ReAct Learning Survey

ReAct: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Google Research & Princeton University

https://arxiv.org/abs/2210.03629
arxiv.org

💡 **Abstract**

Although large language models (LLMs) have shown impressive performance in tasks such as Natural-language Understanding and interactive decision-making, their reasoning (such as thought chain prompts) and action (such as action plan generation) capabilities are mainly studied as separate topics. In this paper, we explore the use of LLMs to generate inference trajectories and task-specific actions in an interleaved manner, thereby achieving greater synergy between the two: inference trajectories help the model induce, track, and update action plans and handle exceptions, while actions allow it to interact with external sources (such as Knowledge Base or environment) and collect additional information.

We apply a method called ReAct to multiple language understanding and decision-making tasks, demonstrating its effectiveness compared to SOTA base lines and improving human interpretability and credibility. Specifically, on the HotpotQA and Fever datasets, ReAct overcomes the common problems of illusion and error propagation in thought chain reasoning by interacting with the simple Wiki Lingo API , and generates human-like task-solving trajectories that are more interpretable than base lines without reasoning trajectories. In addition, on two interactive decision benchmarks (ALFWorld and WebShop), ReAct only requires one or two contextual examples for prompting, and its success rates are 34% and 10% better than those of imitation and reinforcement learning methods, respectively.

Motivations and main contributions

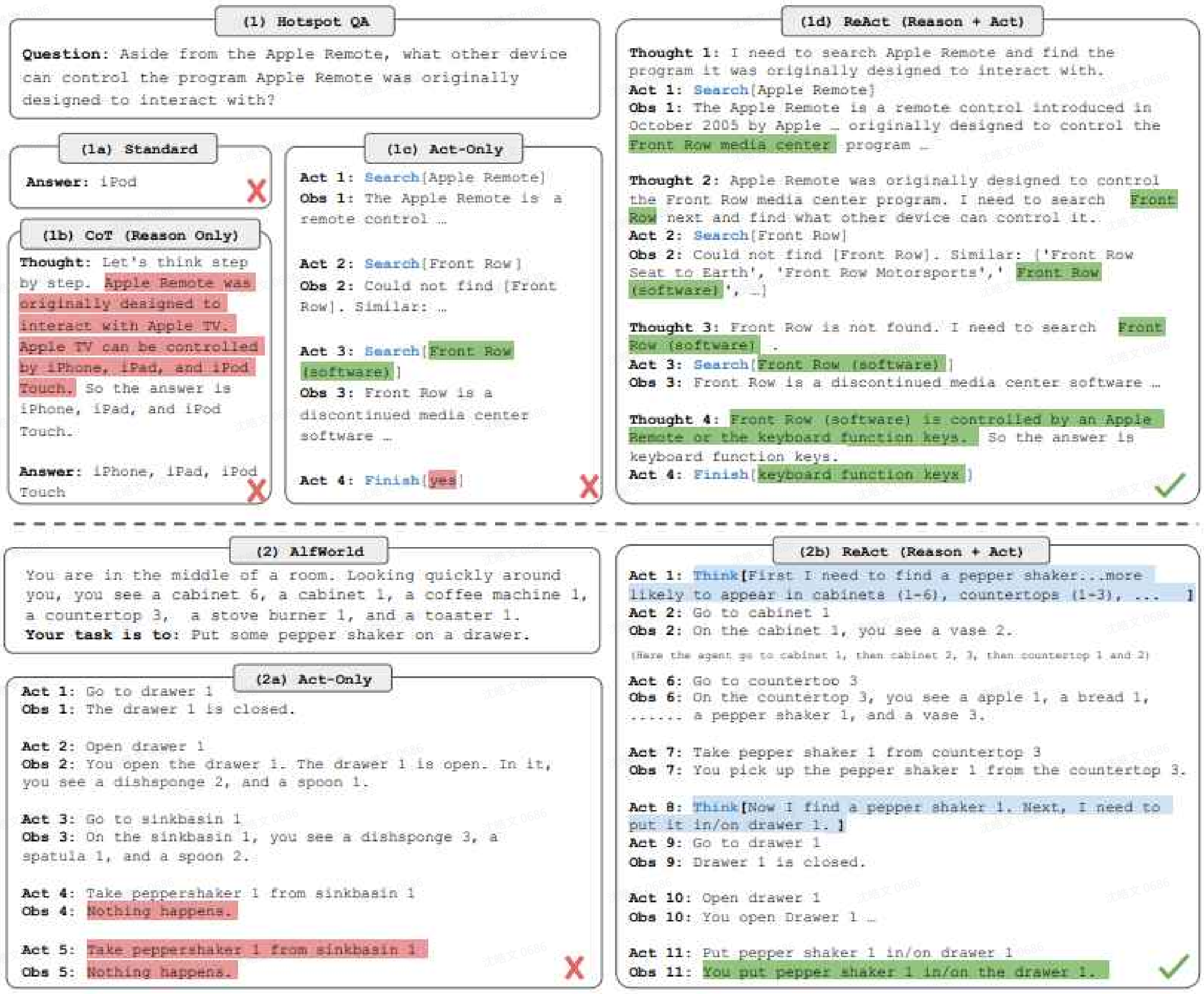
A unique feature of human intelligence is the ability to **seamlessly combine task-oriented actions with verbal reasoning (or inner speech)** The close synergy between "action" and "reasoning" enables humans to quickly learn new tasks and perform robust decisions or reasoning, even in previously unseen situations or when faced with information uncertainty.

