Prospector: Improving LLM Agents with Self-Asking and Trajectory Ranking

Byoungjip Kim[1], **Youngsoo Jang**[1], **Lajanugen Logeswaran**[1],
Geon-Hyeong Kim[1], **Yujin Kim**[1], **Honglak Lee**[1], **Moontae Lee**[1,2]
[1] LG AI Research
[2] University of Illinois Chicago

Abstract

Large language models (LLMs) have shown the ability to solve complex decision-making tasks beyond the natural language processing tasks. Current LLM agents such as ReAct can solve interactive decision-making tasks by imitating the few-shot demonstrations given in the prompt. The LLM agents based on few-shot in-context learning (ICL) achieve surprisingly high performance without training. Despite the simplicity and generalizability, the ICL-based approaches lack optimizing trajectories based on the reward from an environment. In this paper, we introduce Prospector, a LLM agent that consists of two complementary LLMs such as the LLM Actor and LLM Critic. To elicit more proper actions from the LLM Actor, we provide AskAct prompting that interleaves additional self-asking steps in the few-shot demonstrations. Furthermore, to take advantages of the stochasticity of LLMs, we provide Trajectory Ranking in which the LLM Actor generates diverse (creative) trajectories at high temperature and the LLM Critic selects the most rewarding trajectory by predicting the expected total reward of each trajectory. On the representative decision-making benchmark environments such as ALFWorld and WebShop, we empirically demonstrate that Prospector can considerably increase the success rate of given tasks, while outperforming recent advancements such as ReAct and Reflexion.

1 Introduction

Although large language models (LLMs) [7, 17, 3, 18, 23] have recently shown remarkable success, it is still challenging to solve complex interactive decision-making problems that require reasoning and planning abilities [26, 14]. Fine-tuning the LLMs using reinforcement learning (RL) [15, 8, 1] is one of the representative approaches to improve the reasoning and planning abilities of LLM agents. However, RL-based LLM fine-tuning methods require separate expensive training costs for each task, which are unsuitable for training LLMs with an enormous number of parameters. Recently, few-shot prompting approaches (e.g., chain-of-thought [26]) have achieved significant improvement in various natural language processing tasks [30, 20, 25], and are considered a promising direction because they do not require any fine-tuning costs of LLMs.

ReAct [30] is one of the notable few-shot prompting approaches, which prompts LLMs to generate both verbal reasoning traces and actions in an interleaved manner. This allows the LLM agent to perform dynamic reasoning and high-level planning. However, this few-shot prompting alone may not be sufficient to generate optimal trajectories since it does not consider the task feedback signal (i.e. reward) from an environment. To leverage the task feedback signal, [20] presents Reflexion which converts the reward from the environment into verbal feedback and then uses this self-reflective feedback as additional context in the next episode. However, since Reflexion explores and reflects on task feedback signals in subsequent trials, it cannot efficiently search the diverse trajectories.

Foundation Models for Decision Making Workshop at NeurIPS 2023.

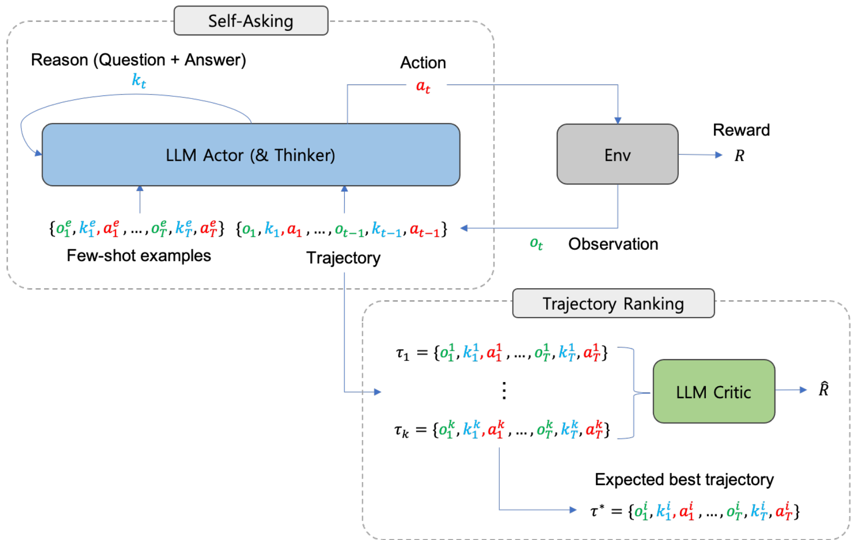
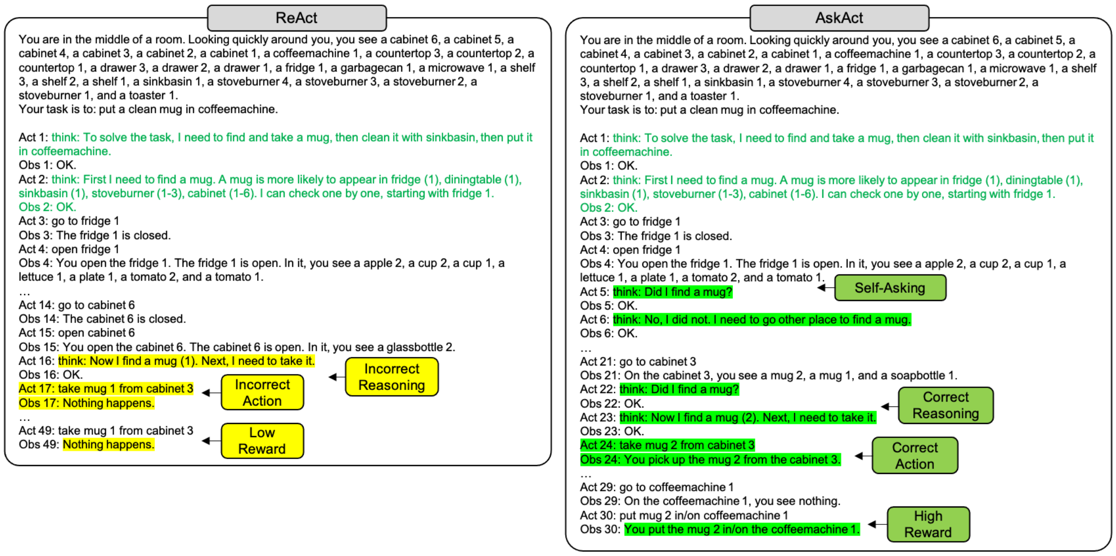


Figure 1: **Overview of Prospector.** Prospector is a LLM agent that consists of two complementary LLMs such as LLM Actor and LLM Critic for solving complex interactive decision-making tasks. The LLM Actor generates actions based on the few-shot demonstrations and the history of observations and actions. To elicit the more proper actions, Prospector interleaves self-asking steps in the few-shot demonstrations. Furthermore, Prospector takes the advantages of the stochasticity of LLMs, and generates diverse trajectories at high temperature. Then, the LLM Critic selects the most rewarding trajectory by predicting the expected reward. The LLM critic can operate either in the few-shot ICL mode or fine-tuning mode.

