

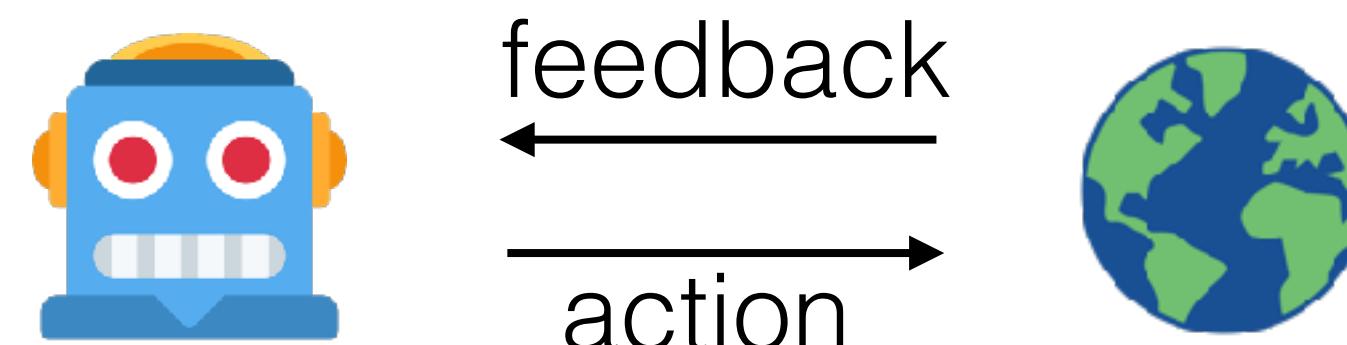
Language Agents

From next token prediction to digital automation



Shunyu Yao

Autonomous agents to interact with the world



Agent

Rule-based agents: manual design

Learning-based agents: trial-and-error

Language agents: reasoning to act

Environment

Interact with humans / physical world

Interact with games / simulation

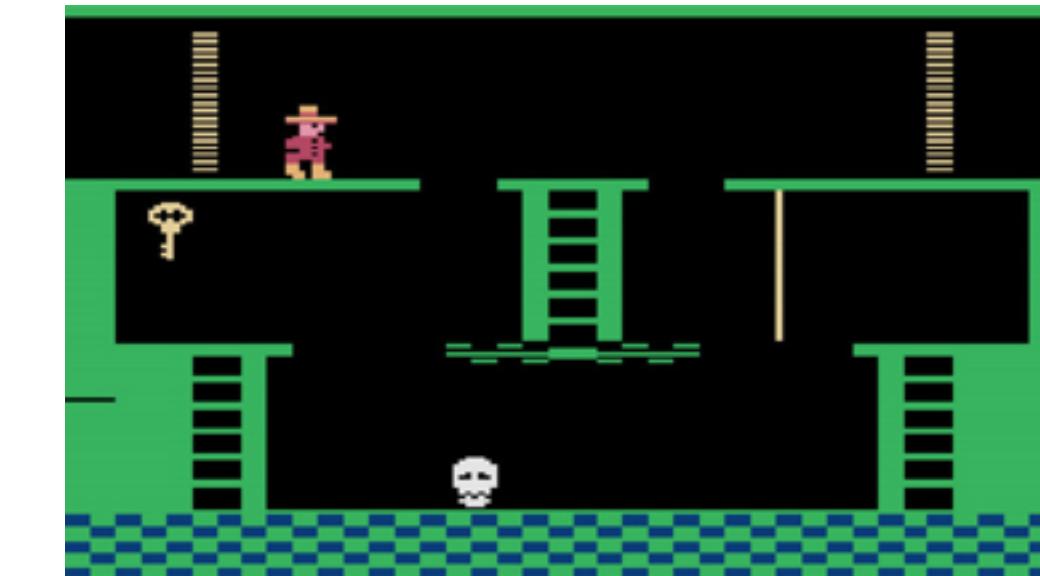
Interact with digital world (e.g., Internet)

Challenge 1: Accessible methods for general agents

Intensive to build
(Even for experts)

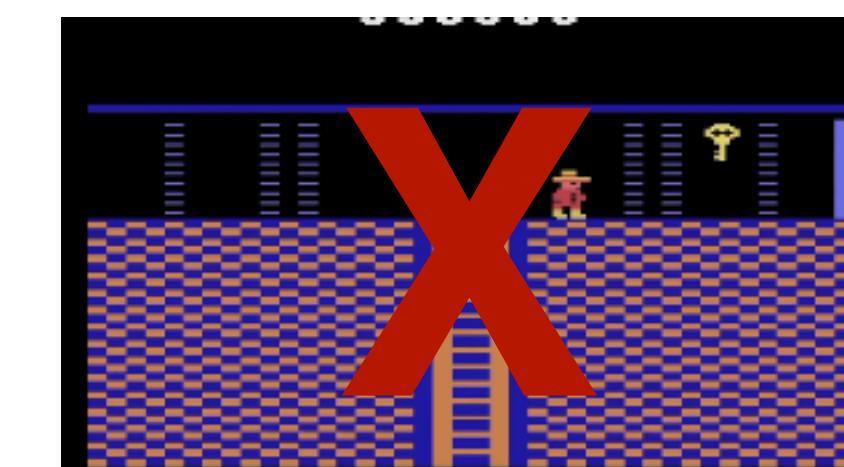


Takes millions of
lines of rules (by
domain experts)



Takes millions of
training iterations
(by RL experts)

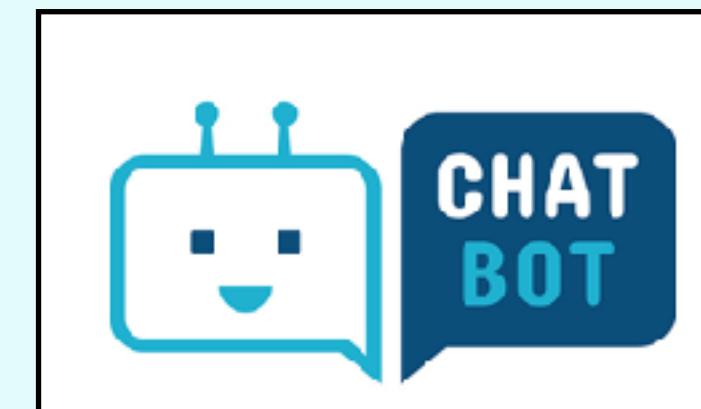
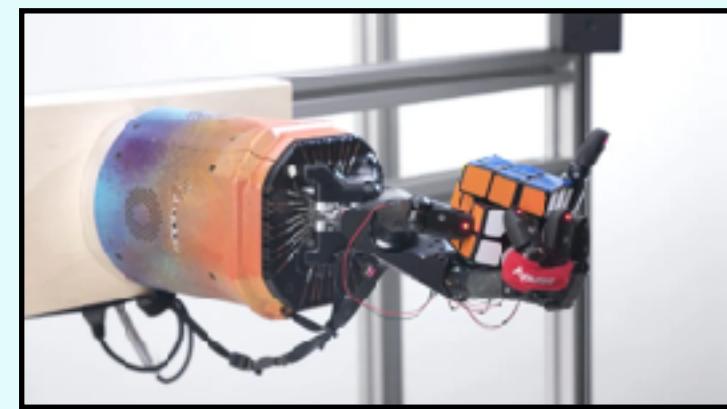
Hard to generalize



Challenge 2: Scalable benchmarks for practical tasks

Practical

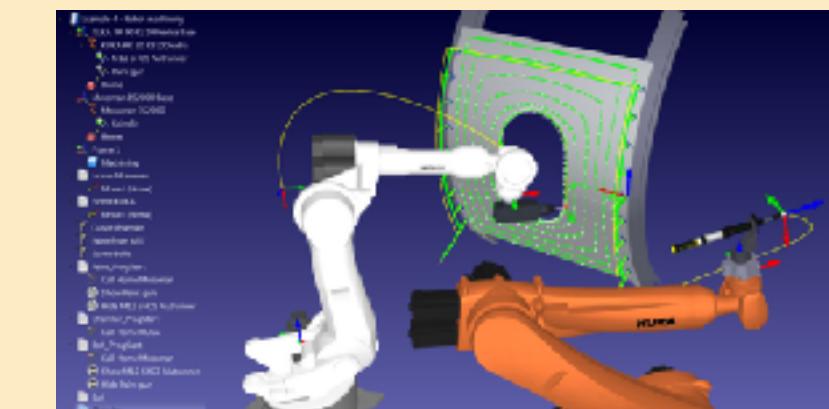
(Can build agents for useful tasks)



(But not scalable)

Scalable

(Easy data/reward collection)

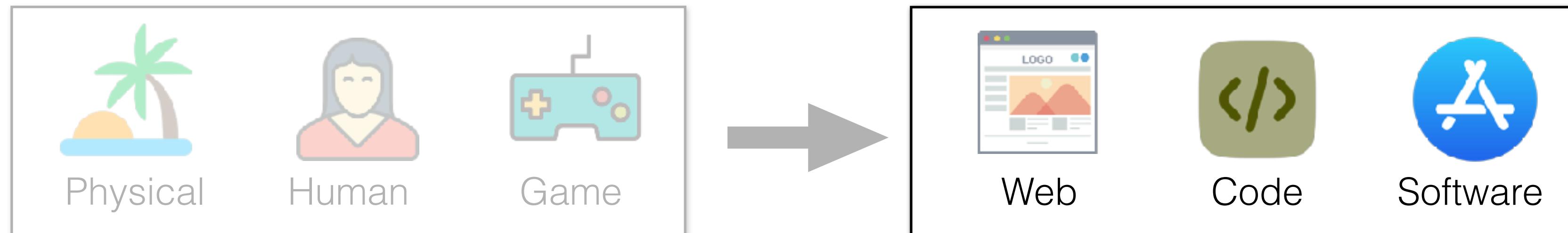


(But not practical)

My research

Part 1. Benchmarking agents via **digital automation**

[NeurIPS'22, NAACL'22, ACL'23, NeurIPS'23, ICLR'24, ICLR'24]



Practical

Scalable

Challenging

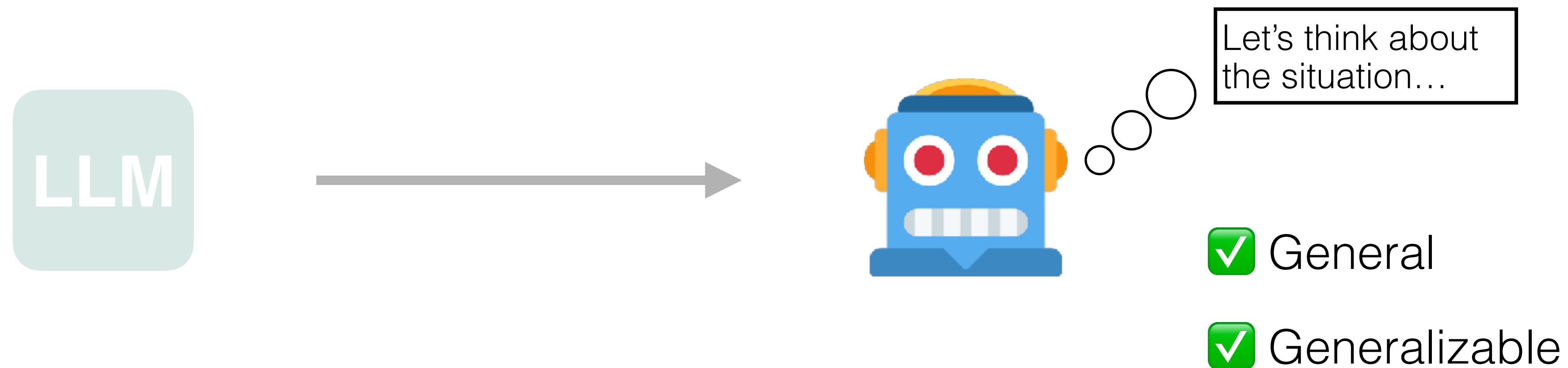
My research

Part 1. Benchmarking agents via **digital automation**

[NeurIPS'22, NAACL'22, ACL'23, NeurIPS'23, ICLR'24, ICLR'24]

Part 2. Building **language agents** that reason to act

[EMNLP'20, ICLR'23, NeurIPS'23, NeurIPS'23]



My research

Part 1. Benchmarking agents via **digital automation**

[NeurIPS'22, NAACL'22, ACL'23, NeurIPS'23, ICLR'24, ICLR'24]

Part 2. Building **language agents** that reason to act

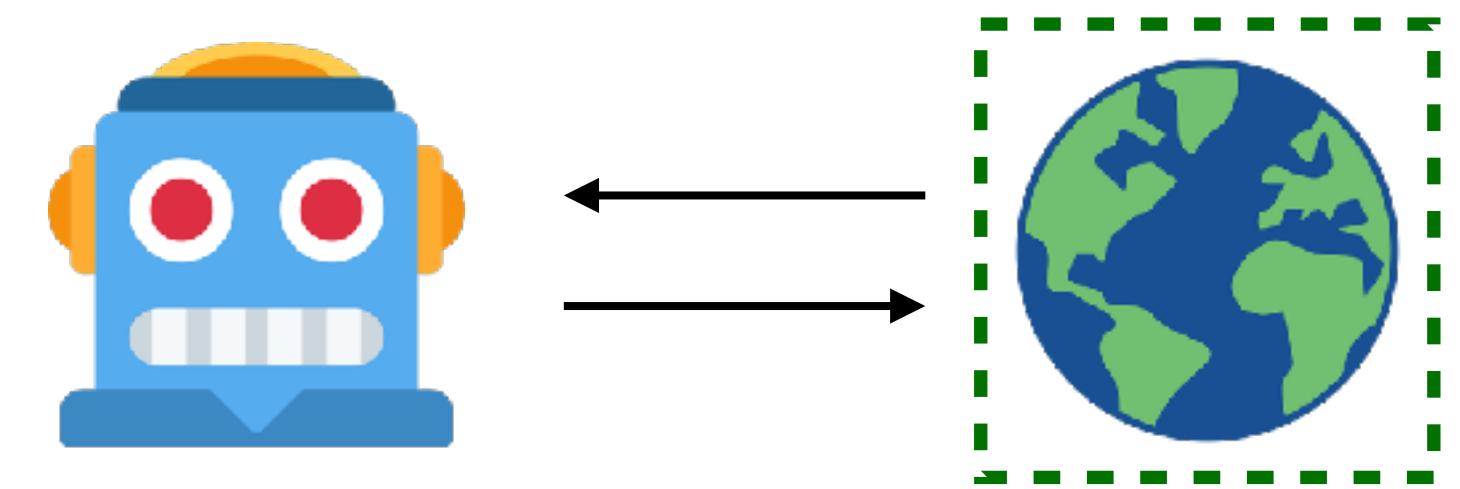
[EMNLP'20, ICLR'23, NeurIPS'23, NeurIPS'23]

Part 3. Principled **framework** for language agents

[TMLR'24]

1

Benchmarking agents via digital automation

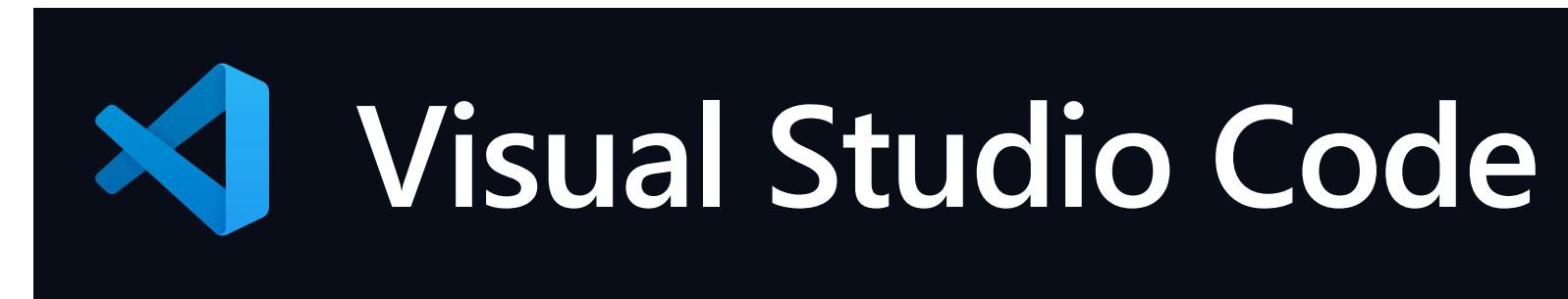


WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents
Yao*, Chen*, Yang, Narasimhan. NeurIPS 2022

Digital automation



File reports



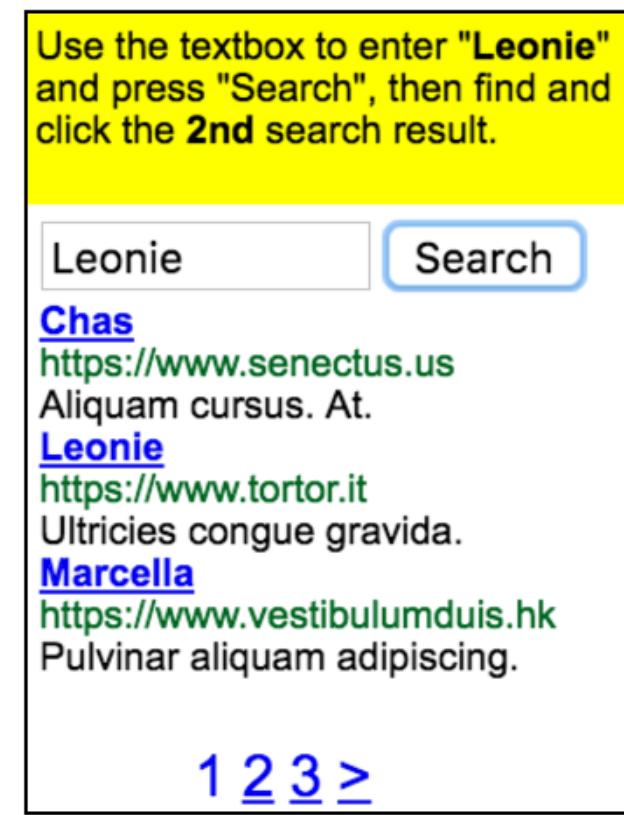
Code experiments



Explore papers

- Tremendous practical values, but little progress (think about Siri)
- Underlying research challenges:
 - Reasoning over **real-world language** (and other modalities)
 - Decision making over **open-ended actions** and **long horizon**
- Solving these is also key for robot navigation, planning, coordination

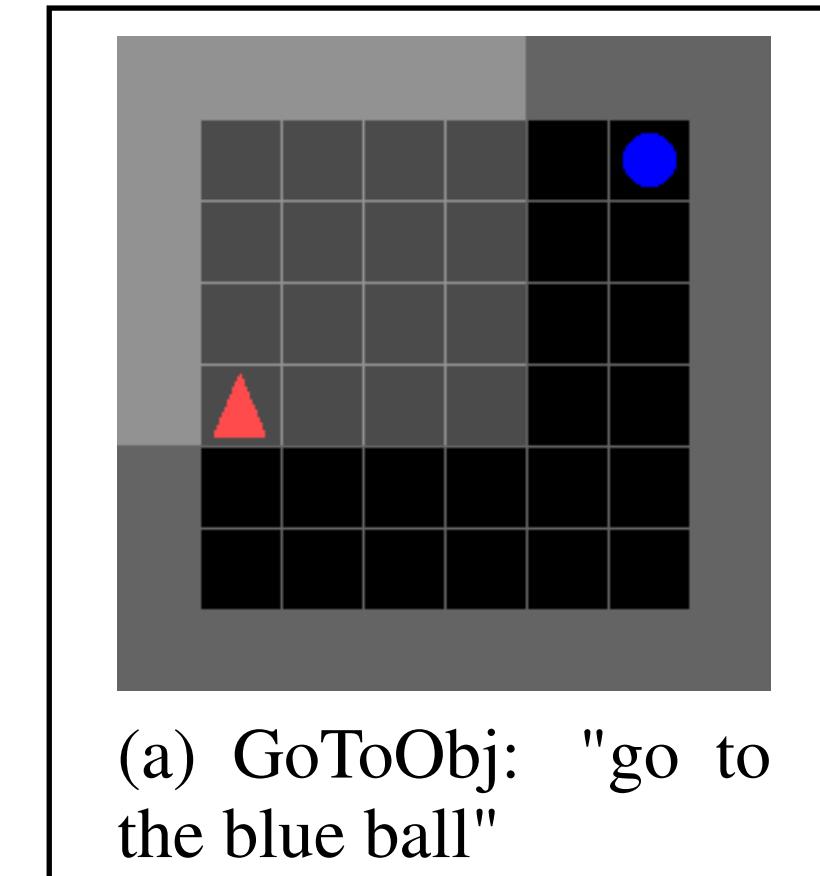
Agent benchmarks without these challenges



MiniWoB
(Shi et al., 2017)

```
> take sandwich  
Taken.  
  
> inventory  
You are carrying:  
a chilled sandwich  
a large stick of butter  
  
> eat it  
You eat the chilled sandwich.
```

TextWorld
(Côté et al., 2019)



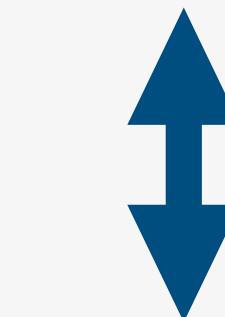
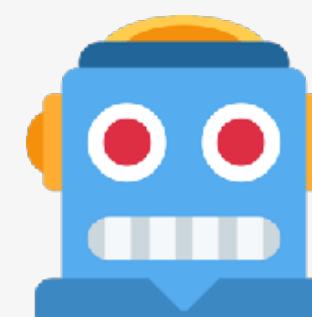
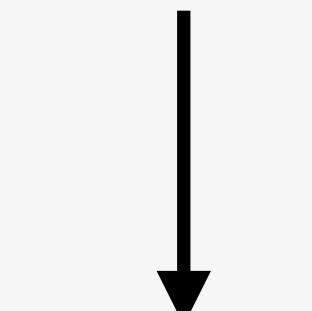
BabyAI
(Chevalier-Boisvert et al., 2019)

- Simulation environment
- Synthetic text (if any)

- Small action space
- Short-horizon tasks

Task input

How do neural networks work?



