



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

DDA6307/CSC6052/MDS6002: Natural Language Processing

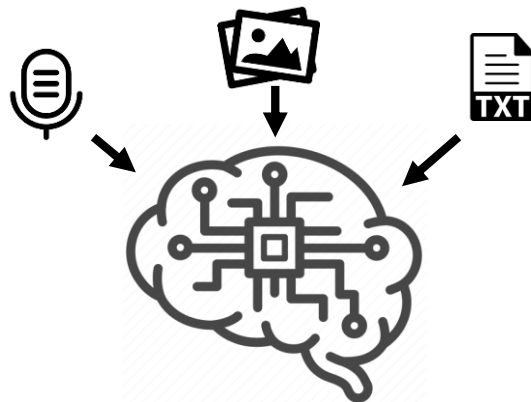
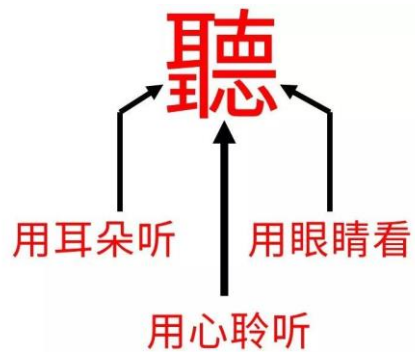
Lecture 10: LLM Agents

Spring 2024
Benyou Wang
School of Data Science

Final project

- 19 April > Lecture 11: guest lecture
- Submit your proposal no later than April 20th
- 26 April > Lecture 12: Some practices for final projects.

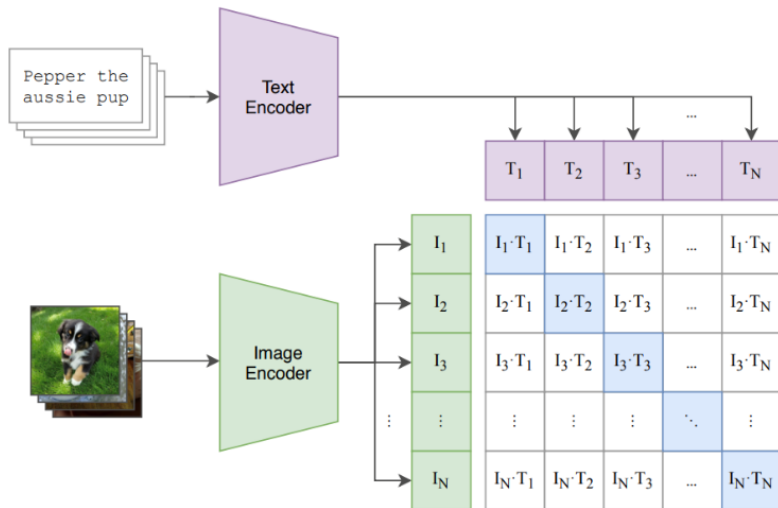
Recap...



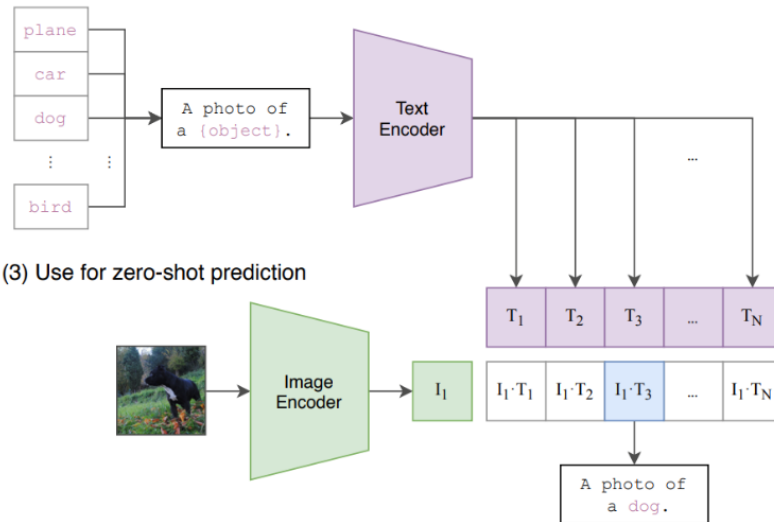
Human processes **multimodal** infos simultaneously

CLIP

(1) Contrastive pre-training



(2) Create dataset classifier from label text



(3) Use for zero-shot prediction

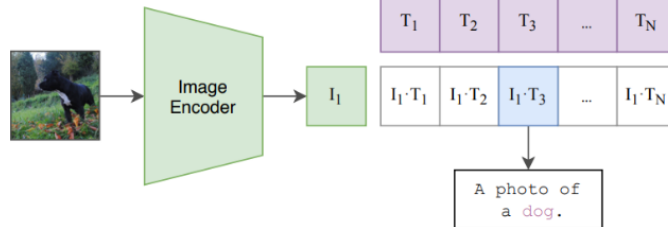


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

Flamingo's high-level architecture

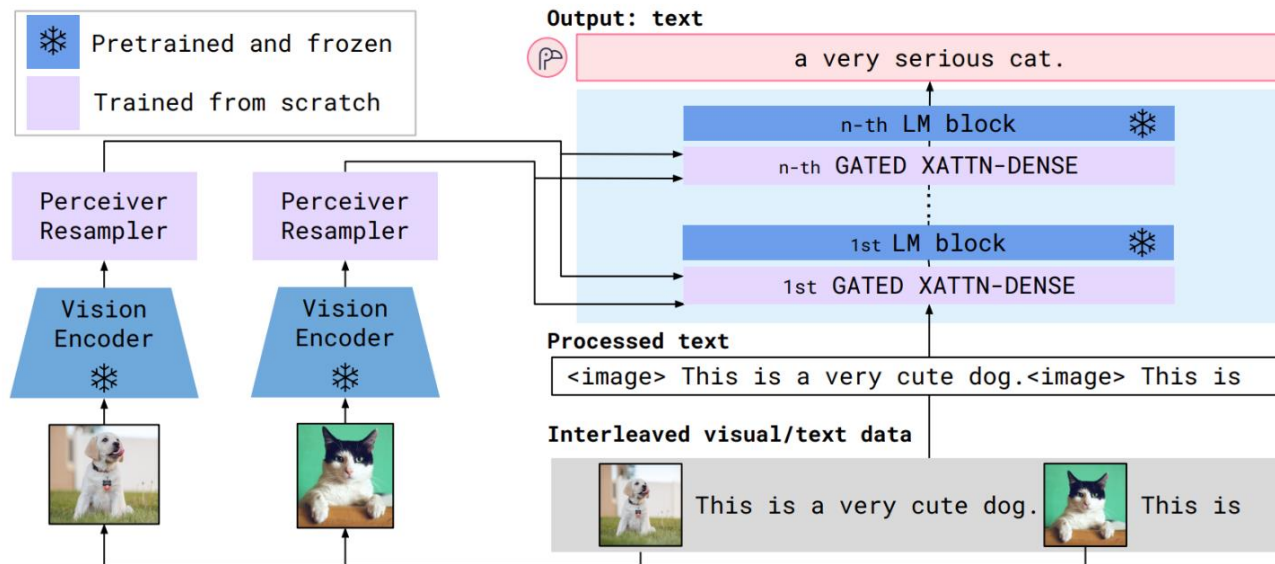







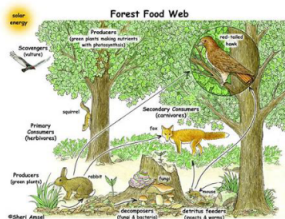
Figure 3: **Flamingo architecture overview.** Flamingo is a family of visual language models (VLMs) that take as input visual data interleaved with text and produce free-form text as output.

MLLM-Bench: evaluating Multi-modal LLMs using GPT-4V

Table 2: Sample questions in MLLM-Bench.

Capability Level	Capability	Image	Sample Questions
Perception	General Object Recognition		Identify the objects in the image and indicate the use case of the item.
Understanding	Scene Understanding		Infer the activities that are likely to take place during the scene based on the objects present.
Applying	Object Counting		How many pears in the picture have an upright stem?
Analyzing	Visual Math Reasoning	<p>Factorise and evaluate:</p> $\frac{3^{25} - 3^{24}}{3^{21} + 3^{20}}$	Compute the solution to this question using a straightforward method, and illustrate the step-by-step process.
Evaluation	Fake image detection		Determine the likelihood that this photo is real, considering both the technical aspects of photography and the plausibility of the scene depicted. Provide a detailed rationale for your conclusion.
Creation	Visual Story Telling		Write a winter ballad on the solitude and muffled world under snow, reflecting life's resilience in a monochrome setting.

MLLM-Bench: evaluating Multi-modal LLMs using GPT-4V



Question: Map out the complex interdependencies between species in this food web and identify any potential cascade effects if one species were to become extinct.

Answer 1 (llava-v1.5-13b):

In the image, there is a complex interdependent system involving a forest, trees, rabbits, squirrels, insects (including beetles), and a bird. The food web shows that these different species interact and play roles in the ecosystem. For instance, rabbits and squirrels are herbivores, consuming plants, while insects like beetles help in decomposing plant matter, and the bird is a predator that feeds on insects and small mammals.

If one species were to become extinct, it could lead to a cascade effect where other species are affected as well. For example, if the rabbits were to become extinct, the herbivore population would decrease, impacting the plants they were consuming. This would lead to an increase in plant growth and change the availability of food for other herbivores, such as squirrels and insects. Additionally, there could be imbalances in predator-prey relationships, leading to overpopulation of certain species, and consequently, a decline in the biodiversity and ecosystem functioning. The ecosystem would need to adapt to these changes, either by altering the populations of the remaining species or by introducing new species into the area.

Answer 2 (llava-instruct4v-llava-7b):

In the image, there is a forest food web featuring various animals, including a bird, a fox, a rabbit, a squirrel, and a frog. The bird is perched on a tree branch, while the other animals are scattered around the forest floor. The interdependencies between these species are numerous, as they all play essential roles in maintaining the ecosystem's balance.

For example, the bird feeds on insects, which helps control their population and maintain a healthy environment. The fox is a predator, feeding on smaller animals such as rabbits, squirrels, and insects, thus controlling their populations and maintaining the balance of the food web. The rabbit and squirrel are herbivores, feeding on grass, plants, and seeds, contributing to the forest's overall health. The frog, as an amphibian, is also a part of the food web, feeding on insects and small aquatic creatures.

If one species were to become extinct, it could lead to a cascade effect on the food web, as the loss of a key species may disrupt the balance of the ecosystem. This could result in an overpopulation of other species, which could then lead to a decline in the populations of those species that depend on them for food. In turn, this could have a negative impact on the overall health of the forest ecosystem and potentially affect other species in the food web.

GPT-4V Judgment³

Answer 1 and Answer 2 both provide a general outline of the relationships and potential cascade effects within the food web depicted in the image. Answer 1 starts by describing the food web including a variety of species and their roles in the ecosystem, such as herbivores (rabbits and squirrels), decomposers (insects like beetles), and predators (birds). It then moves on to discuss the potential cascade effects if one species, specifically the rabbits, were to become extinct. The explanation includes the potential increase in plant growth, changes in the availability of food for other herbivores, imbalances in predator-prey relationships, overpopulation of certain species, a decline in biodiversity, and the need for ecosystem adaptation.

Answer 2 also describes the forest food web, identifying the animals present and their roles, such as the bird controlling insect populations, the fox as a predator, and the herbivorous activities of the rabbit and squirrel. It also acknowledges the frog's role in this ecosystem. The answer then hypothesizes about the potential cascade effects if a key species were to become extinct, including overpopulation of other species, a decline in populations of dependent species, and a negative impact on the overall health of the forest ecosystem.

Both answers are good in terms of providing an overview of the food web's complexity and the interdependencies among the species. However, Answer 1 is slightly more aligned with the image as it specifically names the insects as beetles and discusses the role of birds in greater detail, while Answer 2 incorrectly identifies a frog, which is not present in the image. Therefore, in terms of accuracy and alignment with the image, Answer 1 is superior.

Large space to be improved in MLLMs, w.r.t. GPT-4V

Model	Perception	Understanding	Applying	Analyzing	Evaluation	Creation	Avg
LLaVA-v1.5	0.63	0.77	0.65	0.69	0.83	0.83	0.71
Qwen-VL-Chat	0.65	0.77	0.68	0.61	0.83	0.80	0.70
LVIS	0.61	0.75	0.65	0.63	0.82	0.77	0.69
mPLUG-Owl2	0.58	0.73	0.61	0.61	0.78	0.80	0.67
CogVLM-Chat	0.59	0.74	0.62	0.59	0.78	0.72	0.66
IDEFICS-9B	0.34	0.58	0.49	0.50	0.57	0.63	0.53
MiniGPT-v2	0.44	0.54	0.44	0.50	0.65	0.71	0.52
InstructBLIP	0.45	0.57	0.37	0.43	0.57	0.39	0.47
Fuyu-8B	0.39	0.55	0.33	0.38	0.52	0.42	0.43
SEED-LLaMA	0.39	0.43	0.42	0.40	0.48	0.51	0.43
kosmos-2	0.38	0.53	0.41	0.38	0.43	0.36	0.42
BLIP-2	0.24	0.24	0.24	0.24	0.29	0.19	0.24

Today's lecture

Contents in this lecture

- **Overall framework of agents**
- **Four elements**
 - Planning
 - Tools
 - Memory
 - Action
- **Recap of agent**
- **Proof of concepts**
- **Our research**

What is LLM Agents



Explorations of Visual-Language Model on Autonomous Driving

Sec. 2.1 Traffic signs understanding

Prompt:

This is a front-camera view of a driving car. Please identify all the traffic signs in the image, and explain each of them.



GPT-4V:

There are three traffic signs in the image:

A sign under **the overhead structure displaying "4.5m"**: This sign indicates the maximum height of vehicles that can safely pass underneath the structure. Vehicles taller than 4.5 meters should not proceed.

A triangular sign with a curved arrow pointing left: This is a warning sign that indicates there is a left bend or curve ahead. Drivers should be cautious and possibly reduce their speed.

A rectangular sign with the word "SLOW": This sign instructs drivers to reduce their speed, typically because of potential hazards or a condition ahead that requires slower driving.

Prompt:

This is a front-camera view of a driving car. Please identify all the traffic signs in the image, and explain each of them.



GPT-4V:

There are three traffic signs displayed:

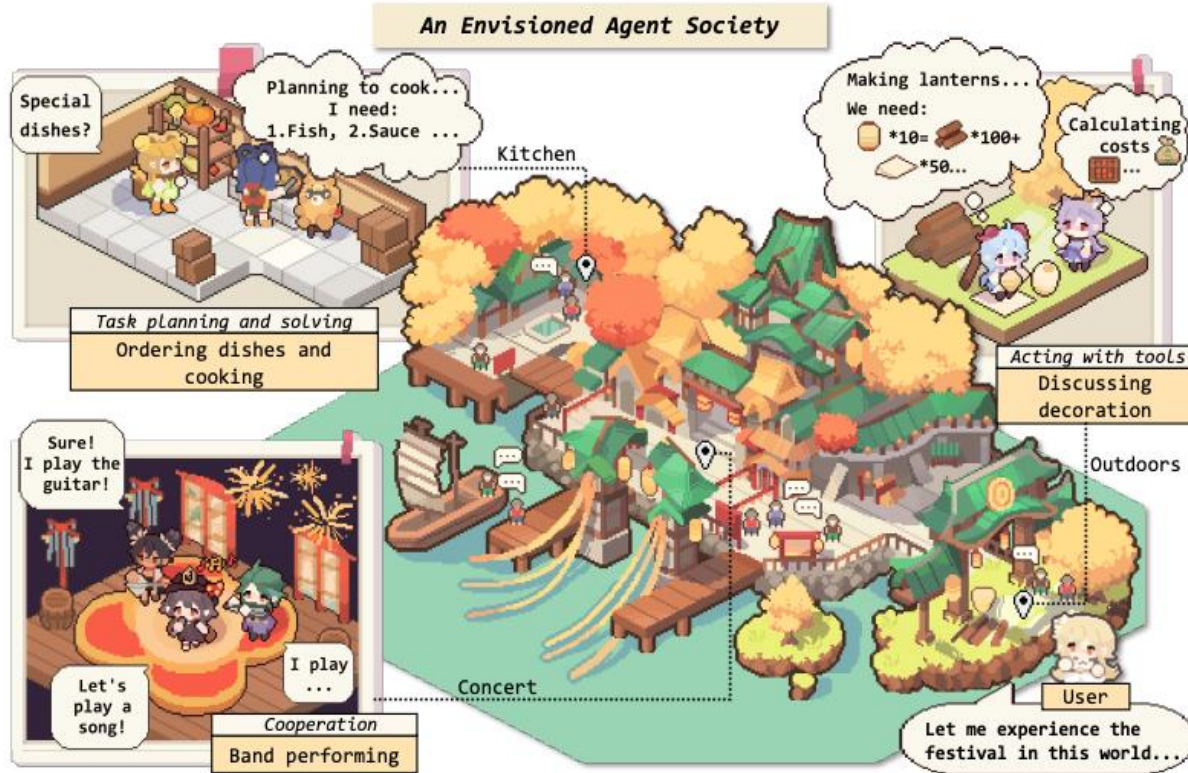
60: This is a speed limit sign, indicating that the maximum allowed speed on this road or section of the road is 60 kilometers per hour (km/h).

Truck with a slash through it: This is a no trucks or no heavy vehicles sign, indicating that trucks or heavy vehicles are not permitted on this road or section of the road.

Car with a horn and a slash through it: This signifies a no honking or horn prohibited zone. Drivers are not allowed to use their vehicle's horn in the area where this sign is displayed.

Figure 6: Results on the ability to understand the traffic signs. **Green** highlights the right answer in understanding, **Red** highlights the wrong answer in understanding. Check Section 2.1 for detailed discussions.

What is LLM Agents



Scenario of an envisioned society composed of AI agents

In the kitchen, one agent **orders dishes**, while another agent is responsible for **planning and solving the cooking task**. At the concert, three agents are collaborating to **perform in a band**. Outdoors, two agents are **discussing lantern-making, planning the required materials, and finances by selecting and using tools**. Users can participate in any of these stages of this social activity

What is LLM Agents

```
> python scripts/main.py
Welcome back! Would you like me to return to being BlogAI?
Continue with the last settings?
Name: BlogAI
Role: an Ai designed to autonomously create a blog post in Ge
Goals: ['research the topic thoroughly', 'write an article tha
st practice examples', "make the blog post interesting by link
he image", 'save article in file as markdown']
Continue (y/n): n
Welcome to Auto-GPT! Enter the name of your AI and its role b
Name your AI: For example, 'Entrepreneur-GPT'
AI Name: NewsAI
NewsAI here! I am at your service.
Describe your AI's role: For example, 'an AI designed to auto
NewsAI is: an AI designed to write news articles
Enter up to 5 goals for your AI: For example: Increase net wo
Enter nothing to load defaults, enter nothing when finished.
Goal 1: find an interesting news topic that includes "AI" and
Goal 2: find an interesting and unusual angle to the topic
Goal 3: write an article from that unusual angle
Goal 4: save the article in a file, in markdown format
Goal 5:
Using memory of type: LocalCache
Thinking...
```

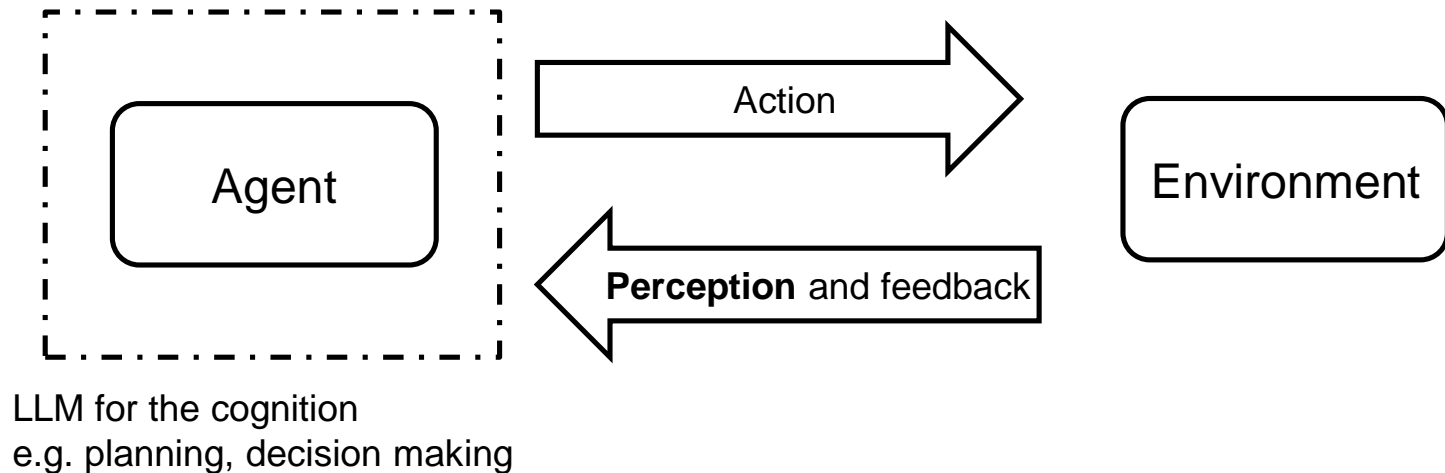
Let an LLM **decide what to do over and over**, while feeding the results of its actions back into the prompt. This allows the program to iteratively and incrementally work towards its objective.