> For example, in the example of cooking in the kitchen: between any two specific actions, we may use language to reason in order to track progress ("Since all the ingredients are cut, I should heat the water in the kettle"), deal with abnormal situations or adjust the plan according to the actual situation ("The salt is used up, use soy sauce and pepper instead"), and realize when external information is needed ("How to prepare the dough? Let me search online"). We can also take actions such as opening the recipe to read the recipe, opening the refrigerator, checking the ingredients, etc. to support reasoning and answer questions such as "What dish can I make now?"

Because of the emergent power of large models, the chain of thought (COT) method can be used to get answers to tasks such as algorithms, common sense, and symbolic reasoning, but this chain of thought method is a static black box, and the **model uses its own internal representation to generate ideas, so it may lead to problems such as factual illusions and error propagation in the reasoning process** (example 1b in the figure below).

At present, there are also many works that use large models for planning and acting in interactive environments, but these methods do not consider using large models to reason about higher-order tasks.

So this article **proposes that the ReAct framework combines reasoning and action with language models. It prompts LLM to generate task-related verbal reasoning trajectories and actions in an interactive way. This enables the model to perform dynamic reasoning to create, maintain, and adjust high-level action plans (reasons for action), while interacting with external environments (such as Wiki Lingo) to incorporate additional information into reasoning (behavior-to-reasoning).**



```
(1) Hotspot QA
Question: Aside from the Apple Remote, what other device
can control the program Apple Remote was originally
designed to interact with?
```

```
(1a) Standard
Answer: iPod                                                    ✗
```

```
(1b) CoT (Reason Only)
Thought: Let's think step
by step. Apple Remote was
originally designed to
interact with Apple TV.
Apple TV can be controlled
by iPhone, iPad, and iPod
Touch. So the answer is
iPhone, iPad, and iPod
Touch.

Answer: iPhone, iPad, iPod
Touch                                                           ✗
```

```
(1c) Act-Only
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a
remote control …

Act 2: Search[Front Row]
Obs 2: Could not find [Front
Row]. Similar: …

Act 3: Search[Front Row
(software)]
Obs 3: Front Row is a
discontinued media center
software …

Act 4: Finish[yes]                                              ✗
```

```
(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 ]                          ✓
```

```
(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 dishsponge 2, and a spoon 1.

Act 3: Go to sinkbasin 1
Obs 3: On the sinkbasin 1, you see a dishsponge 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.
(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.             ✓
```