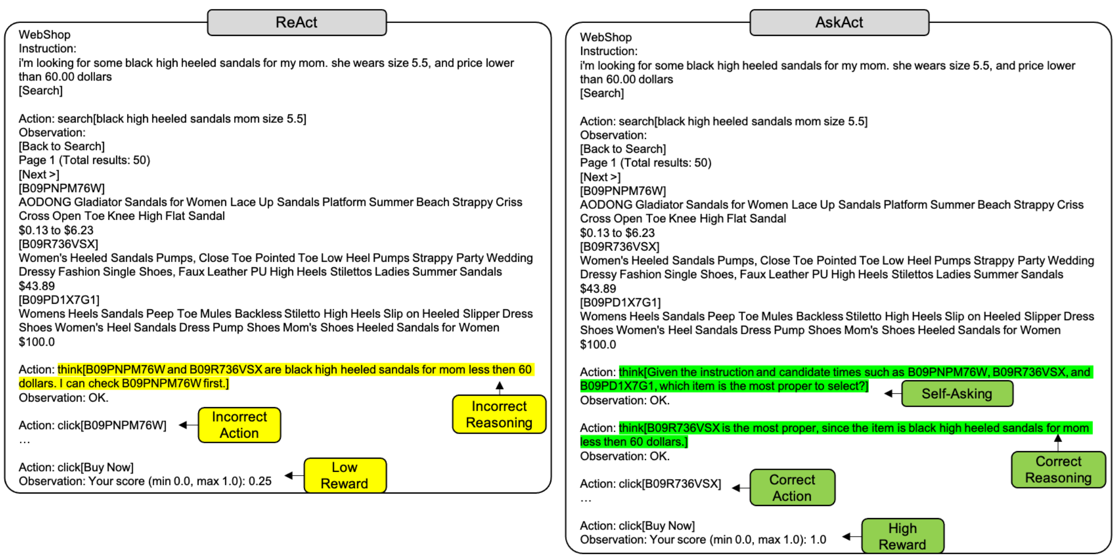
To address the aforementioned limitations, we introduce **Prospector**, a powerful LLM agent for decision-making tasks, which reinforces the ability to generate strategic actions without updating the model parameters. Prospector consists of two complementary LLMs such as LLM Actor and LLM Critic. To improve the baseline performance of the LLM Actor, we propose *AskAct prompting* that introduces additional self-asking steps in the few-shot demonstrations. This allows the LLM Actor to collect the information necessary for decision-making on its own, and generate more strategic actions based on it. Then, to take advantage of the stochastic generation of LLMs, we provide *Trajectory Ranking* in which the LLM Actor generates diverse trajectories with a number of trials and then selects the most rewarding trajectory as the final action. Prospector can achieve high performance through the synergy of 1) *AskAct prompting*, which can generate promising trajectory candidates, and 2) *Trajectory Ranking*, which can select the most rewarding trajectory from the candidates. In the experiments, we demonstrate that Prospector outperforms recent advancements such as ReAct and Reflexion on the representative language-based interactive decision-making benchmarks including ALFWorld [22] and WebShop [28].

The contributions of this paper can be summarized as follows:

- We introduce **Prospector**, a LLM agent that consists of two complementary LLMs such as *LLM Actor* and *LLM Critic* to improve the performance of ICL-based LLM agents (see Figure 1).
- We provide *AskAct prompting* that introduces additional self-asking steps in a ReAct prompt to improve the performance of LLM Actor (see Figure 2). Also, we provide *Trajectory Ranking* that selects the most rewarding trajectories among diverse trajectories generated by LLM Actor (see Table 1).
- We empirically demonstrate that Prospector can provide better success rate than ReAct and Reflexion on two representative decision-making environments such as ALFWorld and WebShop (see Table 2 and Table 7).



(a) ALFWorld



(b) WebShop

Figure 2: **Comparison of ReAct and AskAct.** AskAct is a prompting method that introduces additional self-asking steps in a ReAct prompt. (a) In ALFWorld, the self-asking step checks if a target object is found. This can elicit a correct action by alleviating hallucination. (b) In WebShop, the self-asking step explicitly tries to determine which item is the most proper. This can elicit a better item selection.

2 Preliminaries

2.1 Language-based interactive decision-making tasks

We consider language-based interactive decision-making tasks [28, 22, 6, 21], where reasoning and planning abilities are key challenges in solving the task. At each time step t, the LLM agent receives a text observation o_t from the environment, generates a text action a_t, and receives the associated reward r_t. LLM agent aims to generate the action from the context c_t that maximizes the expected cumulative rewards, where $c_t := (o_1, a_1, \dots, o_{t-1}, a_{t-1}, o_t)$. These interactive decision-making tasks can be naturally formulated as reinforcement learning (RL) problems with partially observable Markov decision processes (POMDPs) [27]. However, optimizing the LLM agent $\pi(a_t|c_t)$ with

Table 1: **Critic prompt template for few-shot reward prediction.**

ALFWorld	WebShop

`Evaluate if the instruction given in the input is accomplished by performing a sequence of actions (fail/success).`
 	
`### Input:`
`{Example success trajectory}`
`### Response: success`
 	
`### Input:`
`{Example fail trajectory}`
`### Response: fail`
 	
`### Input:`
`{Input trajectory}`
`### Response:`
	`Evaluate if the instruction given in the input is accomplished by selecting the proper item (low/middle/high).`
	
	`### Input:`
	`{Example high-reward trajectory}`
	`### Response: high`
	
	`### Input:`
	`{Example low-reward trajectory}`
	`### Response: low`
	
	`### Input:`
	`{Input trajectory}`
	`### Response:`

RL-based LLM fine-tuning methods requires separate expensive training costs for each task, which are unsuitable for LLMs with an enormous number of parameters.

2.2 Few-shot prompting methods for LLM agents

Few-shot prompting [30, 20] is one of the representative methods that can improve the performance of LLMs without additional fine-tuning. ReAct [30], which is the most relevant to Prospector, leverages few-shot prompting to improve reasoning skills in interactive decision-making tasks. Instead of directly generating the action from the current context c_t, it generates *thought* k_t corresponding to the reasoning trace based on given few-shot demonstrations including the reasoning trace, and then finally generates the action from the updated context $\hat{c}_t$ augmented with generated *thought* (i.e. $\hat{c}_t = (c_t, k_t)$). However, reasoning the useful information directly from the context is often challenging, and this few-shot prompting alone may not be sufficient to generate optimal trajectories since it does not consider the task feedback signal (i.e. reward) from an environment.