Complete Guide To Setup AutoGPT

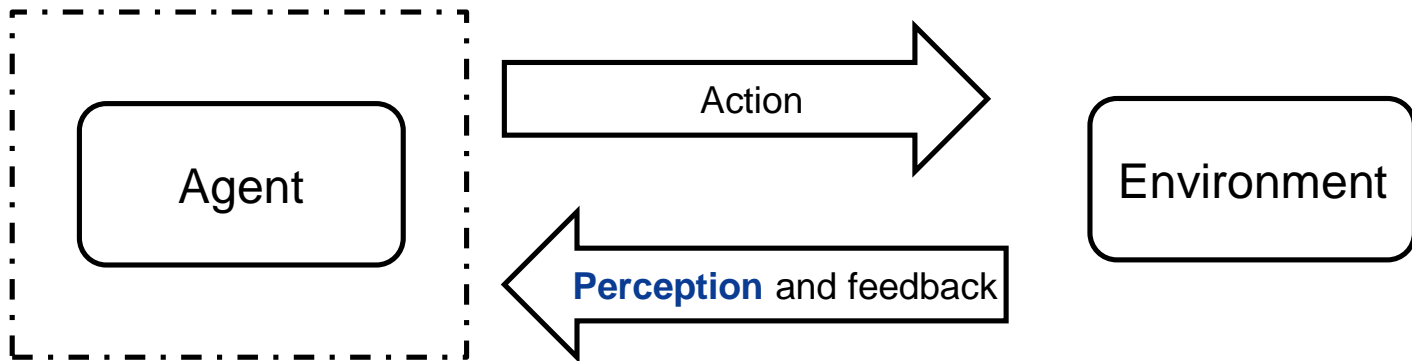
<https://docs.agpt.co/>

The framework of agents

A high-level picture



A high-level picture

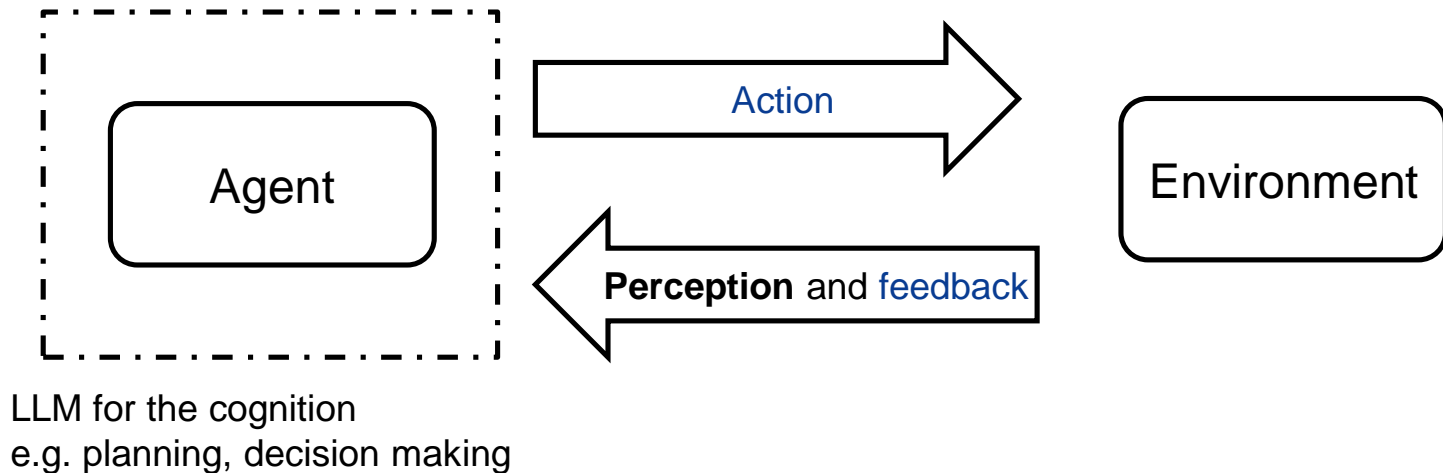


LLM for the cognition
e.g. planning, decision making



Perception

A high-level picture



Action and feedback helps evolution of LLM agents

Use cases of LLM agents

Category

The use cases for LLM agents, or Language Model-based agents, are vast and diverse. These agents, powered by large language models (LLMs), can be used in various scenarios, including:

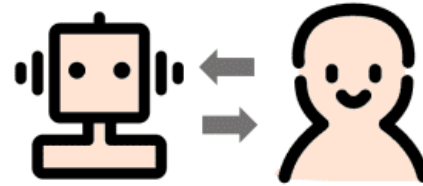
1. Single-agent applications
2. Multi-agent systems
3. Human-Agent cooperation



Single Agent



Agent-Agent

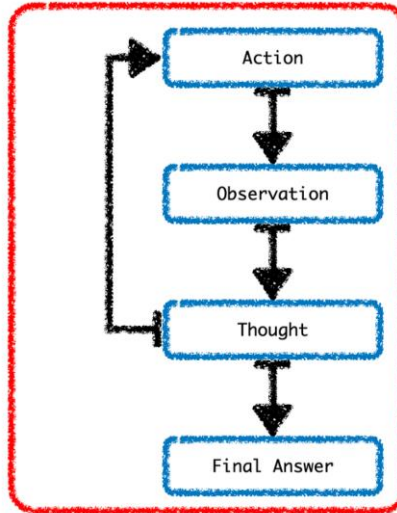


Agent-Human

Single-agent applications

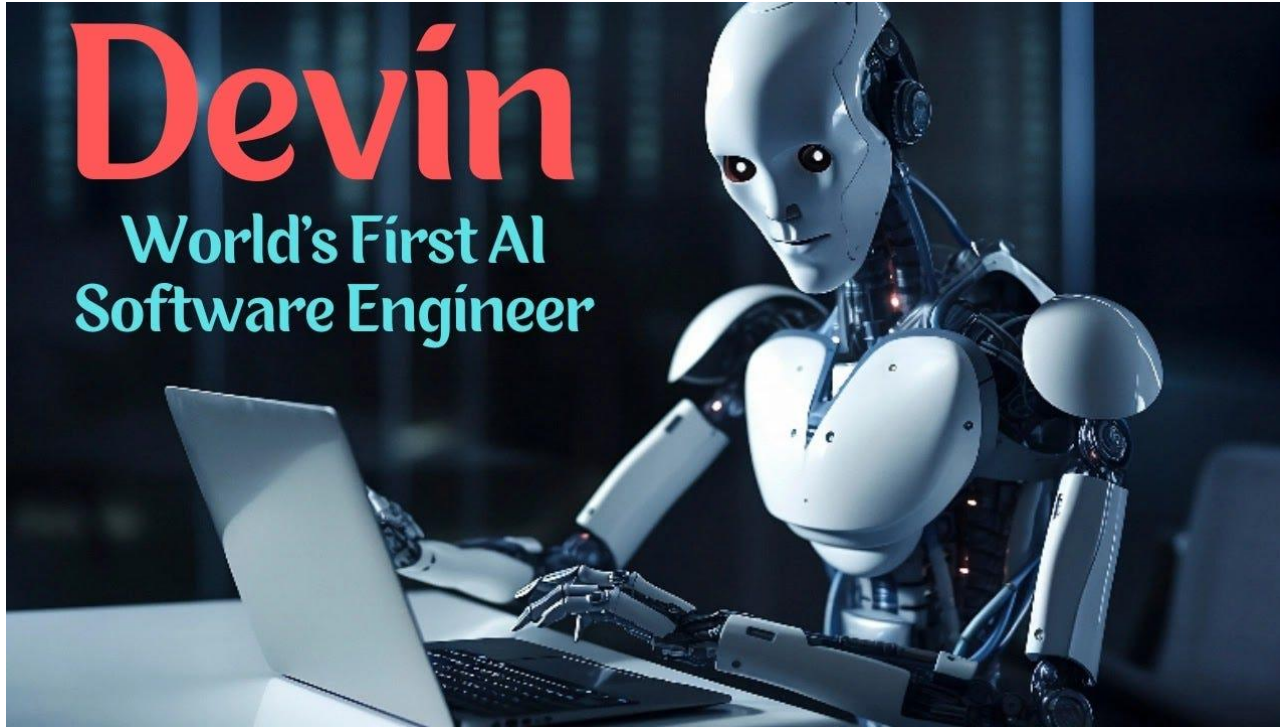


LangChain Agent - Sequence Of Events



LLM agents can be utilized as personal assistants to assist users in **breaking free from daily tasks and repetitive labor**. They can analyze, plan, and solve problems independently, reducing the work pressure on individuals and enhancing task-solving efficiency.

The World's First AI Software Engineer



<https://www.cognition-labs.com/introducing-devin>

Multi-agent systems



Multi-agent systems: LLM agents can interact with each other in a collaborative or competitive manner. This enables them to achieve advancement through teamwork or adversarial interactions. In these systems, agents can **work together on complex tasks** or **compete against each other** to improve their performance.

Play Werewolf (狼人杀)

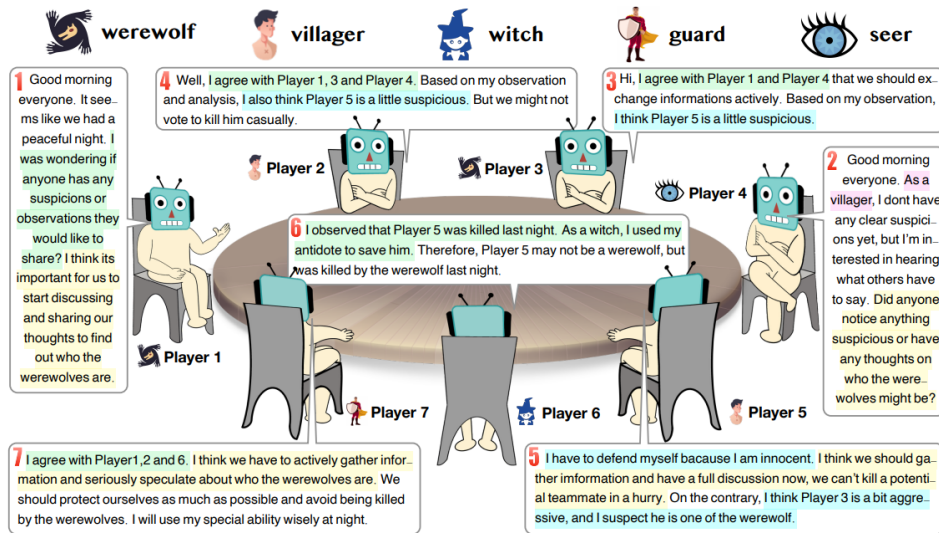
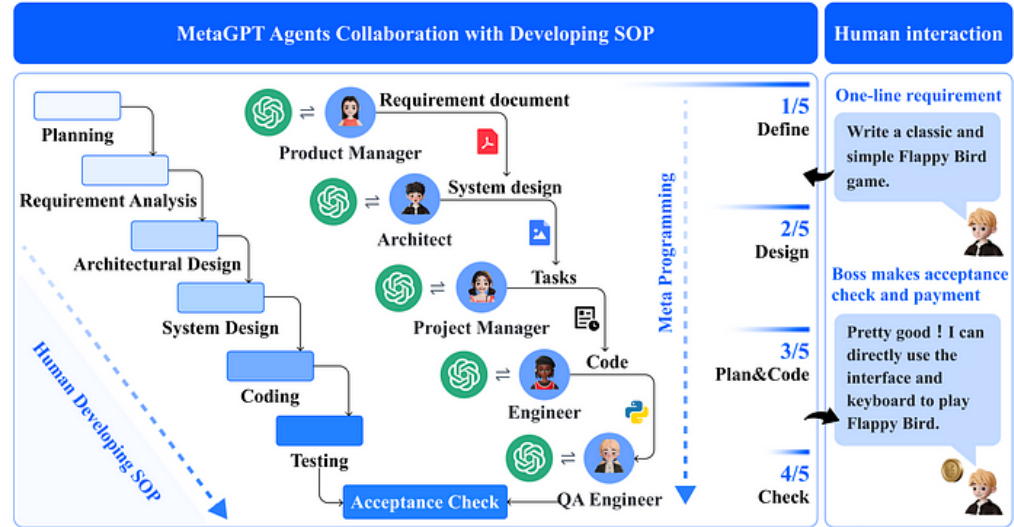


Figure 1: A snapshot of our implemented Werewolf game. There are 5 roles and 7 players, and each of them is acted by an LLM autonomously. The number before each talking denotes the speaking order. Some social behaviors can be primarily observed in this figure, including trust, confrontation, camouflage, and leadership.

Human-Agent cooperation



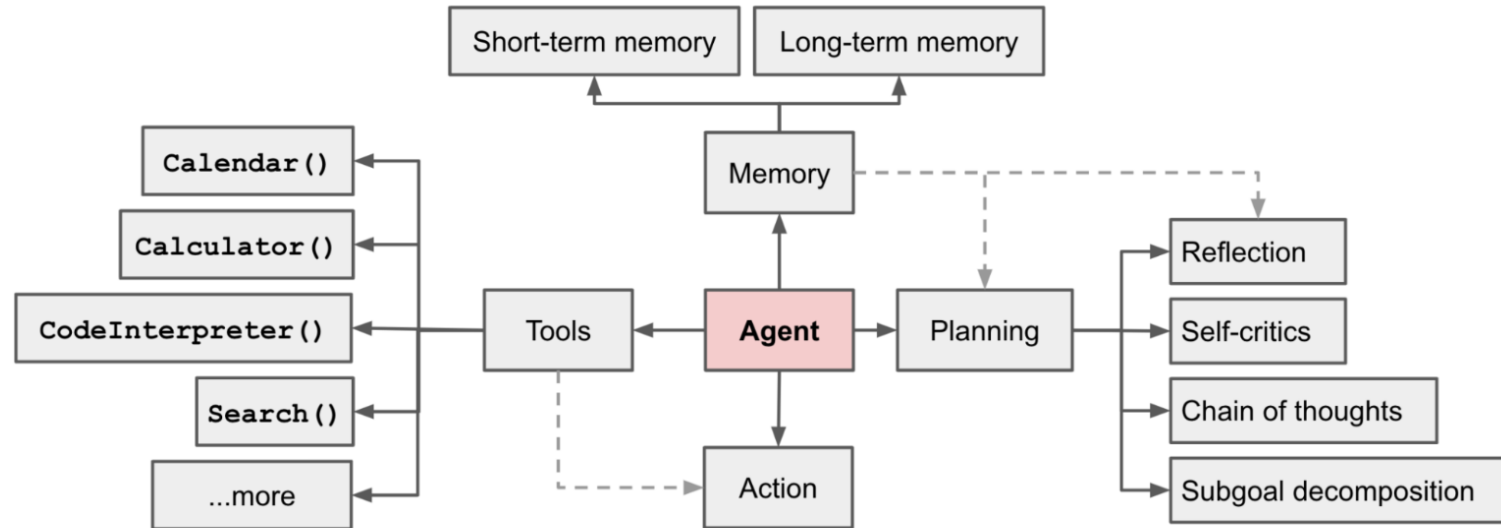
Human-Agent cooperation: LLM agents can **interact with humans**, providing them with assistance and performing tasks more efficiently and safely.

Example: interactively write code together with ChatGPT.

The four Elements

What is LLM Agents

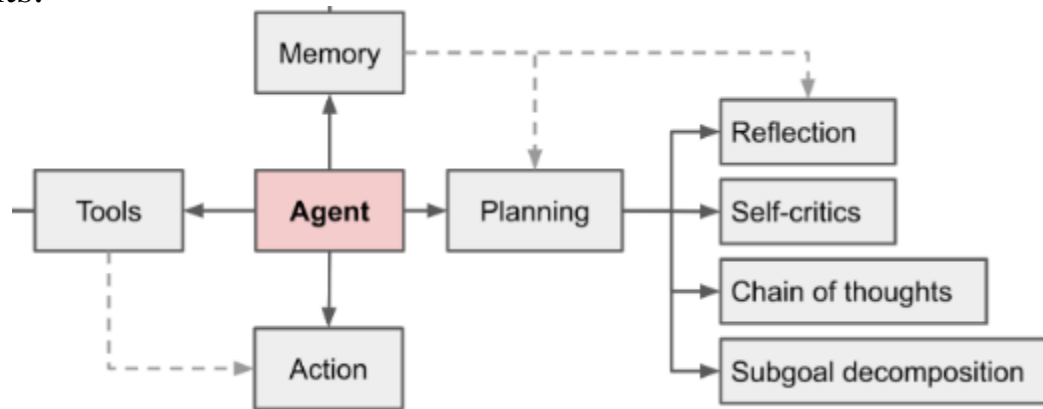
Here is a famous picture from Lilian Weng (from OpenAI)



What is LLM Agents

Planning:

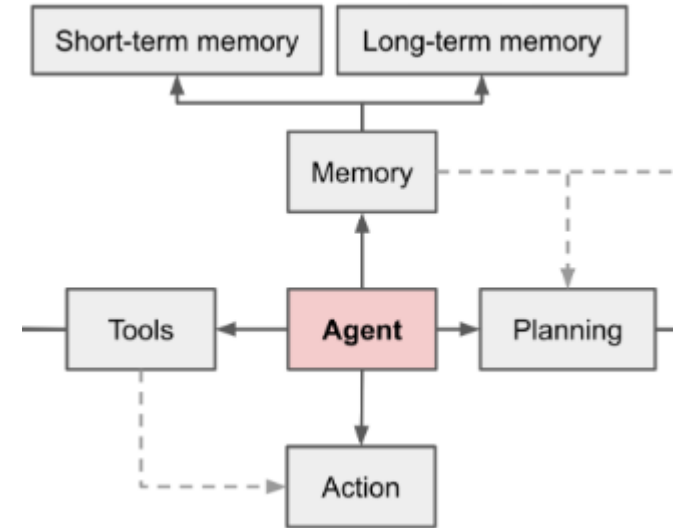
- **Subgoal and decomposition:** The agent breaks down large tasks into smaller, manageable subgoals, enabling efficient handling of complex tasks.
- **Reflection and refinement:** The agent can do self-criticism and self-reflection over past actions, learn from mistakes and refine them for future steps, thereby improving the quality of final results.



What is LLM Agents

Memory:

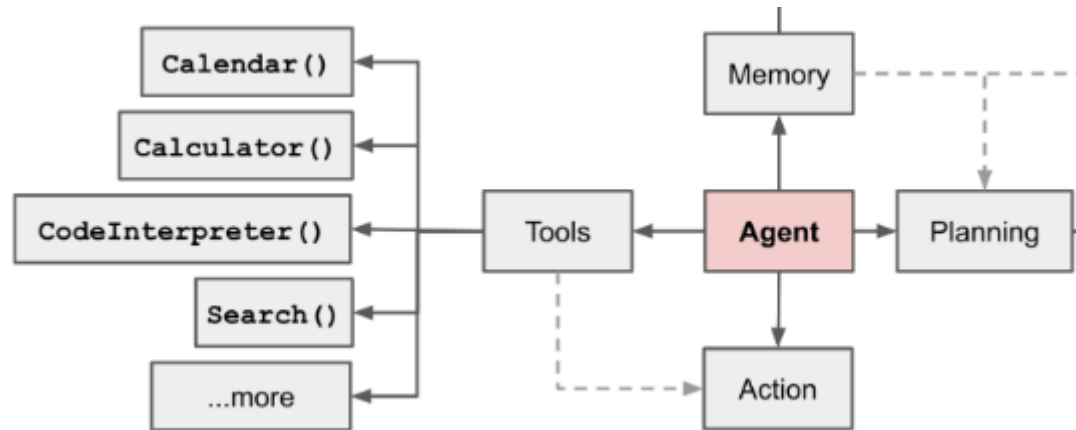
- **Short-term memory:** all the in-context learning is utilizing short-term memory of the model to learn.
- **Long-term memory:** this provides the agent with the capability to retain and recall (infinite) information over extended periods, often by leveraging an external vector store and fast retrieval.



What is LLM Agents

Tool use:

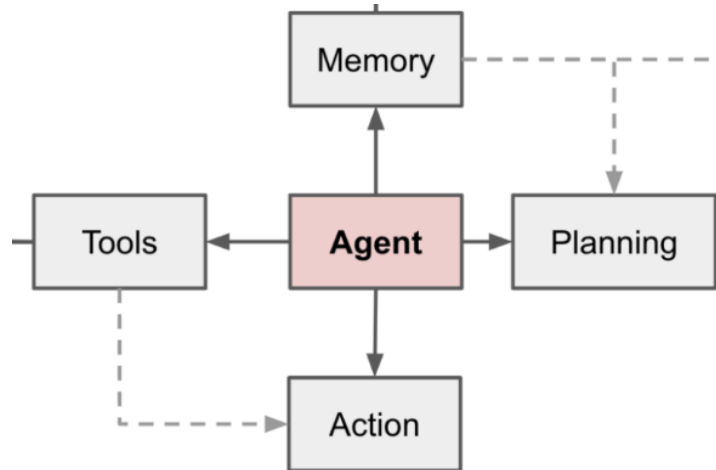
- The agent learns to call **external APIs** for extra information that is missing from the model weights (often hard to change after pre-training), including current information, code execution capability, access to proprietary information sources and more.



