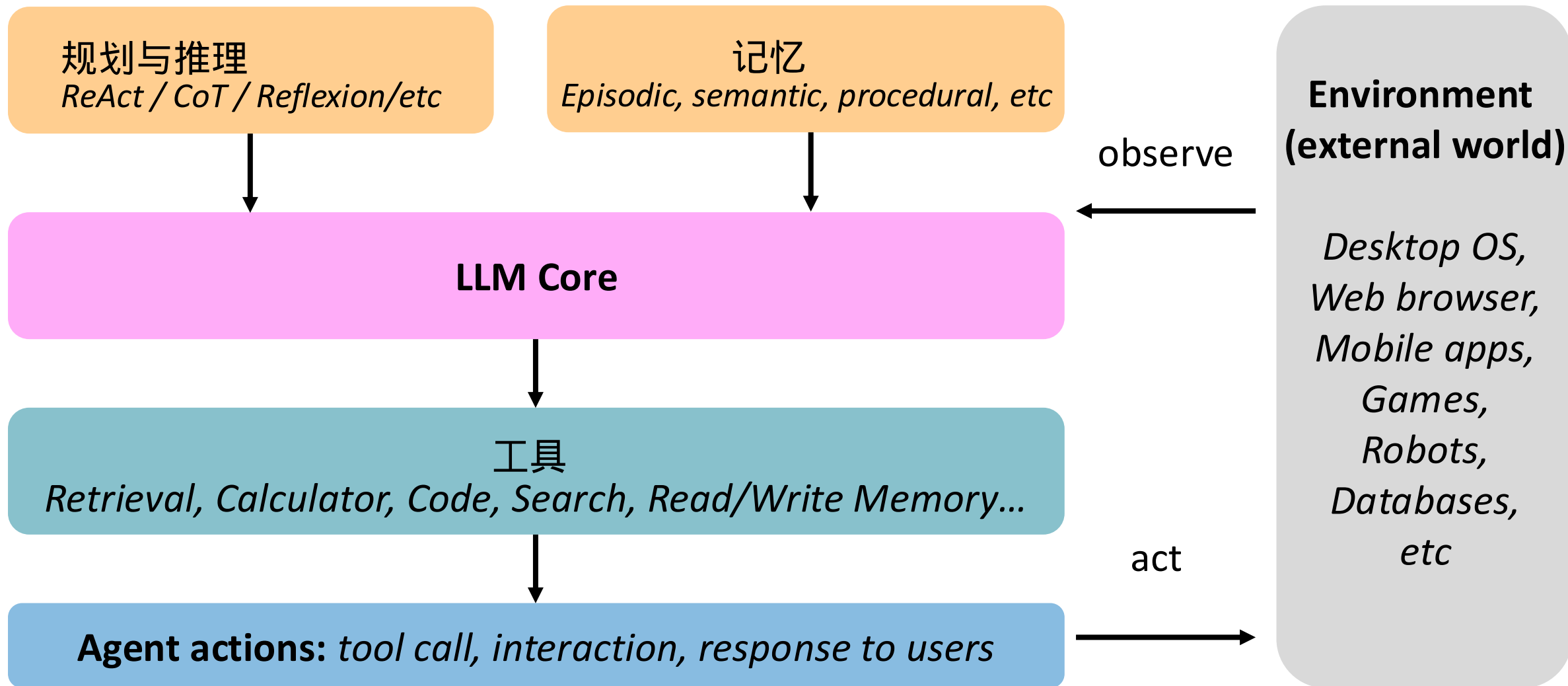
## Challenges in agent evaluation

- 现实世界环境设置的复杂性
- Task coverage
- 开放性成功标准
  - 多个有效解决路径
  - 无法脚本化评估指标，需要人类判断
- 超越任务成功的评估

## Agent evaluation:

- 通过基准测试 / L L M / V L M / 众包
- See lecture on “Benchmarking and Evaluation”

# 语言 Agent : 从词语到行动



# Bonus: building a coding agent (step by step)



John Yang

At its simplest, A I 编程 Agent 就是一个大循环

Given a task, e.g. “create a web app for me”, we 启动以下循环：

1. The LM proposes an action
2. You execute the action in a code environment (e.g., bash terminal, docker container, python interpreter)
3. Tell the LM the output
4. And then repeat steps 2-4!

## Thought & Action

Our reproduction script confirms the reported issue: Max & Min are not being converted to R. Let's search for files related to R code generation.

```
$ search_dir "rcode"
```

## Observation

```
Found 16 matches for "rcode":
reproduce_issue.py (3 matches)
sympy/printing/lambdarepr.py (1 matches)
sympy/printing/rcode.py (12 matches)
```

## Thought & Action

The responsible file is likely rcode.py. We should open and inspect this file.

```
$ open sympy/printing/rcode.py
```

## Observation

```
[File: sympy/printing/rcode.py (412 lines)]
1: """The RCodePrinter converts sympy ...
2: from sympy.core import S
...
20: # dictionary of sympy to C functions
21: known_functions = {
22:     "Abs": "abs",
23:     "gamma": "gamma",
```

# Bonus: building a coding agent (step by step)

At its simplest, A I 编程 Agent 就是一个大循环 pseudocode:

```
messages = [{"role": "user", "content": "Help me fix the ValueError in main.py"}]
while True:
    lm_output = query_lm(messages)
    print("LM output", output)
    messages.append({"role": "assistant", "content": lm_output}) # remember what the LM said
    action = parse_action(lm_output) # separate the action from output
    print("Action", action)
    if action == "exit":
        break
    output = execute_action(action)
    print("Output", output)
    messages.append({"role": "user", "content": output}) # send command output back
```

Let's implement 3 things: **query\_lm**; **parse\_action**; **execute\_action**

# Bonus: building a coding agent (step by step)

## query\_lm

Input: messages

Output: LM's response

```
from openai import OpenAI

client = OpenAI(
    api_key="your-api-key-here"
) # or set OPENAI_API_KEY env var

def query_lm(messages):
    response = client.responses.create(
        model="gpt-5.1",
        input=messages
    )
    return response.output_text
```

## parse\_action

Input: LM's response

Output: bash action

```
import re

def parse_action(lm_output: str) -> str:
    """Take LM output, return action"""
    matches = re.findall(
        r"<bash_action>(.*?)</bash_action>",
        lm_output,
        re.DOTALL
    )
    return matches[0].strip() if matches else ""
```

## execute\_action

Input: base action

Output: std. output

```
import subprocess
import os

def execute_action(command: str) -> str:
    """Execute action, return output"""
    result = subprocess.run(
        command,
        shell=True,
        text=True,
        env=os.environ,
        encoding="utf-8",
        errors="replace",
        stdout=subprocess.PIPE,
        stderr=subprocess.STDOUT,
        timeout=30,
    )
    return result.stdout
```

Full tutorial here: <https://minimal-agent.com/>