Figure 1: (1) Comparison of 4 prompting methods, (a) Standard, (b) Chain-of-thought (CoT, Reason Only), (c) Act-only, and (d) ReAct (Reason+Act), solving a HotpotQA (Yang et al., 2018) question. (2) Comparison of (a) Act-only and (b) ReAct prompting to solve an AlfWorld (Shridhar et al., 2020b) game. In both domains, we omit in-context examples in the prompt, and only show task solving trajectories generated by the model (Act, Thought) and the environment (Obs).

Main contributions:

- Introduced ReAct, a novel prompt-based paradigm that enables collaborative reasoning and action in language models to solve general tasks.
- A large number of experiments have been conducted on different benchmarks to demonstrate the advantages of ReAct in a feed-shot learning setting compared to methods that perform inference or action generation alone.
- Demonstrate systematic ablation and analysis experiments to understand the importance of reasoning in action and interaction tasks.

- The limitations of ReAct under prompt settings (limited support for reasoning and action behavior) were analyzed, and preliminary fine-tuning experiments were conducted, showing the potential of improving ReAct through additional training data. Applying ReAct to more tasks and combining it with reinforcement learning can further unleash the potential of large language models.

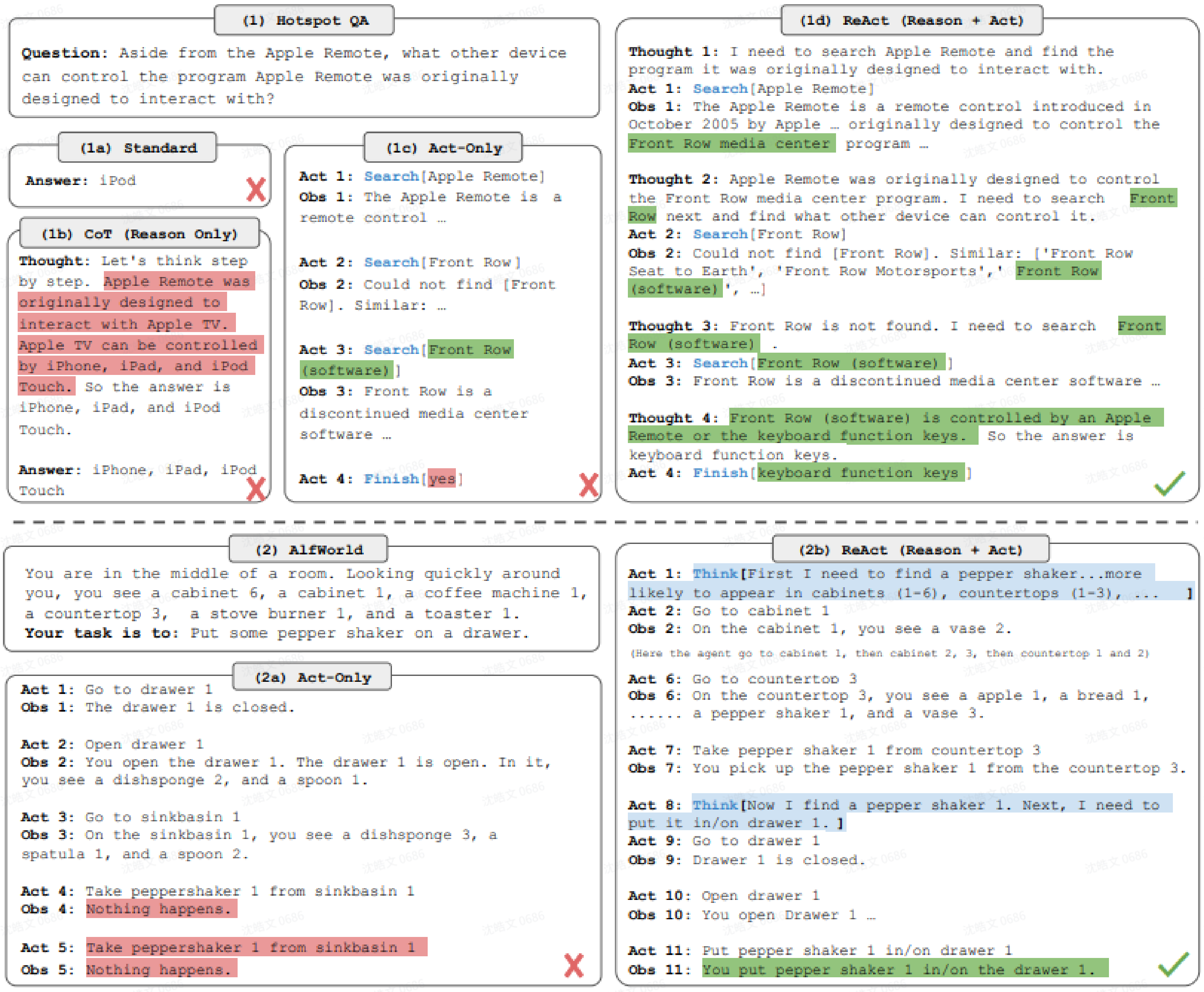
REACT : SYNERGIZING *RE*ASONING + *ACT*ING