3 Method

The overivew of Prospector is shown in Figure 1. Prospector mainly consists of two complementary LLMs such as LLM Actor and LLM Critic. To improve the baseline performance of the LLM Actor, we propose *AskAct prompting* which elicits more proper actions from the LLM Actor by generating a question and answer itself. Also, to explore diverse trajectories, we provide *Trajectory Ranking* which the LLM Actor generates diverse trajectories with high temperature and the LLM Critic selects the most rewarding trajectory by predicting the expected total reward. Combining these components allows Prospector to trial promising actions and generate the most strategic actions. Our algorithm can be adopted for any LLM and decision-making task, given few-shot demonstrations.

3.1 AskAct Prompting

To improve the baseline performance of the LLM Actor, we present *AskAct prompting* that introduces additional self-asking steps in a ReAct prompt. Figure 2 compares AskAct and ReAct by showing examples trajectories. Unlike ReAct [30], which performs the *reasoning* as an intermediate step, we attempt to perform question-and-answering as an intermediate step towards more strategic decision-making. AskAct performs the *asking* about necessary information for strategic decision-making and *answering* them, before generating final action. This sophisticated process of information collecting encourages to generate more promising actions to help achieve the task. Concrete examples of AskAct prompts can be found in Table 11 and Table 13 of the Appendix.

3.2 Trajectory Ranking

Since generating trajectory from an LLM is relatively much cheaper than training an LLM, we consider generating trajectories and selecting the best trajectory among them instead of training

Table 2: **Performance comparison of LLM agents on ALFWorld.** Prospector with AskAct and Trajectory Ranking (TR) considerably improves the success rate on ALFWorld, compared to the recent advancements such as ReAct [30] and Reflexion [20]

Method	LLM Actor	LLM Critic	Success Rate (%)
BUTLER	-	-	37.0
ReAct	text-davinci-002	-	78.4
ReAct + Reflexion ($k = 5$)	text-davinci-002	-	86.0
ReAct + TR ($k = 5$) (Prospector)	text-davinci-002	text-davinci-002	**91.0**
ReAct	Llama-2-70B	-	41.0
ReAct + TR ($k = 5$)	Llama-2-70B	FLAN-T5-3B (SFT)	77.6
AskAct	Llama-2-70B	-	56.7
AskAct + TR ($k = 5$) (Prospector)	Llama-2-70B	FLAN-T5-3B (SFT)	<u>86.6</u>

the LLM. To this end, we present *Trajectory Ranking* in which the LLM agent generates diverse trajectories with a number of trials and then selects the most rewarding trajectory as the final action. Thanks to AskAct, which allows the LLM Actor to generate more promising actions, Prospector can consider high-quality trajectories as candidates for final actions. However, most real-world scenarios allow the agent to interact with the environment (i.e. simulation) but not receive rewards. For example, in a shopping scenario such as WebShop, buyers can browse various products, but they cannot check their satisfaction (i.e. reward) by purchasing the products themselves. Therefore, we investigate two methods to estimate the trajectory reward from a given dataset: (1) Few-shot LLM Critic, and (2) Fine-tuned LLM Critic.

Few-shot LLM Critic. Motivated by recent methods of using LLMs as an evaluator [11, 31], we attempt to use LLMs as reward estimators for interactive decision-making tasks. To evaluate the trajectories without additional training of the reward model, we use few-shot in-context learning with reward-labeled trajectories. We provide the critic prompt template used for few-shot reward prediction in Table 1. More concrete examples of critic prompts can be found the Table 12 and Table 14 in the Appendix.

Fine-tuned LLM Critic. In some complex environments such as WebShop [28], one of the most powerful LLMs such as GPT-3 have difficulty in reward prediction in a few-shot manner (see Table 8). In this case, open-source LLMs fine-tuned on trajectory data can help to increase the performance of Prospector agents. The details can be found in Table 7 and Table 9 in the Experiment section.

4 Experiments

4.1 ALFWorld

ALFWorld [22] is a multi-modal interactive decision-making benchmark that is specialized on embodied reasoning tasks such as solving house-holding tasks. It is designed by aligning TextWorld [6], an interactive text-based game, and ALFRED [21], a representative embodied AI benchmark. It includes 6 types of tasks such as (1) pick and place, (2) examine in light, (3) clean and place, (4) heat and place, (5) cool and place, and (6) pic two and place. The ALFRED dataset provides 3,553 tasks for training, 140 tasks for seen testing, and 134 tasks for unseen testing. In this paper, we perform the experiments in the text-mode of ALFWorld where a natural language instruction is given and the agent is requested to generate text-based actions by interacting the environment. We evaluate LLM agents on the unseen 134 tasks in the ALFRED dataset. For fine-tuning open-sourced LLMs for Trajectory Ranking, we use 3K training tasks in the ALFRED dataset.

4.1.1 Success rate

Comparison. In Table 2, we compare the success rate of Prospector with the recent LLM agents such as ReAct [30] and Reflexion [20] on ALFWorld. To show the difficulty of the tasks of ALFWorld, we also provide the performance of BUTLER, an agent that does not use LLMs.

Table 3: **Success rate with regard to the number of trajectories.**

Method	LLM Actor	LLM Critic	$k = 1$	2	3	4	5
ReAct + TR	Llama-2-70B	FLAN-T5-3B (SFT)	33.6	59.0	69.4	73.1	77.6
AskAct + TR	Llama-2-70B	FLAN-T5-3B (SFT)	53.7	76.1	80.6	84.3	86.6
ReAct + TR	text-davinci-002	text-davinci-002	71.6	-	90.0	-	**91.0**

Table 4: **Comparison of task-level success rate on ALFWorld.**

Method	LLM Actor	Pick	Clean	Heat	Cool	Look	Pick2	All (%)
BUTLER	-	46	39	74	100	22	24	37
Act	PaLM-540B	88	42	74	67	72	41	45
ReAct	PaLM-540B	92	58	96	86	78	41	71
ReAct	text-davinci-002	88	61	78	86	89	71	78
	(#success/#tasks)	21/24	19/31	18/23	18/21	16/18	12/17	104/134
ReAct + TR ($k = 5$)	text-davinci-002	**100**	84	91	**95**	**100**	**76**	**91**
		24/24	26/31	21/23	20/21	18/18	13/17	122/134
ReAct + TR ($k = 5$)	Llama-2-70B	92	74	91	86	61	53	78
		22/24	23/31	21/23	18/21	11/18	9/17	104/134
AskAct + TR ($k = 5$)	Llama-2-70B	92	**87**	**96**	**95**	94	47	87
		22/24	27/31	22/23	20/21	17/18	8/17	116/134

The comparison on `text-davinci-002` the LLM Actor is shown in the middle of the table. Prospector (ReAct + TR) outperforms ReAct and Reflexion. ReAct only uses few-shot demonstrations for solving house-holding tasks on ALFWorld. Reflexion further improves the success rate of the ReAct agent by using iterative refinements of the LLM output. Here, $k = 5$ refinements are performed. For the purpose of comparison, we use ReAct as the base LLM Actor. In Trajectory Ranking (TR), the LLM Actor of Prospector generates diverse trajectories and the LLM Critic selects the expected best trajectory. Here, $k = 5$ trajectories are generated, and TR selects the expected best trajectory by using the 2-shot LLM Critic.