What is LLM Agents

Action:

- The agent's ability to execute actions in the real or virtual world is crucial. This can range from performing tasks in a digital environment to controlling physical robots or devices. The execution phase relies on the agent's planning, memory, and tool use to carry out tasks effectively and adaptively.



Why LLM Agents stand out?

- **Language Mastery:** Their inherent capability to both comprehend and produce language ensures seamless user interaction.
- **Decision-making:** LLMs are equipped to reason and decide, making them adept at solving intricate issues.
- **Flexibility:** Their adaptability ensures they can be molded for diverse applications.
- **Collaborative Interactions:** They can collaborate with humans or other agents, paving the way for multifaceted interactions.



	LLM
	LLM + Planning
	LLM + Planning + Memory
	Agents (LLM + Planning + Memory + Tools)

Element 1: Planning

What is planning

How to solve a complicated task **sequentially**?

One-step task: translate an paragraph

- simple
- usually without interaction

 You

translate "I think therefore I am"



ChatGPT


"I think therefore I am" translates to "我思故我在" in Chinese.

Multi-step task: How to put an elephant into a fridge?

- complicated
- it involves multiple steps
- it usually uses **external tools (e.g., operate the fridge)**

 You

How to put an elephant into a fridge?

 ChatGPT

The phrase "How to put an elephant into a fridge?" is often used as a humorous or rhetorical question, as it's a task that's obviously impractical and absurd in real life. This question is typically found in jokes or as a form of light-hearted riddle.

The usual joke answer to this is a three-step process:

1. Open the fridge.
2. Put the elephant in.
3. Close the fridge.

This kind of joke plays on the absurdity and simplicity of the solution to an apparently complex problem. It's a classic example of lateral thinking and humor in riddles.

Two simple examples

- GSM8K (math word problem)

Problem: Beth bakes 4, 2 dozen batches of cookies in a week. If these cookies are shared amongst 16 people equally, how many cookies does each person consume?

Solution: Beth bakes 4 2 dozen batches of cookies for a total of $4*2 = \ll 4*2=8 \gg 8$ dozen cookies

There are 12 cookies in a dozen and she makes 8 dozen cookies for a total of $12*8 = \ll 12*8=96 \gg 96$ cookies

She splits the 96 cookies equally amongst 16 people so they each eat $96/16 = \ll 96/16=6 \gg 6$ cookies

Final Answer: 6

- GAME24

Input: 4 4 6 8

Steps:

$4 + 8 = 12$ (left: 4 6 12)

$6 - 4 = 2$ (left: 2 12)

$2 * 12 = 24$ (left: 24)

Answer: $(6 - 4) * (4 + 8) = 24$

They are both multi-step problems!

Examples of Planning

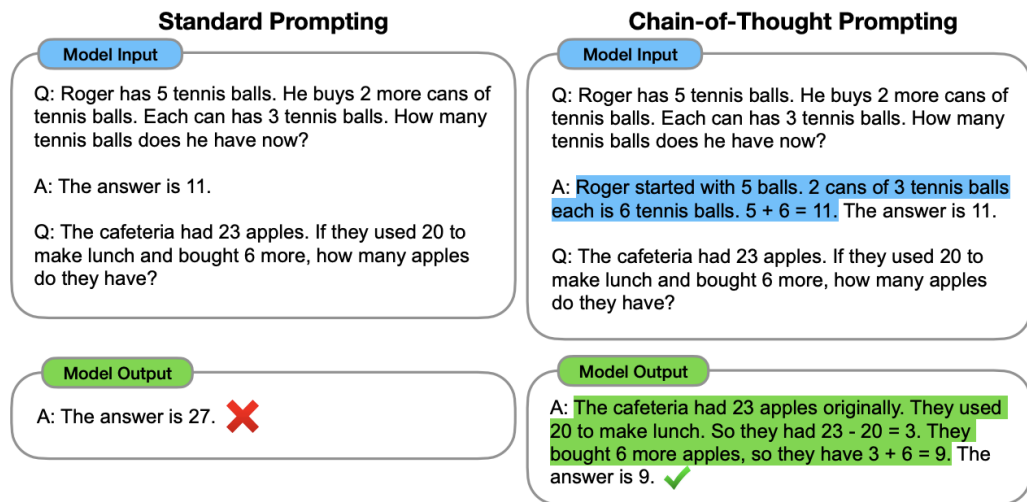
Task Decomposition

Self-Reflection/self-refinement

Planning with Task Decomposition

Task Decomposition: Chain of thought

Chain of Thought (CoT) has become a standard prompting technique for enhancing model performance on complex tasks. The model is instructed to “*think step by step*” to utilize more test-time computation to **decompose hard tasks into smaller and simpler steps**. CoT transforms big tasks into multiple manageable tasks and shed lights into an interpretation of the model’s thinking process.



[Chain-of-Thought Prompting Elicits Reasoning in Large Language Models](#)

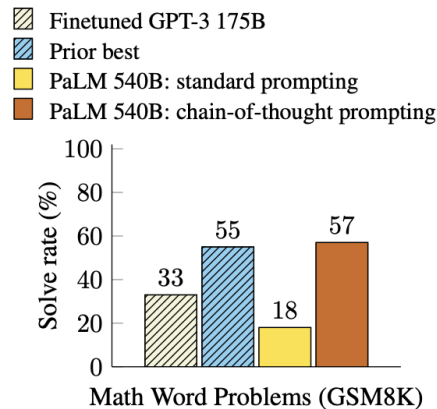
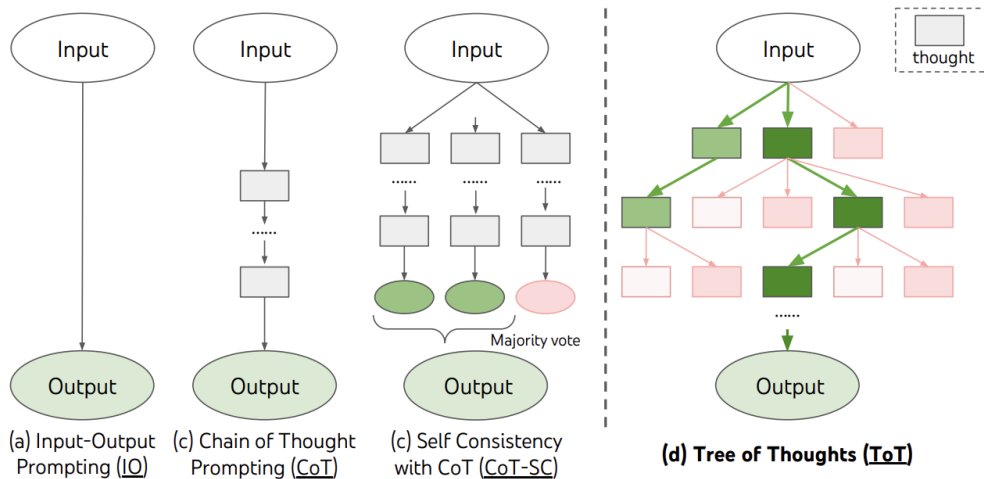


Figure 2: PaLM 540B uses chain-of-thought prompting to achieve new state-of-the-art performance on the GSM8K benchmark of math word problems. Finetuned GPT-3 and prior best are from Cobbe et al. (2021).

Task Decomposition: **Tree of Thoughts**

Tree of Thoughts extends CoT by exploring multiple reasoning possibilities at each step. It first decomposes the problem into multiple thought steps and generates multiple thoughts per step, creating a tree structure. The search process can be BFS (breadth-first search) or DFS (depth-first search) with each state evaluated by a classifier (via a prompt) or majority vote.



Task Decomposition: Tree of Thoughts

- Resource Intensity:** Implementing search methods like ToT is more resource-intensive, incurring higher costs compared to simpler sampling methods.

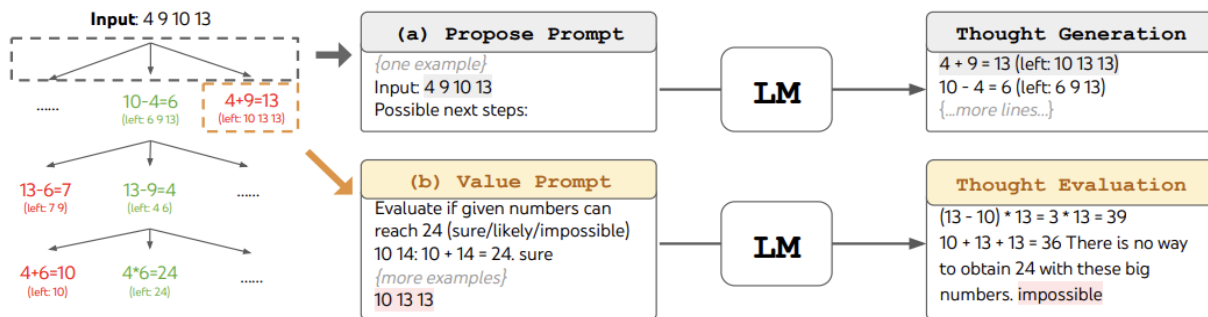
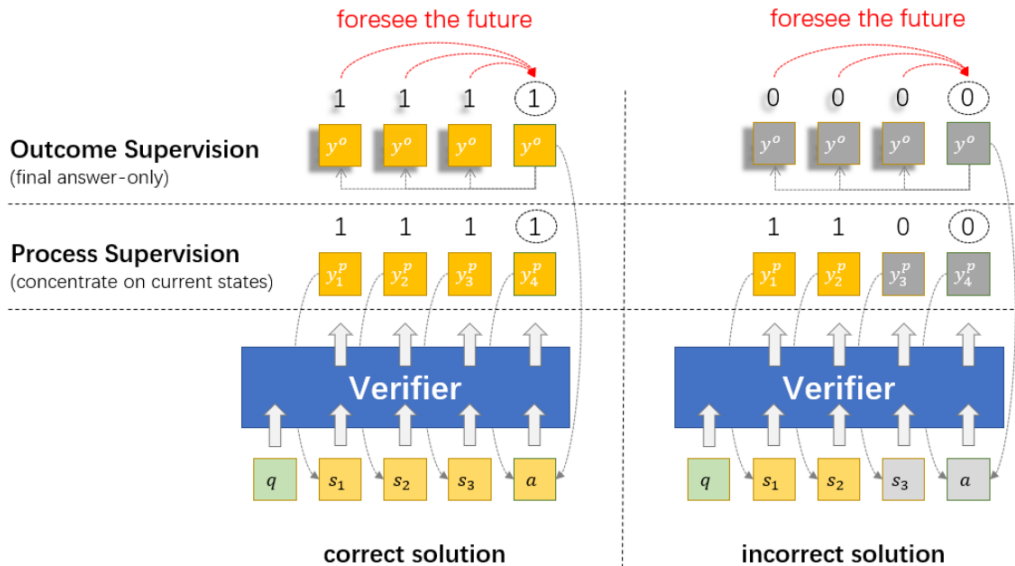


Figure 2: ToT in a game of 24. The LM is prompted for (a) thought generation and (b) valuation.

Method	Success
IO prompt	7.3%
CoT prompt	4.0%
CoT-SC (k=100)	9.0%
ToT (ours) (b=1)	45%
ToT (ours) (b=5)	74%
IO + Refine (k=10)	27%
IO (best of 100)	33%
CoT (best of 100)	49%

Table 2: Game of 24 Results.

Our research: better verification

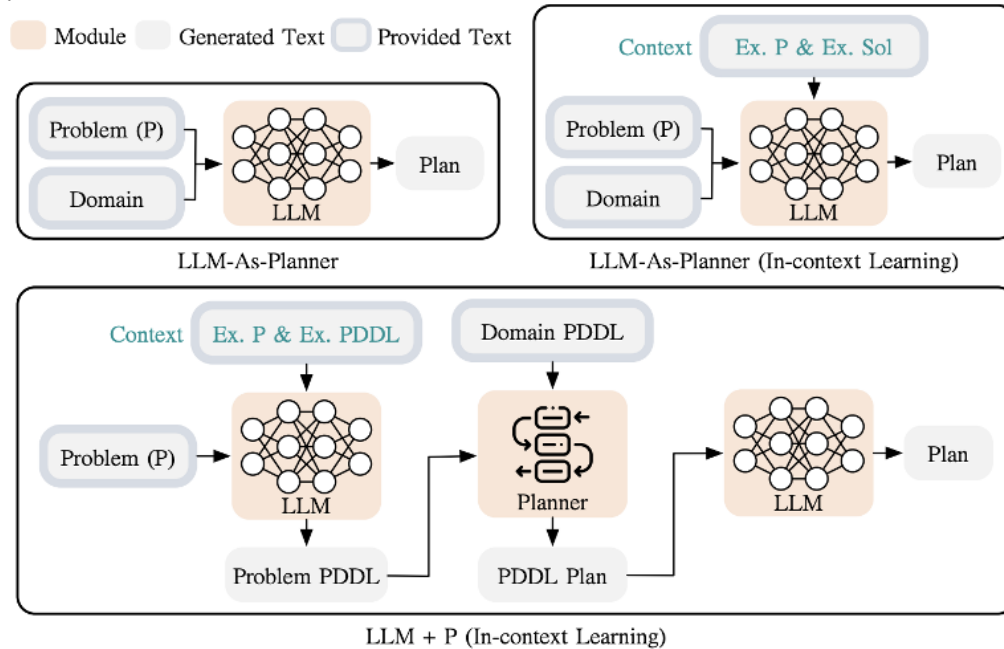


SOTA performance on mathematical reasoning

Model	Size	GPT-3.5/GPT-4	Data Augmentation	Accuracy
Open-Source Models without Code Execution				
RFT (Yuan et al., 2023)	7B			51.2%
WizardMath (Luo et al., 2023)	7B		✓	54.9%
MetaMath (Yu et al., 2023)	7B	✓	✓	66.4%
MuggleMATH (Li et al., 2023a)	7B	✓	✓	68.4%
Arithmo-Mistral	7B	✓	✓	74.7%
MetaMath-Mistral (Yu et al., 2023)	7B	✓	✓	77.7%
RFT (Yuan et al., 2023)	13B			55.3%
WizardMath (Luo et al., 2023)	13B		✓	63.9%
MetaMath (Yu et al., 2023)	13B	✓	✓	71.0%
MuggleMATH (Li et al., 2023a)	13B	✓	✓	74.0%
RFT (Yuan et al., 2023)	70B			64.8%
WizardMath (Luo et al., 2023)	70B		✓	81.6%
MuggleMATH (Li et al., 2023a)	70B	✓	✓	82.3%
MetaMath (Yu et al., 2023)	70B	✓	✓	84.3%
Ours – OVM (Llama2-7B, K=100)	7B			73.7% ± 0.4%
Ours – OVM (Mistral-7B, K=100)	7B			84.7% ± 0.3%
Open-Source Models with Code Execution				
ToRA-Code (Gou et al., 2023)	7B	✓	✓	72.6%
ToRA-Code (Gou et al., 2023)	13B	✓	✓	75.8%
ToRA-Code (SC, K=50) (Gou et al., 2023)	34B	✓	✓	85.1%
ToRA (Gou et al., 2023)	70B	✓	✓	84.3%
ToRA (SC, K=50) (Gou et al., 2023)	70B	✓	✓	88.3%

Task Decomposition: LLM+P

LLM+P involves relying on an external classical planner to do long-horizon planning. This approach utilizes the Planning Domain Definition Language (PDDL) as an intermediate interface to describe the planning problem.



Task Decomposition: LLM+P

In the PDDL process, LLM

- 1) **translates** the problem into “Problem PDDL”;
- 2) requests a classical planner to generate a **PDDL plan** based on an existing “Domain PDDL”;
- 3) translates the PDDL plan back into **natural language**.

Essentially, the planning step is outsourced to an **external tool**, assuming the availability of domain-specific PDDL and a suitable planner.

An Example PDDL Problem File Written by GPT-4 with Context

Context (shaded):

An example planning problem is: You have 5 blocks. b2 is on top of b5. b5 is on top of b1. b1 is on top of b4. b3 is on top of b2. b4 is on the table. b3 is clear. Your arm is empty. Your goal is to move the blocks. b4 should be on top of b3. The problem PDDL is:

```
(:objects b1 b2 b3 b4 b5)
(:init (arm-empty) (on b1 b4) (on
b2 b5) (on b3 b2)
      (on-table b4) (on b5 b1)
(clear b3))
(:goal (and (on b4 b3)))
```

Prompt: Description of Problem (P1) + Context + Provide me with the problem PDDL file that describes the planning problem directly without further explanations.

GPT-4 (the generated problem PDDL):

```
(:objects b1 b2 b3 b4 b5 )
(:init (arm-empty) (on-table b1)
(on b2 b1) (on b3 b4)
      (on b4 b2) (on b5 b3) (clear
b5)))
(:goal (and (on b1 b2) (on b3 b5)
(on b4 b1)))
```

Planner:

```
(unstack b5 b3) (putdown b5)
(unstack b3 b4) (stack b3 b5)
(unstack b4 b2) (putdown b4)
(unstack b2 b1) (putdown b2)
(pickup b1) (stack b1 b2) (pickup
b4) (stack b4 b1)
```

From 机器之心

6.3 程序语言作为心智语言

上述基于 LLM 的数学解题和自然语言理解方法中，一个自然的想法是用程序语言表示心智语言。这是因为 LLM 一般使用程序训练，也能生成程序。

我们最近做了大规模的实验，验证了 Python 程序比英语（自然语言）作为“心智语言”，在数学解题中更有优势的事实 [12]。这个方法的一个优点是，LLM 理解题意后，得到的程序可以直接通过解释器执行，验证解题步骤的正确性。在 Python 程序上进行推理，也比在自然语言上进行推理更为容易。



[12] Jie, Z., Luong, T.Q., Zhang, X., Jin, X. and Li, H., 2023. Design of a Chain-of-Thought in Math Problem Solving. arXiv preprint arXiv:2309.11054.

Python > Wolfram

Input

Question: a is two years older than b who is twice as old as c. If the total of the ages of a, b and c be 27, then how old is b ?

a) 7, b) 9, c) 8, d) 11, e) 10

Answer reasoning:

NL CoT

Let's start by assigning variables to the ages of A, B, and C. Let A = age of A. Let B = age of B. Let C = age of C

From the problem, we know that:

A = B + 2 (A is two years older than B)

B = 2C (B is twice as old as C)

A + B + C = 27 (The total of their ages is 27)

Now we can substitute the second equation into the first equation to get: A = 2C + 2

We can then substitute both equations into the third equation to get: (2C + 2) + 2C + C = 27

Simplifying this equation, we get: 5C + 2 = 27; 5C = 25; C = 5

Now that we know C is 5, we can use the second equation to find B: B = 2C = 2(5) = 10

Therefore, the answer is <E>. B is 10 years old.

