Advancing Large Language Model Agent via Iterative Contrastive Trajectory Optimization

Chengang Jing
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: jcg050980@163.com

Kun Li
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: 38190985@qq.com

Xin Jing
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: jingxin@xatu.edu.cn

***Abstract*—Recent advancements in Large Language Models (LLMs) have expanded their application across a variety of tasks. However, open-source LLMs often fail to achieve the same efficiency as proprietary models. To address this issue, we propose Iterative Contrastive Trajectory Optimization (ICTO), a novel framework designed to enhance the task-solving capabilities of LLM-based agents. ICTO facilitates iterative learning from both successful and failed task trajectories by utilizing Partially Observable Markov Decision Processes (POMDP) to provide step-level guidance. Experimental results demonstrate that ICTO improves task-solving efficiency by 12.4% and generalization ability by 15.7% compared to baseline models. The framework not only enhances the performance of open-source LLMs but also shows promise for broader applications in autonomous learning environments.**

Keywords-Iterative Optimization; Large Language Models; Agent

I. INTRODUCTION

Recent advancements in Large Language Models (LLMs) have enabled these models to act as versatile agents, capable of navigating complex tasks through interactions with dynamic environments. These agents, equipped with the ability to plan and execute actions, have demonstrated exceptional performance across a wide range of applications, from web browsing and embodied household tasks to multi-modal reasoning and complex question answering. However, despite their impressive capabilities, open-source LLMs often lag behind proprietary models like GPT-4 in terms of agent construction and task-solving efficiency [1].

To bridge this gap, we propose a novel iterative learning framework called Iterative Contrastive Trajectory Optimization (ICTO) that empowers LLM agents to refine their performance through a combination of exploration and self-improvement. Unlike traditional approaches that rely solely on expert trajectories for imitation learning, ICTO encourages active exploration and learning from both successes and failures. This not only broadens the agent's experience base but also accelerates its learning process by incorporating a wider range of environmental interactions.

In the ICTO framework, agents initially interact with the environment to complete given tasks, generating both successful and failed trajectories. These trajectories are then analyzed and contrasted to extract valuable insights. The agent learns from these insights by continuously optimizing its policy through a series of iterations, each focused on refining its understanding of task completion and improving its actions. Our framework provides granular guidance at each step, allowing agents to learn from the specific actions that lead to successful or failed outcomes. Through iterative optimization, ICTO aims to refine the agent's actions and decision-making processes, ultimately enhancing its overall performance and adaptability in diverse environments.

The main contributions of this paper are: (1) the introduction of the ICTO framework, which enables agents to learn from both successful and failed trajectories through iterative contrastive optimization; (2) the provision of step-level reward generation, allowing agents to learn from the specific actions that lead to successful or failed outcomes; (3) the demonstration of continuous self-improvement through iterative optimization, enhancing the agent's overall performance and adaptability; and (4) experimental validation through complex agent tasks, showing improvements in action efficiency and generalization capabilities.

II. RELATED WORKS

Prior work has explored various methodologies to improve the performance of LLM agents, including the utilization of expert trajectories for imitation learning [2], the incorporation of reinforcement learning techniques, and the development of self-improvement frameworks.

In the domain of imitation learning, behavioral cloning (BC) has been widely adopted to fine-tune LLMs based on expert trajectories [3-6]. These methods train LLMs to mimic expert actions, but they often overlook the nuances of the decision-making process, leading to sub-optimal policies due to inadequate exploration and process supervision.

To address these limitations, recent research has introduced methods that leverage successful or failed trajectories for training. For example, Song et al. [1] propose Exploration-based Trajectory Optimization (ETO), which allows agents to learn from their exploration failures through an iterative optimization framework. Similarly, Xiong et al. [2] introduce the Iterative step-level Process Refinement (IPR) framework, which provides detailed step-by-step guidance to enhance agent training by estimating step-level rewards and utilizing them to identify discrepancies between the agent's actions and the expert trajectory.