对于一个与环境交互来解决任务的agent，在时刻t，一个agent从环境中收到了观察$o_t \in \mathcal{O}$，并且遵循策略$\pi(a_t | c_t)$采取动作$a_t \in \mathcal{A}$，其中$c_t = (o_1, a_1, \cdots, o_{t-1}, a_{t-1}, o_t)$是agent的上下文(context)。当$c_t \rightarrow a_t$的映射是隐形的并且需要大量计算时，学习一个策略是有挑战的。 比如在上图的(1c)和(2a)都没有执行最终的正确动作。

而ReAct的思想很简单：将agent的动作空间增强变成$\hat{\mathcal{A}} = \mathcal{A} \cup \mathcal{L}$，其中的$\mathcal{L}$是语言空间。一个属于语言空间的动作$\hat{a_t} \in \mathcal{L}$，被称为thought 或reasoning trace，它不影响外部环境，因此也不会有观察反馈。一个thought $\hat{a_t}$ 意在对现有上下文c_t推理来组合出有用的信息，并更新上下文$c_{t+1} = (c_t, \hat{a_t})$以支持将来的推理或动作。

The main types of thoughts include:

- Split task objectives and create action plans (2b, Act 1; 1d, Thought 1)
- Inject common sense knowledge related to solving tasks (2b, Act 1).
- Extracting important information from observations (1d, Thought2,4)
- Tracking progress and action plans (2b, Act8)
- Handling exceptions and adjusting action plans (1d, Thought 3), etc



```
(1) Hotspot QA
Question: Aside from the Apple Remote, what other device
can control the program Apple Remote was originally
designed to interact with?

(1a) Standard
Answer: iPod

(1b) CoT (Reason Only)
Thought: Let's think step
by step. Apple Remote was
originally designed to
interact with Apple TV.
Apple TV can be controlled
by iPhone, iPad, and iPod
Touch. So the answer is
iPhone, iPad, and iPod
Touch.

Answer: iPhone, iPad, iPod
Touch

(1c) Act-Only
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a
remote control …

Act 2: Search[Front Row]
Obs 2: Could not find [Front
Row]. Similar: …

Act 3: Search[Front Row
(software)]
Obs 3: Front Row is a
discontinued media center
software …

Act 4: Finish[yes]

(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 ]
```

```
(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 dishsponge 2, and a spoon 1.

Act 3: Go to sinkbasin 1
Obs 3: On the sinkbasin 1, you see a dishsponge 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.
(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.
```

Because the language space L is infinite, learning from this reinforcement space is difficult and requires strong language prior knowledge. In this article, the author mainly uses a language model: PaLM-540B, and uses the feed-shot context example prompt model to generate domain-related actions or thoughts when solving tasks, such as (1d) and (2b) in the figure above.