Therefore, the answer is: E

Wolfram CDP CoT

```
1.v1 = a == b + 2 (* a is two years older than b *)
2.v2 = b == 2*c (* who is twice as old as c *)
3.v3 = a + b + c == 27 (* the total of the ages of a , b and c
be 27 *)
4.option = <| A -> 7, B -> 9, C -> 8, D -> 11, E -> 10 |>
5.v4 = Solve[{v1, v2, v3}, {a, b, c}] (* Step 1: Solve the syst
em of equations *)
6.v5 = b /. v4[[1]] (* Step 2: Get the value of b *)
7.Keys[Select[option, # == v5 &]] (* So the correct option is *
```

Wolfram SDP CoT

```
1.a_is_two_years_older_than_b = a - 2 == b
2.b_is_twice_as_old_as_c = b == 2*c
3.total_age = a + b + c == 27
4.option = <| A -> 7, B -> 9, C -> 8, D -> 11, E -> 10 |>
5.solution = Solve[{a_is_two_years_older_than_b, b_is_twice_as_old_as_c
total_age}, {a, b, c}]
6.b_age_value = b /. solution[[1]]
7.Keys[Select[option, # == b_age_value &]]
```

Python CDP CoT

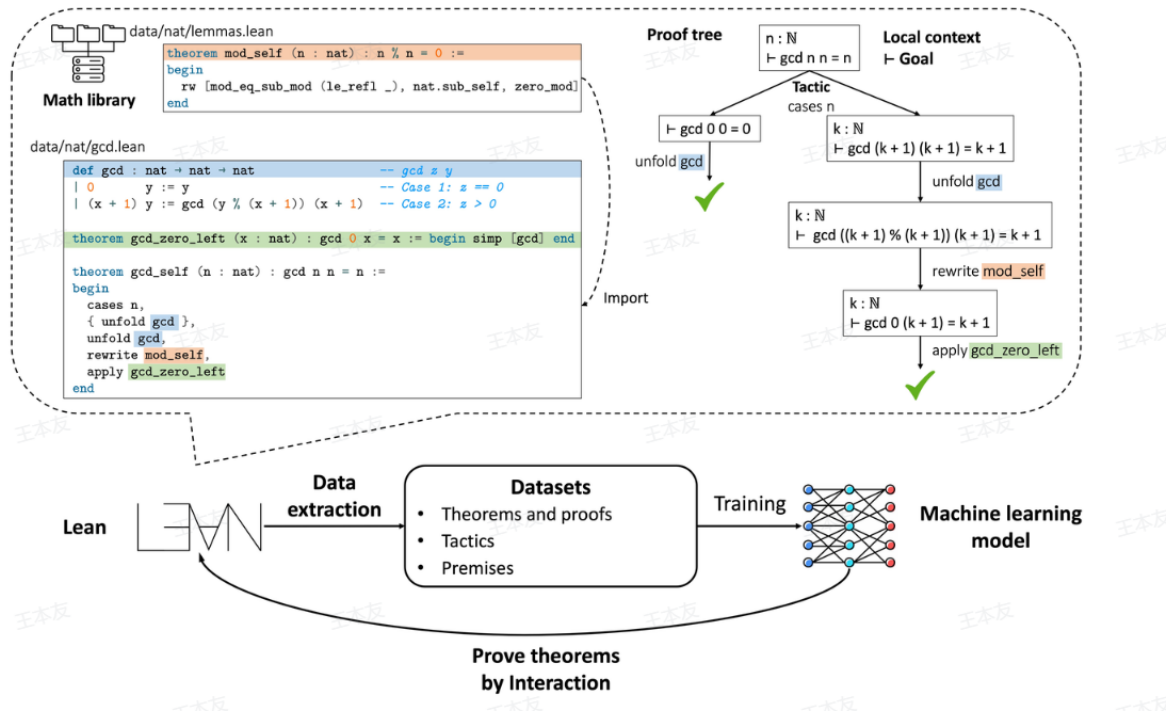
```
1.def solution():
2.     import math
3.     import sympy
4.     v1 = sympy.Symbol('a')
5.     v2 = sympy.Symbol('b')
6.     v3 = sympy.Symbol('c')
7.     options = [7, 9, 8, 11, 10]
8.     v4 = sympy.Eq(v1, v2 + 2) #a is two years older than b
9.     v5 = sympy.Eq(v2, v3 * 2) #b is twice as old as c
10.    v6 = sympy.Eq(v1 + v2 + v3, 27) #the total of the ages
of a , b and c be 27
11.    v7=sympy.solve([v4,v5,v6])[v2] #how old is b
12.    correct_option = None
13.    for i, option in enumerate(options):
14.        if math.fabs(option - v7) < 1e-4:
15.            correct_option = chr(ord('A') + i)
16.            break
17.    result = correct_option
18.    return result
```

Python SDP CoT

```
1.def solution():
2.     import math
3.     import sympy
4.     a = sympy.Symbol('a')
5.     b = sympy.Symbol('b')
6.     c = sympy.Symbol('c')
7.     options = [7, 9, 8, 11, 10]
8.     a_two_years_older_than_b = sympy.Eq(a, b + 2)
9.     b_twice_as_c = sympy.Eq(b, c * 2)
10.    sum_of_a_b_c = sympy.Eq(a + b + c, 27)
11.    b_value=sympy.solve([a_two_years_older_than_b,b_twice_as_
c,sum_of_a_b_c])[b]
12.    correct_option = None
13.    for i, option in enumerate(options):
14.        if math.fabs(option - b_value) < 1e-4:
15.            correct_option = chr(ord('A') + i)
16.            break
17.    result = correct_option
18.    return result
```

Figure 1: Examples of CoT representations: Natural Language (NL) CoT, Comment-Describing Program (CDP) and Self-Describing Program (SDP) in both Wolfram and Python.

For Math and auto theorem prove



The Lean language is powerful for logical reasoning

Planning with Self-Reflection

Self-Reflection

Self-reflection is a vital aspect that allows autonomous agents to improve iteratively by refining past action decisions and correcting previous mistakes. It plays a crucial role in real-world tasks where trial and error are inevitable.

Reflective Thinking



Self-Reflection: **ReACT**

ReACT integrates reasoning and acting within LLM by extending the action space to be a combination of **task-specific discrete actions** and **the language space**. The former enables LLM to interact with the environment (e.g. use Wikipedia search API), while the latter prompting LLM to generate reasoning traces in natural language.

The ReAct prompt template incorporates explicit steps for LLM to think, roughly formatted as:

```
Thought: ...  
Action: ...  
Observation: ...  
... (Repeated many times)
```

Self-Reflection: ReACT

HotpotQA

(1d) ReAct (Reason + Act)

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

Act 1: Search[Apple Remote]

Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

Act 2: Search[Front Row]

Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

Thought 3: Front Row is not found. I need to search Front Row (software) .

Act 3: Search[Front Row (software)]

Obs 3: Front Row is a discontinued media center software ...

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

Act 4: Finish[keyboard function keys]



AlfWorld Env

(2b) ReAct (Reason + Act)

Act 1: Think[First I need to find a pepper shaker...more likely to appear in cabinets (1-6), countertops (1-3), ...]

Act 2: Go to cabinet 1

Obs 2: On the cabinet 1, you see a vase 2.

(Here the agent go to cabinet 1, then cabinet 2, 3, then countertop 1 and 2)

Act 6: Go to countertop 3

Obs 6: On the countertop 3, you see a apple 1, a bread 1, a pepper shaker 1, and a vase 3.

Act 7: Take pepper shaker 1 from countertop 3

Obs 7: You pick up the pepper shaker 1 from the countertop 3.

Act 8: Think[Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1.]

Act 9: Go to drawer 1

Obs 9: Drawer 1 is closed.

Act 10: Open drawer 1

Obs 10: You open Drawer 1 ...

Act 11: Put pepper shaker 1 in/on drawer 1

Obs 11: You put pepper shaker 1 in/on the drawer 1.

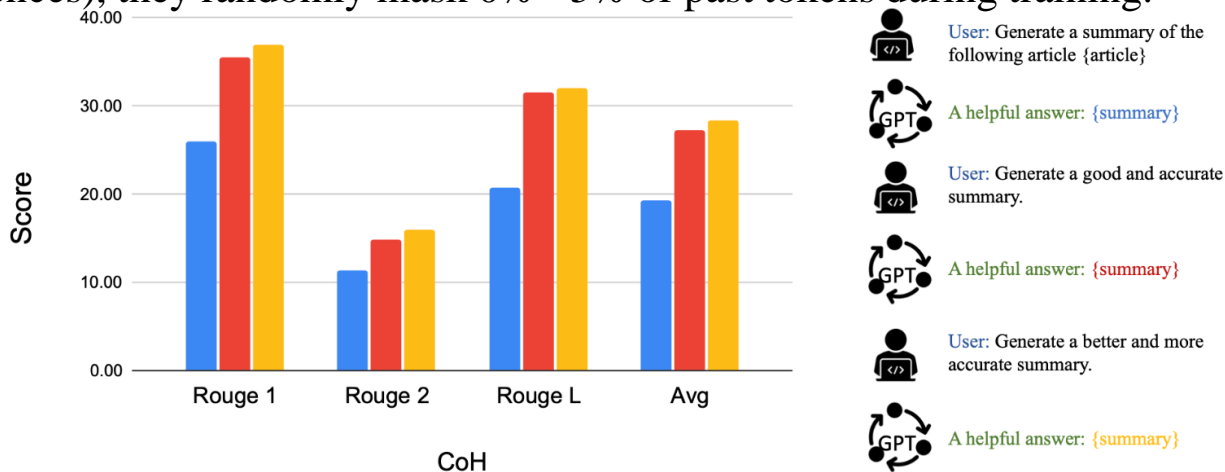


In both experiments on knowledge-intensive tasks and decision-making tasks, *ReAct* works better than the *Act*-only baseline where *Thought*: ... step is removed.

Self-Reflection: Chain of Hindsight

Chain of Hindsight (CoH) encourages the model to improve on its own outputs by explicitly presenting it with a sequence of past outputs, each annotated with feedback.

To avoid overfitting, CoH adds a regularization term to maximize the log-likelihood of the pre-training dataset. To avoid shortcutting and copying (because there are many common words in feedback sequences), they randomly mask 0% - 5% of past tokens during training.



Element 2: tools

Introduction to tools in LLMs

Human + tool use: motivations

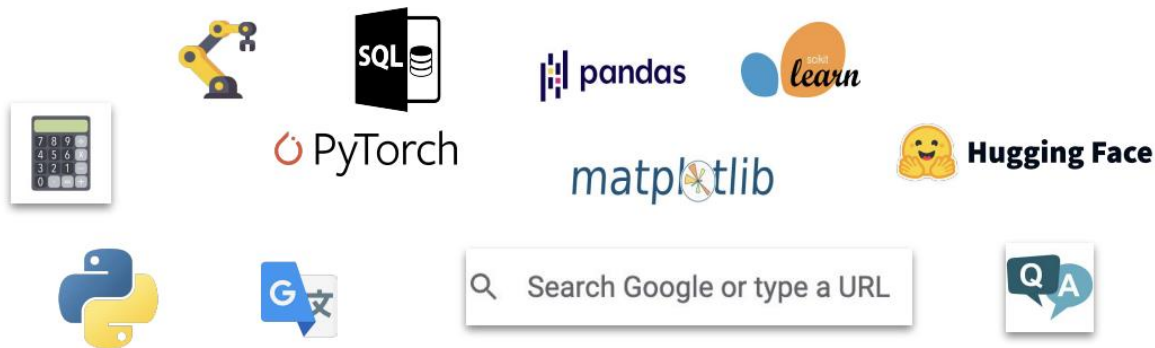
As humans, we have limited time and memory, feel tired, and have emotions.

- Human + tool use
 - Enhanced scalability
 - Improved consistency
 - Greater interpretability
 - Higher capacity and productivity



LLMs + tool use: motivations

- Just like human, LLMs suffer from the similar limitations. But in the same way,
- LLMs + tool use
 - Enhanced scalability
 - Improved consistency
 - Greater interpretability
 - Higher capacity and productivity



Element 2: tools

Recent Works

In the case of calculator

- The early version GPT-4 struggled for numeric calculator

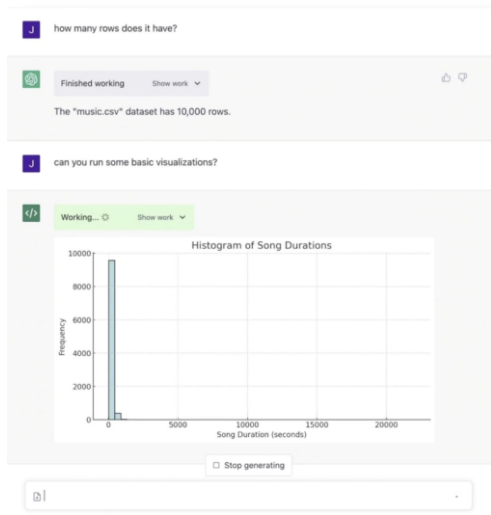


- Using LLM to deal with this, it is a waste of network capacity!

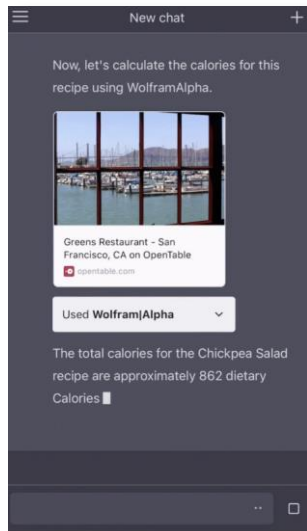
LLMs + tool use in perspective of **executable language grounding**

Ground language models into **executable actions**

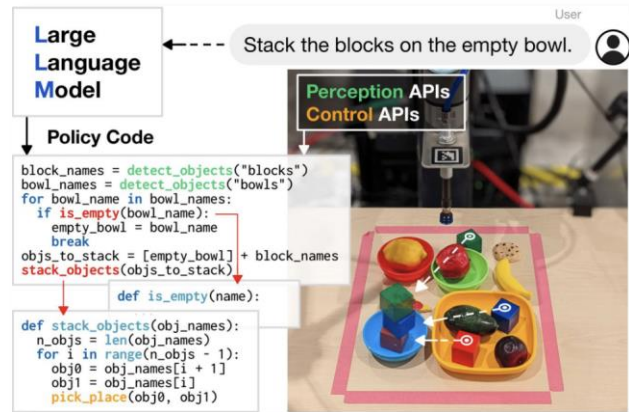
- Mapping natural language instructions into code or actions executable within various environments such as databases, web applications, and robotic physical world.
- LM (planning and reasoning) + **actions**



Data analysis



Web/Apps



<https://openai.com/blog/chatgpt-plugins>

<https://code-as-policies.github.io/>

LLMs + tool use in perspective of **executable language grounding**

LLMs + tool use in executable language grounding tasks

Inputs

- **Language:** user question/request
- **Toolkit:** code, APIs to search engines, self-defined functions, expert models...
- **Environment:** databases, IDE, web/apps, visual and robotic physical world...

Outputs

- Grounded reasoning code/action seq that can be executed in the corresponding environment
 - What tools to select, when and how to use the selected tools

LLMs + tool: PAL, PoT

Chain-of-Thought (Wei et al., 2022)

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 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves. They sold 93 in the morning and 39 in the afternoon. So they sold $93 + 39 = 132$ loaves. The grocery store returned 6 loaves. So they had $200 - 132 - 6 = 62$ loaves left.

The answer is 62.



Program-aided Language models (this work)

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 tennis balls.

tennis_balls = 5

2 cans of 3 tennis balls each is

bought_balls = 2 * 3

tennis balls. The answer is

answer = tennis_balls + bought_balls

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves

loaves_baked = 200

They sold 93 in the morning and 39 in the afternoon

loaves_sold_morning = 93

loaves_sold_afternoon = 39

The grocery store returned 6 loaves.

loaves_returned = 6

The answer is

answer = loaves_baked - loaves_sold_morning
- loaves_sold_afternoon + loaves_returned

```
>>> print(answer)
```

```
74
```



Question: In Fibonacci sequence, it follows the rule that each number is equal to the sum of the preceding two numbers. Assuming the first two numbers are 0 and 1, what is the 50th number in Fibonacci sequence?

The first number is 0, the second number is 1, therefore, the third number is $0+1=1$. The fourth number is $1+1=2$. The fifth number is $1+2=3$. The sixth number is $2+3=5$. The seventh number is $3+5=8$. The eighth number is $5+8=13$ (Skip 1000 tokens)
The 50th number is 32,432,268,459.

CoT

```
length_of_fibonacci_sequence = 50
fibonacci_sequence = np.zeros(length_of_)
fibonacci_sequence[0] = 0
fibonacci_sequence[1] = 1
for i in range(3, length_of_fibonacci_sequence):
    fibonacci_sequence[i] = fibonacci_sequence[i-1] +
    fibonacci_sequence[i-2]
ans = fibonacci_sequence[-1]
```

PoT

32,432,268,459



python

12,586,269,025



Question: Ketty saves 20000 dollars to the bank. After three years, the sum with compound interest rate is 1000 dollars more than the sum with simple interest rate. What is the interest rate of the bank?

Assuming the interest rate is x . The sum after two years with simple interest rate is $20000 + x * 20000 * 3 = 20000 + 60000x$. The sum after two years with compound interest rate is $20000 * (1 + x)^3 = 20000 + 60000 * x + 60000x^2 + 20000x^3$. The difference can be written as $60000x^2 + 20000x^3 = 1000$. In order to solve x , we can use the quadratic formula. $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$, ..., $x = \frac{-20000 \pm \sqrt{61600}}{120000}$, $x = -0.051333$.

CoT

```
interest_rate = Symbol('x')
sum_in_two_years_with_simple_interest = 20000 +
interest_rate * 20000 * 3
sum_in_two_years_with_compound_interest = 20000 * (1 +
interest_rate)**3
# Since compound interest is 1000 more than simple interest.
ans = solve(sum_after_in_yeras_with_compound_interest -
sum_after_two_years_in_compound_interest - 1000,
interest_rate)
```

PoT

-0.051333



python



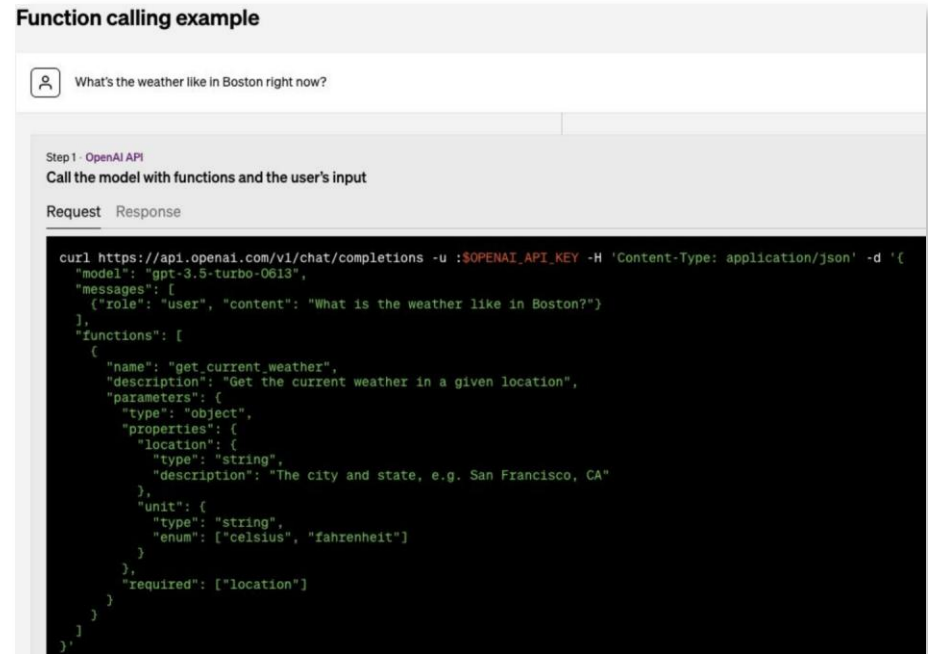
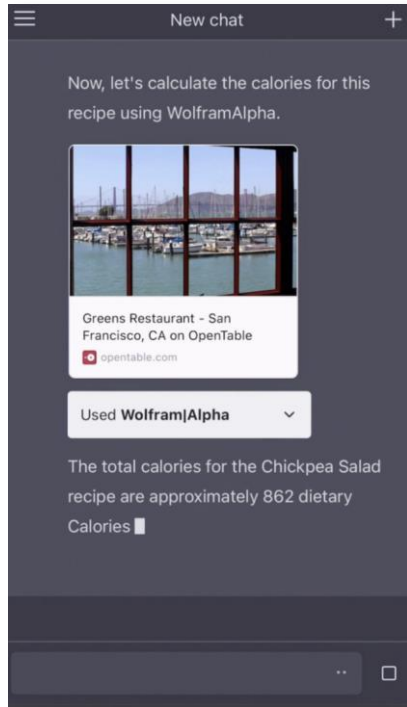
$x = 0.24814$



[PAL: Program-aided Language Models](#)

Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks

LLMs + webs/apps or personalized functions: ChatGPT-Plugins



LLMs + webs/apps or personalized functions: ReAct

(2) AlfWorld

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 1, a coffee machine 1, a countertop 3, a stove burner 1, and a toaster 1.
Your task is to: Put some pepper shaker on a drawer.

(2a) Act-Only

Act 1: Go to drawer 1

Obs 1: The drawer 1 is closed.

Act 2: Open drawer 1

Obs 2: You open the drawer 1. The drawer 1 is open. In it, you see a dish sponge 2, and a spoon 1.

Act 3: Go to sinkbasin 1

Obs 3: On the sinkbasin 1, you see a dish sponge 3, a spatula 1, and a spoon 2.

Act 4: Take peppershaker 1 from sinkbasin 1

Obs 4: Nothing happens.

Act 5: Take peppershaker 1 from sinkbasin 1

Obs 5: Nothing happens.



(2b) ReAct (Reason + Act)

Act 1: Think[First I need to find a pepper shaker...more likely to appear in cabinets (1-6), countertops (1-3), ...]

Act 2: Go to cabinet 1

Obs 2: On the cabinet 1, you see a vase 2.

observations

(Here the agent go to cabinet 1, then cabinet 2, 3, then countertop 1 and 2)

Act 6: Go to countertop 3

Obs 6: On the countertop 3, you see a apple 1, a bread 1, a pepper shaker 1, and a vase 3.

Act 7: Take pepper shaker 1 from countertop 3

Obs 7: You pick up the pepper shaker 1 from the countertop 3.

Act 8: Think[Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1.]

Act 9: Go to drawer 1

Obs 9: Drawer 1 is closed.

Act 10: Open drawer 1

Obs 10: You open Drawer 1 ...