The comparison on Llama-2-70B as the LLM Actor is shown in the bottom of the table. We conduct experiments with four different settings: (1) ReAct only, (2) AskAct only, (3) ReAct + Trajectory Ranking (TR), and (4) AskAct + TR. AskAct effectively improves the success rate of ReAct (from 41.0 to 56.7).

Effect of the number of trials. In Table 3, we show the change in the success rate with reward to the number of generated trajectories. As shown in the figure, the success rate of Prospector (AskAct + TR) increases as the number of generated trajectories (k) increases. To generate diverse (creative) trajectories, the LLM Actor of Prospector sets the temperature to 0.8. For Trajectory Ranking (TR), 2-shot LLM Critic (`text-davinci-002`) is used, and its temperature is set to 0.2. Since AskAct provides a better baseline, AskAct + TR can achieve much better performance with less sampling (e.g., AskAct only (56.7) comparable with ReAct + TR (k=2) (56.0)). We emphasize that AskAct and TR can make an effective synergy in improving LLM agents in terms of both performance and efficiency.

Task-level success rate. In Table 4, we provide the detailed success rate for each task type in the ALFWorld benchmark.

4.1.2 Accuracy of LLM Critic

Few-shot accuracy. In Table 5, we show the few-shot reward prediction accuracy of LLM Critics on ALFWorld. The few-shot accuracy is high enough to be used in Trajectory Ranking without the need for find-tuning open-sourced LLMs on AFLWorld trajectory data. Since the reward prediction

Table 5: **Few-shot reward prediction accuracy of LLM Critics on ALFWorld.**

LLM Critic	1-shot	2-shot	3-shot
text-davinci-002	94.8	**97.0**	95.5
text-davinci-003	93.3	95.5	94.0

Table 6: **Fine-tuning reward prediction accuracy of LLM Critics on ALFWorld.**

LLM Critic	Param.	Adaptation Method	# Trainable Param.	Accuracy (success/fail)
text-davinci-003	-	2-shot ICL	0	95.5
text-davinci-002	-	2-shot ICL	0	**97.0**
Bloom	7.1B	LoRA FT on 3K data	3.9M	79.1
Llama-2-Chat	7B	LoRA FT on 3K data	4.2M	94.8
Bloomz	7.1B	LoRA FT on 3K data	3.9M	95.5
Llama-2	7B	LoRA FT on 3K data	4.2M	96.3
GPT-J	6B	LoRA FT on 3K data	3.7M	97.3
T5	3B	LoRA FT on 3K data	5.9M	98.5
FLAN-T5	3B	LoRA FT on 3K data	4.7M	**98.5**

accuracy of 2-shot LLM Critic is very high (97%), the LLM Critic of Prospector can select the highly-rewarding trajectory from diverse trajectories and considerably increase the success rate.

Fine-tuning accuracy. In Table 6, we show the fine-tuning reward prediction accuracy of LLM Critics on ALFWorld. We finetune open-sourced LLMs on 3K ALFWorld trajectory data. For decoder-only models, we choose GPT-J [24], Bloom [19], Bloomz [13], and Llama-2 [23]. For encoder-decoder models, we choose T5 [18] and FLAN-T5 [5]. For parameter-efficient fine-tuning, we use LoRA [9]. By fine-tuning open-sourced LLMs on 3K ALFWorld trajectory data, they can achieve comparable or better reward prediction accuracy with the closed LLMs such as `text-davinci-002`. The hyperparameters used for fine-tuning LLM Critics can be found in Table 17 in the Appendix.

4.2 WebShop

WebShop [28] is a large-scale online shopping environment with more than 1M real-world products crawled from Amazon. The agent is given a natural language instruction (e.g., "I would like 3 ounce bottle of bright citrus deodorant for sensitive skin, and price lower than 50.00 dollars."), and required to make a sequence of actions (e.g., querying the search engine with keywords and clicking on a product title) to accomplish the given instruction. More specifically, the task mainly consists of five stages: (1) searching products with query words, (2) selecting a product in the search results, (3) selecting proper options, (4) reviewing the product details, and (5) clicking on the "Buy Now" button. WebShop provides two modes: (1) multi-modal mode with product figures, and (2) text-based mode. Also, WebShop provides about 12K human instructions, and reserves 500 instructions for tasting. In this paper, we perform experiments in the text-based mode and evaluate LLM agents on the official 500 test instructions. We use 12K human instructions (without test instruction) for generating trajectories and fine-tuning LLM Critics on them.

4.2.1 Success rate

In Table 7, we compare the performance of Prospector with the recent LLM agents such as ReAct [30] and Reflexion [20]. In addition to these methods, to assess the difficulty of the environment, we also provide the performance of human as upper bound, and the performance of the traditional methods that use Imitation Learning (IL) or Reinforcement Learning (RL) as strong baselines. These results are quoted from the WebShop [28] paper. As shown in the table, Prospector achieves better success rate (43.0%) than the recent advancements such as ReAct (35.8%) and Reflexion(35.0%). Compared to the traditional IL (29.1%) and RL (28.7%) methods, ReAct agents based on `text-davinci-002` surprisingly achieve high success rate without training. However, there is a gap with the human performance (50%). Prospector can considerably reduce this gap by using Self-Asking and Trajectory Ranking (TR). Prospector using AskAct prompting can increase the success rate up to 39.8%

Table 7: **Performance comparison of LLM agents on WebShop.** Prospector with AskAct and Trajectory Ranking (TR) can improves the success rate on WebShop, compared to the recent advancements such as ReAct [30] and Reflexion [20].

Method	LLM Actor	LLM Critic	Reward	Success Rate
Human (expert)	-	-	82.1	59.6
Human (average)	-	-	75.5	50.0
IL	-	-	59.9	29.1
IL + RL	-	-	62.4	28.7
ReAct	text-davinci-002	-	63.3	35.8
ReAct + Reflexion ($k = 8$)	text-davinci-002	-	-	35.0
AskAct	text-davinci-002	-	66.5	39.8
AskAct + TR ($k = 8$) (Prospector)	text-davinci-002	text-davinci-002	69.3	41.4
AskAct + TR ($k = 8$) (Prospector)	text-davinci-002	Llama-2-7B-Chat (SFT)	**70.8**	<u>43.0</u>
AskAct + TR ($k = 8$)	text-davinci-002	Oracle (w/ reward)	71.3	47.0
ReAct	Llama-2-70B	-	62.3	37.6
AskAct	Llama-2-70B	-	68.6	42.2
ReAct + TR ($k = 8$)	Llama-2-70B	FLAN-T5-3B (SFT)	69.3	42.2
AskAct + TR ($k = 8$) (Prospector)	Llama-2-70B	FLAN-T5-3B (SFT)	<u>70.2</u>	**43.6**

Table 8: **Few-shot reward prediction accuracy of LLM Critics on WebShop.** Few-shot LLM Critics have some difficulty in predicting the reward of the agent's trajectory in a complex environment such as WebShop. This requires LLM Critics fine-tuned on WebShop trajectory data.