Web browser

Task output

In its most basic form, a neural network has two layers: an input layer and an output layer^[3]. The output layer is the component of the neural network that makes predictions^[3]. In a feedforward network, information flows through the network in the following way: patterns of information are fed into the network via the input units, which trigger the layers of hidden units, and these in turn arrive at the output units^[1]. The network learns by a feedback process called backpropagation, which involves comparing the output a network produces with the output it was meant to produce, and using the difference between them to modify the weights of the connections between the units in the network, working from the output units through the hidden units to the input units, going backward^{[2][4]}. Over time, backpropagation causes the network to learn, reducing the difference between actual and intended output to the point where the two exactly coincide, so the network figures things out exactly as it should^[2].

1. [How neural networks work - A simple introduction \(www.explainthatstuff.com\)](#) ↗
2. [How neural networks work - A simple introduction \(www.explainthatstuff.com\)](#) ↗
3. [How Do Neural Networks Really Work? | Nick McCullum \(nickmccullum.com\)](#) ↗
4. [How Do Neural Networks Really Work? | Nick McCullum \(nickmccullum.com\)](#) ↗



Reward via professional annotators

Desired benchmark

- Large complex environment
- Automatic reward function
- Research challenges

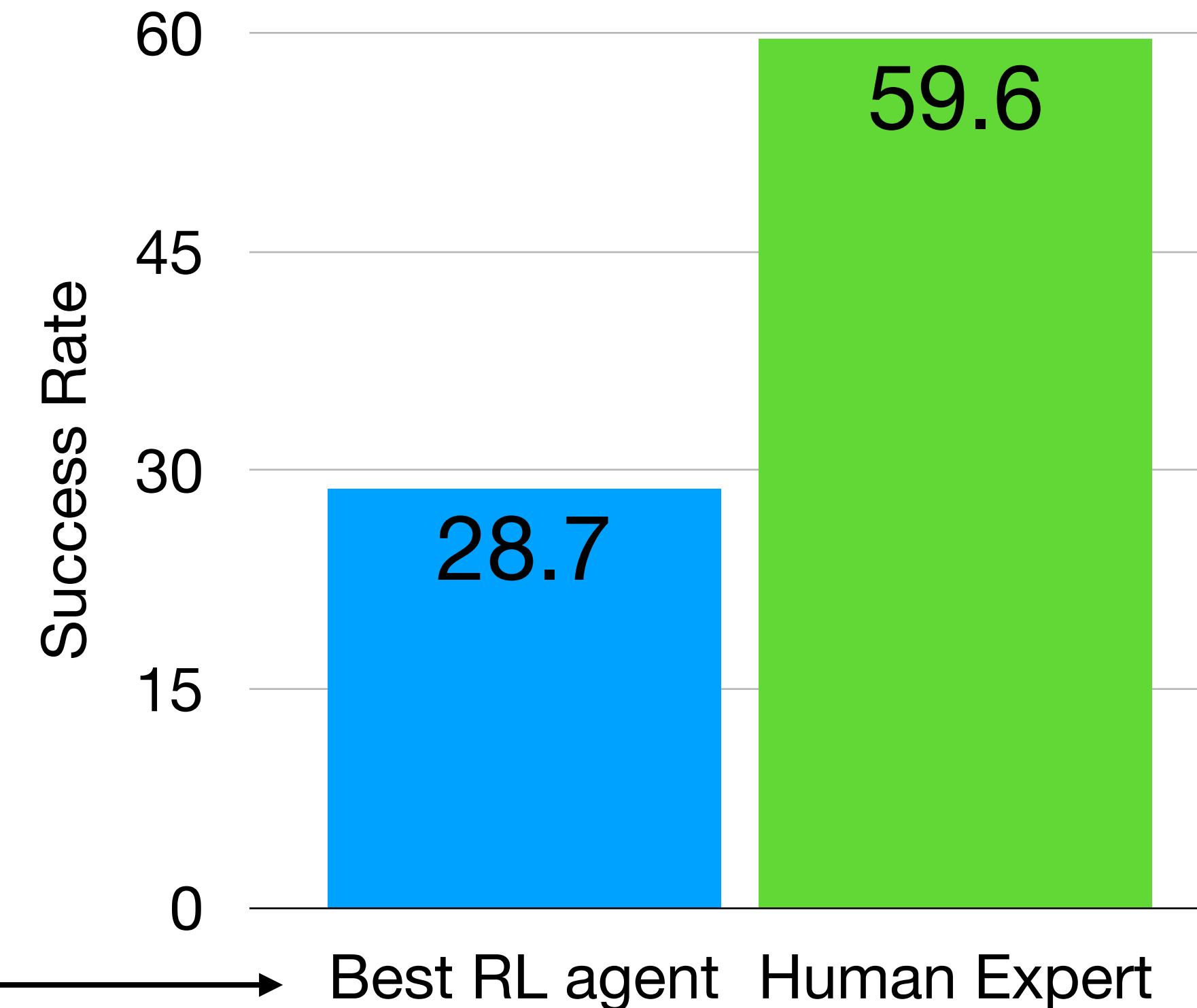


WebShop

- Large-scale complex environment based on 1.16M Amazon products
- Automatic reward based on instruction and product attribute matching
- Challenges language and visual understanding, and decision making

WebShop is challenging

- Pre-trained image model (ResNet)
- Pre-trained language models (BERT, BART)
- Imitation learning
- Reinforcement learning



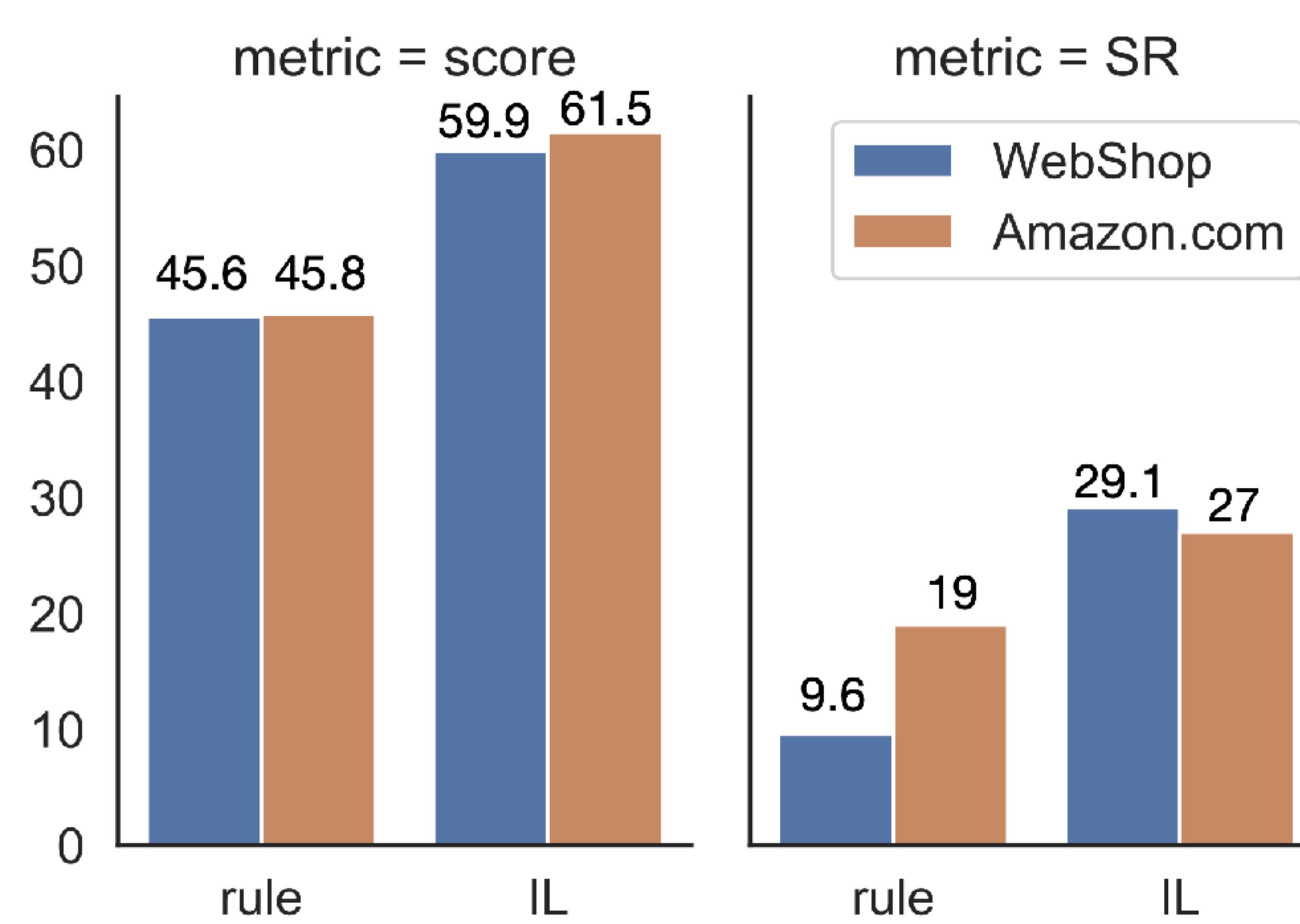
Trajectory length:

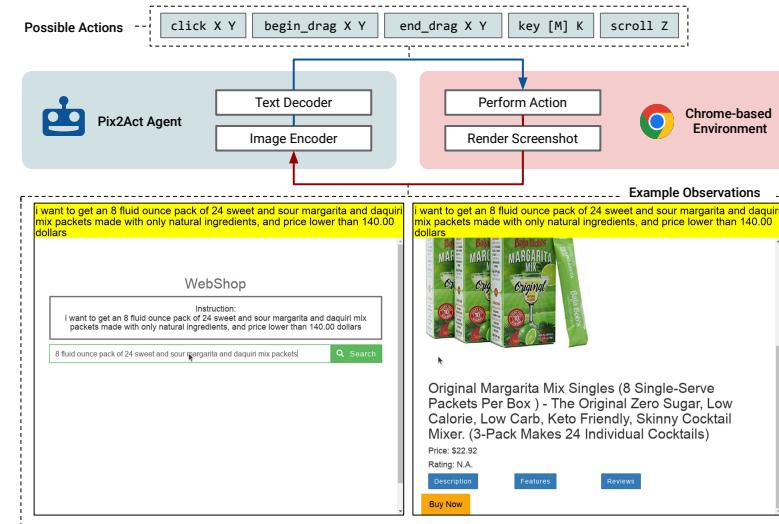
4.5

11.3

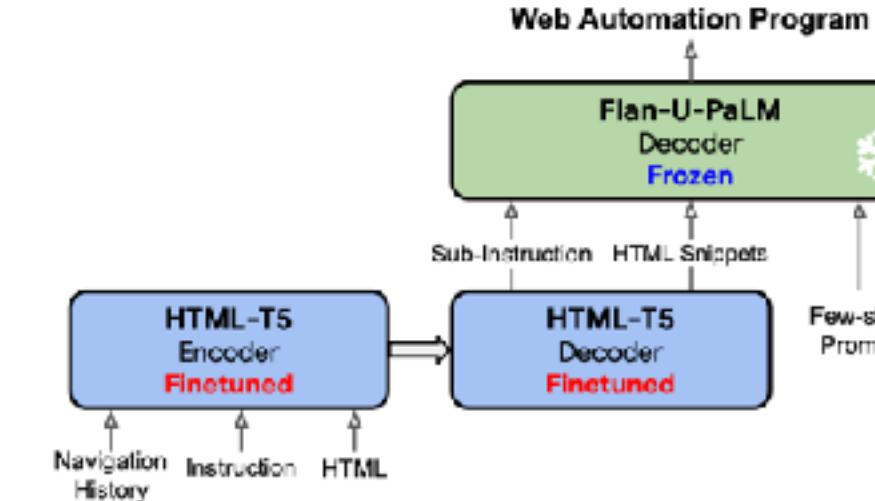
Getting all attributes requires long-horizon exploration!

WebShop enables sim-to-real transfer

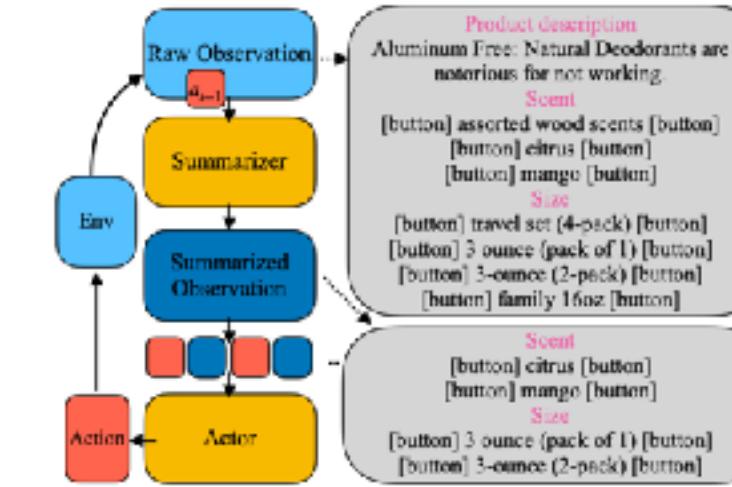




Pix2Act (Shaw et al., 2023)



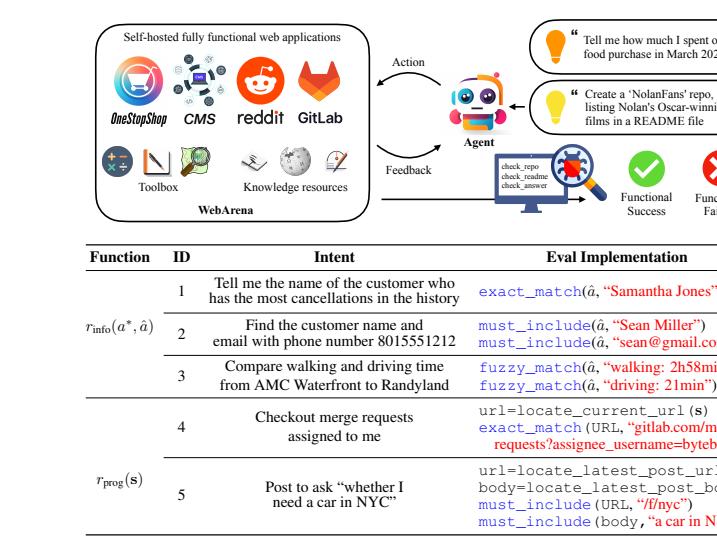
WebAgent (Gur et al., 2023)



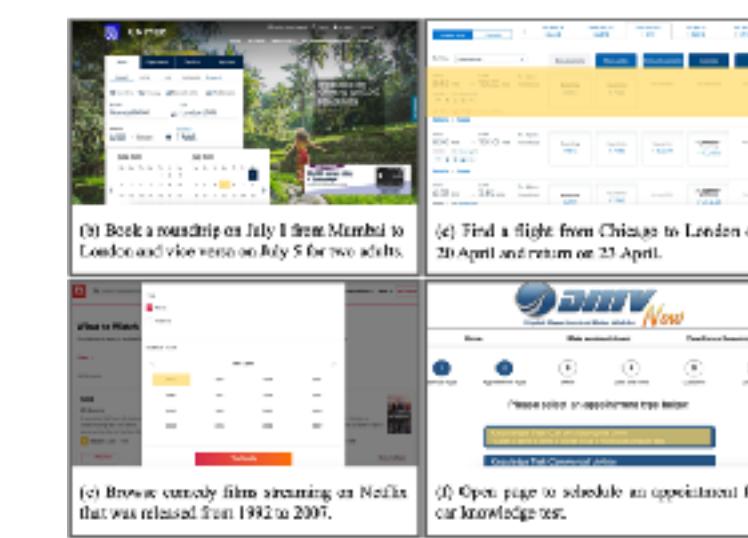
Ash (Sridhar et al., 2023)



SeeAct (Zheng et al., 2024)



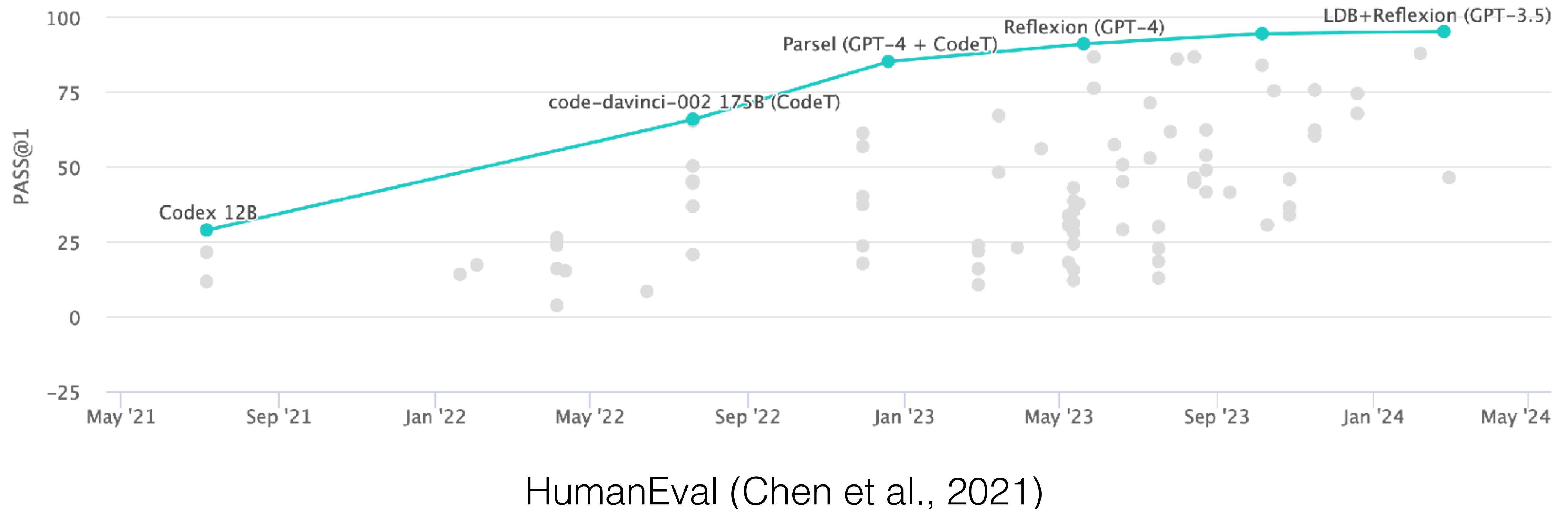
WebArena (Zhou et al., 2023)



Mind2Web (Deng et al.)

- Various followup methods and benchmarks for web interaction
- Testbed for industrial developments (e.g., Google, OpenAI)
- Inspired research on other real-world digital tasks (e.g., coding)

Coding benchmarks are becoming easy



HumanEval (Chen et al., 2021)

Our work [SCBGNY, Reflexion, NeurIPS'23] has reached >95%...

SWE-Bench

Metadata

Repo	scikit-learn/scikit-learn	Issue #s	[14858]
Instance ID	scikit-learn__scikit-learn-14869	Pull Number	14869
Created At	Aug 31, 2019	Base Commit	1018f9f...

Problem Statement

HGBC with categorical_crossentropy fails silently on binary classification

```
import numpy as np
from sklearn.experimental import enable_hist_gradient_boosting
from sklearn.ensemble import HistGradientBoostingClassifier

X = [[1, 0], [1, 0], [1, 0], [0, 1], [1, 1]]
y = [1, 1, 1, 0, 1]
gb = HistGradientBoostingClassifier(loss='categorical_crossentropy',
                                    min_samples_leaf=1)

gb.fit(X, y)
print(gb.predict([[1, 0]]))
print(gb.predict([[0, 1]]))
```

gives:

```
[0]
[0]
```

And binary_crossentropy works fine. categorical_crossentropy
should either generalize or raise an error on binary classification.