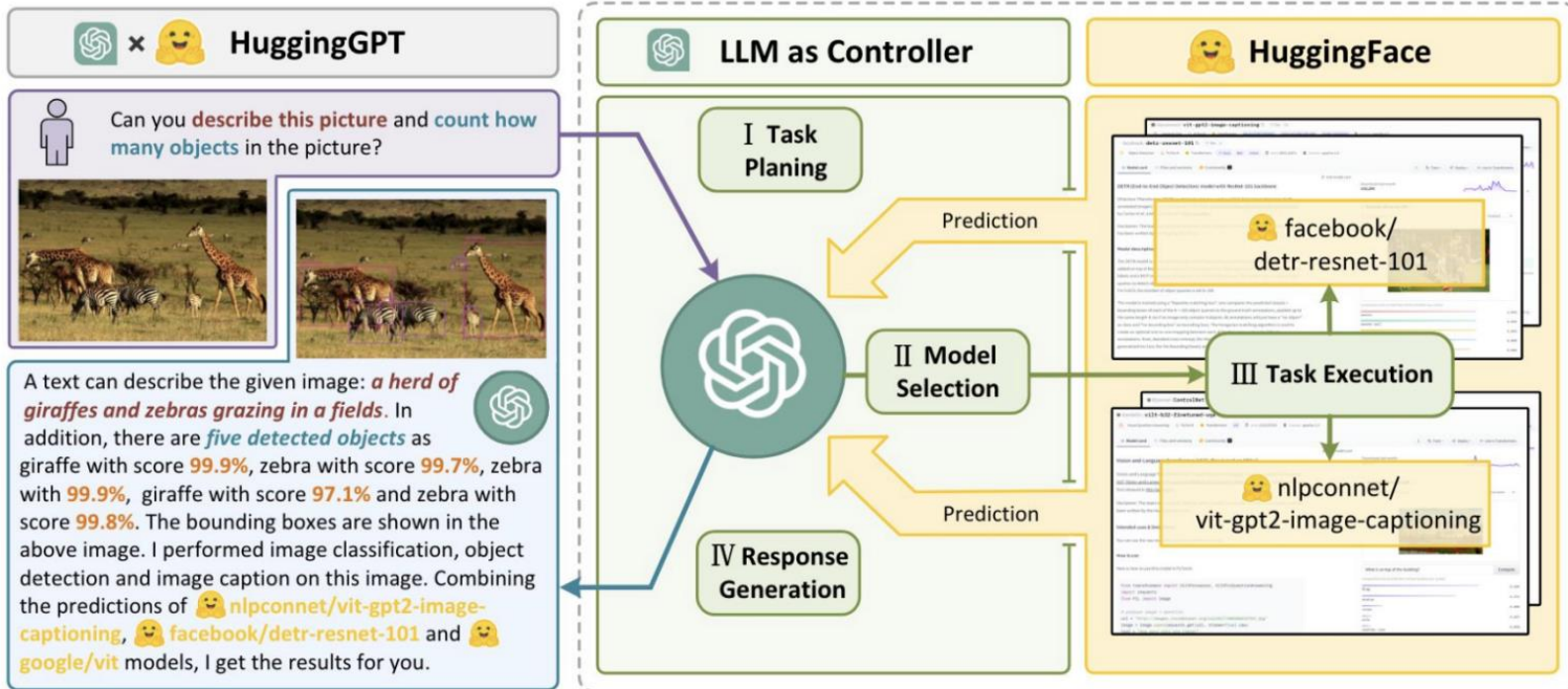
Act 11: Put pepper shaker 1 in/on drawer 1

Obs 11: You put pepper shaker 1 in/on the drawer 1.

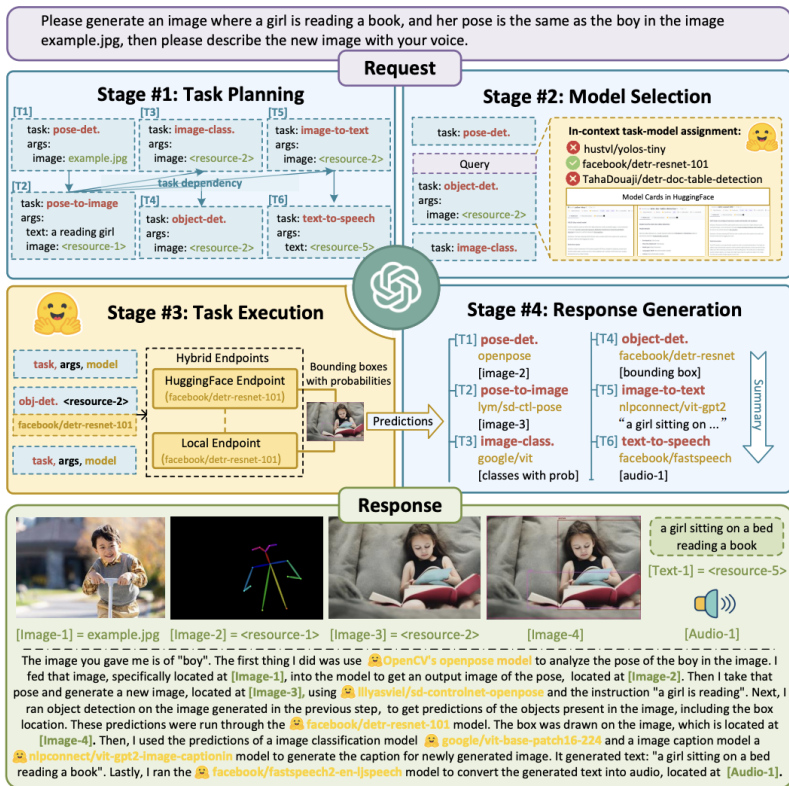


LLMs + APIs to expert models: HuggingGPT

Lots of AI models are available in different fields and modalities, but cannot handle complex artificial intelligence tasks.



LLMs + APIs to expert models: HuggingGPT



The system comprises of 4 stages:

- **Task Planning:** LLM analyze the user's requests, breaking them down into solvable tasks through prompts.
- **Model Selection:** LLM is presented with a list of models to choose from and distributes the tasks to expert models. LLM.
- **Task Execution:** Expert models execute on the specific tasks and log results.
- **Response Generation:** LLM receives the execution results and provides summarized results to users.

LLMs + APIs to expert models: HuggingGPT

Evaluation for task planning abilities:

- Single Task: The user request involves only one task.
- Sequential Task: The user's request needs to be broken down into a sequence of multiple subtasks.
- Graph Task: The user's request needs to be decomposed into a directed acyclic graph.



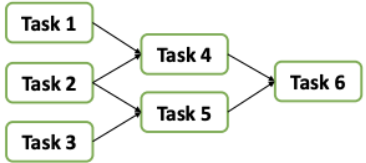
Task Type	Diagram	Example	Metrics
Single Task		Show me a funny image of a cat	Precision, Recall, F1, Accuracy
Sequential Task		Replace the cat with a dog in example.jpg	Precision, Recall, F1 Edit Distance
Graph Task		Given a collection of image A: a.jpg, B: b.jpg, C: c.jpg, please tell me which image is more like image B in terms of semantic, A or C?	Precision, Recall, F1 GPT-4 Score

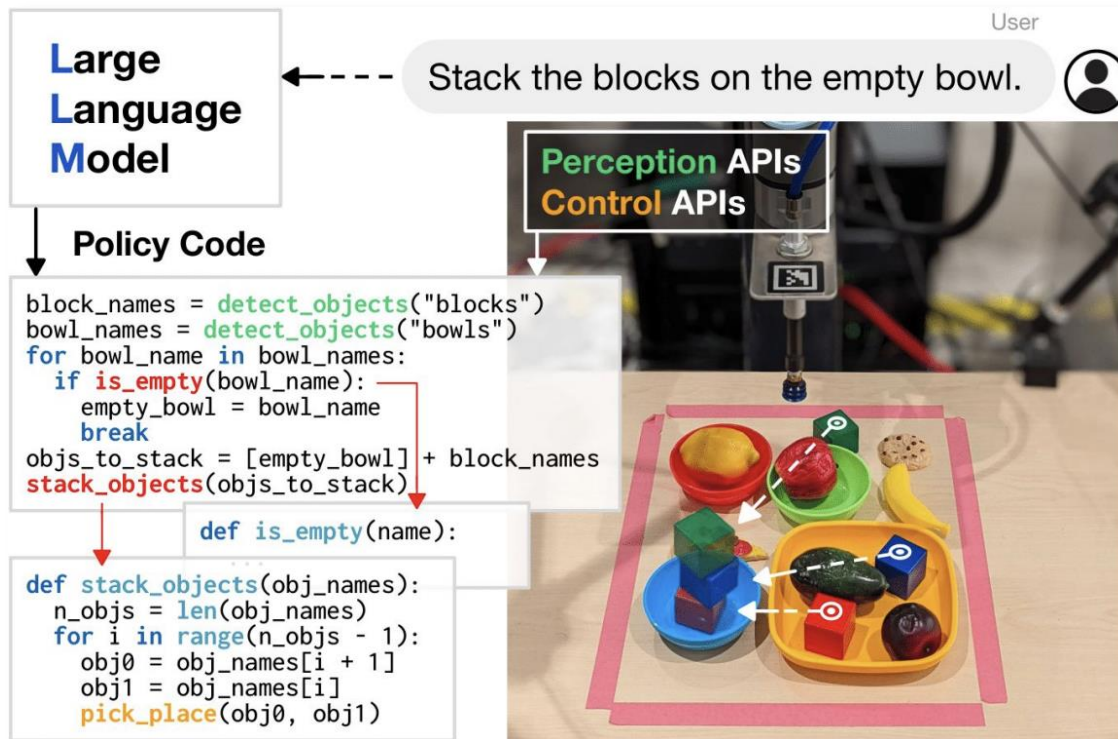
Table 2: Evaluation for task planning in different task types.

LLMs + APIs to expert models: **HuggingGPT**

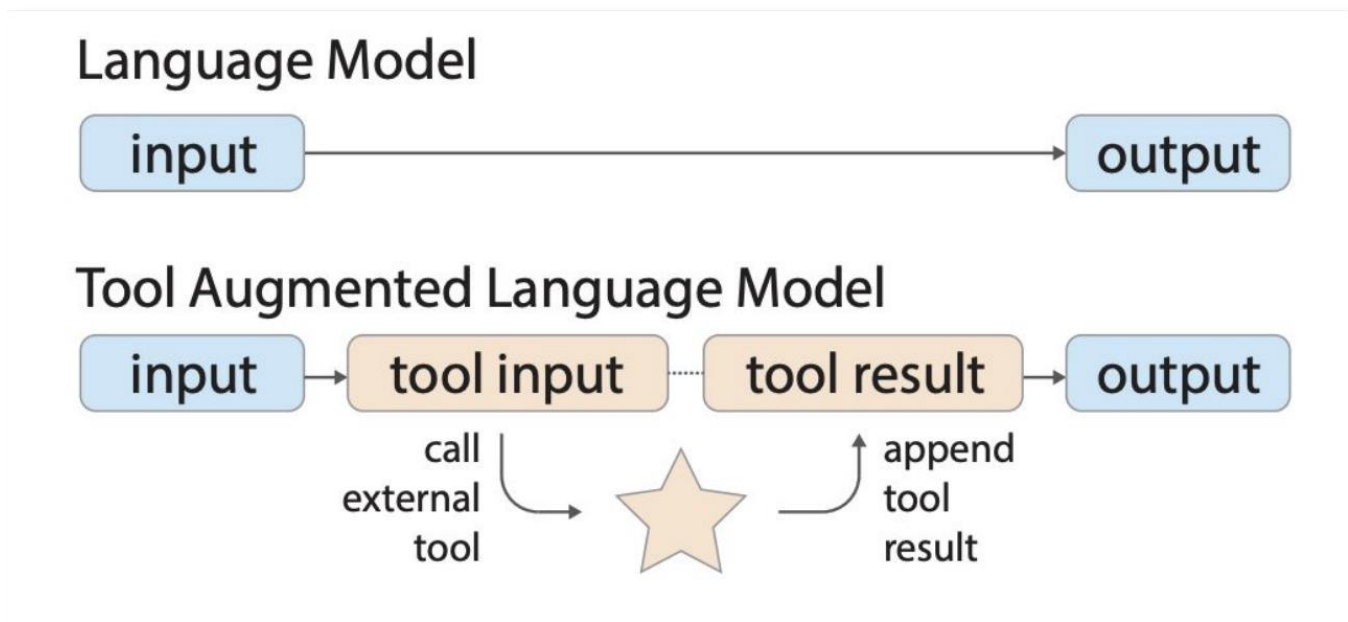
Challenges to put HuggingGPT into real world usage

- 1) Efficiency improvement is needed as both LLM inference rounds and interactions with other models slow down the process;
- 2) It relies on a long context window to communicate over complicated task content;
- 3) Stability improvement of LLM outputs and external model services.

LLMs + code, robotic arm, expert models: Code as Policies



LLMs + training for tool use: TALM



TALM: Tool Augmented Language Models

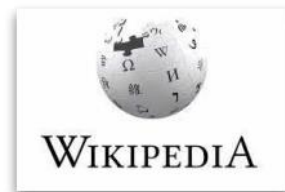
LLMs + training for tool use: Toolformer

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.

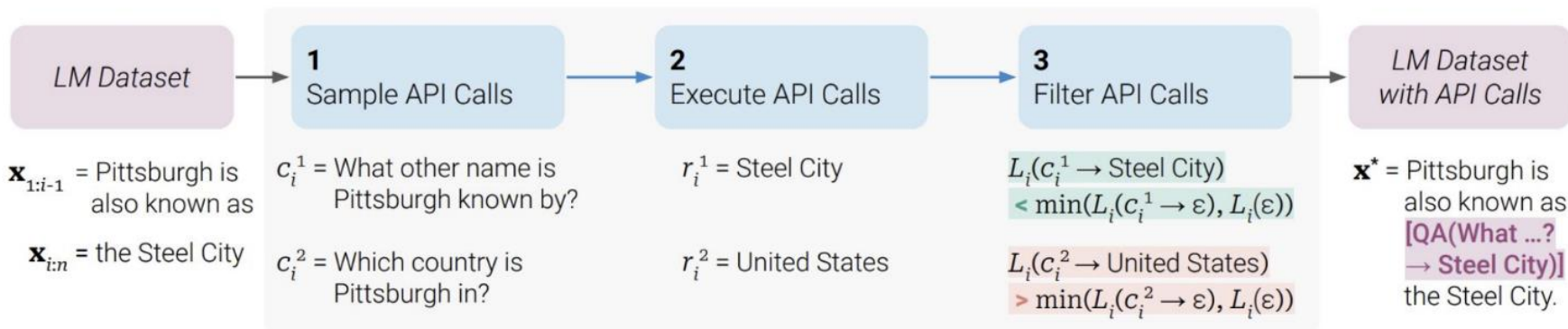
Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

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.



LLMs + training for tool use: Toolformer



Element 2: tools

Evaluation of tools in LLMs

Evaluation: **API-Bank**

API-Bank is a benchmark for evaluating the performance of tool-augmented LLMs. It contains 53 commonly used API tools, a complete tool-augmented LLM workflow, and 264 annotated dialogues that involve 568 API calls.

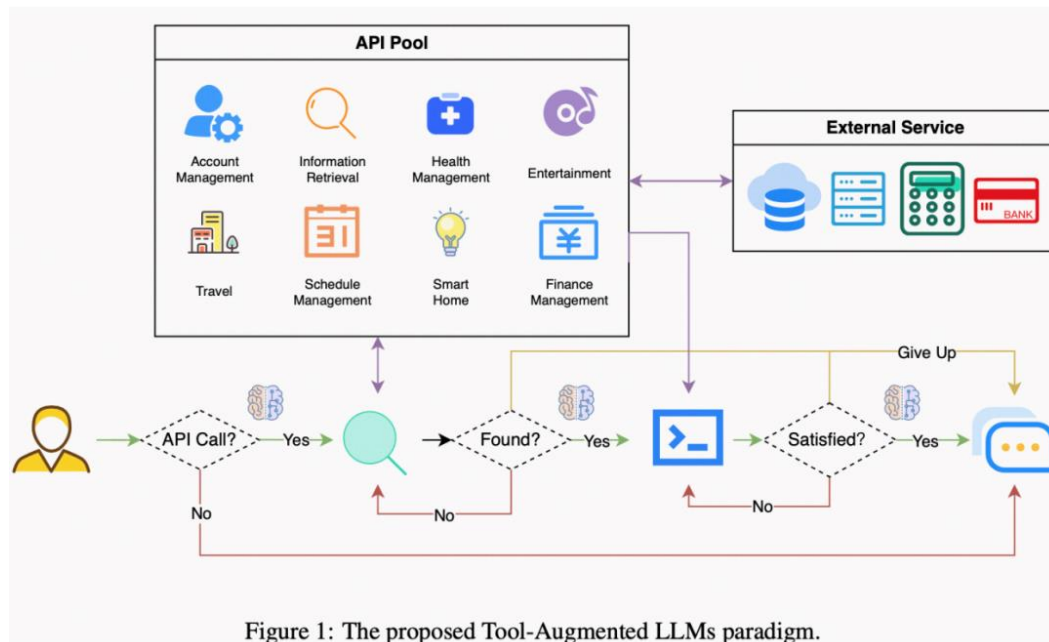


Figure 1: The proposed Tool-Augmented LLMs paradigm.

Evaluation: **API-Bank**

Evaluation index

Level-1: Evaluate LLM's ability to *call the API* (Accuracy); given a description of the API, the model needs to determine whether to call the API.

Level-2: Further evaluate LLM's ability to *retrieve APIs* (Rouge); the model needs to retrieve APIs that may solve user needs.

Level-3: Examine the ability of LLM *planning API* (number of turns).

	level-1	level-2	level-3
Num of Dialogues	214	50	8
Num of API calls	399	135	34

Table 1: The statistics of API-Bank.

Evaluation: GPT4Tools

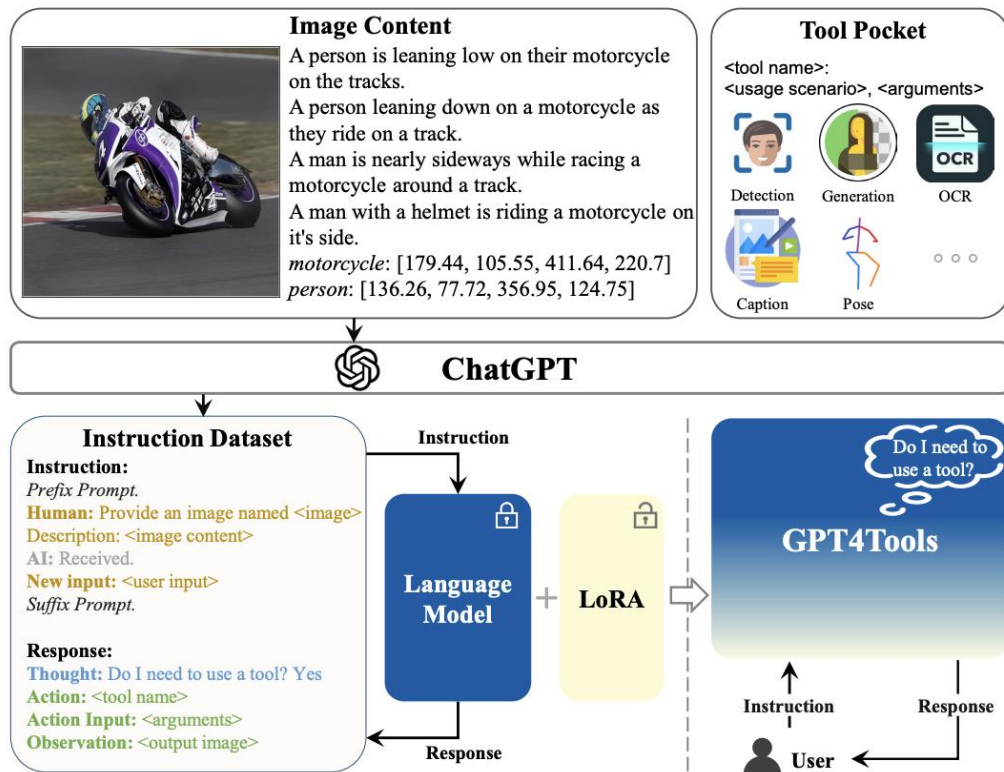


Figure 1: Diagram of the GPT4Tools. We prompt the ChatGPT with image content and definition of tools in order to obtain a tool-related instruction dataset. Subsequently, we employ LoRA [38] to train an open source LLM on the collected instruction dataset, thus adapting the LLM to use tools.

Evaluation: GPT4Tools

- **Successful Rate of Thought** measures whether the predicted decision matches the groundtruth decision.
- **Successful Rate of Action** measures whether the predicted tool name is in agreement with the name of the ground truth tool.
- **Successful Rate of Arguments** evaluates whether the predicted arguments match the ground-truth arguments.
- **Successful Rate** measures whether a chain of actions are executed successfully, which requires the correctness of thought, tool name, and tool arguments.