LLM Critic	1-shot	2-shot	3-shot
text-davinci-002	34.4	**47.0**	42.4
text-davinci-003	37.0	42.2	36.2

compared to ReAct (35.8%). Prospector with AskAct and TR further increase the success rate up to 41.3%. However, since the few-shot LLM Critic based on `text-davinci-002` does not provide high accuracy (47.0%) in reward prediction, the improvement is not significant. In contrast, since the fine-tuned LLM Critic based on Llama-2-7B-Chat [23] provides much higher accuracy in reward prediction, Prospector can achieve better success rate (43.0%). Note that if the oracle with known reward is used, the success rate can be reached by up to 47.0%, while considerably closing the gap with the human performance (50%). Note that the performance of LLM Critic is important to improve the performance of LLM agents. Regarding this, we provide the detailed additional experiments on LLM Critics in Table 8, Table 9, and Table 10 in the following subsection.

On the WebShop environment, Llama-2-70B, one of representative open-source LLMs can achieve comparable performance with text-davinci-002, one of the most powerful LLMs. In both cases of text-davinci-002 and Llama-2-70B, AskAct meaningfully improves the success rate compared to ReAct: from 35.8 to 39.8 on text-davinci-002, and from 37.6 to 42.2 on Llama-2-70B. This means that AskAct, a simple prompting method that adds extra question prompts on ReAct, can be effective. ReAct + TR can improve ReAct from 37.6 to 42.2 in the success rate. AskAct + TR further improves the success rate of AskAct (from 42.2 to 43.6), and provides better performance than ReAct + TR (42.2).

4.2.2 Accuracy of LLM Critic

Few-shot accuracy. In Table 8, we provide the few-shot reward prediction accuracy of API-based LLM Critics such as `text-davinci-002` on WebShop. We find that few-shot LLM Critics have some difficulty in predicting the reward of a given trajectory in a complex environment such as WebShop. LLM Critics with low reward prediction accuracy can not be used for reliable Trajectory Ranking (TR). This result requires us to fine-tune open-sourced LLMs such as Llama-2 on WebShop trajectory data.

Table 9: **Fine-tuning reward prediction accuracy of LLM Critics on WebShop.** Fine-tuned LLM Critics (e.g., Llama-2 fine-tuned on 12K trajectory data) provide significantly improved reward prediction accuracy, compared to few-shot LLM Critics (e.g., `text-davinci-002`) in WebShop. Improved prediction accuracy of fine-tuned LLM Critics help to increase the success rate of Prospector

LLM Critic	Param.	Adaptation Method	# Trainable Param.	Accuracy (hi/mi/lo)
text-davinci-003	-	2-shot ICL	0	42.2
text-davinci-002	-	2-shot ICL	0	47.0
Bloom	7.1B	LoRA FT on 12K data	3.9M	67.2
GPT-J	6B	LoRA FT on 12K data	3.7M	72.0
Llama-2	7B	LoRA FT on 12K data	4.2M	73.8
Bloomz	7.1B	LoRA FT on 12K data	3.9M	75.8
Llama-2-Chat	7B	LoRA FT on 12K data	4.2M	76.2
T5	3B	LoRA FT on 12K data	5.9M	77.0
FLAN-T5	3B	LoRA FT on 12K data	4.7M	**78.0**

Table 10: **Fine-tuning accuracy over the dataset size.**

LLM Critic	3K	6K	9K	12K
Llama-2-7B-Chat (LoRA)	70.0	71.1	76.2	**76.2**

Fine-tuning accuracy. In Table 9, we compare the reward prediction accuracy of fine-tuned LLM Critics. We finetune open-sourced LLMs on 3K ALFWorld trajectory data. For parameter-efficient fine-tuning, we use LoRA [9]. Fine-tuned LLM Critics (e.g., Llama-2 fine-tuned on 12K trajectory data) provide significantly improved reward prediction accuracy, compared to few-shot LLM Critics (e.g., `text-davinci-002`) in the WebShop environment. Improved prediction accuracy of fine-tuned LLM Critics help to increase the success rate of Prospector.

In Table 10, we provide the change in reward prediction accuracy with regard to the size of trajectory data. We can see that the reward prediction accuracy increases as the data size increases. The hyperparameters used for fine-tuning LLM Critics can be found in Table 17 in the Appendix.

5 Related Work

Reasoning in LLMs. Few-shot in-context learning (ICL) is one of the representative methods, that achieves high performance in various NLP tasks. However, ICL-based approaches are known to struggle in reasoning tasks. To address this shortcoming, Wei et al. [26] introduced chain-of-thoughts (CoT) that generates a series of short sentences that mimic the human reasoning process. CoT with Self-Consistency (CoT-SC) [25] samples k diverse reasoning paths instead of selecting the greedy one and subsequently returns the most frequent answer. However, since this approach is only applicable when the output space is limited, Tree-of-Thoughts (ToT) [29] overcomes this limitation by generalizing CoT prompting and further enhancing local exploration of thought. On the other hand, Self-Ask [16] improves CoT on QA tasks by transforming a chain-of-thought into a multi-turn self-question-answering process. This study also introduced the concept of conducting reasoning through question-answering concurrently with our work, but we want to emphasize that while Self-Ask focuses on QA tasks, our work enhances LLM agents for interactive decision-making tasks through synergizing self-asking and trajectory ranking.

LLM-based agents. The use of reasoning prompts for LLM agents also enables achieving high performance in text-based interactive decision-making tasks without training. ReAct [30] is an algorithm that integrates reasoning and action within language models to tackle a diverse range of language reasoning and decision-making tasks. When task feedback is accessible, Reflexion [20] and Self-Refine [12] reinforce LLM agents by learning a linguistic reward model to verbally reflect the task feedback signal. We note that Reflexion iteratively generates trajectories and reflects rewards verbally in sequence, while Prospector generates diverse trajectories in parallel and chooses the best one in terms of rewards. On the other hand, when human or other external knowledge sources are available, Asking-Before-Action (ABA) [4] incorporates humans into the decision-making process by introducing a contextual MDP with human or external information sources in the loop.

Reward models and rankings. Reward models and rankings are widely employed within the LLM context and their applications. In order to enhance LLM performance, InstructGPT [15] and Llama-2 [23] leverage RL for fine-tuning the LLMs themselves. Furthermore, LLMs have showcased their impressive capability to generate code across diverse programming tasks, highlighting their versatility. Within this domain, a neural ranker, CodeRanker [10], was introduced to improve the accuracy of various code generation models, enabling them to predict the correctness of sampled code without actual execution. On the other hands, to harness the LLM's semantic knowledge about the real world, SayCan [2] proposed an innovative approach to combine LLM and RL.