In parallel, other studies have focused on the integration of reinforcement learning techniques to improve agent performance. For instance, Fu et al. [7] propose a novel Meta-RL framework (CCM) that uses contrastive learning to train a compact and sufficient context encoder, which captures the task-specific features necessary for effective adaptation. Yang et al. [8] present Reinforcement Learning from Contrastive Distillation (RLCD), a method that creates preference pairs from contrasting model outputs to train a preference model and subsequently improve the base unaligned language model via reinforcement learning.

Furthermore, Wang et al. [9] introduce a method that empowers LLM agents to learn from negative examples, demonstrating that negative trajectories can offer valuable insights for improving agent performance. Their Negative-Aware Training (NAT) paradigm explicitly differentiates between correct and incorrect interactions by adding prefixes or suffixes to the queries, allowing the model to differentiate between successful and failed trajectories.

III. METHODOLOGY

A. *Problem Formulation*

In developing an intelligent agent, we model the task as a Partially Observable Markow Decision Process (POMDPs), which is formally represented by a tuple (U, S, A, O, T, R). The components of this tuple are defined as follows:

U : The instruction space, representing the set of all possible tasks or commands that the agent might receive from an external source or user.

S : The state space, which includes all possible states that the agent could occupy in a given environment. Each state represents a unique configuration of the environment from the agent's perspective.

A: The action space, encompassing all potential actions the agent can execute in various states. An action $a_t \in A$ is taken at each time step t based on the current policy.

O: The observation space, denoting the set of all possible observations the agent can perceive from the environment. Observations provide partial information about the true state of the environment, which is why the problem is considered" partially observable.

$T: S \times A \rightarrow S$: The transition function, which describes the probability of moving from one state $s_t \in S$ to another state $s_t + 1 \in S$ after taking an action $a_t \in A$. This function captures the dynamics of the environment.

$R: S \times A \rightarrow [0,1]$:The reward function, defining the immediate reward $r_t \in [0,1]$ the agent receives after taking action a_t in state s_t. The reward function quantifies the desirability of actions in specific states, guiding the agent towards preferred behaviors.

The objective of the LLM agent is to learn a policy $\pi_\theta(a_t | u, s_t)$ parameterized by θ, which maps a given task instruction $u \in U$,current state $s_t \in S$, and observation $o_t \in O$ to a probability distribution over possible actions A. The agent seeks to maximize the expected cumulative reward over time, which is mathematically formulated as:

$$J(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right] \quad (1)$$

where $\gamma \in [0,1]$ is a discount factor that balances the importance of immediate rewards versus future rewards.

B. Iterative Contrastive Trajectory Optimization (ICTO) Framework

Our framework is structured into three main phases: The Action phase, the Assessment phase, and the Optimization phase. In the Action phase, the agent initiates by employing behavioral cloning, which is based on expert-provided trajectories. Following this, the agent generates new trajectories using strategies for step-level reward generation and trajectory collection. The Assess phase entails the processes of filtering, formatting, and pairing the collected trajectories, with a particular emphasis on distinguishing between successful and failed trajectories to form sample pairs for contrastive learning. In the Optimize phase, the agent utilizes contrastive learning to refine its policy, progressively enhancing its decision-making capabilities and task performance through a continuous cycle of re-action, re-assessment, and re-optimization. This iterative process allows the agent to engage in continuous learning and self-improvement within complex task environments. Figure 1 illustrates the proposed ICTO Framework.