Challenges and future work

- **Complexity:** more complex domain professional/unseen tools?
- **Interactivity:** go beyond single turn?
- **Evaluation:** multiple possible solutions? Real-time interactive evaluation?
- **Efficiency:** smaller models?
- **Reliability:** know when to abstain, know its capacity, memorizing and querying tools?
- **Others**
 - Better tool API design/tool making?
 - Personalization?
 -

Element 3: Memory

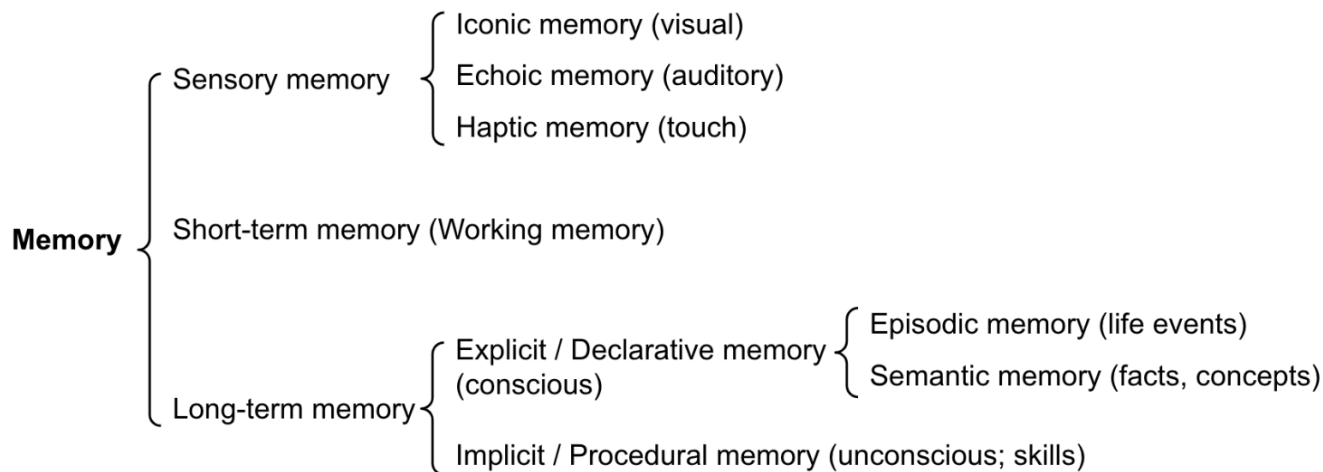
LLM Agent Memory: **Types of Memory in human brains**

- 1. Sensory Memory:** This is the earliest stage of memory, providing the ability to retain impressions of sensory information (visual, auditory, etc) after the original stimuli have ended. Sensory memory typically only lasts for up to a few seconds. Subcategories include iconic memory (visual), echoic memory (auditory), and haptic memory (touch).
- 2. Short-Term Memory (STM) or Working Memory:** It stores information that we are currently aware of and needed to carry out complex cognitive tasks such as learning and reasoning. Short-term memory is believed to have the capacity of about 7 items (Miller 1956) and lasts for 20-30 seconds.
- 3. Long-Term Memory (LTM):** Long-term memory can store information for a remarkably long time, ranging from a few days to decades, with an essentially unlimited storage capacity.

There are two subtypes of LTM:

- a. Explicit / declarative memory:** This is memory of facts and events, and refers to those memories that can be consciously recalled, including episodic memory (events and experiences) and semantic memory (facts and concepts).
- b. Implicit / procedural memory:** This type of memory is unconscious and involves skills and routines that are performed automatically, like riding a bike or typing on a keyboard.

LLM Agent Memory: **Types of Memory in LLMs**



1. **Sensory Memory:** learning embedding representations for raw inputs, including text, image or other modalities; [\[Vision encoder/speech encoder\]](#)
2. **Short-Term Memory (STM):** in-context learning. It is short and finite, as it is restricted by the finite context window length of Transformer. [\[prompt engineering\]](#)
3. **Long-Term Memory (LTM):** the external vector store that the agent can attend to at query time, accessible via fast retrieval. [\[Retrieval-augmented LMs \]](#)

Element 3: memory

Introduction to Retrieval-Augmented LMs (RAG)

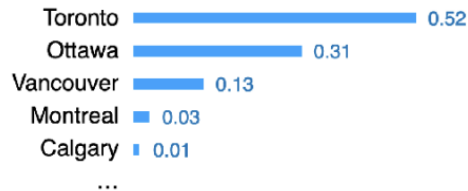
Retrieval-based language models (LMs)

Retrieval-based LMs = Retrieval + LMs

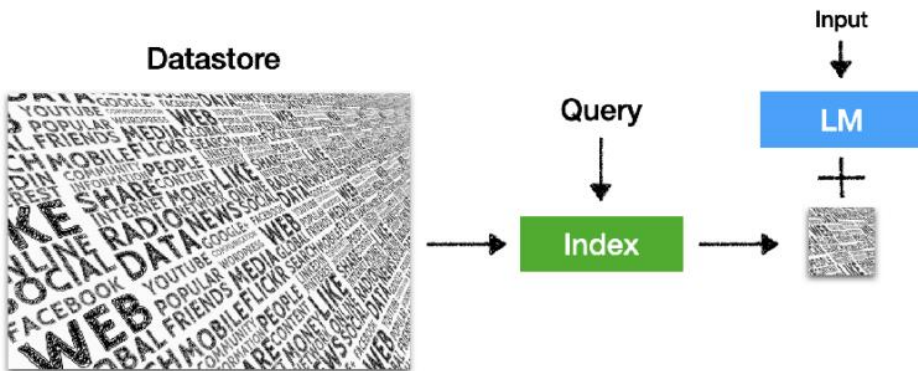
- It is a language model $P(x_n | x_1, x_2, \dots, x_{n-1})$

The capital city of Ontario is ____

(can be broadly extended to masked language models or encoder-decoder models)



- It retrieves from an external datastore (at least during inference time)



(Also referred to semiparametric and non-parametric models)

Why retrieval-based LMs?

LLMs can't memorize all (long-tail) knowledge in their parameters



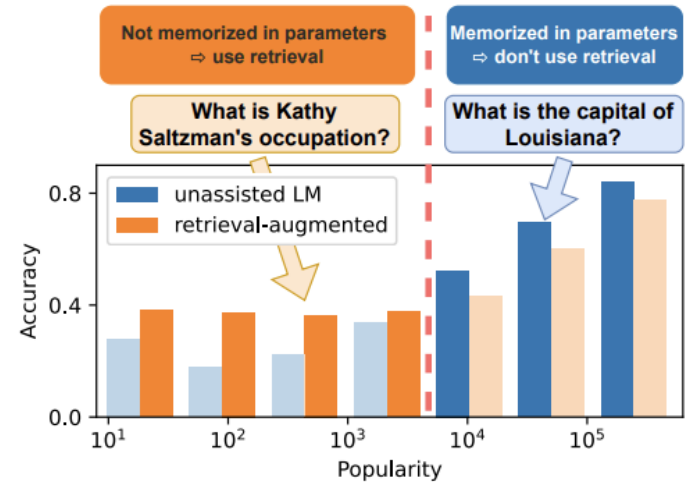
List 5 important papers authored by Geoffrey Hinton



Geoffrey Hinton is a renowned computer scientist ... Here are five important papers authored by him:

1. "**Learning Internal Representations by Error Propagation**" (with **D. E. Rumelhart** and **R. J. Williams**) - This paper, published in **1986**, ..
2. "**Deep Boltzmann Machines**" (with **R. Salakhutdinov**) - Published in **2009**, ..
- ...
4. "**Deep Learning**" (with Y. Bengio and A. Courville) - Published as a book in **2016**,...
5. "**Attention Is All You Need**" (with V. Vaswani, N. Shazeer, et al.) - Published in **2017**, this paper introduced the Transformer model,...

What is Kathy Saltzman's occupation?



GPT-3 davinci-003: 20%-30% accuracy

Why retrieval-based LMs?

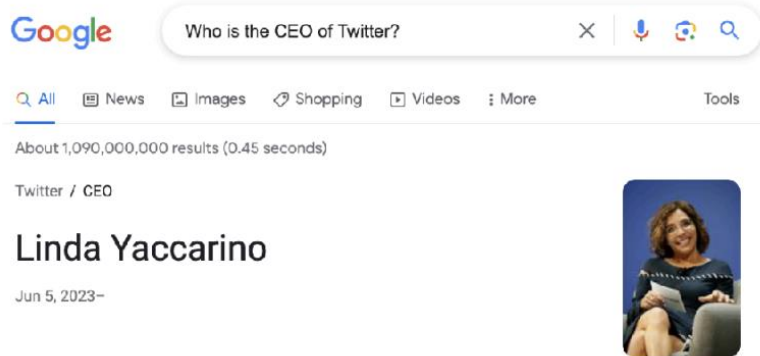
LLMs' knowledge is easily outdated and hard to update



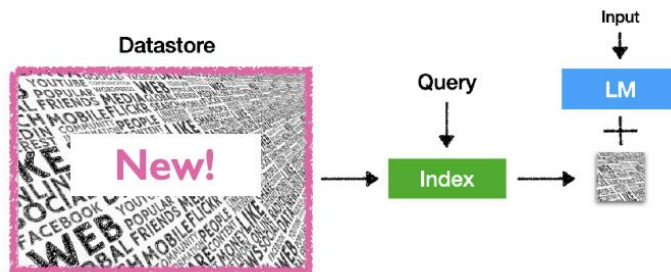
Who is the CEO of Twitter?



As of my **knowledge cutoff in September 2021**, the CEO of twitter is **Jack Dorsey**....

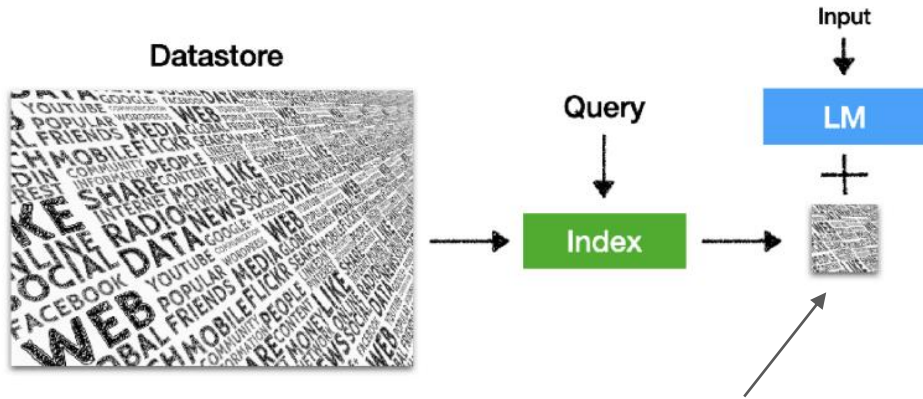


- Existing **knowledge editing** methods are still NOT scalable (**active research!**)
- The datastore can be easily **updated** and **expanded** - even without retraining!



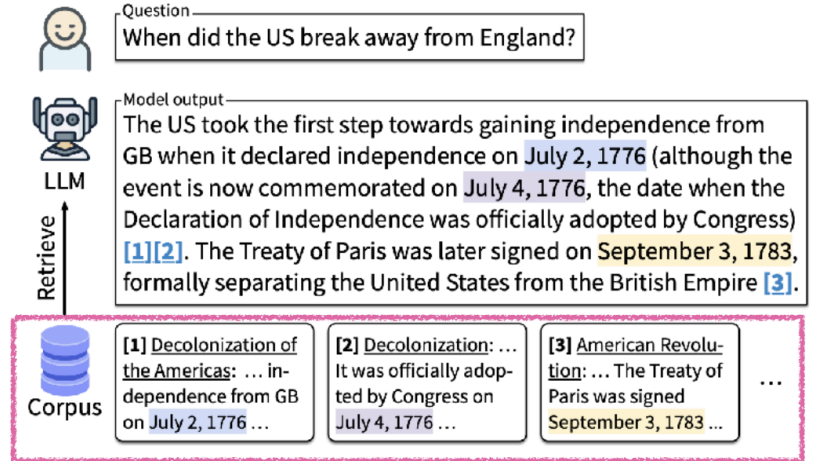
Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify



Can trace knowledge source from retrieval results - better interpretability & control

Generating text with citations



Why retrieval-based LMs?

LLMs' output is challenging to interpret and verify

🔍 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 ¹. You can also experience the thrill of walking on the glass floor or the EdgeWalk, the world's highest hands-free walk ¹. 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 ². After lunch, take a stroll along **Queen West**, a trendy neighborhood with eclectic shops, galleries, cafes and street art ³. 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 ⁴. 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 ³ ⁵. 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 ³.

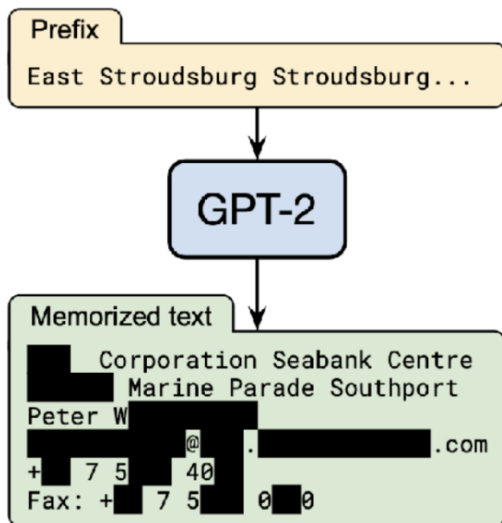
I hope this helps you plan your trip to Toronto. Have fun! 😊

Learn more:

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

Why retrieval-based LMs?

LLMs are shown to easily leak private training data



Category	Count
US and international news	109
Log files and error reports	79
License, terms of use, copyright notices	54
Lists of named items (games, countries, etc.)	54
Forum or Wiki entry	53
Valid URLs	50
Named individuals (non-news samples only)	46
Promotional content (products, subscriptions, etc.)	45
High entropy (UUIDs, base64 data)	35
Contact info (address, email, phone, twitter, etc.)	32
Code	31
Configuration files	30
Religious texts	25
Pseudonyms	15
Donald Trump tweets and quotes	12
Web forms (menu items, instructions, etc.)	11
Tech news	11
Lists of numbers (dates, sequences, etc.)	10

Individualization on private data by storing it in the datastore

Why retrieval-based LMs?

LLMs are large and expensive to train and run



vs.



Long-term goal: can we possibly reduce the **training** and **inference costs**, and scale down the size of LLMs?

e.g., RETRO ([Borgeaud et al., 2021](#)): “obtains comparable performance to GPT-3 on the Pile, despite using **25x fewer parameters**”

Definition of Retrieval-based LM

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

Typical LMs



The capital city of Ontario is **Toronto**



LM

Training time

The capital city of Ontario is _____



LM

Test time

Retrieval-based LMs



The capital city of Ontario is **Toronto**



LM

Training time

The capital city of Ontario is _____



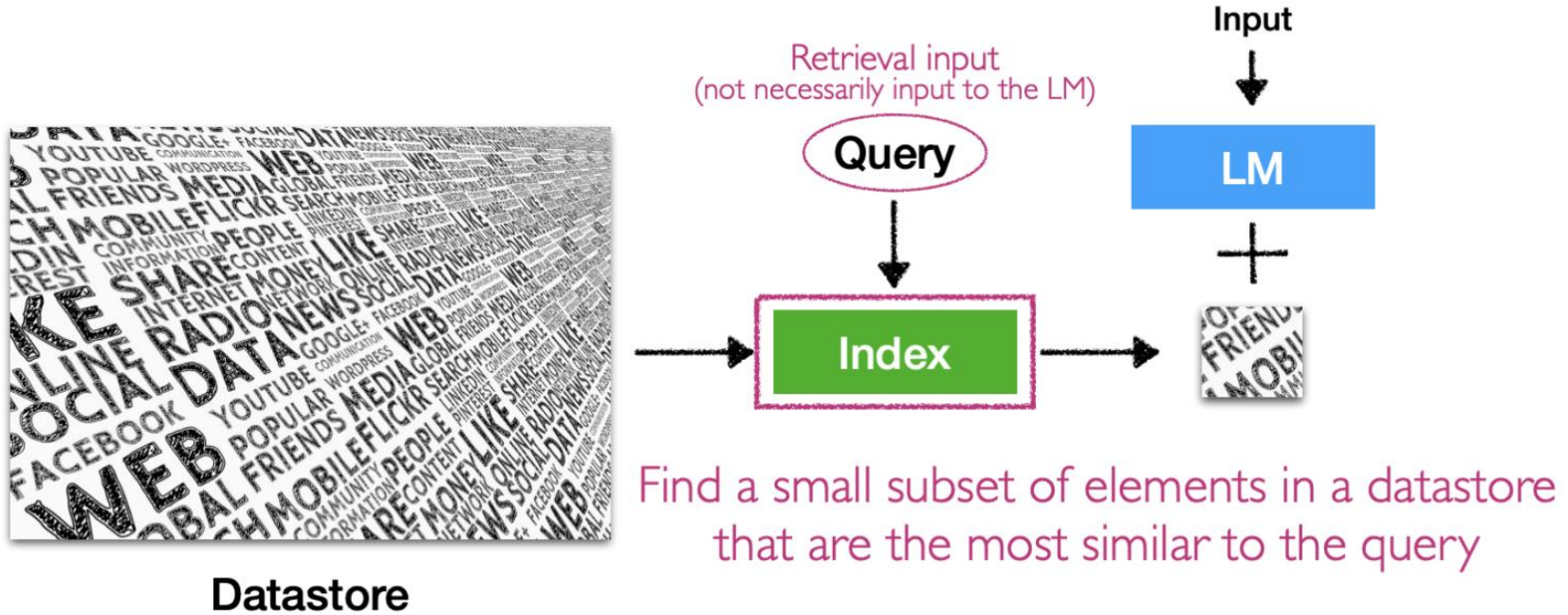
LM

Test time

Definition of Retrieval-based LM

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

Inference: Index



Definition of Retrieval-based LM

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

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Definition of Retrieval-based LM

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

Goal: find a small subset of elements in a datastore that are the most similar to the query

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

df_i # of docs containing i

Definition of Retrieval-based LM

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

Goal: find a small subset of elements in a datastore that are the most similar to the query

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

df_i : # of docs containing i

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

Maps the text into an h -dimensional vector

Definition of Retrieval-based LM

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

Goal: find a small subset of elements in a datastore that are the most similar to the query

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 df_i # of docs containing i

An entire field of study on how to get similarity function better

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

Maps the text into an h -dimensional vector

Definition of Retrieval-based LM

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

Goal: find a small subset of elements in a datastore
that are the most similar to the query

sim: a similarity score between two pieces of text

Index: given q , return $\arg\text{Top-}k_{d \in \mathcal{D}} \text{sim}(q, d)$ through fast nearest neighbor search

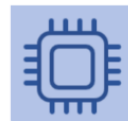
Software: FAISS, Distributed FAISS, ScaNN, etc...

Definition of Retrieval-based LM

Software: FAISS, Distributed FAISS, ScaNN, etc...

Method	Class name	index_factory	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	IndexFlatL2	"Flat"	d	4*d	yes	brute-force
Exact Search for Inner Product	IndexFlatIP	"Flat"	d	4*d	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	IndexHNSWFlat	"HNSW,Flat"	d, M	$4d + x * M * 2 * 4$	no	
Inverted file with exact post-verification	IndexIVFFlat	"IVFx,Flat"	quantizer, d, nlists, metric	4*d + 8	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	IndexLSH	-	d, nbits	ceil(nbits/8)	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	IndexScalarQuantizer	"SQ8"	d	d	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	IndexPQ	"PQx", "PQ"M"x"nbits	d, M, nbits	ceil(M * nbits / 8)	yes	
IVF and scalar quantizer	IndexIVFScalarQuantizer	"IVFx,SQ4" "IVFx,SQ8"	quantizer, d, nlists, qtype	SQp16: 2 * d + 8, SQ8: d + 8 or SQ4: d/2 + 8	no	Same as the IndexScalarQuantizer
IVFADC (coarse quantizer+PQ on residuals)	IndexIVFPQ	"IVFx,PQ"x"nbits	quantizer, d, nlists, M, nbits	ceil(M * nbits/8)+8	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	IndexIVFPQR	"IVFx,PQy+z"	quantizer, d, nlists, M, nbits, M_refine, nbits_refine	M+M_refine+8	no	

Exact Search



CPU vs. GPU

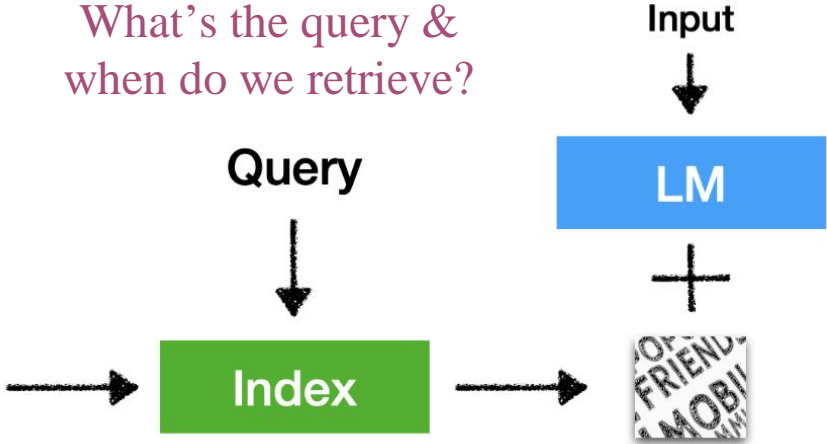
Approximate Search
(Relatively easy to scale to ~1B elements)

Questions to answer

What's the query & when do we retrieve?