6 Discussion

Comparison of AskAct and ReAct. AskAct is a prompting method that interleaves additional self-asking steps in a ReAct [30] prompt. In ALFWorld, the self-asking step checks if a target object is found. This can elicit a correct action by alleviating hallucination. In WebShop, the self-asking step explicitly tries to determine which item is the most proper. This can elicit a better item selection. We empirically show that AskAct considerably improves the success rate compared to ReAct (see Table 2 and Table 5). In ALFWorld [22], AskAct on Llama-2-70B provides 56.7% of success rate, while achieving about 15.0% absolute improvement compared to ReAct on Llama-2-70B (41.0% of success rate). In WebShop [28], AskAct on Llama-2-70B achieves 42.2% of success rate (about 4.6% improvement), while ReAct on Llama-2-70B provides 37.6% of success rate.

Comparison of AskAct and Self-Ask. AskAct and Self-Ack [16] commonly have self-asking steps in few-shot examples in a prompt to improve the LLM response. However, AskAct and Self-Ask are significantly different in the purpose and composition. AskAct is mainly designed to solve sequential decision-making tasks, and consists of a sequence of observation, self-question, self-answer, self-reasoning, and action. In contrast, Self-Ask is mainly designed to provide better answers to knowledge-intensive questions, and consist of question, follow-up question, follow-up answer, and answer. AskAct (adding extra question and answer steps in a ReAct prompt) and Self-Ask (adding extra question and answer steps in a CoT prompt) can be seen as having a relationship similar to ReAct (adding intermediate reasoning steps in an Act prompt) and Chain-of-Thoughts (adding intermediate reasoning steps in a direct prompt).

Comparison of Trajectory Ranking (TR) and Reflexion. In Prospector, the LLM agent can generate multiple trajectories in a parallel manner. Given multiple trajectories, Trajectory Ranking (TR) predicts the expected reward of each trajectory and selects a trajectory which expected reward is the maximum. In contrast, Reflexion [20] iteratively refines a trajectory generated by the LLM agent. Since the trajectory improvement is done in a sequential manner, exploration ability is significantly limited. Furthermore, for trajectory improvement, Reflexion requires carefully-crafted verbal feedback prompts. Due to these limitations, Reflexion does not improve the success rate in WebShop.

Synergy of combining AskAct and Trajectory Ranking. Finally, we would like to emphasize that AskAct and TR can make an effective synergy in improving LLM agents in terms of both performance and efficiency. Since AskAct provides a better baseline, AskAct and TR can achieve much better performance with less sampling. For example, in ALFWorld, AskAct only achieves 56.7% of success rate, while ReAct and TR (k=2) reaches 56.0% of success rate.

7 Conclusion

In this paper, we introduce Prospector, a LLM agent that consists two complementary LLM agents such as LLM Actor and LLM Critic. To improve the baseline performance of LLM Actor, we provide AskAct promptint that introduces addtional self-asking steps in the few-shot demonstrations. Also, we provide Trajectory Ranking that selects the most rewarding trajectory from diverse trajectories generated by LLM Actor by predicting the expected reward.

References

[1] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv:2204.05862*, 2022.

[2] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *CoRL*, 2023.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[4] X. Chen, S. Zhang, P. Zhang, L. Zhao, and J. Chen. Asking before action: Gather information in embodied decision making with language models. *arXiv:2305.15695*, 2023.

[5] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, et al. Scaling instruction-finetuned language models. *arXiv:2210.11416*, 2022.

[6] M.-A. Côté, A. Kádár, X. Yuan, B. Kybartas, T. Barnes, E. Fine, J. Moore, M. Hausknecht, L. El Asri, M. Adada, et al. Textworld: A learning environment for text-based games. In *IJCAI Computer Games Workshop*, 2018.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[8] A. Glaese, N. McAleese, M. Trebacz, J. Aslanides, V. Firoiu, T. Ewalds, M. Rauh, L. Weidinger, M. Chadwick, P. Thacker, et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv:2209.14375*, 2022.

[9] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. 2022.

[10] J. P. Inala, C. Wang, M. Yang, A. Codas, M. Encarnación, S. Lahiri, M. Musuvathi, and J. Gao. Fault-aware neural code rankers. In *NeurIPS*, 2022.

[11] X. Li, P. Yu, C. Zhou, T. Schick, L. Zettlemoyer, O. Levy, J. Weston, and M. Lewis. Self-alignment with instruction backtranslation. *arXiv:2308.06259*, 2023.

[12] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, et al. Self-refine: Iterative refinement with self-feedback. *NeurIPS*, 2023.

[13] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf, et al. Crosslingual generalization through multitask finetuning. In *ACL*, 2023.

[14] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv:2112.00114*, 2021.

[15] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[16] O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, and M. Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

[17] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[18] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[19] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv:2211.05100*, 2022.

[20] N. Shinn, B. Labash, and A. Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. In *NeurIPS*, 2023.

[21] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020.

[22] M. Shridhar, X. Yuan, M.-A. Côté, Y. Bisk, A. Trischler, and M. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *ICLR*, 2021.

[23] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*, 2023.

[24] B. Wang and A. Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

[25] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2023.

[26] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[27] J. D. Williams and S. Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.

[28] S. Yao, H. Chen, J. Yang, and K. Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*, 2022.

[29] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*, 2023.

[30] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

[31] S. Ye, D. Kim, S. Kim, H. Hwang, S. Kim, Y. Jo, J. Thorne, J. Kim, and M. Seo. Flask: Fine-grained language model evaluation based on alignment skill sets. *arXiv:2307.10928*, 2023.

A Appendix

A.1 Example prompts

A.1.1 Example AskAct prompt on ALFWorld

I this section, we provide the prompts used for the LLM Actor and few-shot LLM Critic to solve the tasks in the ALFWorld. The ReAct prompt for the LLM Actor is provided in Table 11. The Critic prompt for the LLM Critic is provided in Table 12.

A.1.2 Example AskAct prompt on WebShop

I this section, we provide the prompts used for the LLM Actor and few-shot LLM Critic to solve the decision-making tasks in the WebShop. The AskAct prompt for the LLM Actor is provided in Table 13. The Critic prompt for the few-shot LLM Critic is provided in Table 14.

A.2 Example trajectories

A.2.1 Example Prospector trajectory on ALFWorld

In Table 15, we provide an example success trajectory generated by Prospector on ALFWorld.

A.2.2 Example Prospector trajectory on WebShop

In Table 16, we provide an example high-reward trajectory generated by Prosepctor on WebShop.

A.3 Hyper-parameters of LLM Critic fine-tuning

In Table 17, we provide the hyper-parameters used for fine-tuning the LLM Critic on the trajectory data.