Ping @NicolasHug @ogrissel

Input: a GitHub repo and an issue

Output: a file diff to resolve the issue

Evaluation: unit tests from pull request

Test Patch

```
sklearn/ensemble/_hist_gradient_boosting/tests/test_gradient_boosting.py

418     assert stump_clf.fit(X, y_isnan).score(X, y_isnan) == 1
419
420 + def test_crossentropy_binary_problem():
421 +     # categorical_crossentropy should only be used if there
422 +     # are more than two classes present. PR #14869
423 +     X = [[1], [0]]
424 +     y = [0, 1]
425 +     gbdt = HistGradientBoostingClassifier(loss='categorical_crossentropy')
426 +     with pytest.raises(ValueError, match="'crossentropy' not suitable"):
427 +         gbdt.fit(X, y)
428
429     @pytest.mark.parametrize("scoring", [None, 'loss'])
```

LLMs cannot solve SWE-Bench

At least not in a sequence-to-sequence setup

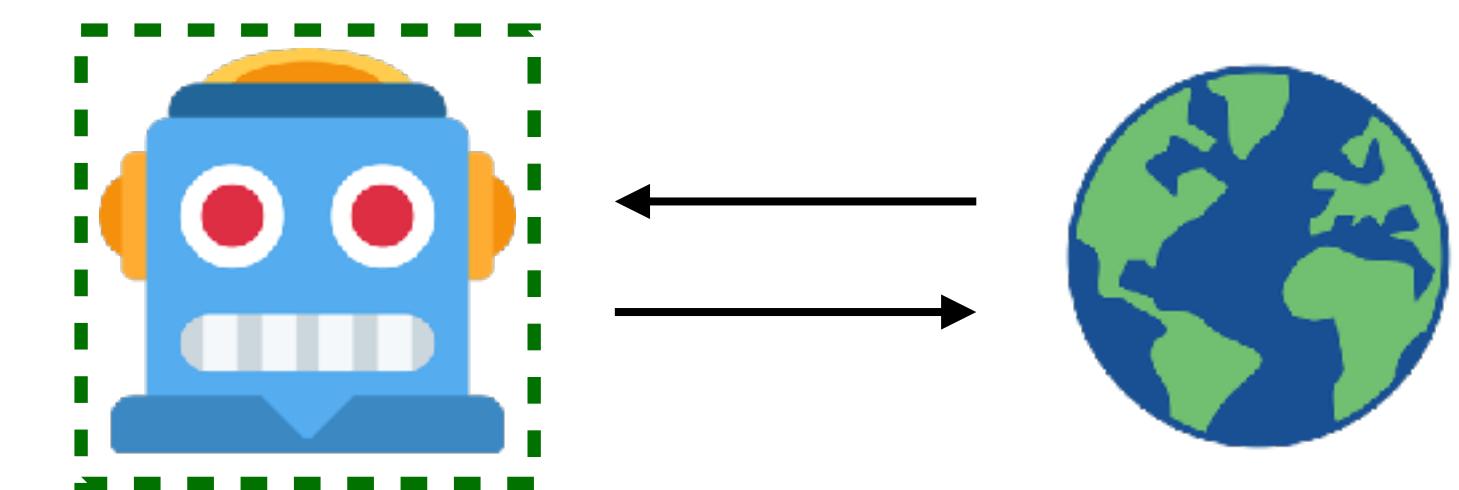
Model	% Resolved
ChatGPT-3.5	0.20
Claude 2	1.96
GPT-4*	0.00

Summary

- Digital automation: a new frontier for autonomous agents
 - Tremendous practical values
 - Scalable environment
 - Bottleneck: scalable evaluation
- It requires sequential decision-making over open-ended language
 - LLMs or RL agents cannot solve it
 - Require a fundamentally new type of agents

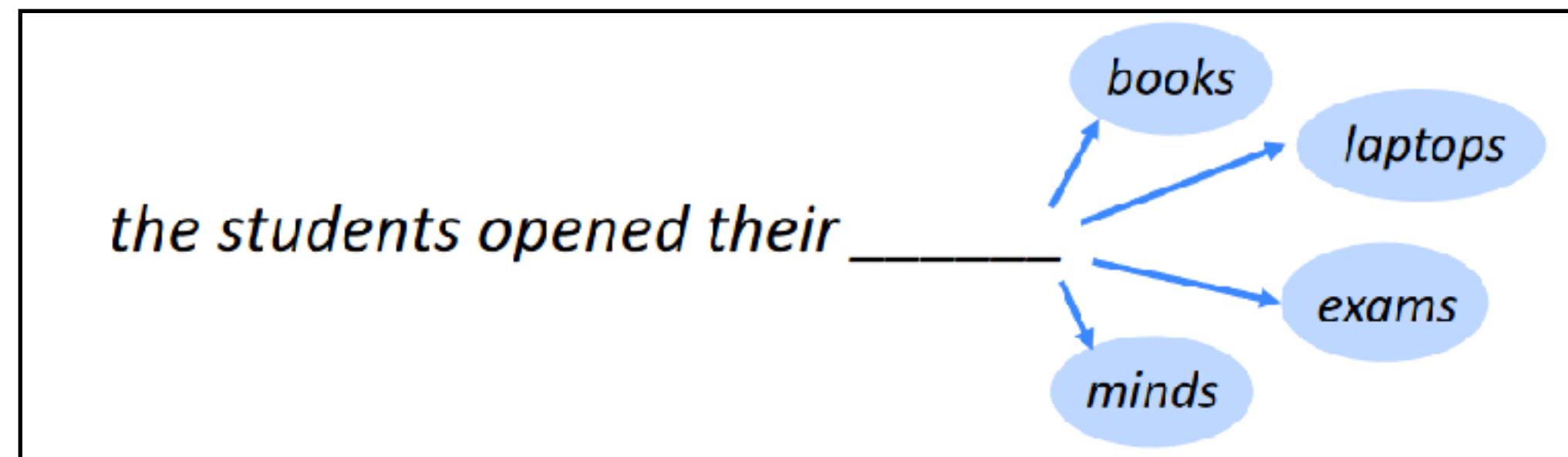
2

Building language agents
that reason to act

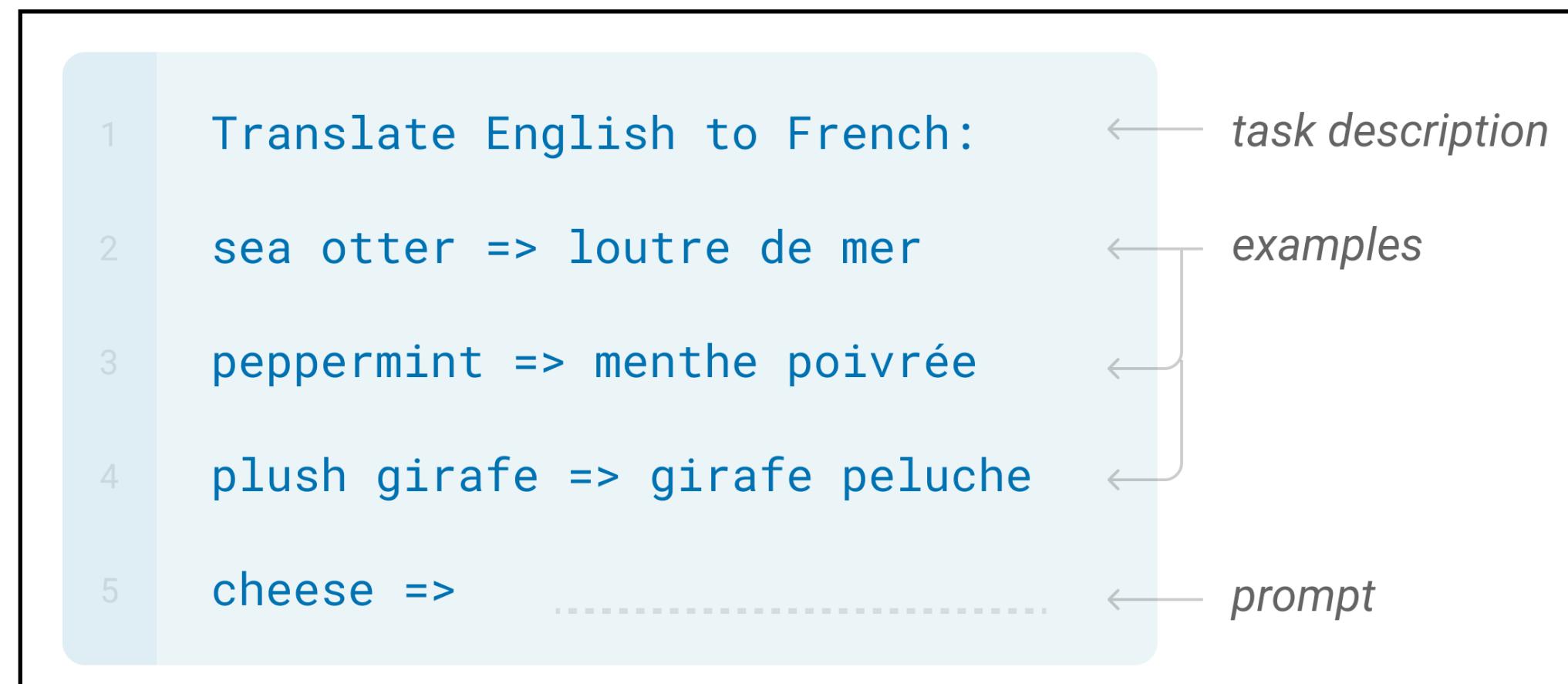


ReAct: Synergizing Reasoning and Acting in Language Models
Yao, Zhao, Yu, Du, Shafran, Narasimhan, Cao. ICLR 2023

LLMs can solve tasks using few examples



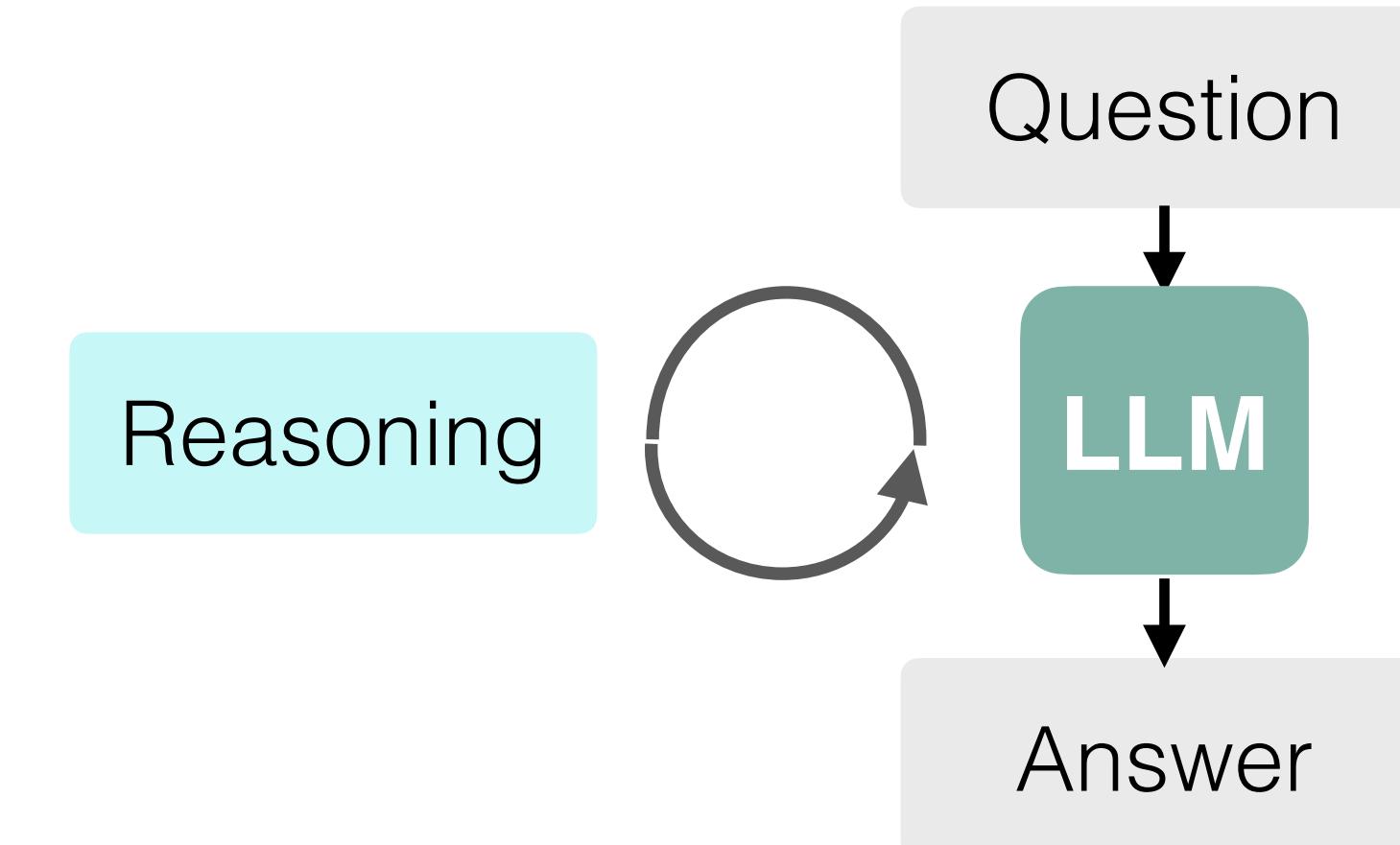
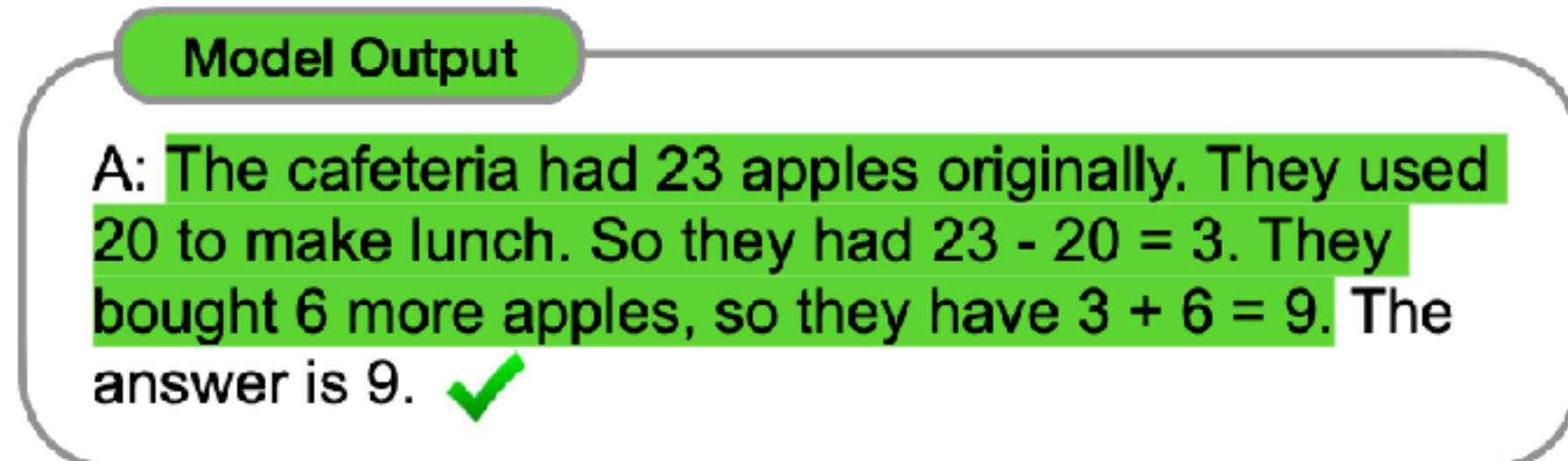
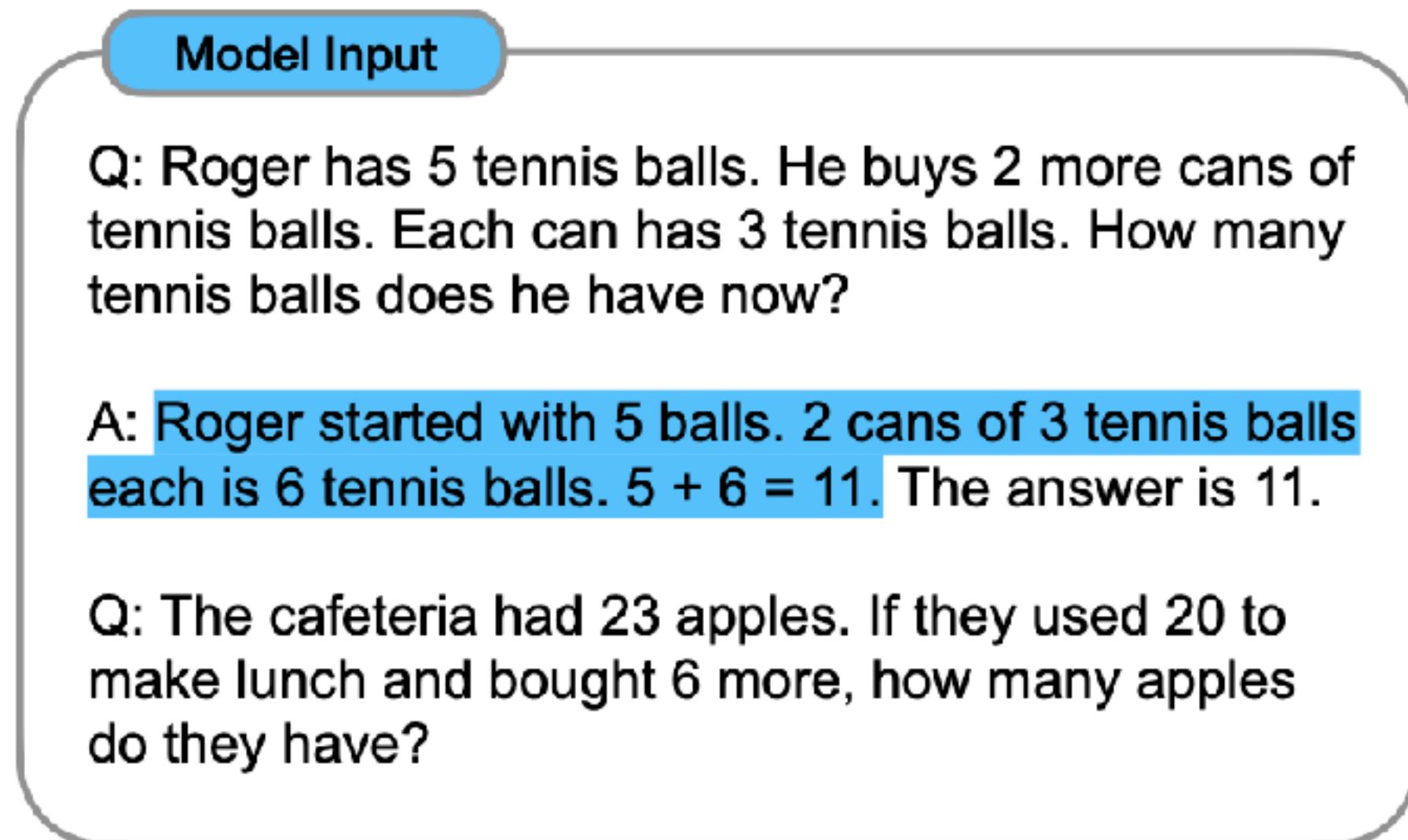
Training:
Next-token prediction



Inference:
Prompting for various NLP tasks

LLMs can reason to answer questions

Chain-of-Thought Prompting



Derive new information and update internal context

LLM reasoning: Lack of ~~knowledge & capabilities~~ Actions & feedback

(Chain-of-thought prompt omitted)

Question: If I have seven trillion dollars, can I buy Apple, Nvidia, and Microsoft? If not, how much more money do I need?

LLM

GPT-4
(Mar 2024)

Apple (AAPL) market cap: \$2.40 trillion
Nvidia (NVDA) market cap: \$1.20 trillion
Microsoft (MSFT) market cap: \$2.50 trillion
Total market cap: $\$2.40 + \$1.20 + \$2.50 = \6.10 trillion
 $\$6.10 \text{ trillion} - \$7 \text{ trillion} = \$0.90 \text{ trillion}$
You would need \$0.90 trillion more.