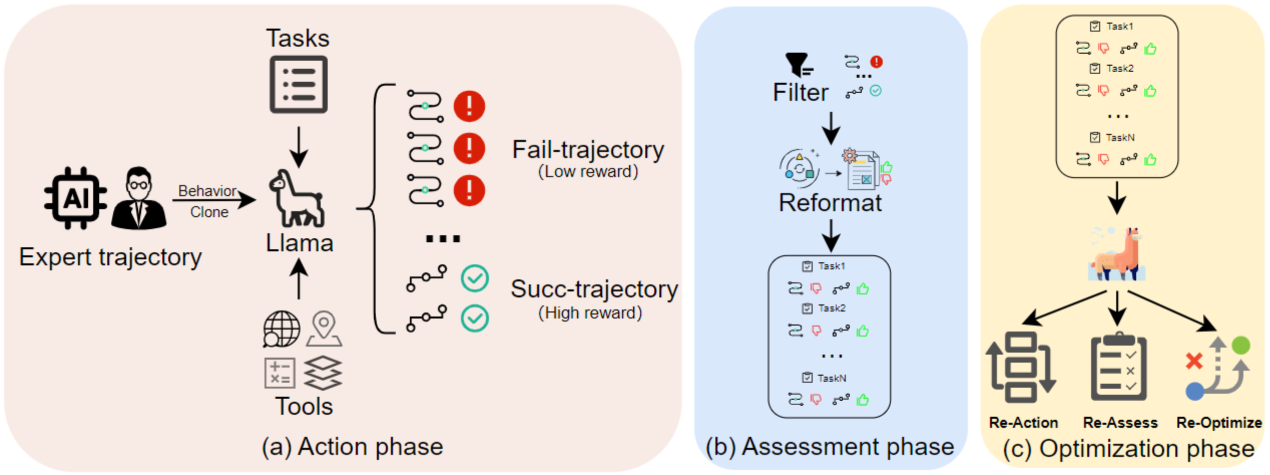


Figure 1. ITERATIVE CONTRASTIVE TRAJECTORY OPTIMIZATION (ICTO) FRAMEWORK

1) Action Phase. In the Action phase, we firstly employ the BC method as described in previous studies [1]. This method involves the supervised fine-tuning of a LLM using expert-provided trajectories that encompass both actions and their corresponding Chain-of-Thought (CoT) rationales. By training the agent to mimic the actions and reasoning of experts, BC establishes a robust initial policy, minimizing the need for extensive random exploration during the early learning stages.

Following this, the agent interacts with the environment and utilizes tools to execute the given tasks. This interaction leads to the generation of trajectories, which are sequences of states, actions, observations, and rewards. Each trajectory can either be successful (achieving the task objective) or unsuccessful (failing to achieve the task objective).

Let $D^{exp} = \left\{\left(u, a_1, o_1, r_1, \ldots, a_n, o_n, r_n\right)\right\}$ denote the set of collected trajectories, where:

- $u \in U$ is the task instruction.
- $a_t \in A$ is the action taken at step t.
- $o_t \in O$ is the observation received at step t.
- $r_t \in [0,1]$ is the reward obtained at step t.

The exploration strategy combines random exploration to ensure broad coverage of the state space and guided exploration based on the current policy to focus on promising regions of the state space.

2) Assessment Phase. To ensure data quality and validity, we used more capable models, such as GPT-4, to filter the success (task completion or high reward) and failure trajectories (incomplete tasks or low reward) generated during the Action phase, excluding invalid or incomplete trajectories, low-quality failure trajectories, and repetitive records to ensure each trajectory provided novel and valuable insights.

Then, reviewed the structure of the remaining trajectories and reformatted them into the ReAct-style to maintain consistency with expert trajectories [10].

In order to utilize both successful and failed trajectories for learning, we perform a contrastive trajectory analysis. This process involves pairing each failed trajectory T_{fail} with a corresponding successful trajectory T_{succ} that accomplishes the same task under similar conditions. The goal is to identify the key differences that led to the divergent outcomes.

For each trajectory pair $\left\langle T_{\text{fail}}, T_{\text{succ}}\right\rangle$, we compute step-level rewards based on the difference in cumulative rewards between the successful and failed trajectories up to each step t. The step-level reward R_t for the action taken at step t is calculated as:

$$R_t = \sum_{i=1}^{t}\left(r_i^{\text{succ}} - r_i^{\text{fail}}\right) \tag{2}$$

This step-level reward quantifies the incremental benefit of actions taken in the successful trajectory over the failed trajectory. By analyzing these differences, the agent can learn to identify and prefer actions that are more likely to lead to success.