Table 11: **Example AskAct prompt of LLM Actor on ALFWorld.** Reasoning steps are denoted in blue. Additional self-asking steps are denoted in red.

AskAct Prompt for ALFWorld

```
You are in the middle of a room. Looking quickly around you,
you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1,
a countertop 1, a garbagecan 1, a handtowelholder 2,
a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1,
a toiletpaperhanger 1, and a towelholder 1.

Your task is to: put some spraybottle on toilet.

> think: To solve the task, I need to find and take a sparybottle,
then put it on toilet.
OK.
> think: First I need to find a spraybottle. A spraybottle is
more likely to appear in cabinet (1-4), countertop (1), toilet (1),
sinkbasin (1-2), garbagecan (1). I can check one by one,
starting with cabinet 1.
OK.
> go to cabinet 1
On the cabinet 1, you see a cloth 1, a soapbar 1, a soapbottle 1.
> think: Did I find a spraybottle?
OK.
> think: No, I did not. I need to go other place to find a spraybottle.
OK.
> go to cabinet 2
The cabinet 2 is closed.
> open cabinet 2
You open the cabinet 2. The cabinet 2 is open. In it,
you see a candle 1, and a spraybottle 2.
> think: Did I find a spraybottle?
OK.
> think: Now I find a spraybottle (2). Next, I need to take it.
OK.
> take spraybottle 2 from cabinet 2
You pick up the spraybottle 2 from the cabinet 2.
> think: Now I take a spraybottle (2). Next, I need to
put it in/on toilet 1.
OK.
> go to toilet 1
On the toilet 1, you see a soapbottle 2.
> put spraybottle 2 in/on toilet 1
You put the spraybottle 2 in/on the toilet 1.
```

Table 12: **Critic prompt of few-shot LLM Critic on ALFWorld.** The *success* trajectory given as a 1-shot demonstration is generated by `text-davinci-002` with ReAct prompting. Instructions for few-shot reward prediction are denoted in red.

Critic Prompt on ALFWorld

```
Evaluate if the instruction given in the input is accomplished by
performing a sequence of actions (fail/success).

### Input:
You are in the middle of a room.  Looking quickly around you,
you see a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3,
a countertop 2, a countertop 1, a diningtable 1, a drawer 6,
a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1,
a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1,
a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1,
and a toaster 1.

Your task is to:  clean some lettuce and put it in fridge.

> think:  To solve the task, I need to find and take a lettuce,
then clean it with sinkbasin, then put it in fridge.
OK.
> think:  First I need to find a lettuce.  A lettuce is more likely
to appear in fridge (1), diningtable (1), sinkbasin (1),
stoveburner (1-3), cabinet (1-2).  I can check one by one,
starting with fridge 1.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
You open the fridge 1.  The fridge 1 is open.  In it, you see a apple 1,
a cup 1, a lettuce 1, a potato 2, and a potato 1.
> think:  Now I find a lettuce (1).  Next, I need to take it.
OK.
> take lettuce 1 from fridge 1
You pick up the lettuce 1 from the fridge 1.
> think:  Now I take a lettuce (1).  Next, I need to go to
sinkbasin (1) and clean it.
OK.
> go to sinkbasin 1
On the sinkbasin 1, you see a apple 2, a butterknife 1, a cup 2,
a knife 2, a potato 3, and a spoon 3.
> clean lettuce 1 with sinkbasin 1
You clean the lettuce 1 using the sinkbasin 1.
> think:  Now I clean a lettuce (1).  Next, I need to put it
in/on fridge 1.
OK.
> go to fridge 1
The fridge 1 is open.  In it, you see a apple 1, a cup 1, a potato 2,
and a potato 1.
> put lettuce 1 in/on fridge 1
You put the lettuce 1 in/on the fridge 1.
### Response:  success

### Input:
{Input trajectory}
### Response:
```

Table 13: **Example AskAct prompt of LLM Actor on WebShop.** Reasoning steps are denoted in blue. Additional self-asking steps are denoted in red.

AskAct Prompt for WebShop

```
Webshop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for
sensitive skin, and price lower than 50.00 dollars
[Search]

Action: search[3 ounce bright citrus deodorant sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B078GWRC1J]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for
Sensitive Skin, Pregnancy and Breastfeeding,
Contains Organic Calendula 3-Ounce
$10.99
[B078GTKVXY]
Ginger Fresh Deodorant by Earth Mama | Natural and Safe for
Sensitive Skin, Pregnancy and Breastfeeding,
Contains Organic Calendula 3-Ounce
$10.99
[B08KBVJ4XN]
Barrel and Oak - Aluminum-Free Deodorant, Deodorant for Men,
Essential Oil-Based Scent, 24-Hour Odor Protection,
Cedar & Patchouli Blend, Gentle on Sensitive Skin
(Mountain Sage, 2.7 oz, 2-Pack)
$15.95

Action: think[Given the instruction and candidate times
such as B078GWRC1J, B078GTKVXY, and B08KBVJ4XN, which item is
the most proper to select?]
Observation: OK.
Action: think[B078GWRC1J is the most proper, since the item
is bright citrus deodorant less then 50 dollars.]
Observation: OK.

Action: click[B078GWRC1J]
Observation:
[Back to Search]
[< Prev]
scent [assorted scents][bright citrus][calming lavender]
[ginger fresh][simply non-scents]
size [travel set (4-pack)][3 ounce (pack of 1)][3-ounce (2-pack)]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for
Sensitive Skin, Pregnancy and Breastfeeding,
Contains Organic Calendula 3-Ounce
Price: $10.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Buy Now]

Action: think[For 3 ounce bottle of bright citrus deodorant for
sensitive skin, the item has options 'bright citrus' and
'3 ounce (pack of 1)' and seems good to buy.]
Observation: OK.

Action: click[bright citrus]
Observation: You have clicked bright citrus.
Action: click[3 ounce (pack of 1)]
Observation: You have clicked 3 ounce (pack of 1).
Action: click[Buy Now]
```

Table 14: **Critic prompt of few-shot LLM Critic on WebShop.** The *high-reward* trajectory given as a 1-shot demonstration is generated by `text-davinci-002` with ReAct prompting. Instructions for few-shot reward prediction are denoted in red.