X

LLM acting: Lack of ~~examples~~ Reasoning

WebShop observation

[B0061IVFZE]

Brothers-ALL-Natural Fruit Crisps, Strawberry Banana, 0.42 - Ounce (Pack of 100)

\$85.0

[B096H2P6G2]

Moon Fruit Freeze Dried Fruit Snacks. Fruit Snacks for Kids - (Variety Pack)

\$18.99

[B092JLLYK6]

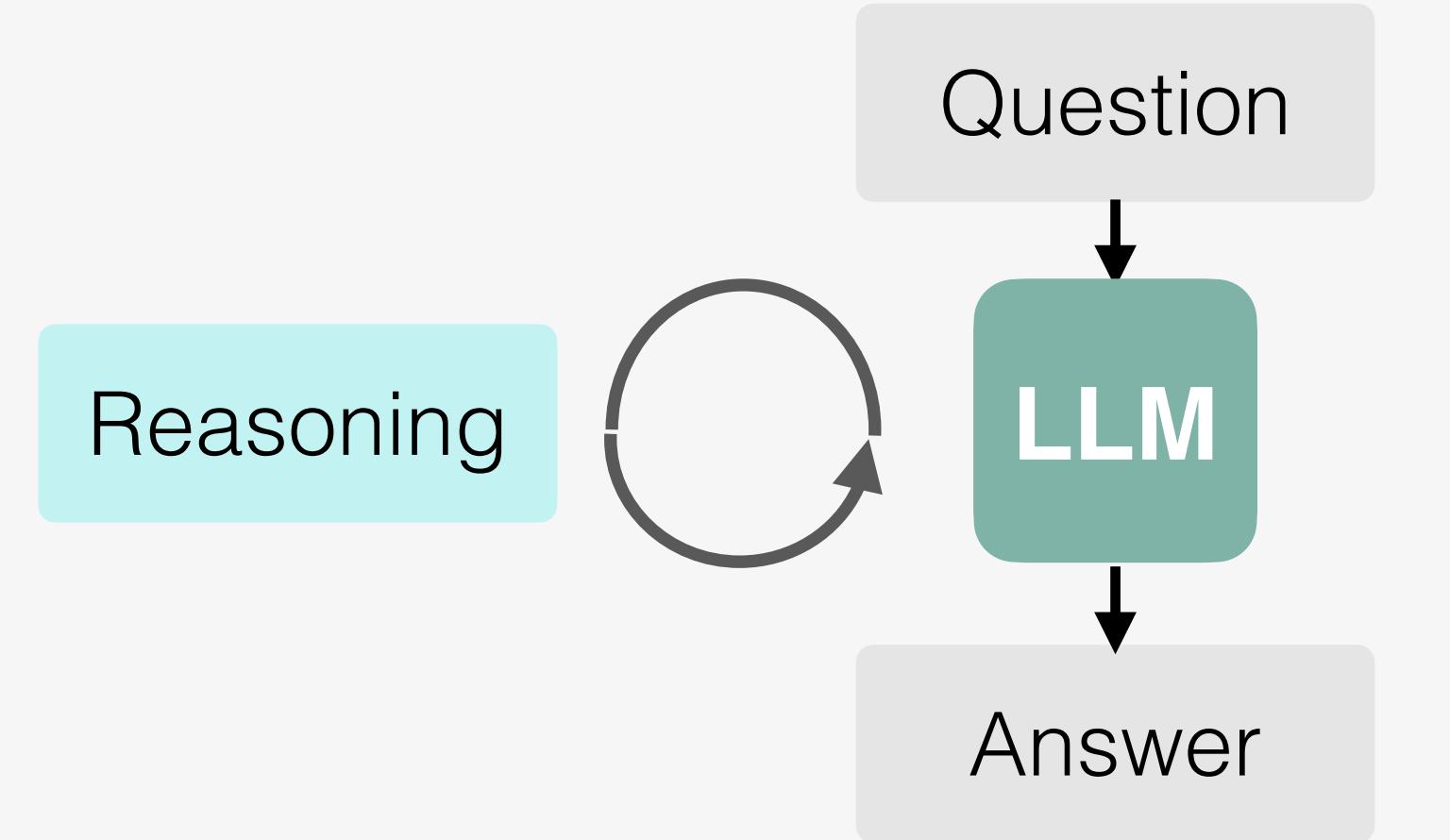
Nature's Turn Freeze-Dried Fruit Snacks - Banana Crisps - Perfect For School Lunches or an On-The-Go Snack - No Sugar Added, Non GMO, Gluten Free, Nothing Artificial (0.53oz) 6-Pack

\$12.99

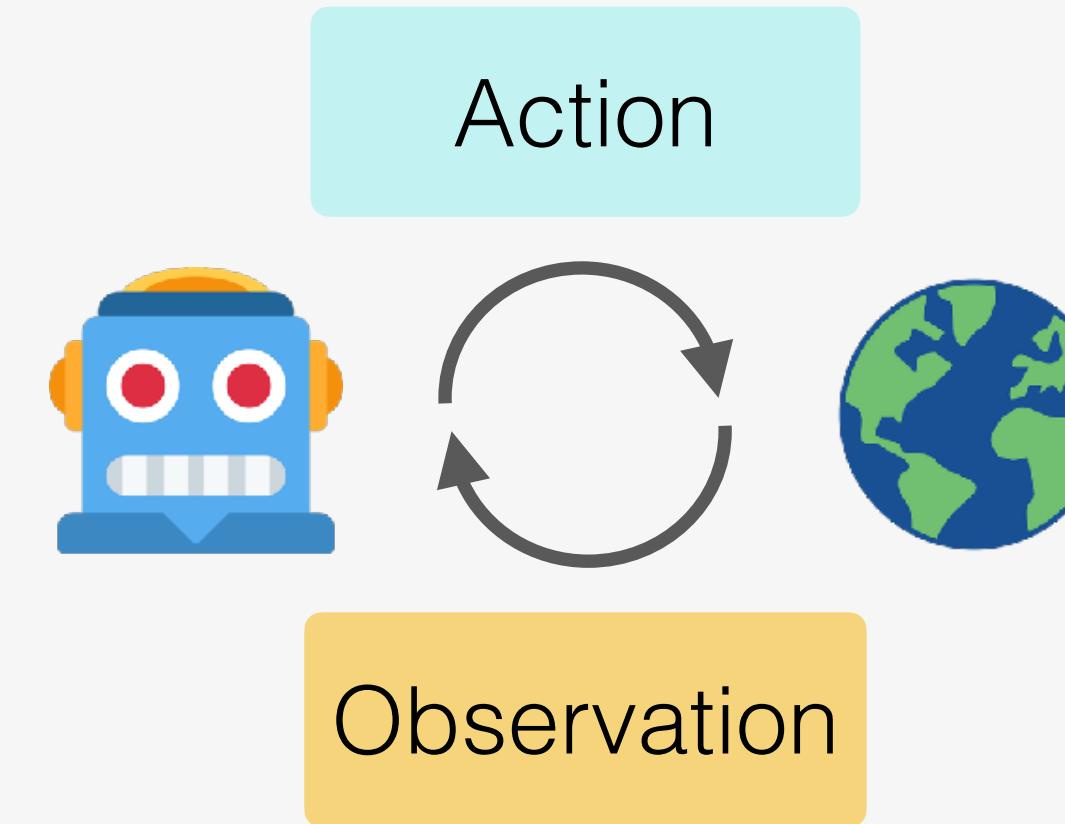
WebShop action

Action: click [B092JLLYK6]

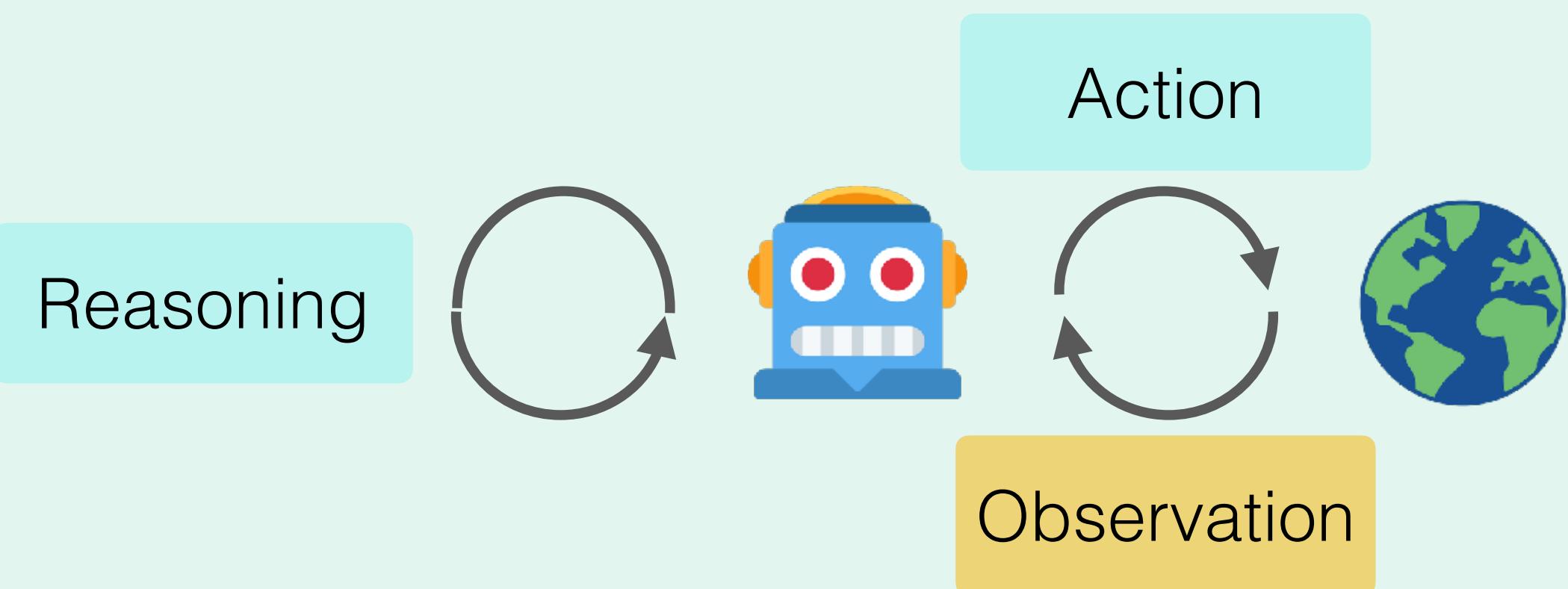
Reasoning (update internal belief)



Acting (obtain external feedback)

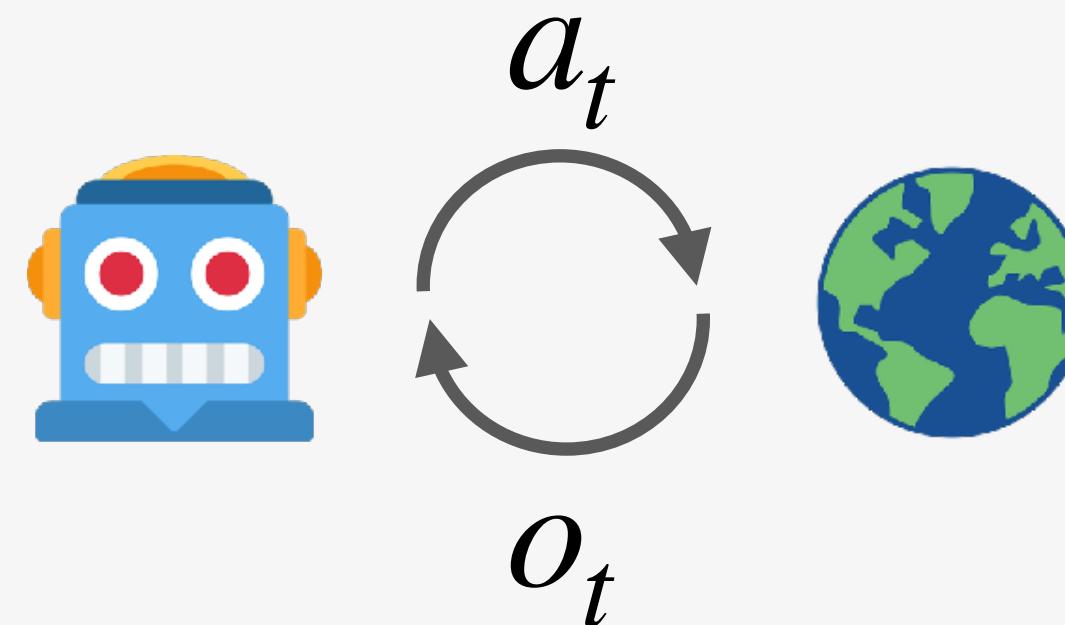


ReAct: a new paradigm of agents that **reason and act**



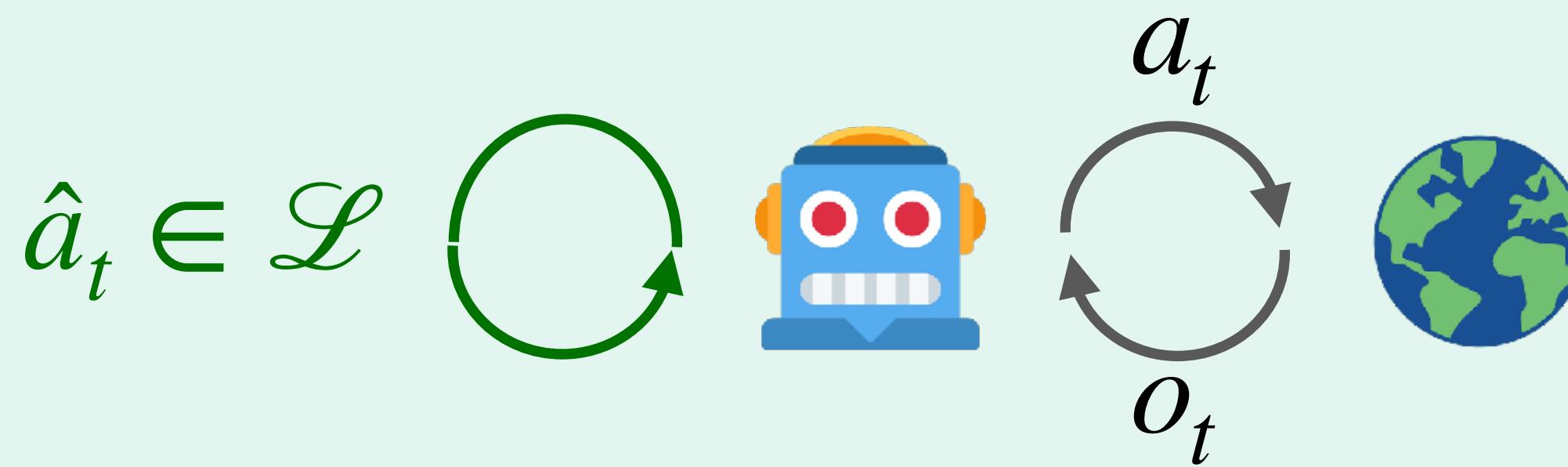
- **Synergy** of reasoning and acting
- **Simple** and intuitive to use
- **General** across domains

Traditional agents: action space A defined by the environment



- **External feedback** o_t
- Agent context $c_t = (o_1, a_1, o_2, a_2, \dots, o_t)$
- Agent action $a_t \sim \pi(a | c_t) \in A$

ReAct: action space $\hat{A} = A \cup \mathcal{L}$ augmented by reasoning



- $\hat{a}_t \in \mathcal{L}$ can be any language sequence
- Agent context $c_{t+1} = (c_t, \hat{a}_t, a_t, o_{t+1})$
- $\hat{a}_t \in \mathcal{L}$ only updates **internal context**

ReAct is simple and intuitive to use

ReAct supports

- One-shot prompting
- Few-shot prompting
- Fine-tuning

Example ReAct
trajectory

Task: xxxxxxx

Thought: xxx
Action: xxx

Observation:
xxxxxxxx

Thought: xxx
Action: xxx

.....

Human prompt

LLM output

Environment
feedback

A zero-shot ReAct prompt

You are an agent that answers questions by using two actions:

- search[query]: Google search the query. You can also use it to calculate math.
- finish[answer]: return the answer

Your generation should have the following format:

Thought: to analyze the observation and inform the action.

Action: your action

Question: If I have seven trillion dollars today, can I buy Apple, Nvidia, and Microsoft? If not, how much more money do I need?

LLM

(GPT-4)

Thought: I need to find the current market capitalization of Apple, Nvidia, and Microsoft to understand how much it would cost to buy all of them.
Action: search["current market capitalization of Apple, Nvidia, and Microsoft"]



(Mar 2024)

Observation: "Nvidia's market cap stood at \$1.832 trillion as of about 10 a.m. ET Wednesday, while Alphabet's was at \$1.814 trillion, bringing Nvidia behind just Microsoft (MSFT) at \$3.015 trillion and Apple (AAPL) at \$2.836 trillion."

Thought: Now that I have the market cap for all companies, let's add them together and see if \$7 trillion is enough.
Action: search["1.832 trillion + 3.015 trillion + 2.836 trillion"]

Observation: 7683000000000

Thought: The amount of money needed in addition to the \$7 trillion is \$683 billion.
Action: finish["You need an additional \$683 billion to be able to buy Apple, Nvidia, and Microsoft."]



LLM

(GPT-4)

Thought: I need to find the current market capitalization of Apple, Nvidia, and Microsoft to understand how much it would cost to buy all of them.
Action: search["current market capitalization of Apple, Nvidia, and Microsoft"]

Observation: nothing found!

(My adversarial observation)

Thought: The search did not give any results. I will need to search for each company's market capitalization individually.

Action: search["current market capitalization of Apple"]

Observation: 'price': 180.75

(My adversarial observation)

Thought: This is the current price of Apple's stock, not its market capitalization. The market cap is calculated by multiplying the share price by the number of outstanding shares. Thus, I need to find the total number of Apple's outstanding shares to calculate its market cap.

Action: search["total number of Apple's outstanding shares"]



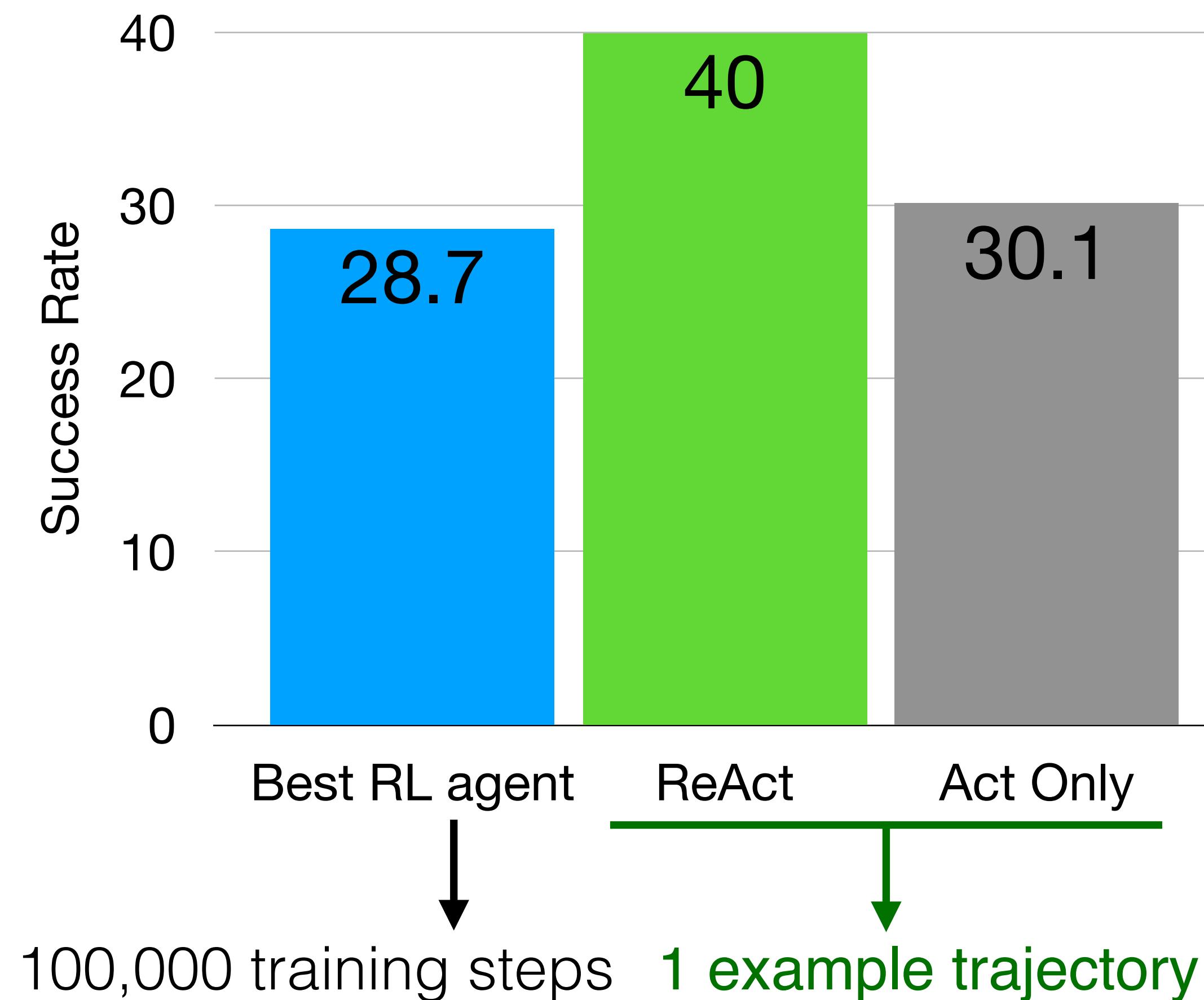
(Continues to solve the task)