Example

Each context example is an artificially generated trajectory of actions, thoughts, and environmental observations used to solve the task instance. It is shown in Appendix C of the paper. For example, the context example for HOTPOTQA is as follows:

```
"""
Question      What is the elevation range for the area that the eastern sector
of the                      Colorado orogeny extends into?
Thought 1     I need to search Colorado orogeny, find the area that the
eastern sector of                     the Colorado orogeny extends into, then find
the elevation range of the                      area.
Action 1       Search[Colorado orogeny]
```

```
Observation 1  The Colorado orogeny was an episode of mountain building (an
orogeny) in                         Colorado and surrounding areas.
Thought 2      It does not mention the eastern sector. So I need to look up
eastern sector. Action 2       Lookup[eastern sector]
Observation 2  (Result 1 / 1) The eastern sector extends into the High Plains
and is called                     the Central Plains orogeny.
Thought 3      The eastern sector of Colorado orogeny extends into the High
Plains. So I                            need to search High Plains and find its
elevation range.
Action 3       Search[High Plains]
Observation 3    High Plains refers to one of two distinct land regions
Thought 4       I need to instead search High Plains (United States).
Action 4       Search[High Plains (United States)]
Observation 4   The High Plains are a subregion of the Great Plains. From east
to west, the                      High Plains rise in elevation from around 1,800
to 7,000 ft (550 to 2,130                     m).[3]
Thought 5       High Plains rise in elevation from around 1,800 to 7,000 ft,
so the answer                       is 1,800 to 7,000 ft.
Action 5       Finish[1,800 to 7,000 ft]
"""

```

For **the more important task of inference** (**(1) in the figure above**), the authors alternately generate thoughts and actions, so the task trajectory consists of multiple "thought-action-observation" steps.

For **tasks involving many actions** (**(2) in the figure above**), thoughts only need to appear sparsely in the most relevant positions in the trajectory, so the authors let the language model decide for itself that thoughts and actions occur asynchronously.

Because ReAct includes decision-making and reasoning abilities, the authors believe it has several unique features:

- Intuitive and easy to design: Human annotators can write down their thoughts when taking action, with no special formatting requirements.
- General and flexible: Because there is no limit to the format and thought space, it is applicable but not limited to various tasks: QA, fact verification, word games, total page browsing, etc.
- Performant and robust: **With only 1-6 contextual examples, ReAct shows strong generalization to new examples and performs better in different domains than inference-only or action-only base lines.**

- Human aligned and controllable: **The trajectory of ReAct is interpretable, and people can easily check the accuracy of reasoning and facts. And in the process, people can also edit thoughts to control and correct agent behavior**

KNOWLEDGE-INTENSIVE REASONING TASKS

Illustrating the knowledge-intensive reasoning task with multiple questions and answers and fact checking, in (1d) of the figure above, the synergy of reasoning and action is demonstrated by interacting with the Wikipedia API . ReAct can retrieve information to support reasoning and reason to decide what content to retrieve.

- Domain: Two datasets: HotPotQA multi-hop question answering dataset and FEVER fact confirmation dataset, considering only questions as input for two tasks.
- Action Space: Using the Wikipedia API to support interactive search, there are three types of actions:

1. search [entity] returns the first 5 sentences corresponding to the entity wiki page, or the 5 most similar entities suggested by the Wikipedia search engine.

2. lookup [string], find the next sentence on the page that includes the string, similar to the Ctrl + F shortcut in browsers.

3. Finish [answer], end the current task with the answer.

- Method

ReAct Promping: Randomly select samples from the training set and manually annotate them as few-shot examples. HotpotQA has 6 examples and Fever has 3 examples. The authors found that more examples will not improve performance. As shown in 1d of the figure above, more examples are in Appendix C of the paper. Note that the authors attempted to write examples.