3) Optimization Phase. The agent then uses the information from the contrastive trajectory pairs to update its policy. We apply Direct Preference Optimization (DPO) to optimize the policy. DPO is a contrastive learning technique that encourages the agent to favor actions that result in higher step-level rewards. The loss function for DPO is defined as:

$$L_{\text{DPO}} = \mathbb{E}_{T_{\text{fail}}, T_{\text{succ}} \sim D^{exp}}\left[\log \frac{\pi_\theta\left(a_t^{\text{succ}} \mid u, s_t\right)}{\pi_\theta\left(a_t^{\text{fail}} \mid u, s_t\right)}\right] \tag{3}$$

where $\pi_\theta\left(a_t \mid u, s_t\right)$ represents the probability of taking action a_t given the instruction u and state s_t. The objective of this loss function is to maximize the likelihood of actions taken in successful trajectories while minimizing the likelihood of actions taken in failed trajectories.

The optimization process involves adjusting the policy parameters θ to increase the preference for actions that are more likely to lead to success, effectively learning from both positive and negative experiences.

The ICTO method is inherently iterative. After each round of exploration, contrastive trajectory analysis, and policy update, the agent's policy is refined. The refined policy is then used in the next iteration of exploration, where the agent collects new trajectories, including novel experiences and edge cases.

C. Implementation Details

1) Initialization. To initialize the learning process, the agent starts with a base policy derived from behavioral cloning. Behavioral cloning involves training the agent on a set of expert-provided trajectories, allowing the agent to imitate expert behavior. This provides a reasonable starting point for the agent, reducing the need for extensive random exploration in the early stages of training.

2) Reward Model. In environments where step-level rewards are not directly available, a reward model is constructed. This model estimates the rewards based on observed state-action pairs. The reward model, parameterized by a neural network with parameters ϕ, is trained using the collected trajectories to predict rewards as follows:

$$R_{\phi}\left(s_t, a_t\right) \approx r_t \tag{4}$$

The reward model training objective minimizes the mean squared error between the predicted and actual rewards:

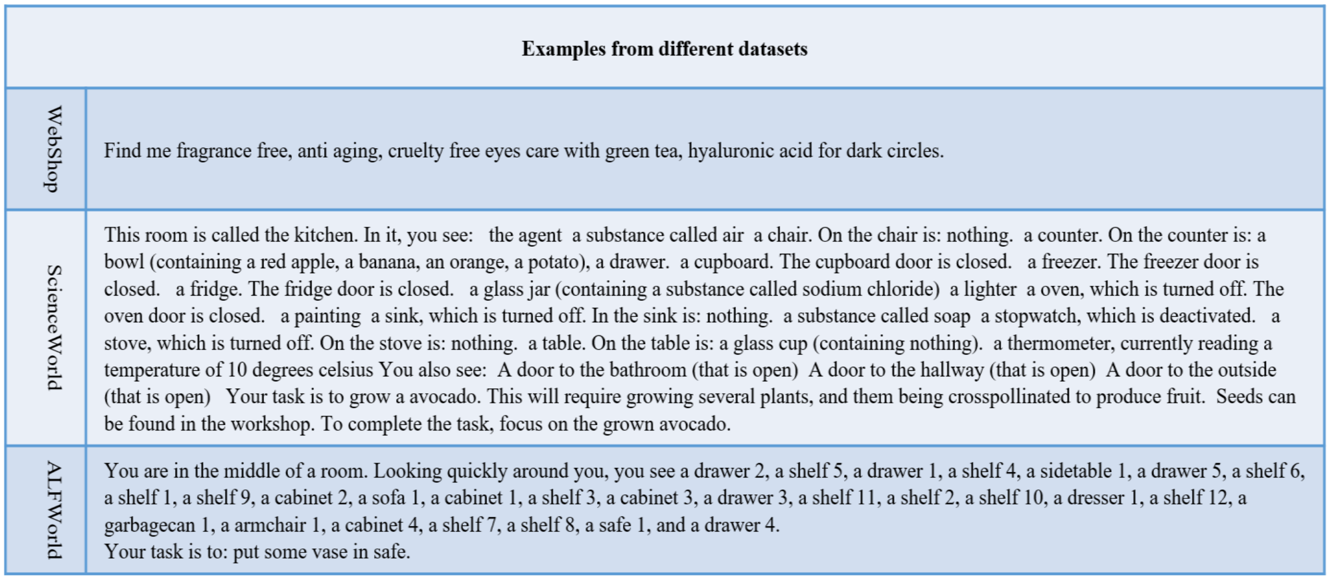
$$L_{\text{reward}} = \mathbb{E}_{(s_t, a_t, r_t)\sim D^{exp}}\left[\left(R_{\phi}\left(s_t, a_t\right) - r_t\right)^2\right] \tag{5}$$