Synergy: acting support reasoning, reasoning guides acting

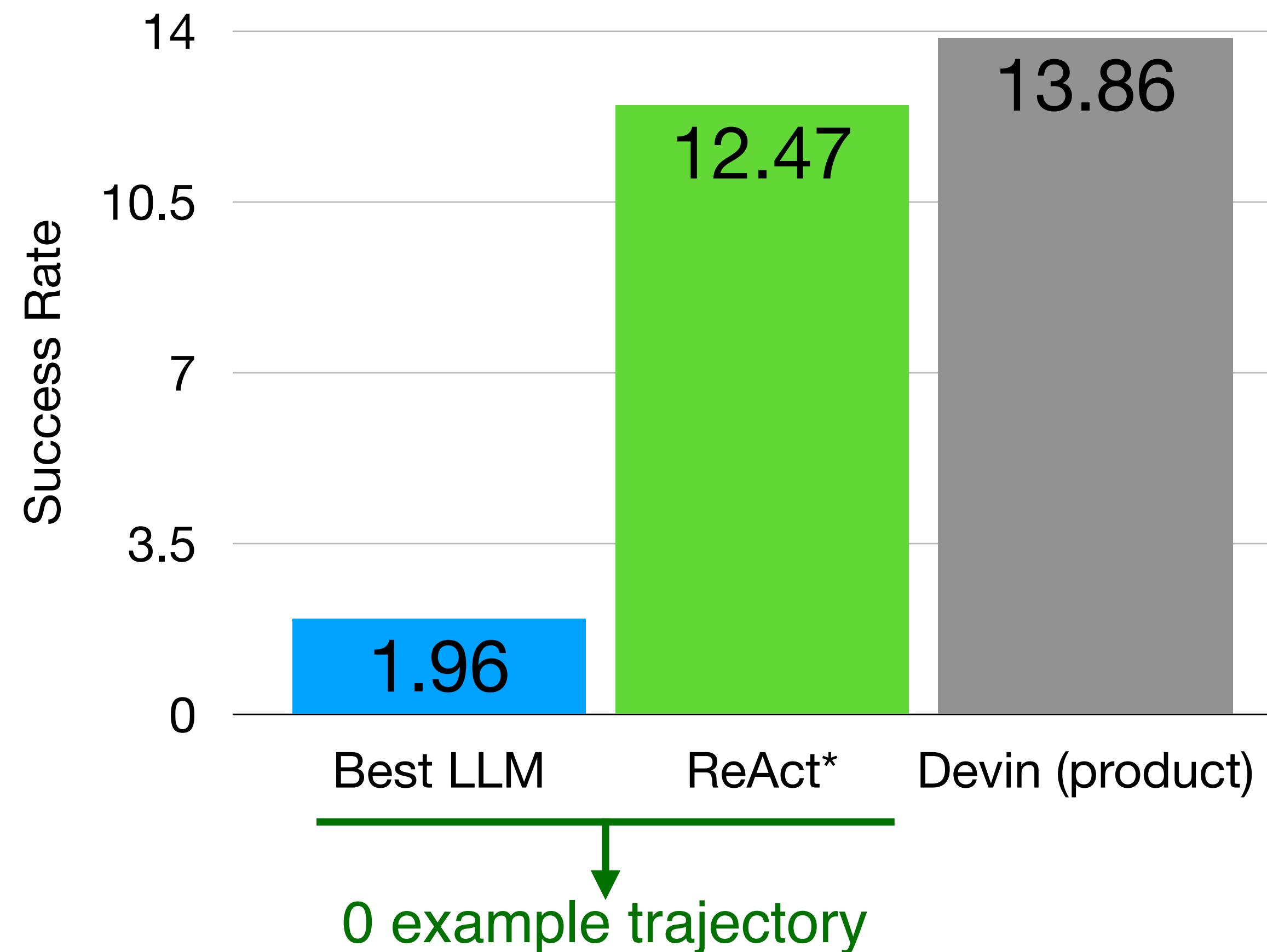
ReAct is general and effective

PaLM-540B	(NLP tasks)		(RL tasks)	
	HotpotQA (QA)	FEVER (<i>fact check</i>)	ALFWorld (<i>Text game</i>)	WebShop (<i>web</i>)
Reason	29.4	56.3	N/A	N/A
Act	25.7	58.9	45	30.1
ReAct	35.1	64.6	71	40

WebShop: One-shot generalization



SWE-Bench: Zero-shot generalization



* SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering

Paper coming out soon :)

Impact in 2023

Followup methods	Interdisciplinary research	Industrial applications
Voyager	(Wang et al.)	Creative art (Sun et al.)
ToolLLM	(Qin et al.)	Healthcare (Imrie et al.)
AutoGen	(Wu et al.)	Robotics (Hireche et al.)
AutoGPT	(Richards et al.)	Education (Tabrizi et al.)
MetaGPT	(Hong et al.)	Diaster control (Colverd et al.)
SwiftSage	(Lin et al.)	Networks (Hamadanian et al.)
MM-ReAct	(Yang et al.)	Chemistry (Bran et al.)
ToolkenGPT	(Hao et al.)	Mathematics (Liao et al.)

ReAct enables discovery of a novel chromophore

a

```
from chemcrow.agents import ChemTools, ChemCrow

chemtools = ChemTools()

# Initialize ChemCrow object with toolset and LLM
crow = ChemCrow(
    chemtools.all_tools,
    model="gpt-4",
    temp=0.1,
)

# Task definition
task = (
    "Find and synthesize a thiourea organocatalyst "
    "which accelerates a Diels-Alder reaction."
)

# Execute ChemCrow
crow.run(task)
```

b **Task:** Find and synthesize a thiourea organocatalyst which accelerates a Diels-Alder reaction.

First, I need to find a thiourea organocatalyst that can accelerate the Diels-Alder reaction. I will perform a web search to find a suitable catalyst.

Web Search tool: Schreiner's thiourea catalyst

Now, I will obtain the SMILES. **Name2Smiles tool:**

FC(F)(F)c1cc(NC(=S)Nc2cc(C(F)(F)F)cc(C(F)(F)F)c2)cc(C(F)(F)F)c1

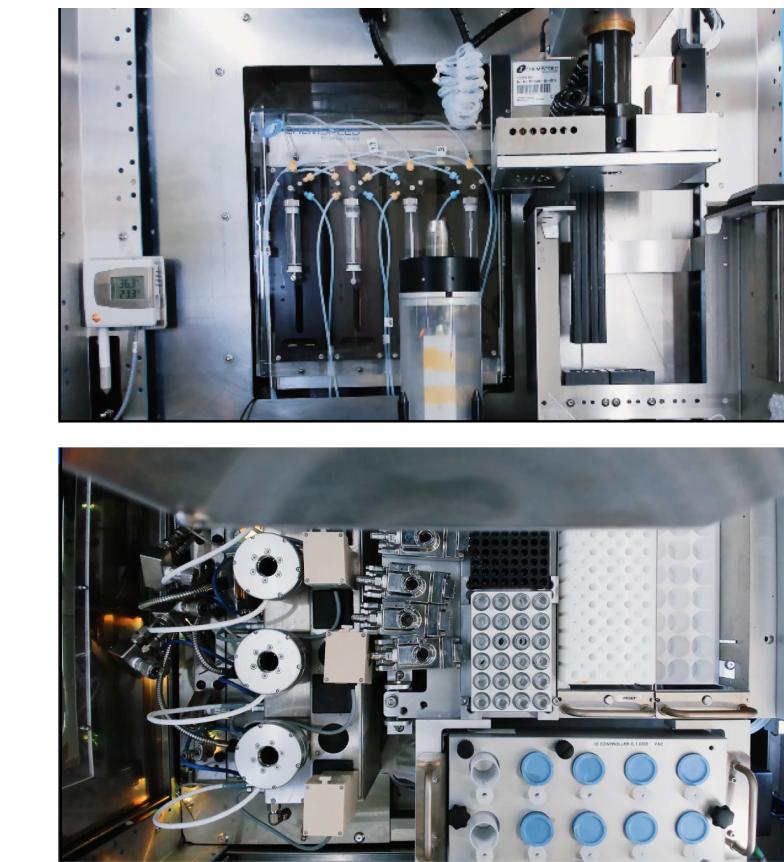
I will plan a synthesis for Schreiner's thiourea catalyst.

SynthesisPlanner tool: Detailed synthesis plan

I will execute the synthesis.

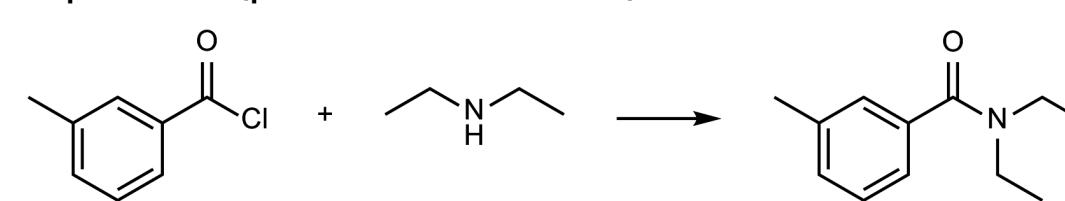
SynthesisExecuter tool: Successful synthesis.

c **RoboRXN synthesis platform**



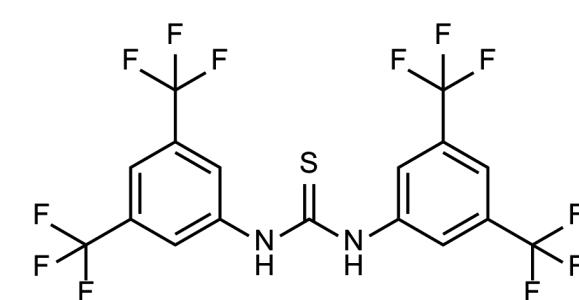
d Chemcrow workflows with experimental validation

Insect repellent (plan and execute)

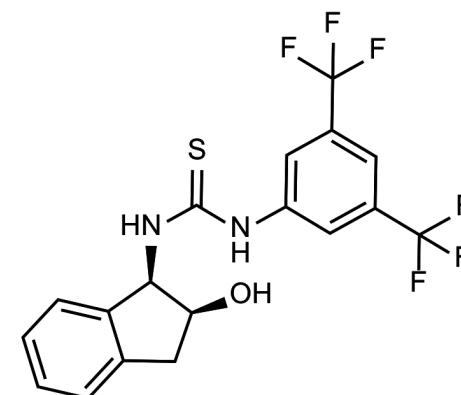


Thiourea organocatalysts (plan and execute)

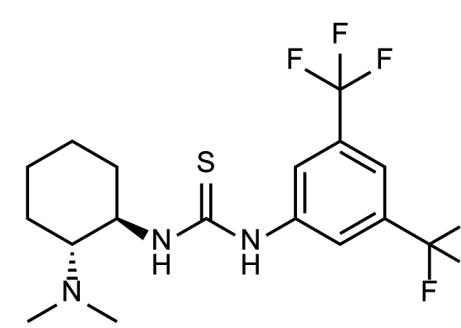
Schreiner's catalyst



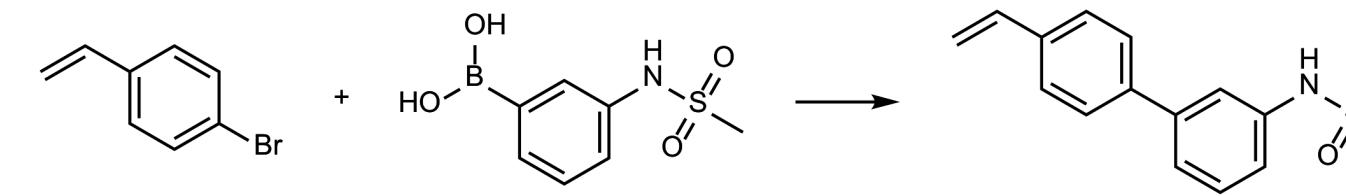
Ricci's catalyst



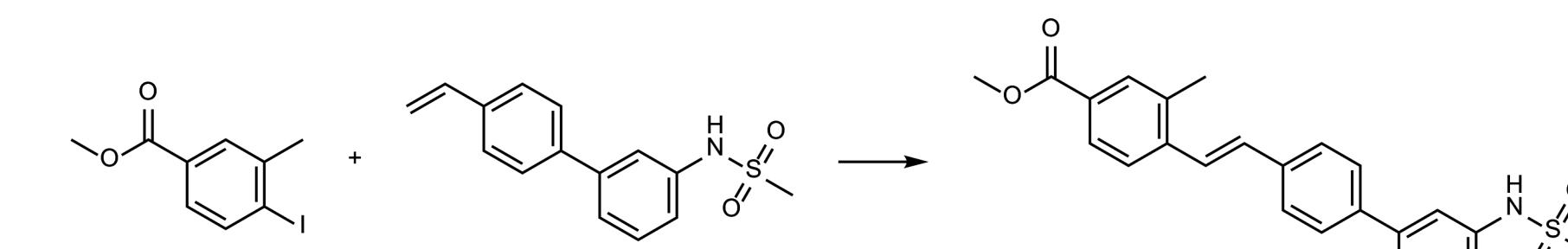
Takemoto's catalyst



Novel chromophore (clean data, train model, and predict)



Synthesis step 1: Bromo Suzuki coupling



Synthesis step 2: Iodo Heck reaction

Language agents

Large language models

Language models

Text generation

Translation, QA, summarization...

Web interaction, SWE, robotics, scientific discovery...

Is next-token prediction enough for general problem solving?

Tree of Thoughts (Yao et al., 2023): no!

Next-token prediction cannot reason deliberately

GPT-4 Input

Question: How to combine 2, 9, 10, 12 to get 24?

Thought: $12 * 2 = 24$; $10 - 9 = 1$; $24 * 1 = 24$.

Answer: $(12 * 2) * (10 - 9) = 24$

Question: How to combine 4, 5, 6, 10 to get 24?

GPT-4 Output

Initial tokens are hard to decide



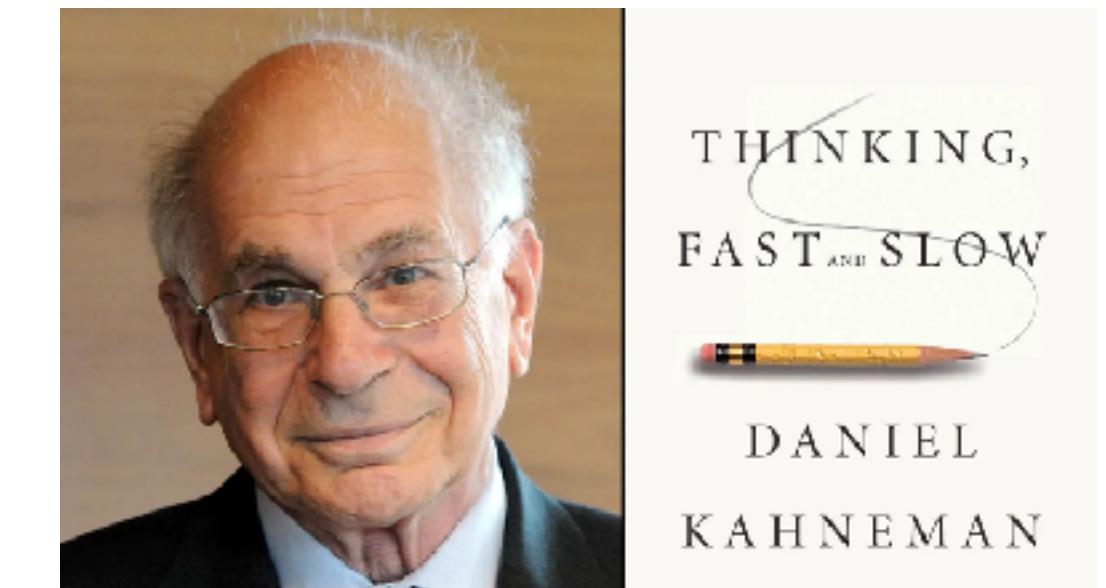
Thought: $10 * 6 = 60$; $60 / 5 = 12$; $12 * 4 = 48$
Answer: $((10 * 6) / 5) * 4 = 24$

X

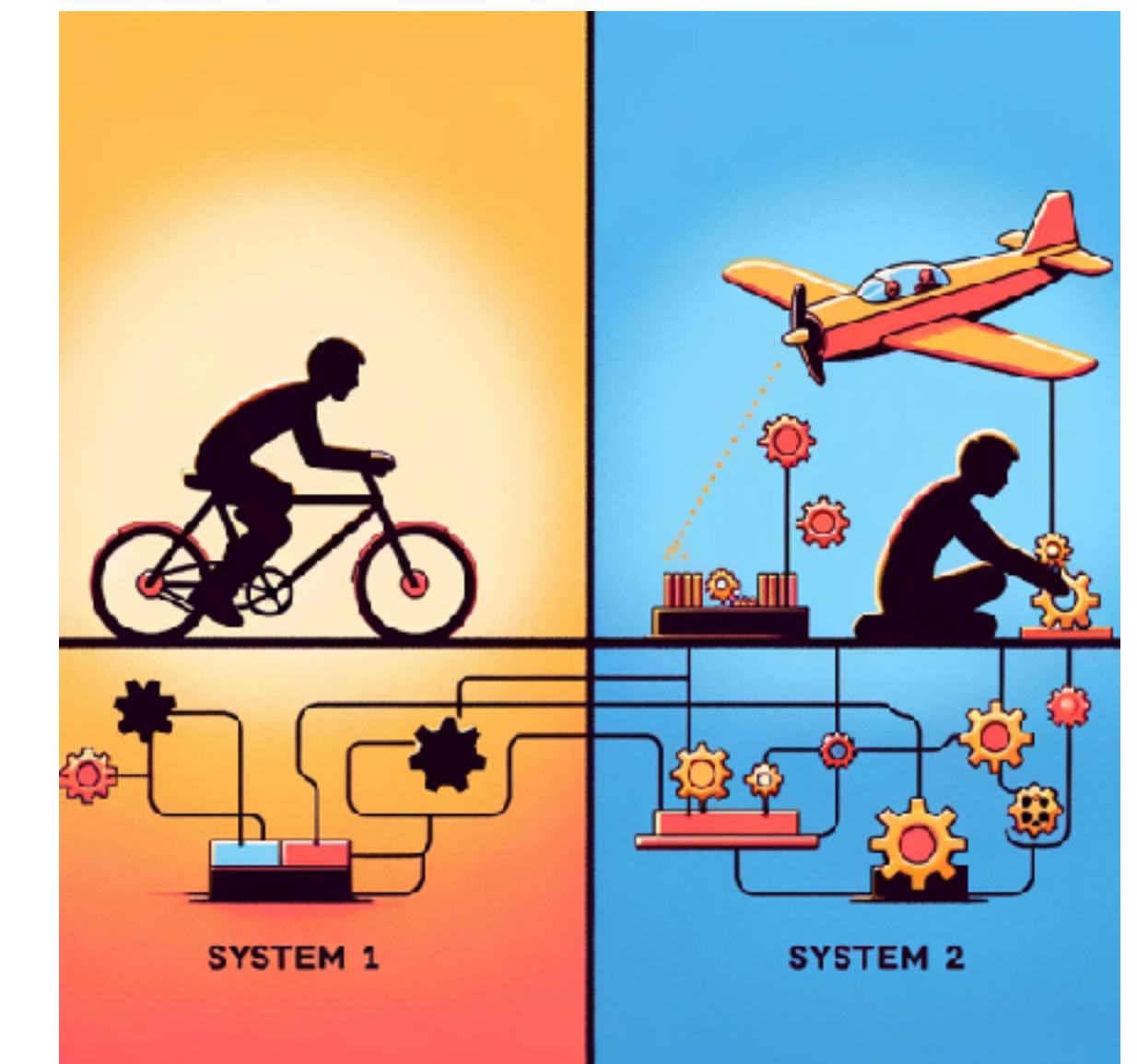
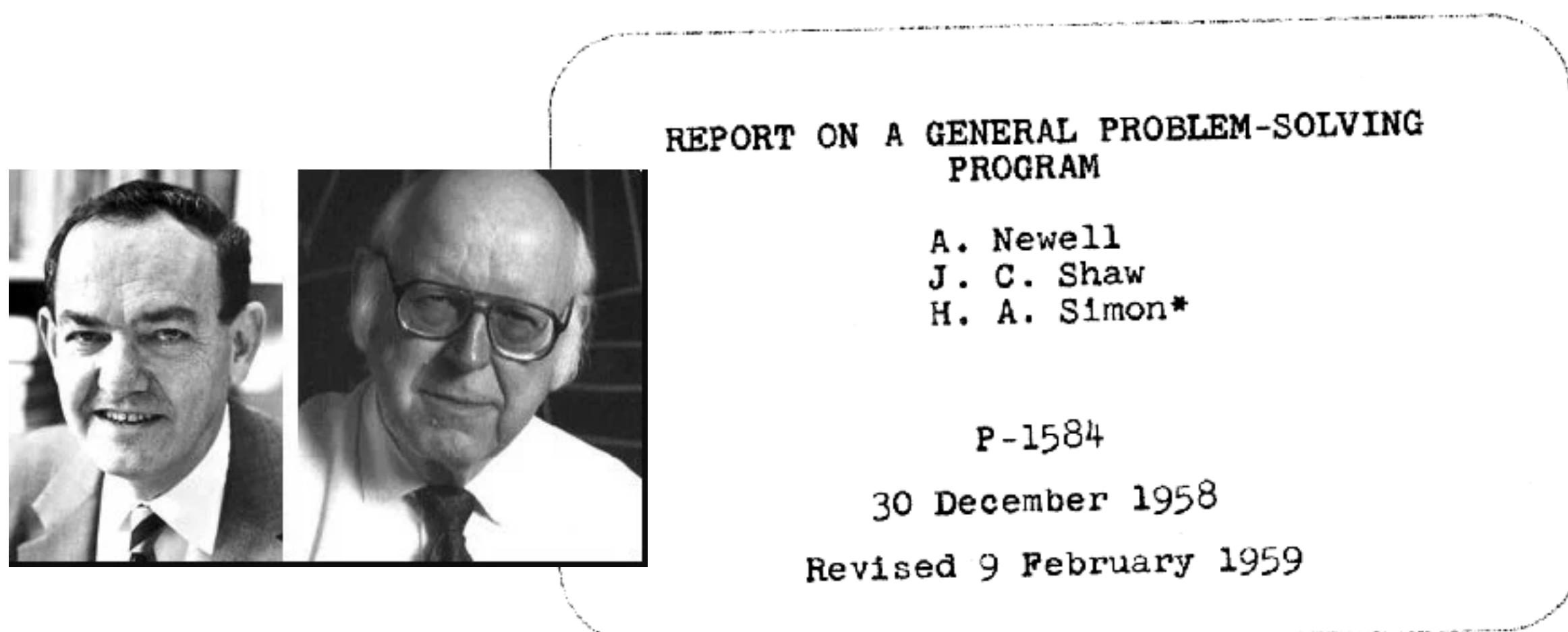
LLMs make linear token decisions without lookahead or backtrack!