Datastore

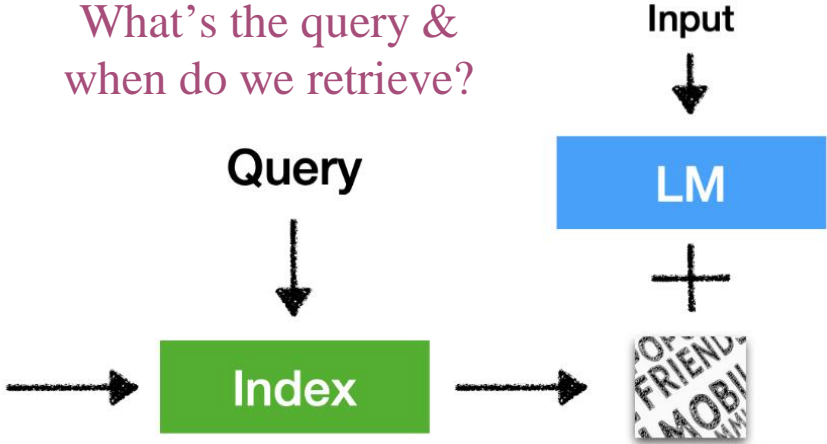


Questions to answer



Datastore

What's the query & when do we retrieve?



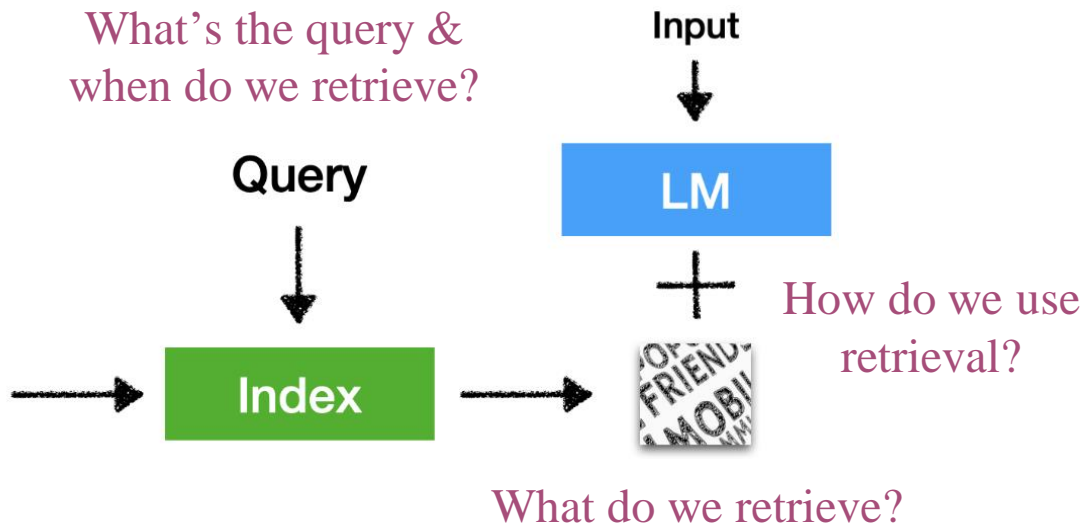
What do we retrieve?

Questions to answer



Datastore

What's the query & when do we retrieve?



Element 4: action

Action: Introduction

In the construction of the agent, the action module receives action sequences sent by the planning module and carries out actions to interact with the environment.

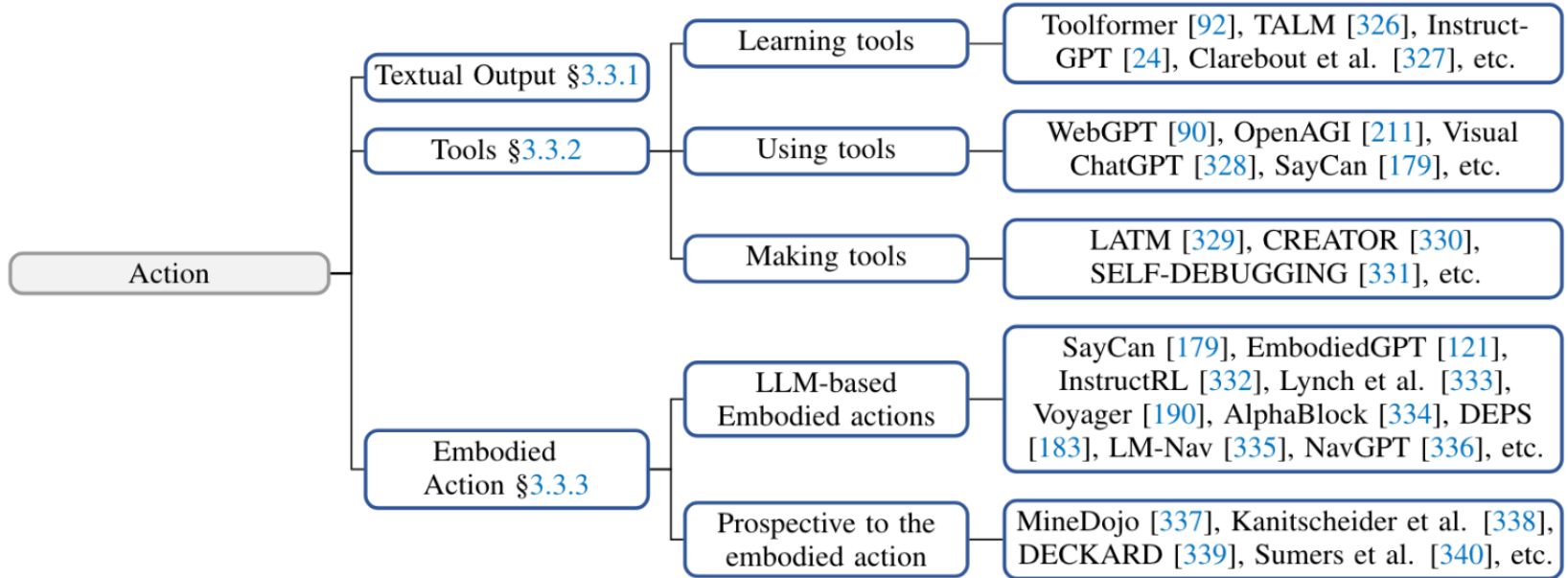
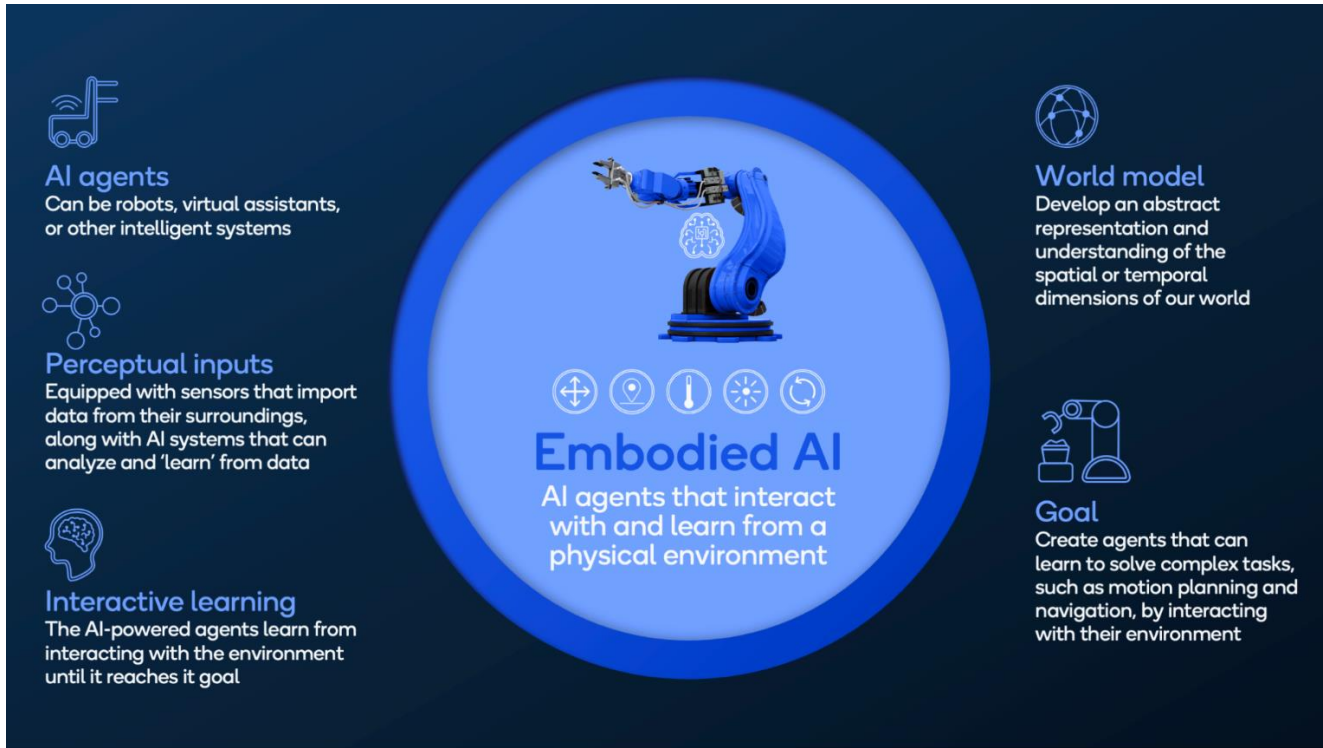


Figure 5: Typology of the action module.

Action: Embodied AI

In the pursuit of Artificial General Intelligence (AGI), the embodied agent is considered a pivotal paradigm while it strives to integrate model intelligence with the physical world.



Action: Embodied AI

Embodied AI should be capable of **actively perceiving**, **comprehending**, and **interacting** with physical environments, making decisions, and generating specific behaviors to modify the environment based on LLM's extensive internal knowledge. We collectively term these as ***embodied actions***, which enable agents' ability to interact with and comprehend the world in a manner closely resembling human behavior



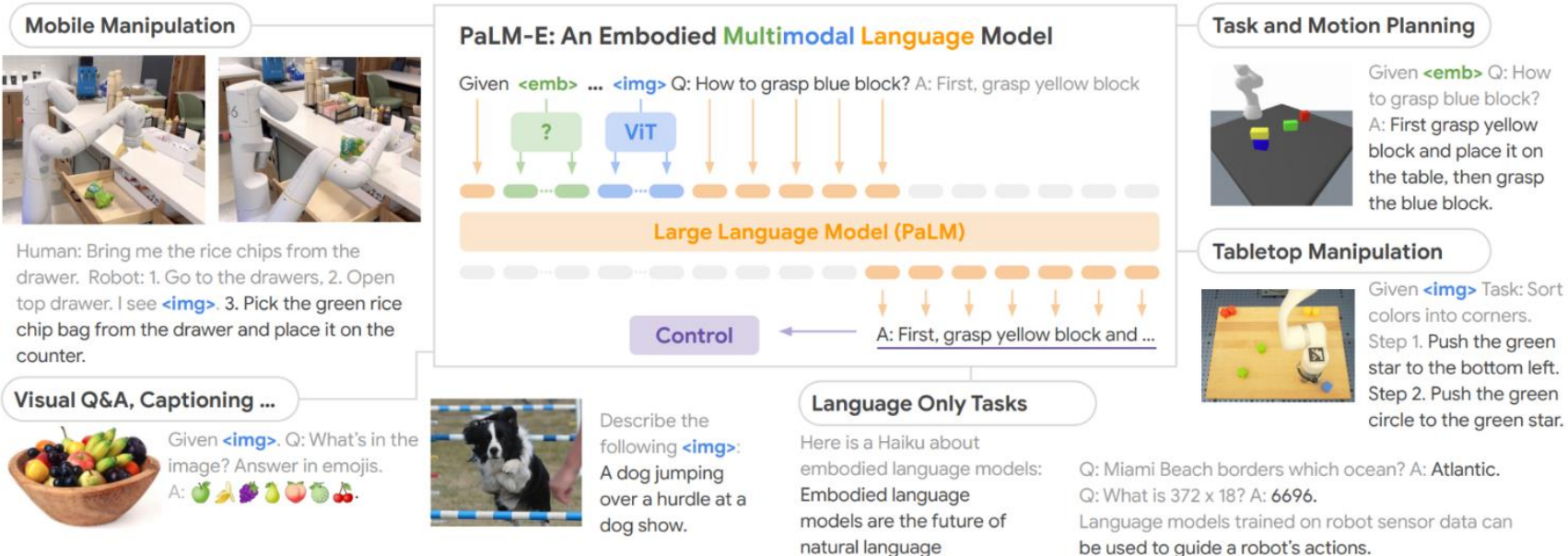
Action: Embodied AI

The potential of LLM-based agents for embodied actions.

- **Cost efficiency:** Some on-policy algorithms struggle with sample efficiency as they require fresh data for policy updates while gathering enough embodied data for high-performance training is costly and noisy.
- **Embodied action generalization:** An agent's competence should extend beyond specific tasks. When faced with intricate, uncharted real-world environments, it's imperative that the agent exhibits dynamic learning and generalization capabilities
- **Embodied action planning:** Planning constitutes a pivotal strategy employed by humans in response to complex problems as well as LLM-based agents.

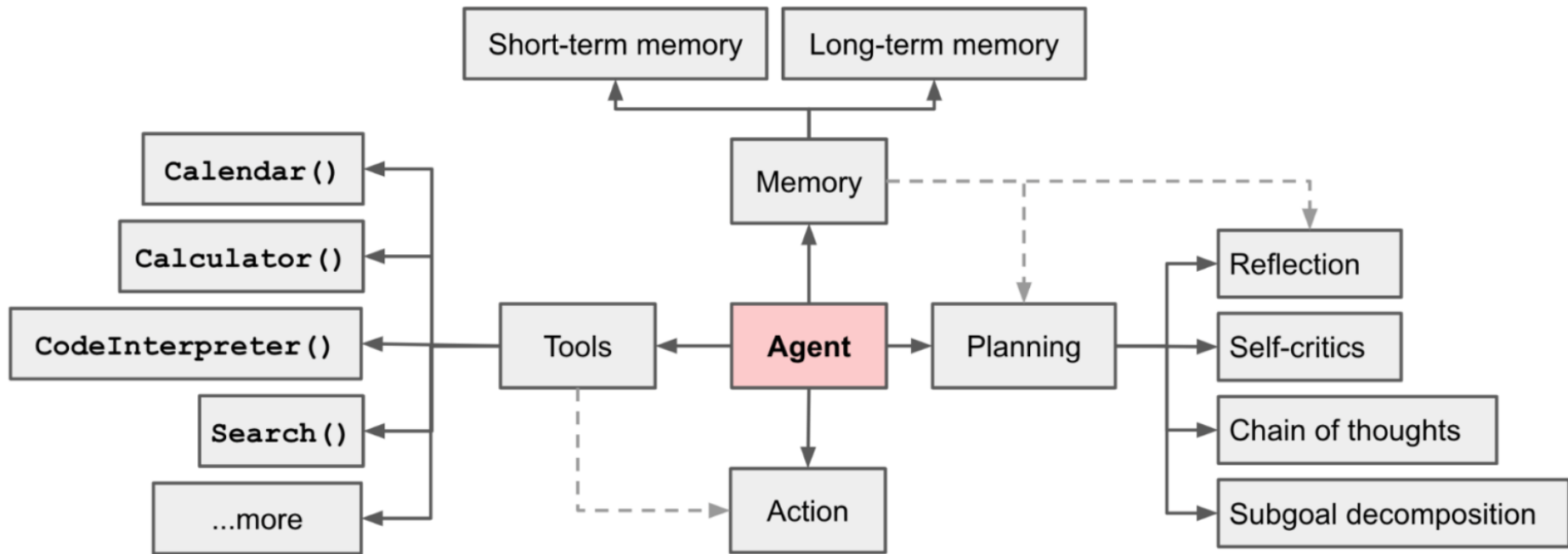
Embodied AI: PaLM-E: An Embodied Multimodal Language Model

PaLM-E transfers knowledge from visual-language domains into embodied reasoning – from robot planning in environments with complex dynamics and physical constraints, to answering questions about the observable world.

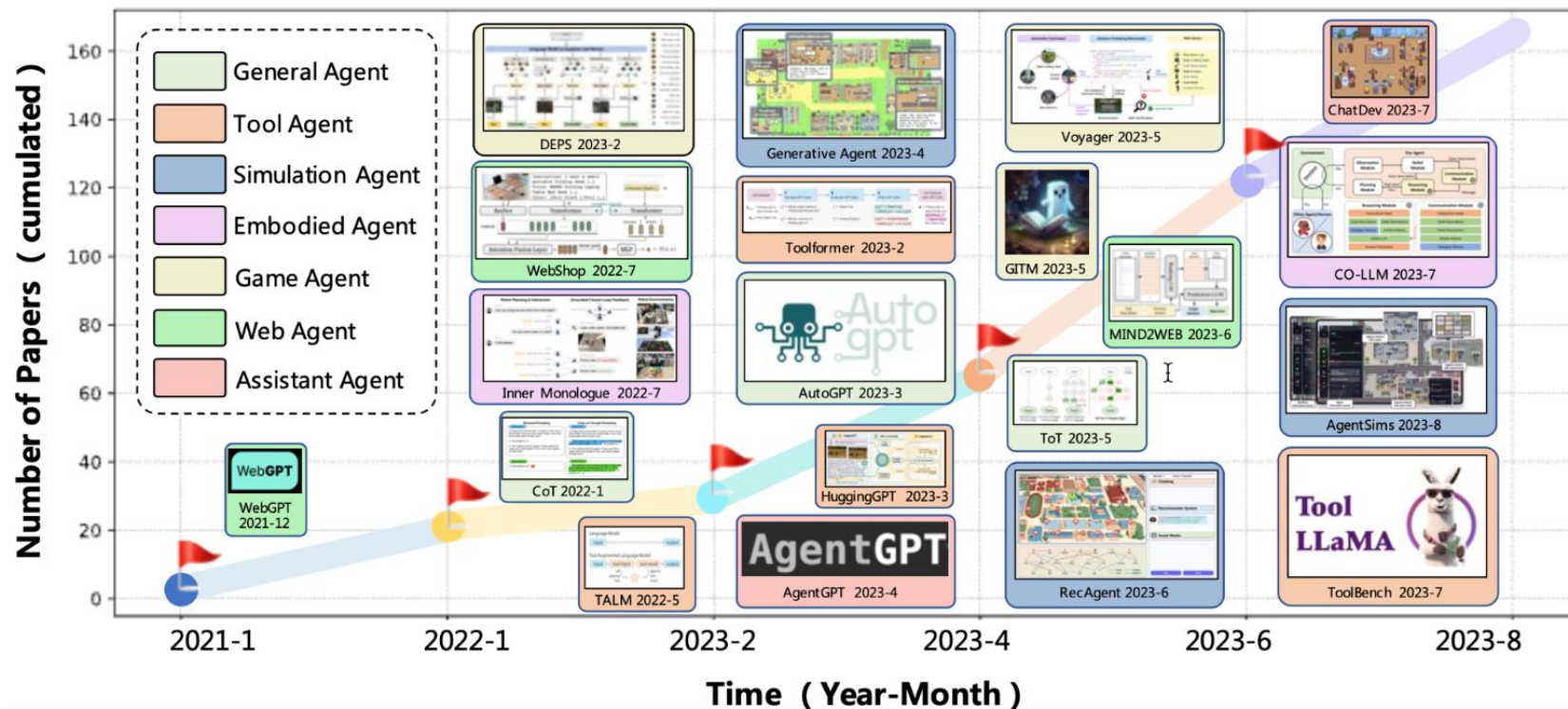


Recap of Agent

Recap1: four key components of LLM Agent



Recap2: the development of LLM agents



Recap3: challenges

After going through key ideas and demos of building LLM-centered agents, here are couple common limitations:

- 1. Finite context length:** The restricted context capacity limits the inclusion of historical information, detailed instructions, API call context, and responses. The design of the system has to work with this limited communication bandwidth, while mechanisms like self-reflection to learn from past mistakes would benefit a lot from long or infinite context windows. Although vector stores and retrieval can provide access to a larger knowledge pool, their representation power is not as powerful as full attention.
- 2. Challenges in long-term planning and task decomposition:** Planning over a lengthy history and effectively exploring the solution space remain challenging. LLMs struggle to adjust plans when faced with unexpected errors, making them less robust compared to humans who learn from trial and error.
- 3. Reliability of natural language interface:** Current agent system relies on natural language as an interface between LLMs and external components such as memory and tools. However, the reliability of model outputs is questionable, as LLMs may make formatting errors and occasionally exhibit rebellious behavior (e.g. refuse to follow an instruction). Consequently, much of the agent demo code focuses on parsing model output.