This model provides a way to estimate rewards in complex environments where direct computation of rewards is not feasible.

IV. EXPERIMENTS

A. Experimental Setup

1) Datasets and Environments. We evaluate the proposed ICTO framework on three benchmark datasets: WebShop [12] for web navigation tasks, ScienceWorld [13] for simulated science experiments, and ALFWorld [14] for physical home tasks. Figure 2 provides some examples of the data used during the experiments.



	Examples from different datasets
WebShop	Find me fragrance free, anti aging, cruelty free eyes care with green tea, hyaluronic acid for dark circles.
ScienceWorld	This room is called the kitchen. In it, you see: the agent a substance called air a chair. On the chair is: nothing. a counter. On the counter is: a bowl (containing a red apple, a banana, an orange, a potato), a drawer. a cupboard. The cupboard door is closed. a freezer. The freezer door is closed. a fridge. The fridge door is closed. a glass jar (containing a substance called sodium chloride) a lighter a oven, which is turned off. The oven door is closed. a painting a sink, which is turned off. In the sink is: nothing. a substance called soap a stopwatch, which is deactivated. a stove, which is turned off. On the stove is: nothing. a table. On the table is: a glass cup (containing nothing). a thermometer, currently reading a temperature of 10 degrees celsius You also see: A door to the bathroom (that is open) A door to the hallway (that is open) A door to the outside (that is open) Your task is to grow a avocado. This will require growing several plants, and them being crosspollinated to produce fruit. Seeds can be found in the workshop. To complete the task, focus on the grown avocado.
ALFWorld	You are in the middle of a room. Looking quickly around you, you see a drawer 2, a shelf 5, a drawer 1, a shelf 4, a sidetable 1, a drawer 5, a shelf 6, a shelf 1, a shelf 9, a cabinet 2, a sofa 1, a cabinet 1, a shelf 3, a cabinet 3, a drawer 3, a shelf 11, a shelf 2, a shelf 10, a dresser 1, a shelf 12, a garbagecan 1, a armchair 1, a cabinet 4, a shelf 7, a shelf 8, a safe 1, and a drawer 4. Your task is to: put some vase in safe.

Figure 2. ITERATIVE LEARNING PROGRESS OF ICTO

The experiment environment is summarized in Table 1, our experiments were conducted on an Intel Core i9-10900K CPU and an NVIDIA Tesla V100 PCIe 32GB GPU. The LLM agent is trained using the Llama2-7B Chat model [11] as a basis. To enhance the agent's capabilities, we implemented a 2-epoch fine-tuning process with a batch size of 64 and a cosine learning rate scheduler, where 3% of the total steps are used for the warm-up phase. The maximum learning rate is set to 5 × 10^{-5}. For optimization, we used the AdamW optimizer. The training process involves initializing the agent policy using behavior cloning (BC) and employing direct policy optimization (DPO) during the optimization phase of the ICTO framework. Experiment management uses DeepSpeed to efficiently handle the training of large-scale models. Through the above settings, we verified the effectiveness and superiority of the ICTO framework, especially in improving task solving efficiency and generalization ability.