1. Problem decomposition ("I need to search x, find y, then find z")
2. Extract information from Wikipedia pages ("x was started in 1844", "The paragraph does not tell x")
3. Perform common sense reasoning ("x is not y, so z must instead be...")
4. Arithmetic Reasoning ("1844 < 1989")
5. Redefine search content ("maybe I can search/look up x instead")
6. Composite answer ("... so the answer is x").
7. Base line: Compare ReAct with the following base lines:

Baseline comparison

- Standard Promping (Standard): ReAct removes all thoughts, actions, and observations, as shown in 1a in the image above.
- Chain-of-thought prompting (CoT): This is the popular thinking chain, which removes actions and observations from ReAct and serves as the base line for only reasoning, as shown in 1b in the picture above. In addition, the self-consistency baseline (CoT-SC) is compared, which takes temperature = 0.7 and samples 21 CoT trajectories to adopt most answers.
- Action-only prompt (Act), remove the thought in React, like 1c in the picture above.
- Combining internal and external knowledge: Because ReAct is more fact-based, while CoT is more accurate in reasoning structure, but prone to hallucinating facts or thoughts, ReAct and CoT-SC are merged, based on the following heuristic rules to let the model decide what to switch to another method:

  > CoT-SC →\ rightarrow → ReAct: When the majority answer of n CoT-SC samples is less than n/2, switch to ReAct.

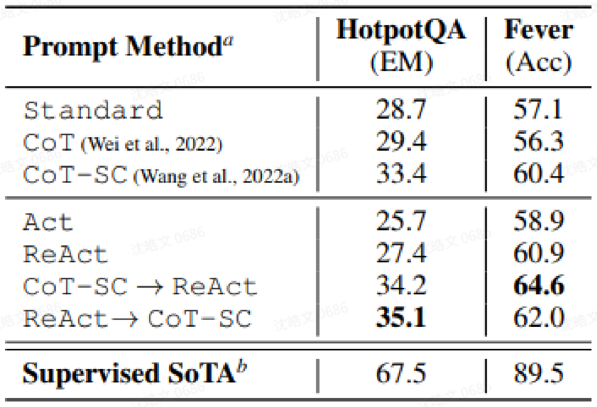
  > ReAct →\ rightarrow → CoT-SC: When ReAct cannot return an answer within the specified number of steps, switch to CoT-SC and set HotpotQA to 7 steps and Fever to 5 steps.

Finetuning

Use the boostraping method to generate 3000 correct React trajectories and fine-tune the language model PaLM-8/62B.

Findings and observations

- ReAct works better than Act: Table 1 in the picture below shows that ReAcT works better than Act on both tasks, especially when synthesizing the final answer (as shown in 1c-d in the picture above). Figure 3 in the later picture also shows that reasoning after Fine-tuning is beneficial.



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

Table 1: PaLM-540B prompting results on HotpotQA and Fever.

[a] HotpotQA EM is 27.1, 28.9, 33.8 for Standard, CoT, CoT-SC in Wang et al. (2022b).

[b] (Zhu et al., 2021; Lewis et al., 2020)

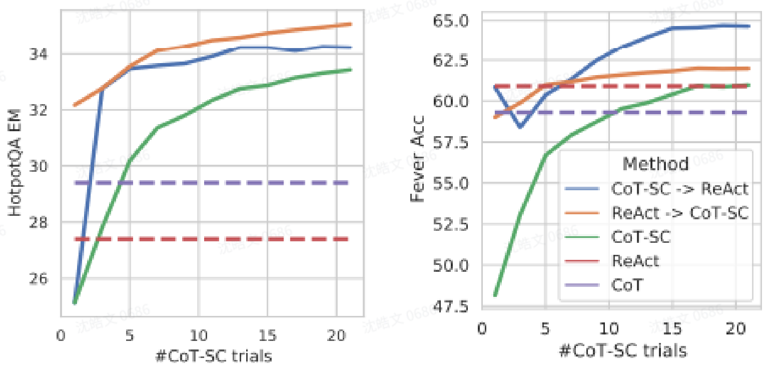


Figure 2: PaLM-540B prompting results with respect to number of CoT-SC samples used.