How do we fix next-token prediction?

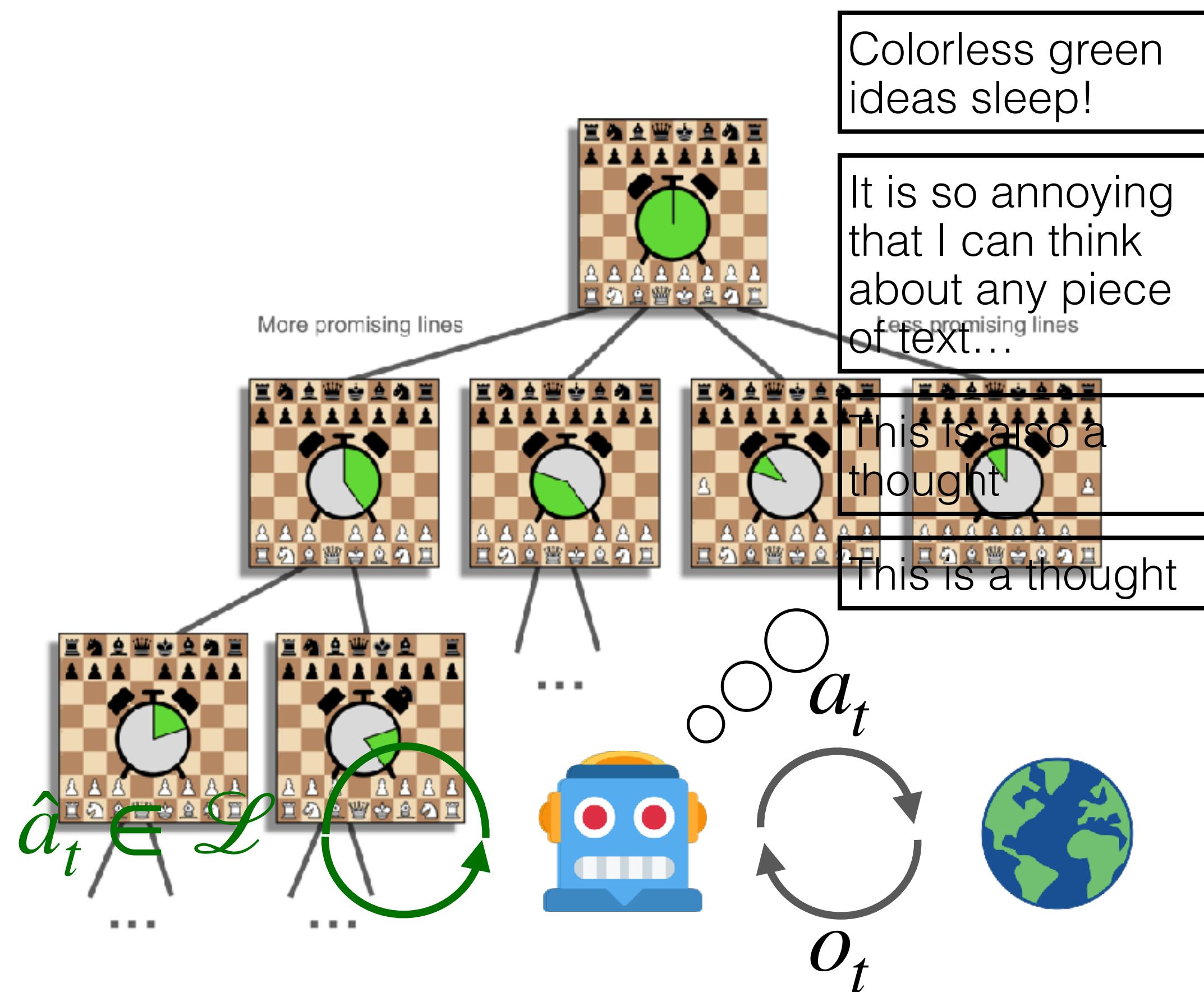
- We took inspirations from human cognition
 - System 1: fast and automatic (~next-token prediction)
 - System 2: slow and deliberate (~control algorithm)



One of the oldest ideas in AI: Tree search



Natural language search: Curse of combinatoriality



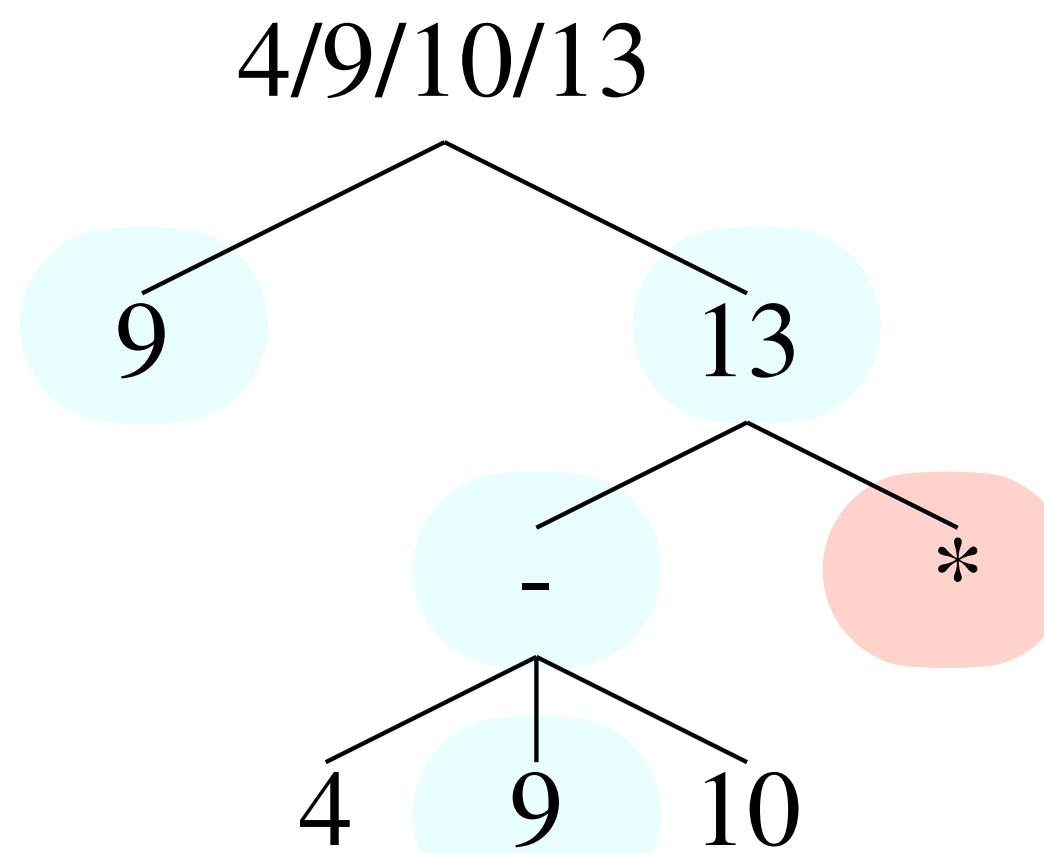
- Classical search (e.g., chess agent):
 - A small, well-defined action space A_{env}
 - Can simulate external feedback $o_t \in O_{env}$
 - Can design/learn evaluation heuristics $f(a_t)$
- Search in the space of thought \mathcal{L} :
 - \mathcal{L} is combinatorial and infinite!
 - No external feedback
 - Hard to enumerate or evaluate thoughts

Tree of Thoughts: Blessing of compositionality

Thought: A semantically coherent unit of text that can be generated/evaluated by LLMs

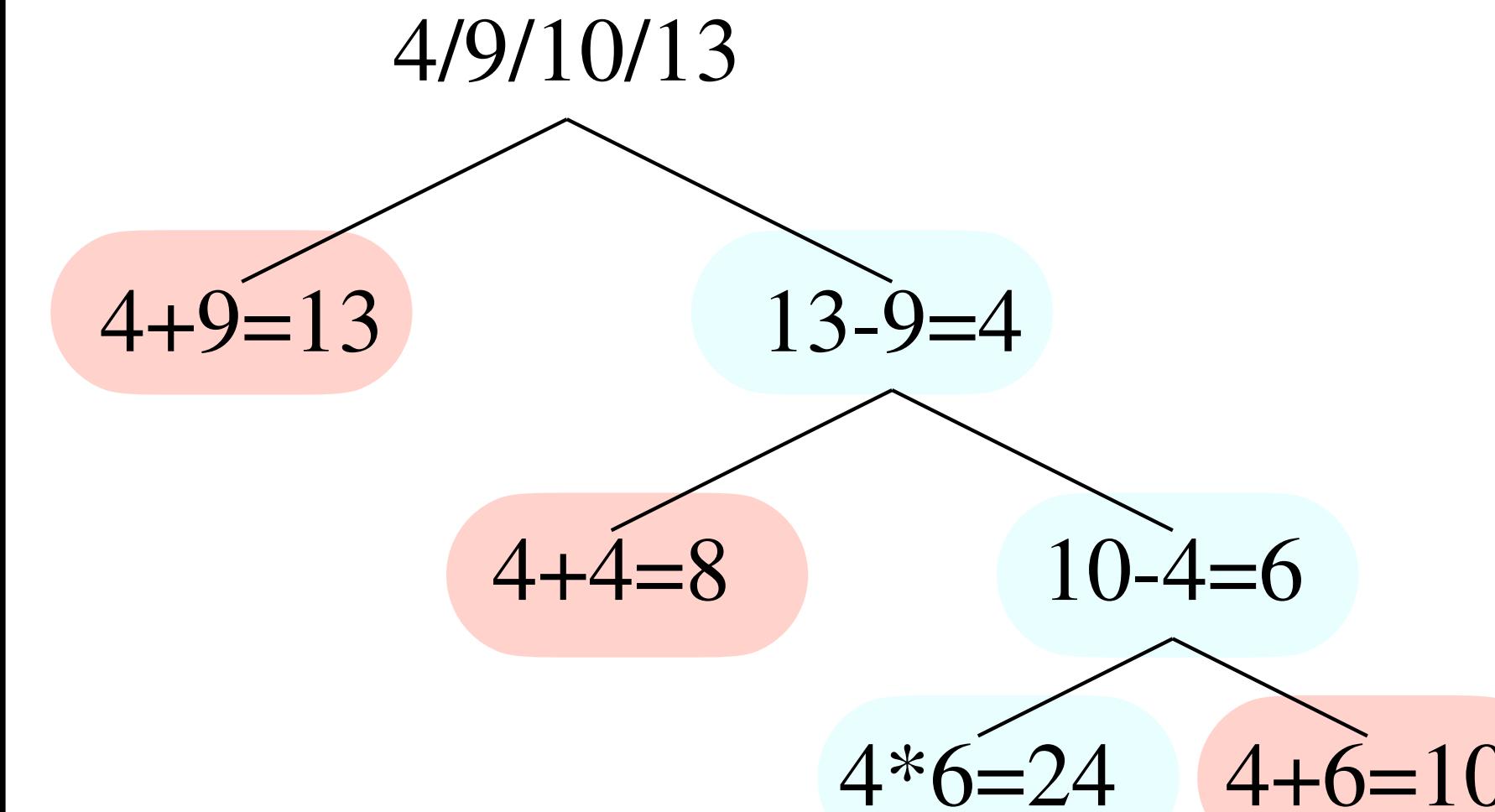
Each token as thought

- Easy to generate
- Hard to evaluate



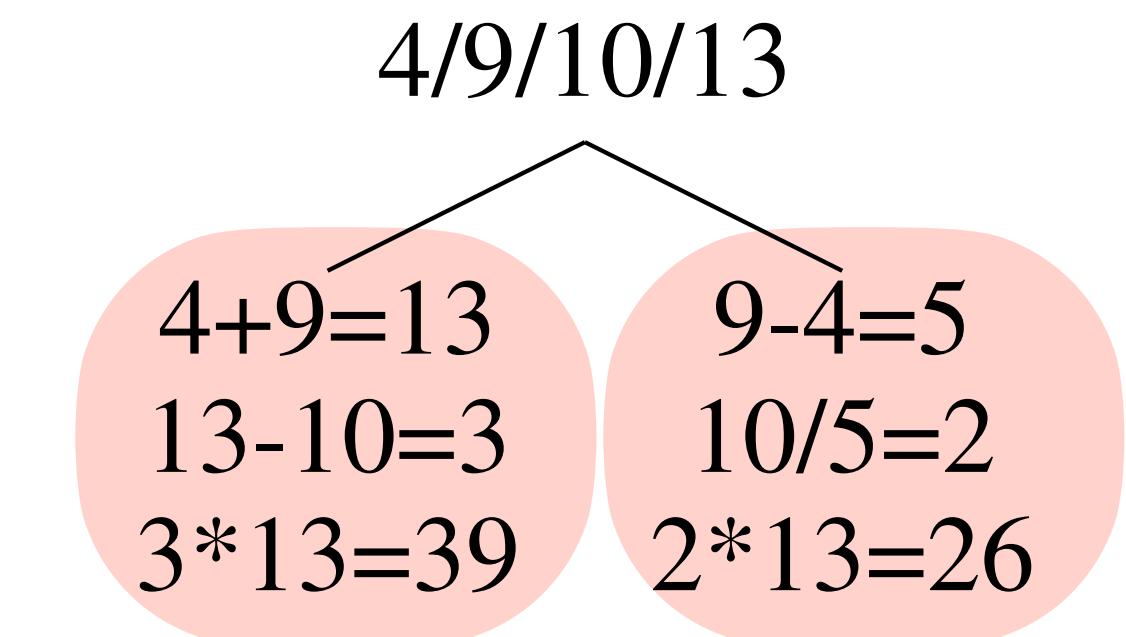
Each equation as thought

- Relatively easy to generate/evaluate
- A problem-specific tradeoff design

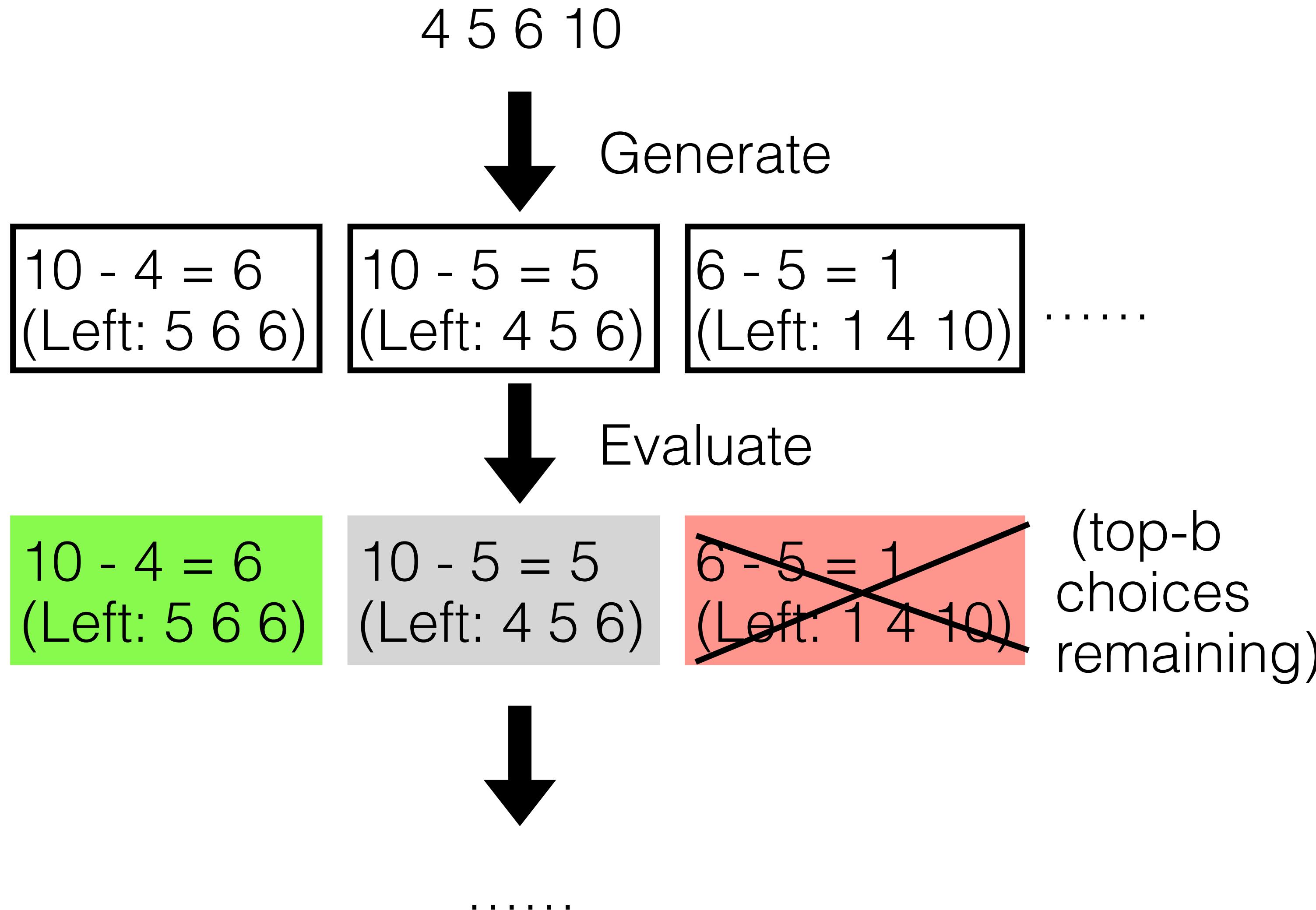


Whole reasoning as thought

- Easy to evaluate
- Hard to generate



Thought-level BFS



Generation Prompt: come up with ways to combine two of these numbers...

Evaluation Prompt: how likely are these 3 numbers to combine to 24...