Extension: frameworks for LLM agent

- **Langchain:** This is a framework that allows you to build applications with LLMs through composability. You can use different agents for different data types, such as
- **AutoGen:** AutoGen is a framework that enables development of LLM applications using multiple agents that can converse with each other to solve tasks.
- **OpenAgents** is an open platform for using and hosting language agents in the wild of everyday life. Language agents are systems that can understand and communicate in natural language, such as chatbots, voice assistants, or conversational AI.
- **ChatDev** is a project that aims to create customized software using natural language idea (through LLM-powered multi-agent collaboration).

The image is a composite of three parts illustrating LLM agent frameworks:

- Left:** Microsoft AutoGen logo and statistics: 153 Contributors, 1 Used by, 56 Discussions, 17k Stars, 2k Forks. Text: "Enable Next-Gen Large Language Model Applications. Join our Discord: <https://discord.gg/pAbnFJrkGZ>"
- Middle:** Diagram of the AutoGen architecture. It shows a "User System" (a. Web UI for Agents) interacting with a "Language Agents" system (c. Language Agents). The Language Agents system includes a "Program" that uses "Data Agent", "Plugins Agent", and "Web Agent (GPT)", which in turn use "Retriever" and "Web Agent (GPT)". The system also includes "API Calling", "Action Parsing", and "Agent Methods". The architecture is split into "Frontend" (UX/UI) and "Backend" (Web Extension, Sandbox).
- Right:** A cartoon illustration of the ChatDev development process. It shows a character thinking "Develop a Gomoku game" and another character using "Code" and "Docs". The process involves "Designing", "Coding", and "Testing" (Test Line). The final output is "Software".

Proof of concepts

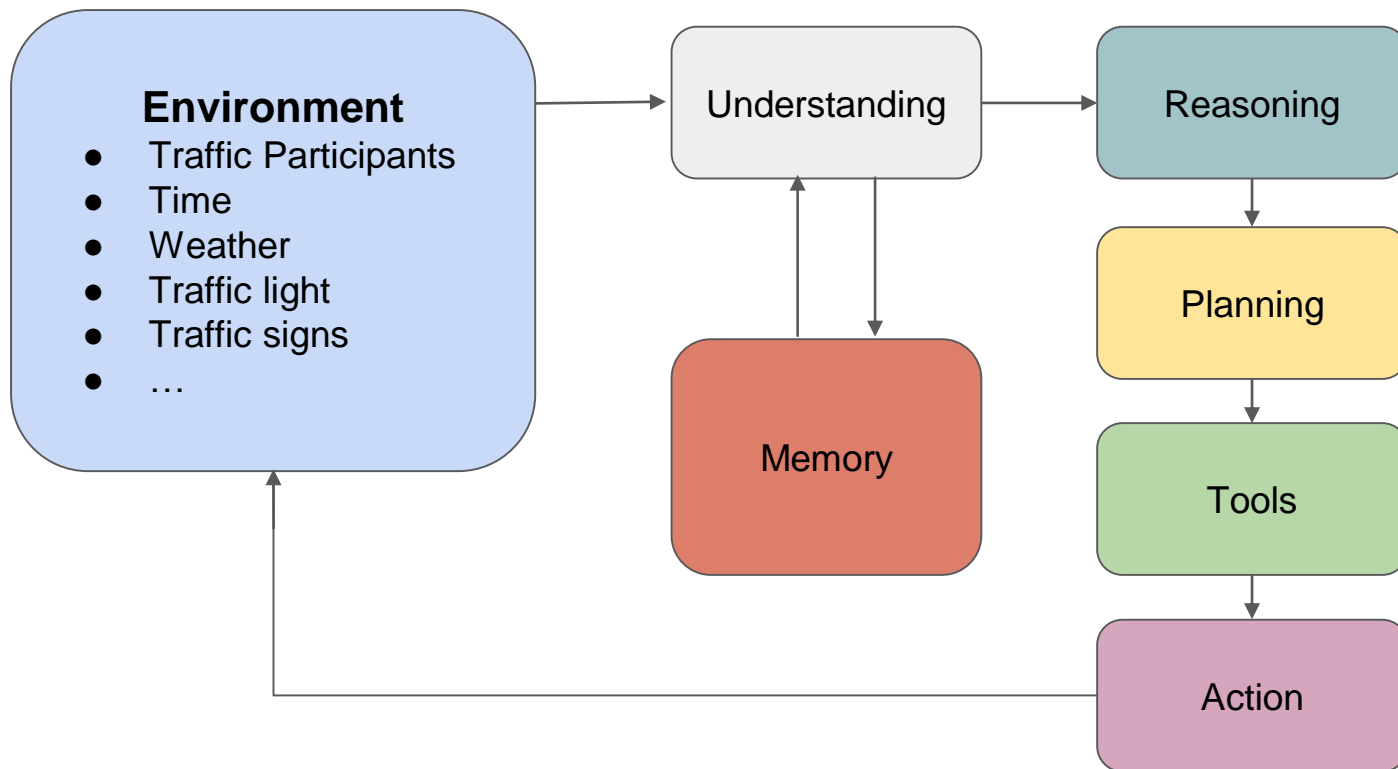
POC1: **Autonomous driving using GPT-4V**



Explorations of Visual-Language Model on Autonomous Driving

[On the Road with GPT-4V\(ision\): Early Explorations of Visual-Language Model on Autonomous Driving](#)

POC1: **Autonomous driving using GPT-4V**



POC1: Autonomous driving using GPT-4V

Understanding

Sec. 2.1 Time understanding throughout the day

Prompt:

This is the front-view image of your car. Please describe the current traffic scenario you're in.



GPT-4V:

You're on a city street with multiple lanes. There are parked cars on both sides of the road. Ahead, there's moderate traffic with vehicles moving in both directions. There's also a pedestrian crossing zone visible. Buildings and trees line the street, and there are various street signs along the way.

Prompt:

This is the front-view image of your car. Please describe the current traffic scenario you're in.



GPT-4V:

You are on a road during twilight or early evening. The road has a pedestrian crossing ahead. There's a vehicle with its tail lights on, indicating it's either stationary or moving away from you. There are traffic signs on both sides of the road, and the area appears to be moderately lit. Trees and other structures are visible in the background.

Figure 2: Results on the ability to comprehensively understand time over the course of a day. Green highlights the right answer in understanding. Check Section 2.1 for detailed discussions.

POC1: Autonomous driving using GPT-4V

Sec. 3.2 Multi-view images

Prompt:

You are a mature driver behind the wheel. These six images were surround view camera in your vehicle. The FOV of each surround view image is 180 degrees. The middle two images are the directly front.

Please describe in detail the scenario you are in, make a special distinction between you and what's on either side of you.



GPT-4V:

You're in a mostly empty urban area with roads bordered by structures and some greenery.

Directly in front: There's a relatively wide road with a few lane markings. Directly ahead, there's a large modern building with a unique architectural design featuring angled glass panels. To the left of the road, there are barriers indicating construction or some restricted area. There's a parked car on the right of these barriers. On the far left of the road, there's another street branching out.

To the left: The beginning of a pedestrian crosswalk is visible, leading towards a traffic island with some trees. On the far side, there's a street that turns to the left with another building in the background.

To the right: There's a continuation of the pedestrian crosswalk. Further right, a road intersects the one you're on, with a sign indicating a left turn. There are barriers similar to the ones directly in front of you. On the far right, you can see the corner of another building.

At the back: It seems like you've just passed a junction. There are more road barriers and markings. On the left side of the road, there's a structure that appears to be a bus stop or a shelter. Further back, there's another large building and a traffic light. On the right side, you can see a continuation of the road you're on, flanked by more buildings.

Prompt:

Please count and describe how many vehicles are in the current scene.

GPT-4V:

There are two vehicles in the current scene:

A red car parked on the right side, near the construction or restricted area barriers in front of you.

Another vehicle (due to the angle, specifics are hard to discern) is seen on the far right image, turning onto the road that intersects the one you're on.

Reasoning

Sec. 3.1 Corner cases

Prompt:

You're the mature driver behind the wheel and this image is what you see now. Describe what you see and what you plan to drive next.



GPT-4V:

As I observe the scene through the rain-speckled windshield, I see a blue truck to my left carrying a load under a white cloth. Directly ahead, there's an area cordoned off with traffic cones and barricades indicating some roadwork or maintenance activity. Near this section, there's a large orange industrial tank-like equipment with workers wearing safety gear actively engaged in their tasks. The road continues forward with buildings on both sides, and a few vehicles and pedestrians can be spotted in the distance. Overhead, trees line the road, providing a canopy.

Given the current conditions, I'd slow down due to the rain and potential slippery roads. I'd approach the work area with caution, ensuring I provide ample space for the workers. If the path is clear and there's no oncoming traffic, I'd move slightly to the left side to maintain a safe distance from the ongoing work. As I pass this area, I'd remain vigilant of the blue truck to my left, making sure it doesn't obstruct my path. After navigating this section, I'd continue driving, keeping in mind the importance of maintaining a safe speed given the rain and surrounding activities.

Prompt:

You're the mature driver behind the wheel and this image is what you see now. Describe what you see and what you plan to drive next



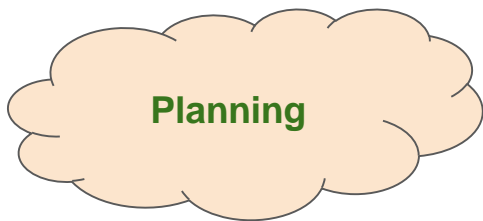
GPT-4V:

From behind the wheel, I see a large construction truck in the lane ahead of me. It's a prominent orange and yellow vehicle with noticeable safety markings, and it seems to be carrying construction equipment. Further ahead, there's a crossroads with traffic lights, and several vehicles are either stopped or in motion. On my right, there's a crane in the distance, indicating ongoing construction work in the area. In addition to cars, there are scooters and cyclists crossing the junction, suggesting that the area is bustling with mixed vehicular and pedestrian traffic.

With safety in mind, I plan to maintain a safe distance from the large construction vehicle and remain alert to the various road users around me. As the traffic light seems to be in favor of our direction, I'll smoothly accelerate once the truck ahead moves, ensuring I watch for any pedestrians or smaller vehicles that might dart out unexpectedly.

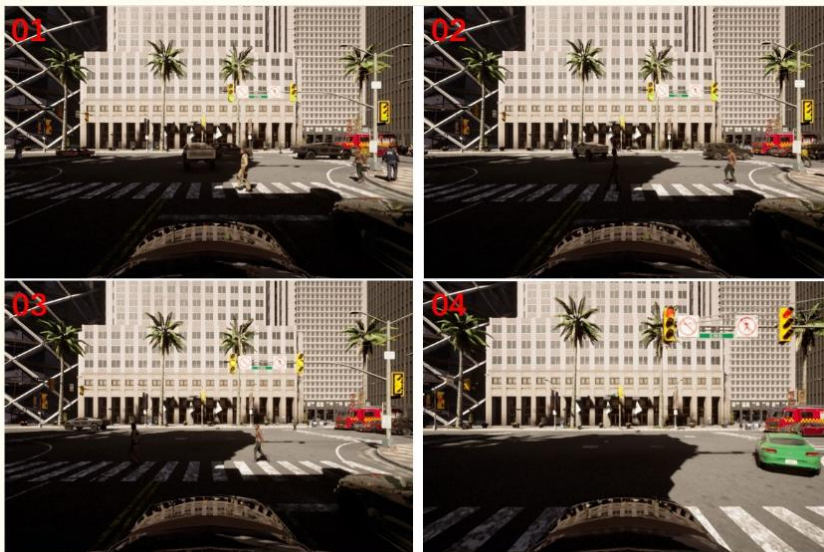
Figure 13: Illustration of GPT-4V's ability to make correct decisions in corner cases. Green highlights the right answer in understanding. Check Section 3.1 for detailed discussions.

POC1: Autonomous driving using GPT-4V



Prompt:

This is a time series of four pictures taken from a driving recorder. The chronological order is marked on the corresponding subfigure. As the driver of the car, please describe in detail what happened in these four seconds, what actions you performed, and why you performed the actions you did.



GPT-4V:

I approached an intersection with a green traffic light ahead. As I neared the crosswalk, I noticed pedestrians beginning to cross the street. I maintained my position, ensuring not to obstruct the crosswalk, and waited for the pedestrians to clear. The light turned yellow, then red, indicating for me to remain stopped. I obeyed the traffic signals and waited for the appropriate time to proceed.

Figure 19: Illustration of GPT-4V's ability to understand temporal sequential images. Green highlights the right answer in understanding. Check Section 3.3 for detailed discussions.

POC1: Autonomous driving using GPT-4V

Navigation Prompt:

You are on a busy road and you want to leave the main road and merge onto the ramp.



GPT-4V:

Observation: [...] The lane markings indicate that the current lane is ending or merging, as suggested by the dashed lines becoming solid and the arrow on the road pointing towards the merge. There is a sign indicating a hospital nearby, which may not directly affect driving decisions but suggests a possible increase in pedestrian and emergency vehicle activity.

Action: Reduce speed, prepare to merge right.

Tool -> Action

Sec. 4.3 Highway ramp turnaround

Navigation Prompt:

You are in a scenario where you come across a turnaround intersection and you need to turn around according to the lane line instructions. You are moving at a normal speed. Please answer me using the prescribed format.



GPT-4V:

Observation: [...] what appears to be a turnaround intersection ahead, indicated by arrow signs and a dividing lane line. There are red rear lights of another vehicle at the intersection, suggesting it is either stopped or moving slowly. The road seems to be wet [...]

Action: Reduce speed, turn left following the lane line instructions.



GPT-4V:

Observation: [...] with several vehicles ahead. Reflective lane markers outline a path that curves to the left. There's a green road sign that appears to indicate the direction for the turnaround, with a no-entry sign for the adjacent lane on the right [...]

Action: Apply the brakes lightly to decelerate in preparation for the left turn. Signal left to inform other drivers of the intention to enter the turnaround.[...]



GPT-4V:

Observation: [...] depicts a vehicle navigating a nighttime road environment, with the focus on a curved section lit by a single streetlight. The road is delineated with yellow lane dividers, and the surrounding area is poorly lit, limiting visibility.

Action: Gently decelerate if necessary to maintain a safe speed that matches the curvature of the road and the existing visibility conditions. Keep the vehicle within the lane, using the yellow lane dividers as a guide. [...]



GPT-4V:

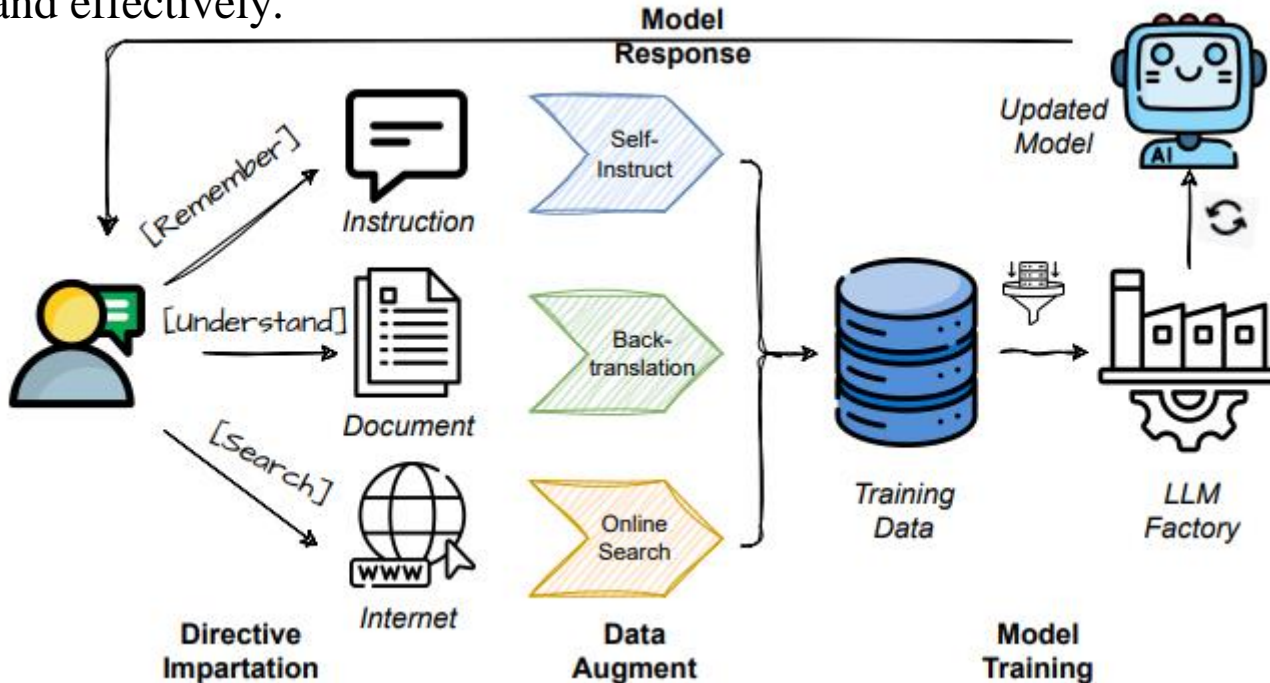
Observation: The image is taken from a vehicle's perspective during nighttime driving on a multi-lane road. There is a white car to the right that appears to be merging or taking an exit, and the glare of streetlights and other vehicle lights is noticeable. The surroundings are dimly lit, and the road seems to curve to the right ahead.

Action: [...] adjust the vehicle's speed or position on the road as necessary to accommodate the traffic flow. Use the vehicle's headlights and consider high beams if appropriate and legal [...].

Our research

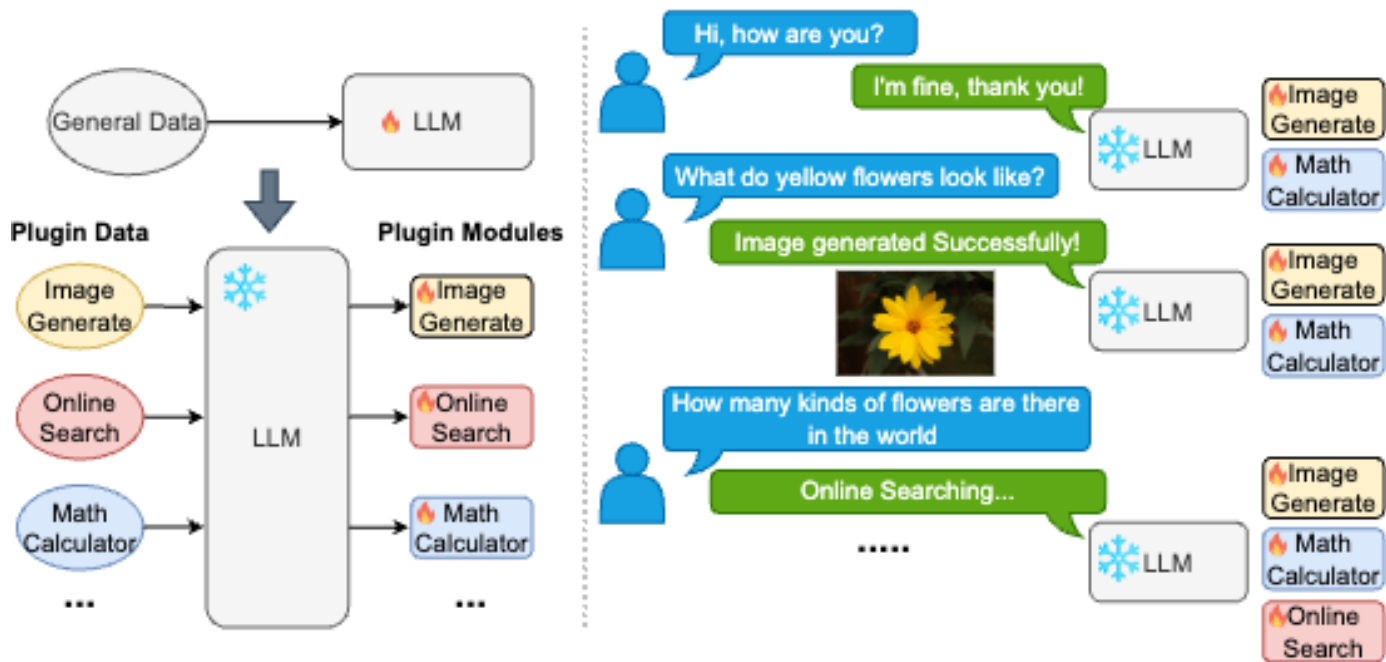
Our research: **Online Training of Large Language Models: Learn while Chatting** (Agent Memory Related)

Online-training can convert specific short-term memory into long-term memory efficiently and effectively.



Our research: **Modularization Makes Language Models Easier to Use Tools.** (Agent Tool Related)

Modularization is an effective way to help models quickly learn from multiple tools



Acknowledgements

- <https://github.com/Paitesanshi/LLM-Agent-Survey>
- <https://github.com/WooooDyy/LLM-Agent-Paper-List>
- [Generative Agents: Interactive Simulacra of Human Behavior.](#)
- <https://wenting-zhao.github.io/complex-reasoning-tutorial/>
- <https://acl2023-retrieval-lm.github.io/>
- <https://github.com/xlang-ai/llm-tool-use>