ReAct vs. CoT

The above image shows that ReAct is better than CoT on the Fever dataset and worse than CoT on HotpotQA. In order to better understand the behavioral differences between ReAct and CoT on HotpotQA, 50 correct and incorrect trajectories (a total of 200) were randomly sampled from each of ReAct and COT, and then their success and error patterns were manually marked, as shown in Table 2 of the following figure. Some key results are as follows, and detailed examples are given in Appendix E.1 of the paper.

- **The illusion of CoT is a serious problem, with higher false positives (14% vs 6%) compared to ReAct in success mode, and causing 56% of errors. ReAct, on the other hand, is more fact-based and credible due to the existence of an external Knowledge Base.**
- **The structure of alternating reasoning, action, and observation steps in React makes it more reliable, but it also limits the flexibility of reasoning steps, so React has more reasoning errors compared to CoT. ReAct has an obvious tendency to repeat previous thoughts and actions, which the author summarizes as "reasoning errors".**
- **For React, it is crucial to successfully retrieve useful knowledge from searches, and useless searches cause 23% of error cases.**

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

Table 2: Types of success and failure modes of ReAct and CoT on HotpotQA, as well as their percentages in randomly selected examples studied by human.

ReAct + CoT-SC is a better prompt method. In Table 1 of the above picture, ReAct →\ rightarrow → CoT-SC and CoT-SC →\ rightarrow → ReAct are the best prompt methods on HotpotQA and Fever, respectively. Figure 2 above shows the effect of different sample sizes in CoT-SC. This result shows the value of appropriately combining internal and external knowledge in inference tasks.

Prompt tuning vs. fine-tuning

The ReAct method performs best in fine-tuning , as shown in Figure 3 below, which shows a comparison of four methods for prompting and finetuning on HotpotQA. The authors believe that finetuning with a larger annotated dataset may be more effective than state-of-the-art supervised methods.