Task success:

CoT	4%
ToT (ours)	74%

Tasks	Game of 24	Crosswords	Creative Writing
“Thought”	An equation	A clue word	A writing plan
Steps	3	5-10	1
Search	BFS	DFS	BFS
Generation	proposal	proposal	sample
Evaluation	simulation/ commonsense	simulation/ commonsense	zero-shot vote
CoT -> ToT	4% -> 74%	1% -> 20%	21% vs 41%



princeton-nlp/tree-of-thought-lm

Public



[NeurIPS 2023] Tree of Thoughts: Deliberate Problem Solving with Large Language Models

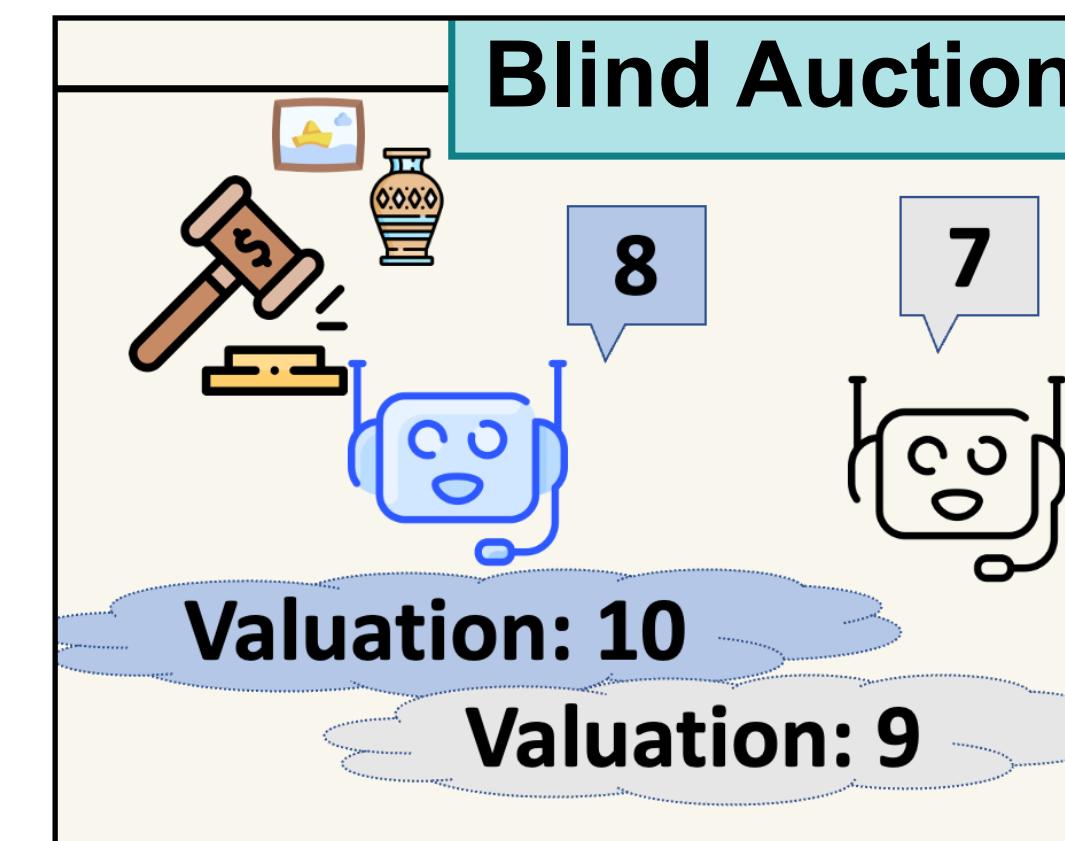
Python

4k

364

Rating Prediction

How will `user_X` rate the item "Kusco-Murphy Tart Hair"?
The rating should be an integer between 1 to 5, with 1 being lowest and 5 being highest.

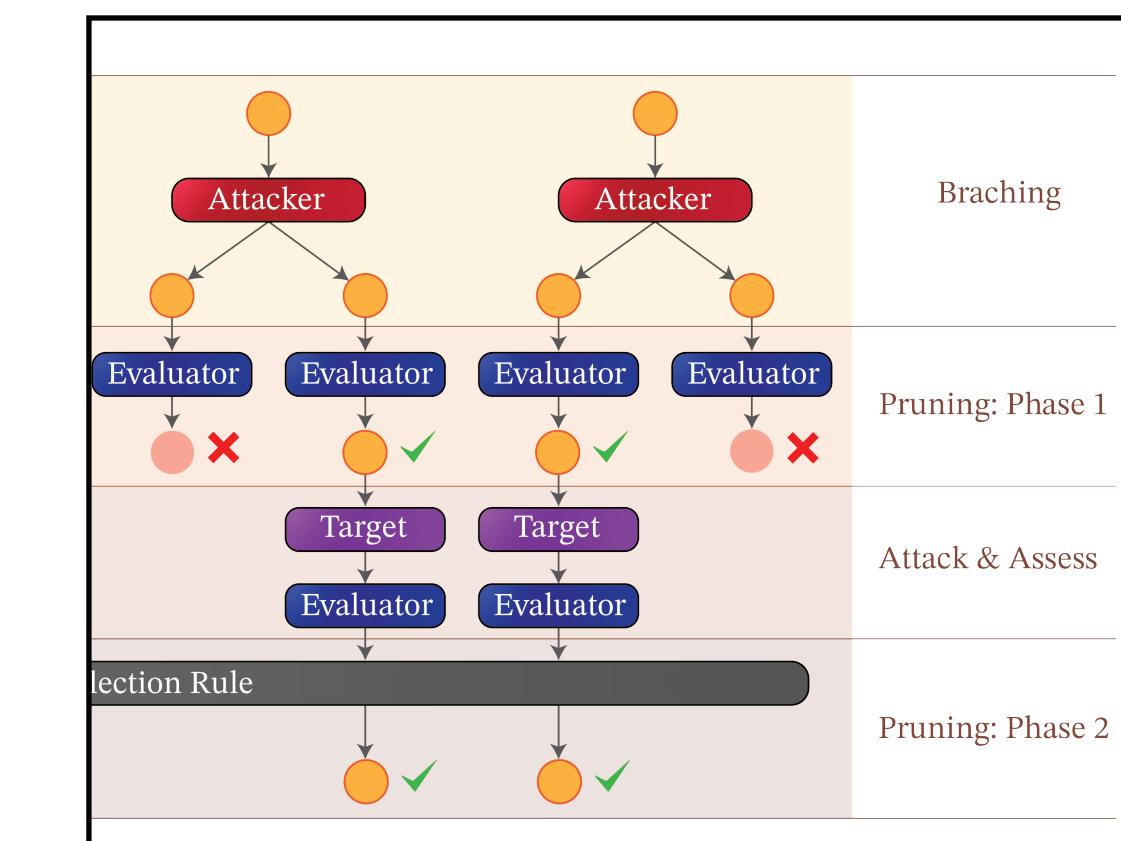


Recommender agent
(Wang et al., 2023)

Evaluator:
simulate humans

Auction agent
(Dean et al., 2024)

Evaluator:
simulate agents



Jailbreak agent
(Mehrotra et al., 2023)

Evaluator:
simulate self

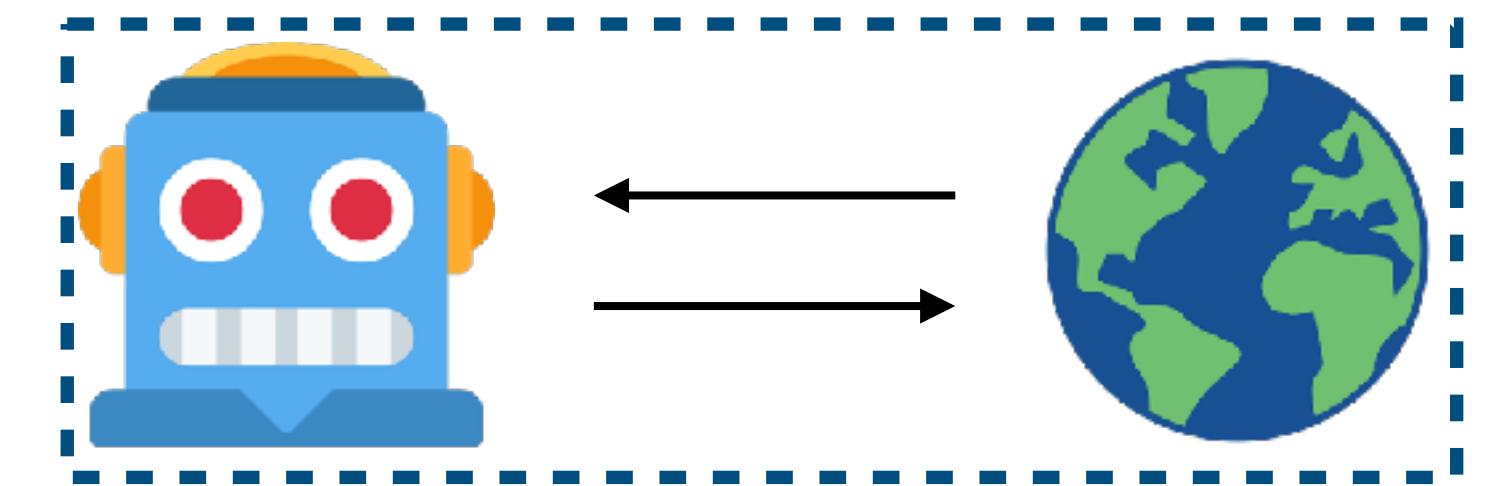
Summary

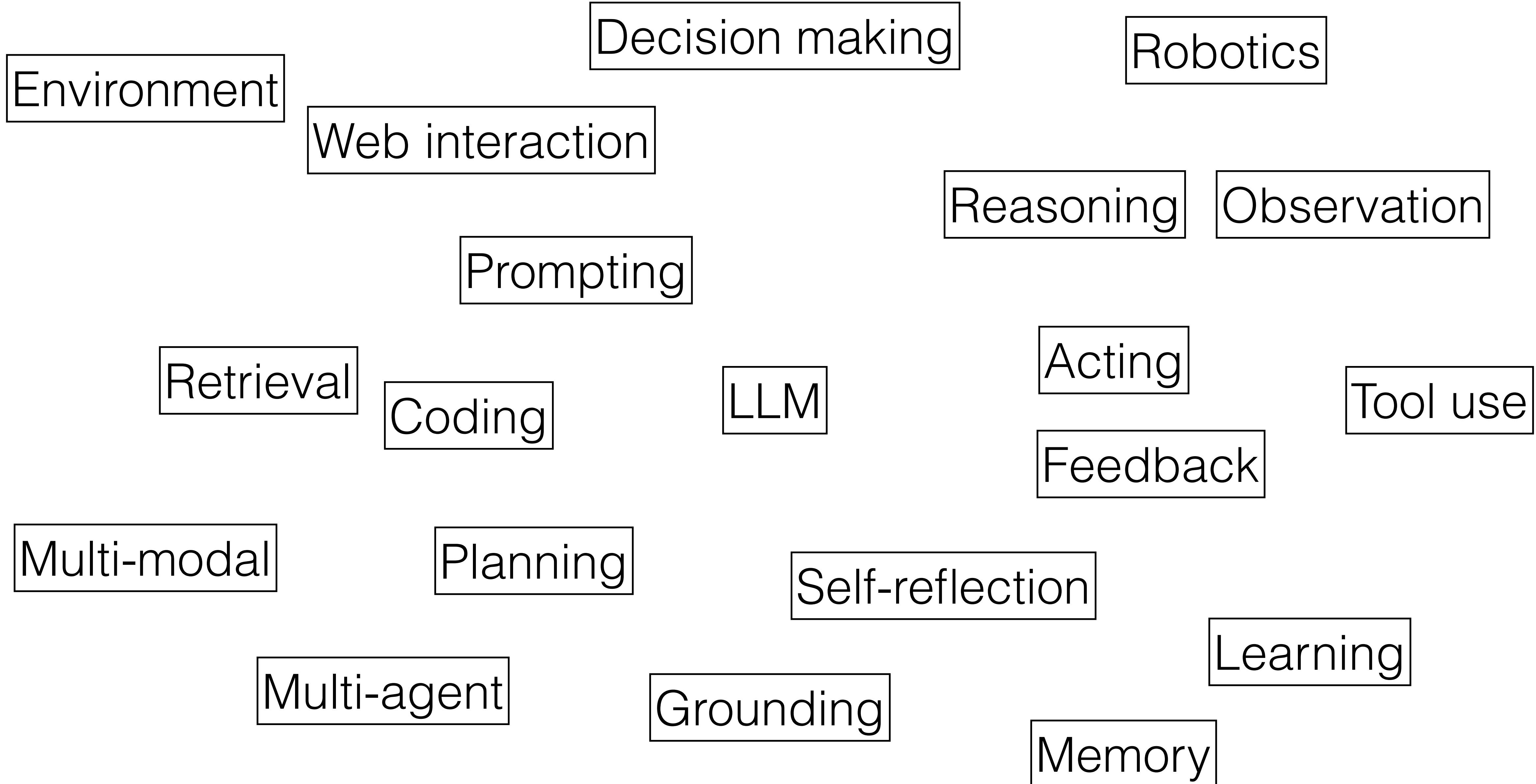
- Language agents: reasoning as internal actions
 - Reasoning and acting can be complementary (ReAct)
 - Reasoning and acting can be similarly planned (ToT)
- They address key limitations of LLMs and traditional agents
 - Ground LLMs with external feedback and internal control
 - Few-shot generalization to act in various new domains

3

Principled framework for language agents

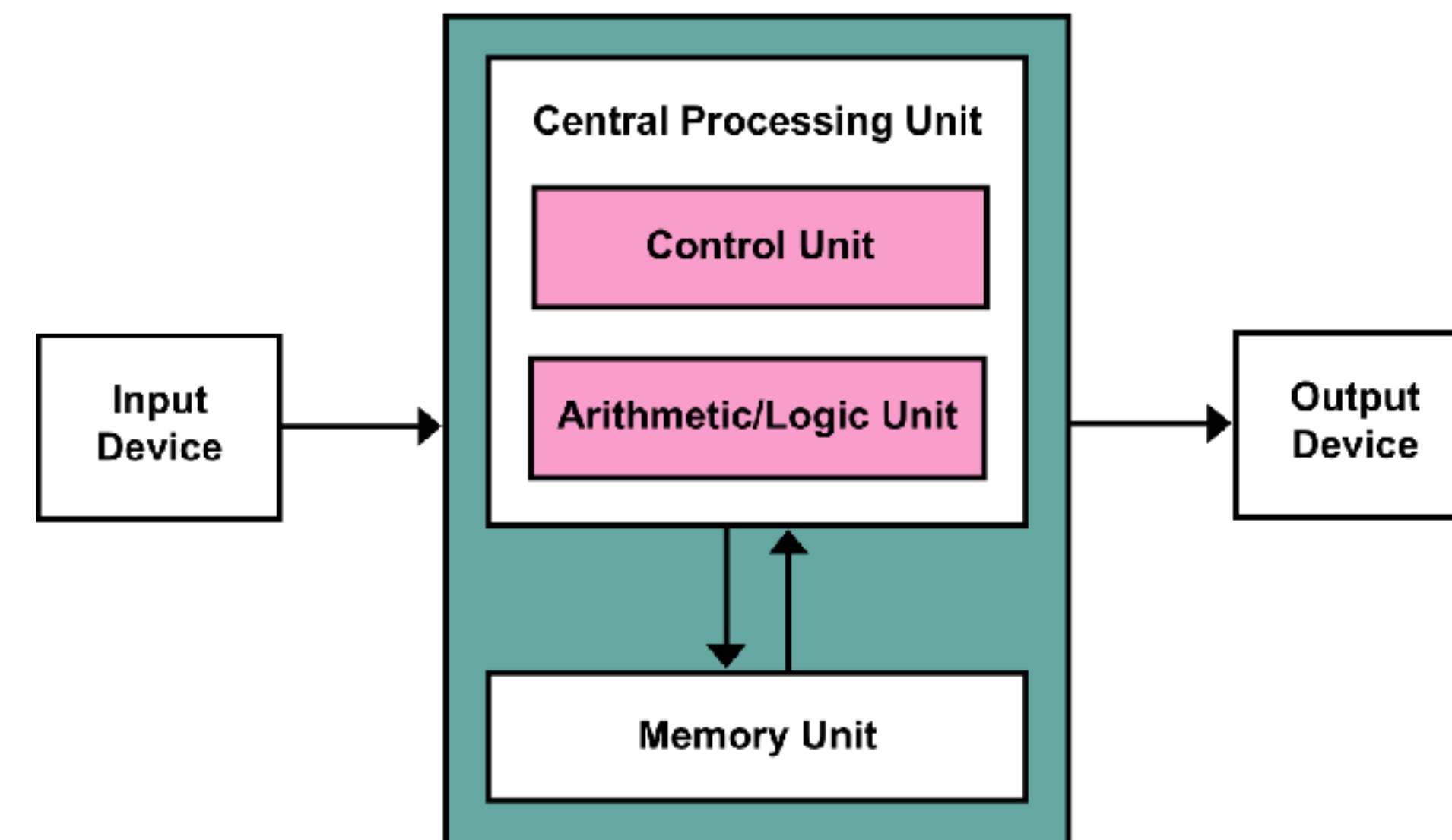
CoALA: Cognitive Architectures for Language Agents
Sumers*, Yao*, Narasimhan, Griffiths. TMLR 2024

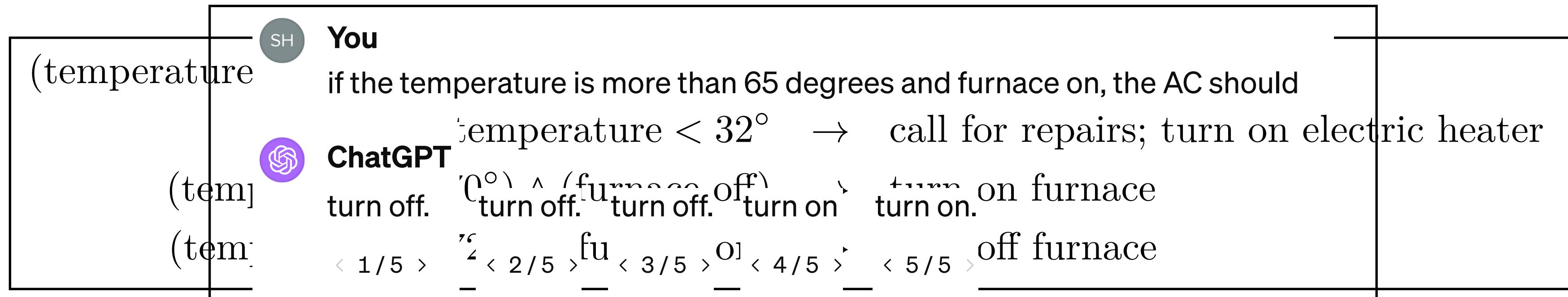




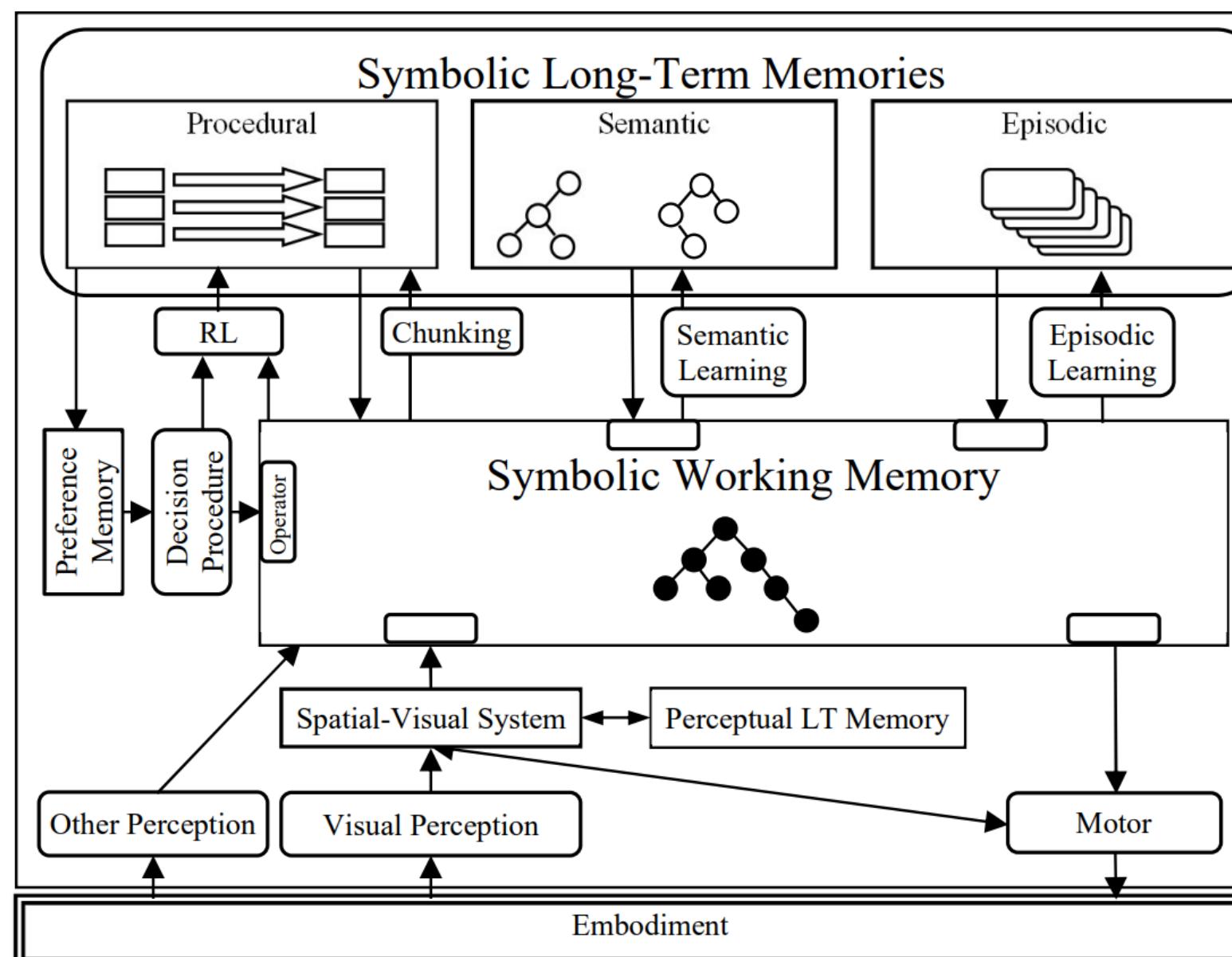
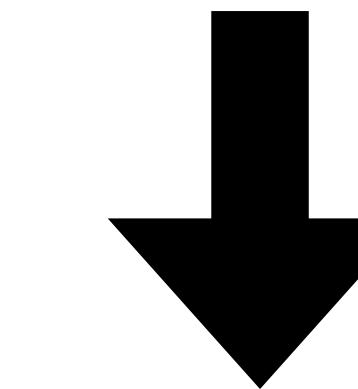
How do we make sense of various ~~LLM~~ systems? digital circuits

Where should the field be going?