Critic Prompt on WebShop

```
Evaluate if the instruction given in the input is accomplished
by selecting the proper item (low/middle/high).

### Input:
WebShop
Instruction:
find me a regular fit machine washable cargo pants with buttoned closure
in 6057 apricot color and 29 size, and price lower than 40.00 dollars
[Search]

Action: search[regular fit machine washable cargo pants buttoned closure
6057 apricot color 29 size]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B00PQFK67G]
Match Men's Athletic-Fit Cargo Pants
$21.99 to $29.99
[B01M4KXXPT]
Vertx Men's Phantom Lt 2.0 Shorts
$21.14 to $76.84
[B01EAP5JD8]
DUNLOP Mens On Site Workwear Trousers Regular Classic
$63.99

Action: think[B00PQFK67G is the one I want.]
Observation: OK.

Action: click[B00PQFK67G]
Observation:
[Back to Search]
[< Prev]
size [29][30][32][34][36][38][40][42][44]
color [6057 apricot][6057 khaki][6062 dark khaki][6531 apricot][6531 black]
 [6531 grayish black][6531 light gray][6531 light khaki][khaki max]
 [6057 light apricot][6069 blue][6531 army green][6531 blue][6531 dark gray]
 [6071 black][6071 dark blue][6071 light khaki][6039m tangerine max]
 [6071 dark apricot][6071 dark khaki][6057 bluish purple][6057 dark gray]
 [6071 brown][6071 gray][6039m multi-max][6039 light khaki][6071 steel gray]
 [6039 dark yellow][6039 blue][6039 apricot][6039 dark gray][6050 peacock blue]
 [6057 silver gray][6066 light apricot]
Match Men's Athletic-Fit Cargo Pants
Price: $21.99 to $29.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Attributes]
[Buy Now]

Action: think[The item has options '29', '6057 apricot', '6057 khaki',
'6062 dark khaki', '6531 apricot', '6531 black', '6531 grayish black',
'6531 light gray', '6531 light khaki', 'khaki max', '6057 light apricot',
'6069 blue', '6531 army green', '6531 blue', '6531 dark gray', '6071 black']
Observation: OK.

Action: click[29]
Observation: You have clicked 29.

Action: click[6057 apricot]
Observation: You have clicked 6057 apricot.

Action: click[Buy Now]
### Response: high

### Input:
{Input trajectory}
### Response:
```

Table 15: **Success trajectory generated by Prospector on ALFWorld.** Reasoning steps are denoted in blue. Additional self-asking steps are denoted in red.

Success Trajectory on ALFWorld

```
You are in the middle of a room. Looking quickly around you,
you see a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2,
a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2,
a countertop 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1,
a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1,
a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2,
a stoveburner 1, and a toaster 1.
Your task is to: put a clean mug in coffeemachine.

> think: To solve the task, I need to find and take a mug,
hen clean it with sinkbasin, then put it in coffeemachine.
OK.
> think: First I need to find a mug. A mug is more likely
to appear in fridge (1), diningtable (1), sinkbasin (1), stoveburner (1-3),
cabinet (1-6). I can check one by one, starting with fridge 1.
OK. > go to fridge 1
The fridge 1 is closed.
> open fridge 1
You open the fridge 1. The fridge 1 is open. In it, you see a apple 2, a cup 2,
a cup 1, a lettuce 1, a plate 1, a tomato 2, and a tomato 1.
> think: Did I find a mug?
OK.
> think: No, I did not. I need to go other place to find a mug.
OK.
> go to diningtable 1
Nothing happens.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 3, a egg 3, and a glassbottle 3.
> think: Did I find a mug?
OK.
> think: No, I did not. I need to go other place to find a mug.
OK.
> go to stoveburner 1
On the stoveburner 1, you see a pot 1.
> think: Did I find a mug?
OK.
> think: No, I did not. I need to go other place to find a mug.
OK.
> go to cabinet 1
On the cabinet 1, you see a dishsponge 1, and a soapbottle 2.
> think: Did I find a mug?
OK.
> think: No, I did not. I need to go other place to find a mug.
OK.
> go to cabinet 2
The cabinet 2 is closed.
> open cabinet 2
You open the cabinet 2. The cabinet 2 is open. In it, you see a peppershaker 1.
> think: Did I find a mug?
OK.
> think: No, I did not. I need to go other place to find a mug.
OK.
> go to cabinet 3
On the cabinet 3, you see a mug 2, a mug 1, and a soapbottle 1.
> think: Did I find a mug?
OK.
> think: Now I find a mug (2). Next, I need to take it.
OK.
> take mug 2 from cabinet 3
You pick up the mug 2 from the cabinet 3.
> think: Now I take a mug (2). Next, I need to go to sinkbasin (1) and clean it.
OK.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 3, a egg 3, and a glassbottle 3.
> clean mug 2 with sinkbasin 1
You clean the mug 2 using the sinkbasin 1.
> think: Now I clean a mug (2). Next, I need to put it in/on coffeemachine 1.
OK.
> go to coffeemachine 1
On the coffeemachine 1, you see nothing.
> put mug 2 in/on coffeemachine 1
You put the mug 2 in/on the coffeemachine 1.
```

Table 16: **High-reward trajectory generated by Prospector on WebShop.** Reasoning steps are denoted in blue. Additional self-asking steps are denoted in red.

High-reward trajectory on WebShop

```
WebShop
Instruction:
i am looking for mn4 color foundation for my sensitive skin,
and price lower than 40.00 dollars
[Search]

Action: search[mn4 color foundation sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B077PR9TL4]
Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz
$7.59
[B07VLKYJNH]
PÜR 4-in-1 Love Your Selfie Longwear Foundation & Concealer - Full Coverage
Foundation & Concealer Makeup For Dark Spots, Blemishes and Imperfections -
Long-Lasting Liquid Makeup & Skincare
$36.0
[B09J1X9M1P]
Neutrogena Healthy Skin Sensitive Skin Serum Foundation with Pro-Vitamin B5,
Color Correcting & Pore Minimizing Liquid Foundation & Face Serum, Buildable
Coverage, Medium 02, 1 oz
$14.39

Action: think[Given the instruction and candidate times such as B077PR9TL4,
B07VLKYJNH, and B09J1X9M1P, which item is the most proper to select?]
Observation: OK.

Action: think[B077PR9TL4 is the most proper, since the item is mn4 color
foundation less then 40 dollars.]
Observation: OK.

Action: click[B077PR9TL4]
Observation:
[Back to Search]
[< Prev]
color [1][dc1][dn3][dn4][lc1][ln3][ln4][lw2][mc1][mn3][mn4][mw2]
size [1 fl oz (pack of 1)][1 ounce (pack of 1)][1 fluid ounce]
Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz
Price: $7.59
Rating: N.A.
[Description]
[Features]
[Reviews]
[Attributes]
[Buy Now]

Action: think[For mn4 color foundation for sensitive skin, the item has
options 'mn4' and '1 fl oz (pack of 1)' and seems good to buy.]
Observation: OK.

Action: click[mn4]
Observation: You have clicked mn4.

Action: click[1 fl oz (pack of 1)]
Observation: You have clicked 1 fl oz (pack of 1).

Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 1.0
```

Table 17: **Hyper-parameters of LLM Critic fine-tuning.**

Hyper-parameter	Value
model max length (context length)	1024
batch size	128
max epochs	50
optimizer	AdamW
learning rate	3e-4
weight decay	0.1
learning rate scheduler	cosine
warm-up steps	50% of the max steps
LoRA r	8
LoRA alpha	32
LoRA drop-out	0.1