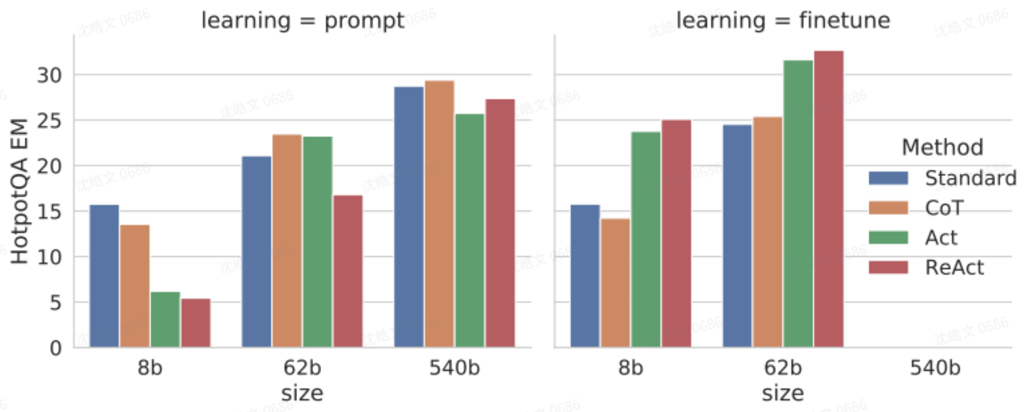
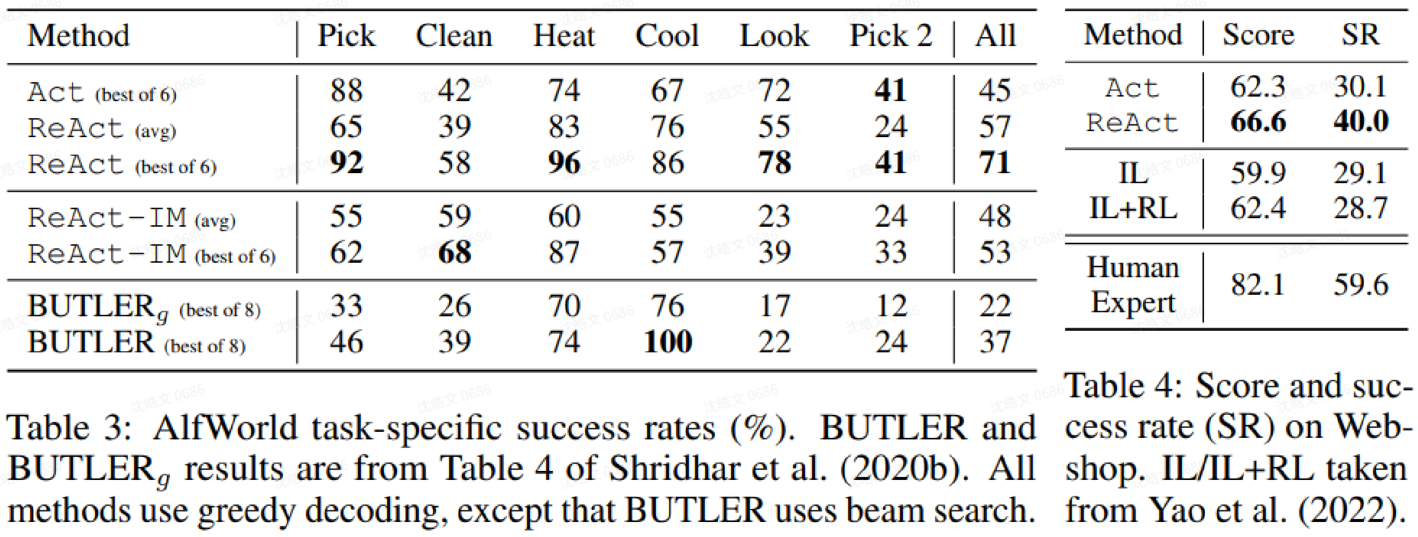


Figure 3: Scaling results for prompting and finetuning on HotPotQA with ReAct (ours) and baselines.

Decision-making tasks

The effect of ReAct was tested on two datasets, ALFWorld and WebShop.

As shown in tables 3 and 4, ReAct outperforms Act on both datasets.



Method	Pick	Clean	Heat	Cool	Look	Pick 2	All
Act (best of 6)	88	42	74	67	72	**41**	45
ReAct (avg)	65	39	83	76	55	24	57
ReAct (best of 6)	**92**	58	**96**	86	**78**	**41**	**71**
ReAct-IM (avg)	55	59	60	55	23	24	48
ReAct-IM (best of 6)	62	**68**	87	57	39	33	53
BUTLER$_g$ (best of 8)	33	26	70	76	17	12	22
BUTLER (best of 8)	46	39	74	**100**	22	24	37

Table 3: AlfWorld task-specific success rates (%). BUTLER and BUTLER$_g$ results are from Table 4 of Shridhar et al. (2020b). All methods use greedy decoding, except that BUTLER uses beam search.

Method	Score	SR
Act	62.3	30.1
ReAct	**66.6**	**40.0**
IL	59.9	29.1
IL+RL	62.4	28.7
Human Expert	82.1	59.6

Table 4: Score and success rate (SR) on Webshop. IL/IL+RL taken from Yao et al. (2022).

Conclusion

As a method proposed in 2022, ReAct actually provides ideas and method paths for many subsequent methods, including tool tuning and agent tuning. At the same time, there are currently many new optimizations for ReAct methods, which can be further read and understood later, but it still inherits the core idea of ReAct.

Reference

- React
- https://blog.csdn.net/beingstrong/article/details/132123996