Key production systems



Cognitive architectures:

frameworks to modularize and build complex symbolic AI agents, using cognitive inspirations



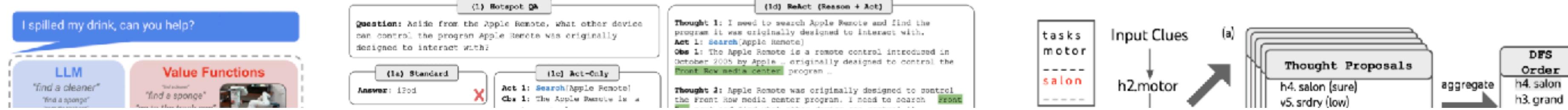
Soar cognitive architecture

Cognitive Architectures for Language Agents (CoALA)

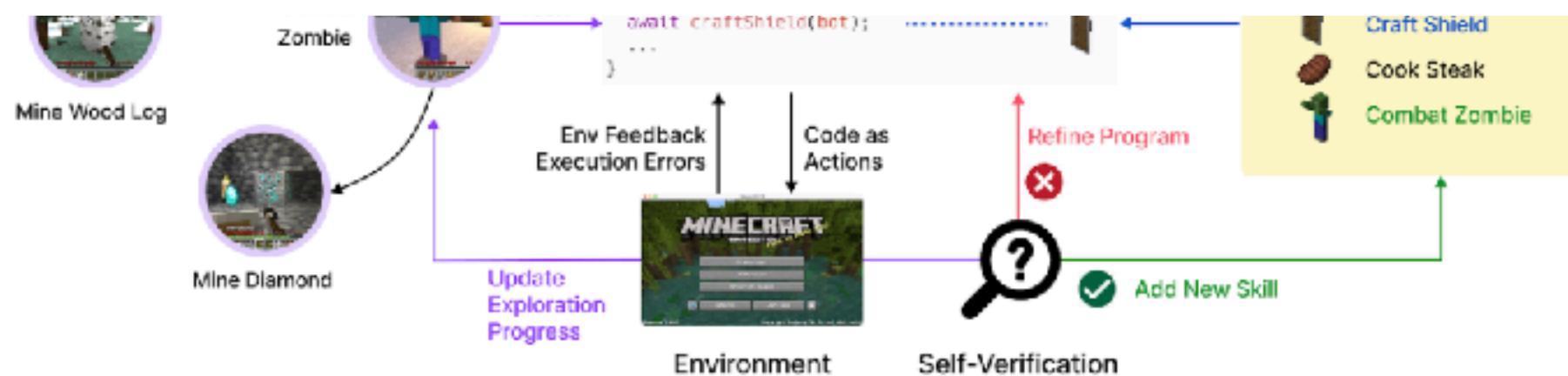
(+ gradients) (+ function) (+ knowledge) (+ task trajectory)

- **Memory:** short and long term
- **Action space:** internal and external
 - 1. Reasoning (update short-term memory)
 - 2. Retrieval (read long-term memory)
 - 3. Learning (write long-term memory)
 - 4. Grounding (update external world)
- **Decision making:** choose an action

Modularize and compare language agents



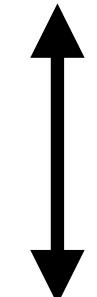
	Long-term Memory ⁵	External Grounding	Internal Actions	Decision Making
SayCan (Ahn et al., 2022)	-	physical	-	evaluate
ReAct (Yao et al., 2022b)	-	digital	reason	propose
Voyager (Wang et al., 2023a)	procedural	digital	reason/retrieve/learn	propose
Generative Agents (Park et al., 2023)	episodic/semantic	digital/agent	reason/retrieve/learn	propose
Tree of Thoughts (Yao et al., 2023)	-	digital ⁶	reason	propose, evaluate, select



Language agents

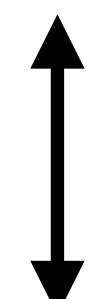
Benchmarks

[NeurIPS'22, NAACL'22, ACL'23,
NeurIPS'23, ICLR'24, ICLR'24]



Methods

[EMNLP'20, ICLR'23,
NeurIPS'23, NeurIPS'23]



Frameworks

[TMLR'24]



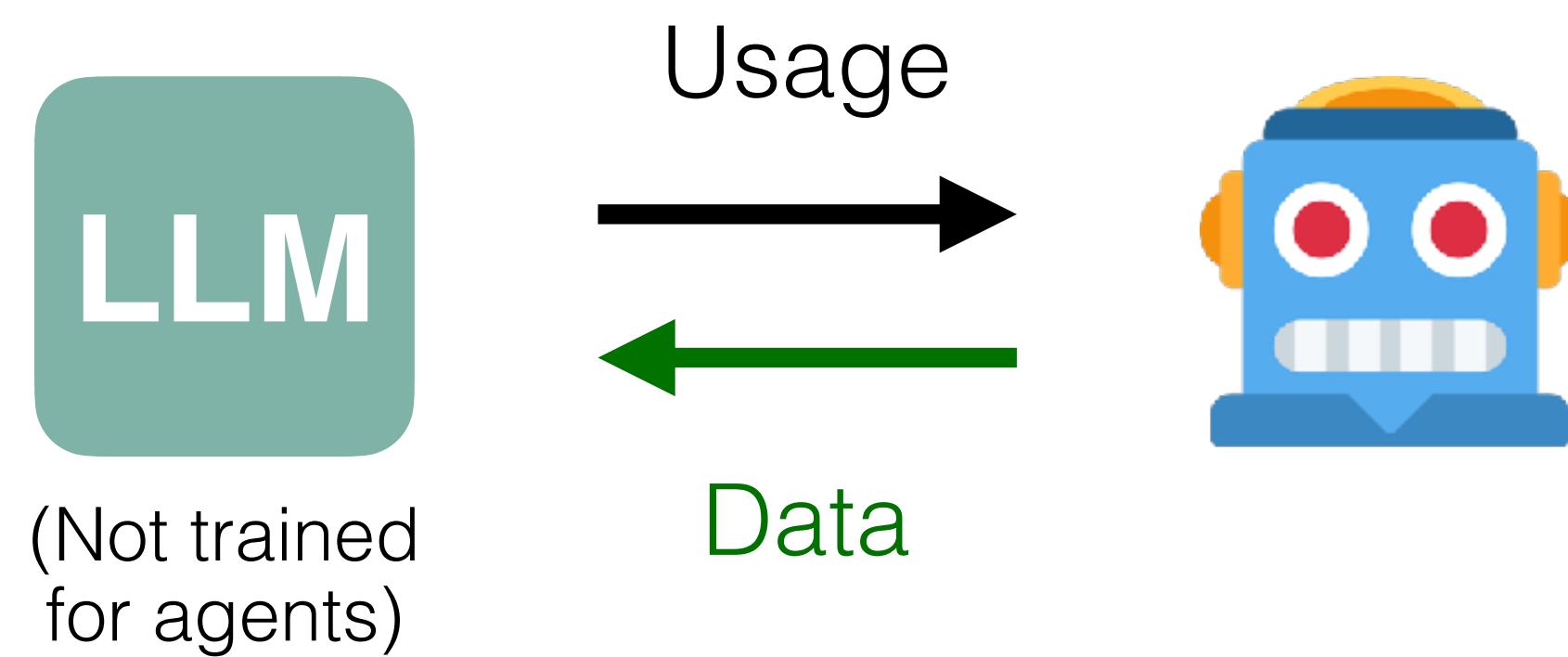
Other topics

- Computer vision and graphics [NeurIPS'18]
- Developmental psychology [NeurIPS'19, CogSci'20]
- Reinforcement learning and control [ICLR'22, CVPR'23]
- Human-computer interaction [DIS'24 submission]
- Information Retrieval [ACL'24 submission]
- Theory [ACL'21]



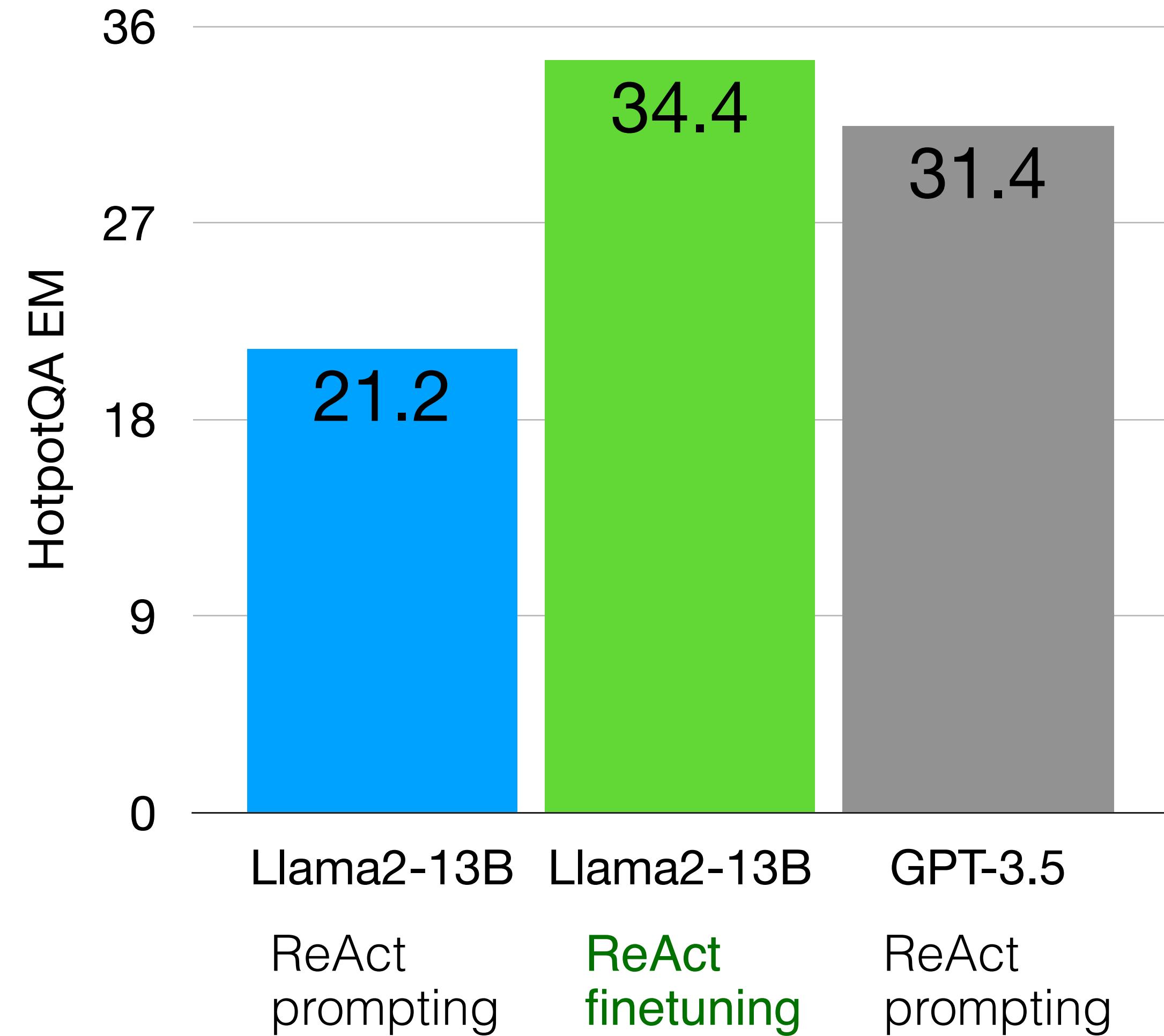
Future work

Future work #1: Train models for agents



Establish model-agent synergy:

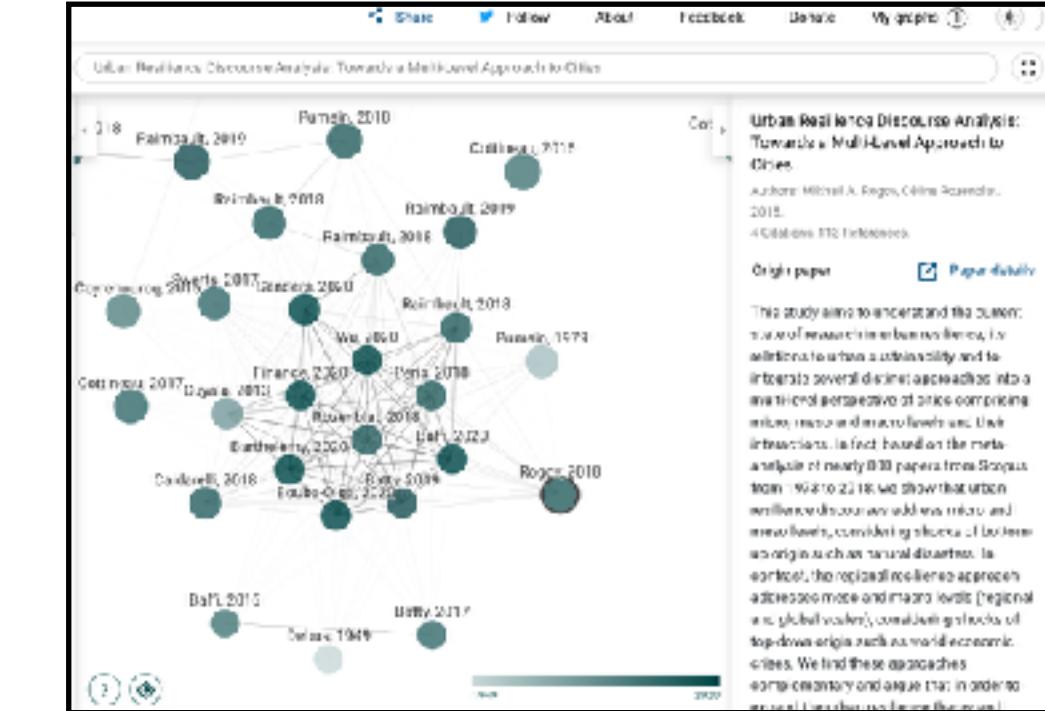
- Improve “agent capabilities” like planning, self-evaluation, calibration..
- Open-source agent backbone model
- Next trillion tokens for model training



Future work #2: Teach and discover knowledge

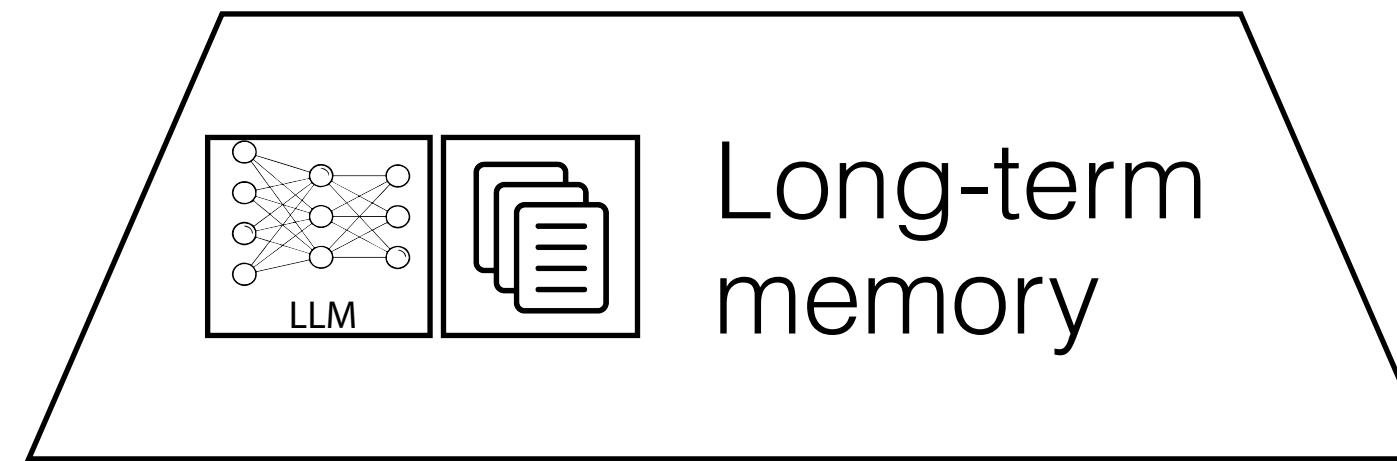


Personalized education

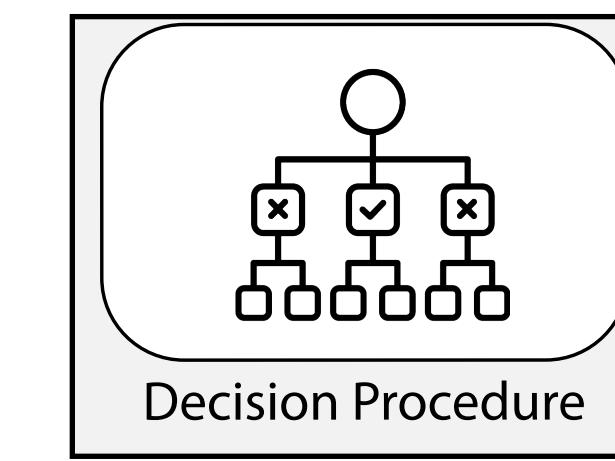


Scientific discovery

Through the lens of CoALA, these new applications require:



Flexible learning and retrieval



Intrinsic motivation (e.g., curiosity)

Parting thoughts

The most powerful neural networks ever built shouldn't just answer questions or draft emails.



They should be used to automate every aspect of our life, society, and science.

Thanks to my committee

- Danqi: thanks for your great students :)
- Tom: thanks for all the classical insights :)
- Sanjeev: thanks for the retreat and retweet :)
- Ben: thanks for making me feel old :)
- Tatsu: thanks for shaping my talk :)

Thanks to my advisor and friends

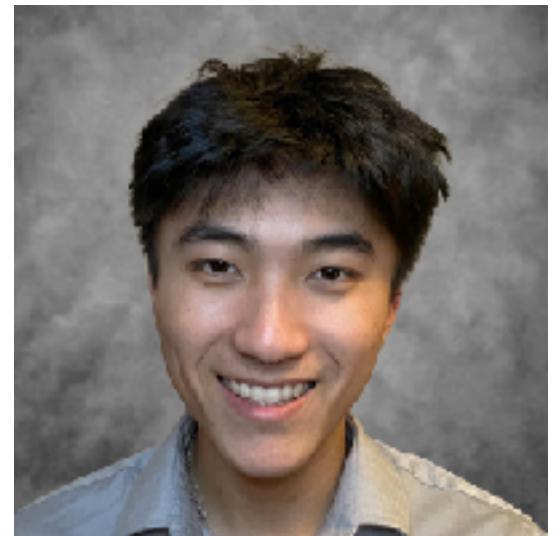


Thanks to my collaborators

Karthik Narasimhan



Noah Shinn



Carlos Jiminez



Tom Griffiths



Howard Chen



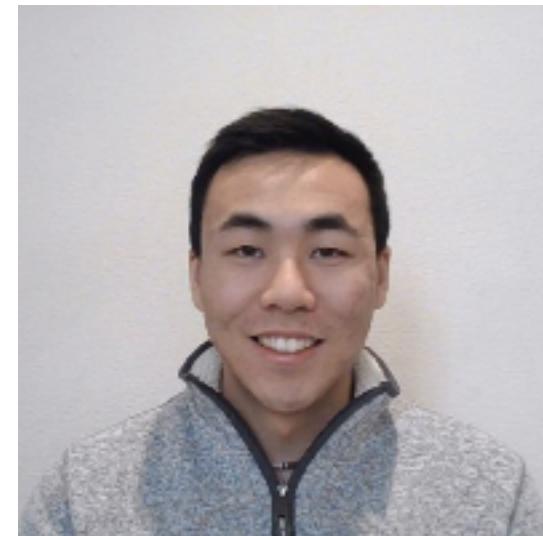
Ted Sumers



Yuan Cao



John Yang



Akshara Prabhakar

Alex Wettig

Ashwin Gopinath

Austin Wang

Baian Chen

Ben Shi

Binghui Peng

Chang Shu

Christos Papadimitriou

Chuang Gan

Dan Friedman

Dian Yu

Edward Berman

Ehsan Shareghi

Fandong Meng

Federico Cassano

Izhak Shafran

Jeffery Zhao

Jeffrey Stanton

Jens Tuyls

Jiangnan Li

Jie Zhou

Jing Li

Josh Tenenbaum

Kexin Pei

Matthew Hardy

Matthew Hausknecht

Michael Tang

Mingyu Ding

Mo Yu

Nan Du

Nigel Collier

Ofir Press

Ping Luo

Rohan Rao

Runzhe Yang

Sham Kakade

Tao Yu

Tom McCoy

Wenjie Pang

Xiangyang Mou

Xiaochen Zhou

Yang Zhang

Yao Mu

Yi Gu

Yisi Sang

Yuqian Sun

Zhiyong Wu

Zhou Xiao