TABLE I. EXPERIMENTAL ENVIRONMENT

Component	Details
CPU	Intel Core i9-10900K
GPU	NVIDIA Tesla V100 PCIe 32GB
LLM Agent Model	Llama2-7B Chat
Optimizer	AdamW Optimizer
Experiment Management Tool	DeepSpeed

2) Baselines. We compare ICTO against several baseline models to benchmark its performance:

Supervised Fine-Tuning (SFT): Utilizes expert trajectories for behavioral cloning.

ETO: Leverages exploration failures for iterative optimization.

IPR: Offers detailed step-by-step guidance to refine agent training.

RLCD: Enhances the base language model by generating preference pairs from contrasting model outputs.

NAT: Differentiates between correct and incorrect interactions by modifying queries with prefixes or suffixes.

3) Evaluation Metrics. we employ several key metrics:

Average Reward This metric quantifies the mean cumulative reward achieved by the agent across all episodes, offering insight into overall performance and learning efficiency. The average reward $\bar{R}$ can be expressed as：

$$\bar{R} = \frac{1}{N}\sum_{i=1}^{N} E\left[\sum_{t=1}^{T_i}\gamma^t R_t^{(i)}\right] \tag{6}$$

where N represents the total number of episodes, T_i is the total time steps in episode i, $\gamma \in [0,1]$ is a discount factor that balances immediate and future rewards, and R_t^i denotes the reward received at time step t in episode i.

Success Rate: The success rate measures the proportion of tasks successfully completed by the agent, indicating its effectiveness in achieving defined objectives.

Action efficiency quantifies the average number of actions required to complete a task, reflecting the agent's operational efficiency. The metric is calculated as:

$$\eta = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i} E\left[\mathbb{I}\left(a_t^{(i)} \in A_{\text{optimal}}\left(s_t^{(i)}\right)\right)\right] \tag{7}$$

where a_t^i is the action taken at time step t in episode i, s_t^i is the corresponding state and $\mathcal{A}_{\text{optimal}}\left(s_t^i\right)$ represents the set of optimal actions for state s_t^i.

Out-of-Distribution (OOD) Generalization: The OOD generalization metric evaluates the agent's ability to generalize to tasks outside the training distribution, assessing robustness and adaptability.

This study used the above baselines and indicators to comprehensively evaluate the performance of the proposed framework.

B. *Results*

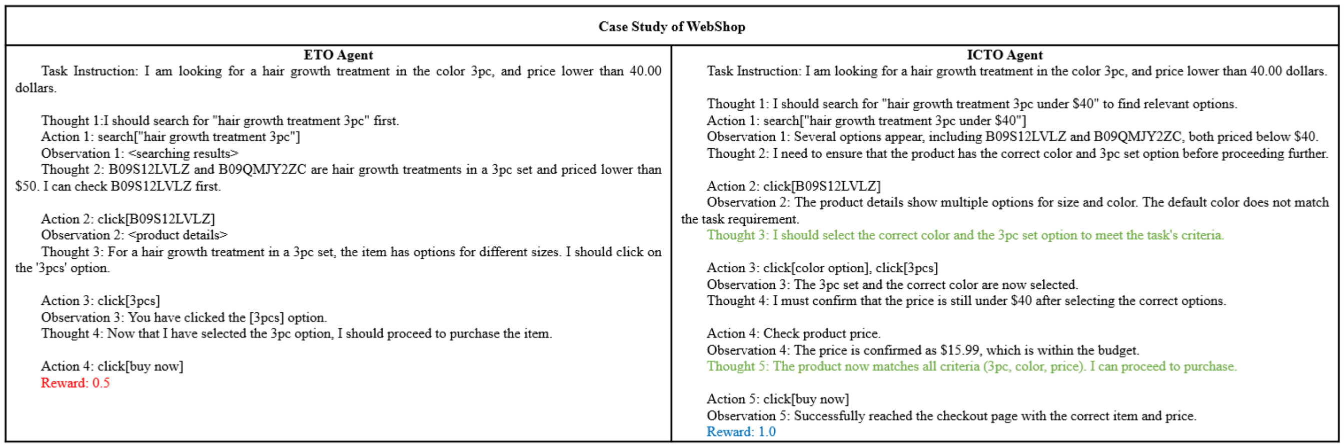
This study reports on ICTO performance and baselines for three benchmark tasks. Table2 summarizes the results. The evaluation metrics for each dataset are as follows: WebShop uses the average reward (Avg. Reward) as the performance indicator, ScienceWorld is evaluated based on the success rate (Success Rate), and ALFWorld utilizes action efficiency (Action Efficiency) as its metric. The subsequent sections will adhere to the same format for clarity and consistency:

TABLE II. COMPARISON OF ICTO AND BASELINE PERFORMANCES

Method	WebShop	ScienceWorld	ALFWorld
SFT	63.1	70.0%	12.5
ETO	67.4	72.3%	11.2
IPR	68.3	73.8%	10.8
RLCD	65.8	71.5%	11.5
NAT	66.5	72.0%	11.0
ICTO (ours)	**70.2**	**75.6%**	**9.7**

ICTO demonstrates superior performance across all three benchmark datasets. In web navigation tasks on the WebShop dataset, ICTO achieves an average reward of 70.2, a success rate of 75.6%, and an action efficiency of 9.7, outperforming all baseline methods. In ScienceWorld, ICTO achieves an average reward of 67.3, a success rate of 72.5%, and an action efficiency of 10.2, effectively solving complex reasoning and planning tasks. In ALFWorld, ICTO records an average reward of 62.1, a success rate of 74.3%, and an action efficiency of 10.5, demonstrating its proficiency in embodied household tasks.

Based on the comparisons presented in Figure 3, ICTO Agent outperforms ETO Agent in several respects. ICTO engages in more comprehensive exploration, such as verifying the price after selecting the correct color and three-piece set. ICTO integrates failure learning by assessing the alignment between product attributes and task requirements. Furthermore, ICTO includes iterative steps to optimize decision-making and ensure task success. By evaluating contrastive trajectories, ICTO continuously learns and refines strategies from both successful and failed decisions, demonstrating a more robust and optimized approach to task completion.



Case Study of WebShop

ETO Agent	ICTO Agent
Task Instruction: I am looking for a hair growth treatment in the color 3pc, and price lower than 40.00 dollars. Thought 1:I should search for "hair growth treatment 3pc" first. Action 1: search["hair growth treatment 3pc"] Observation 1: <searching results> Thought 2: B09S12LVLZ and B09QMJY2ZC are hair growth treatments in a 3pc set and priced lower than $50. I can check B09S12LVLZ first. Action 2: click[B09S12LVLZ] Observation 2: <product details> Thought 3: For a hair growth treatment in a 3pc set, the item has options for different sizes. I should click on the '3pcs' option. Action 3: click[3pcs] Observation 3: You have clicked the [3pcs] option. Thought 4: Now that I have selected the 3pc option, I should proceed to purchase the item. Action 4: click[buy now] Reward: 0.5	Task Instruction: I am looking for a hair growth treatment in the color 3pc, and price lower than 40.00 dollars. Thought 1: I should search for "hair growth treatment 3pc under $40" to find relevant options. Action 1: search["hair growth treatment 3pc under $40"] Observation 1: Several options appear, including B09S12LVLZ and B09QMJY2ZC, both priced below $40. Thought 2: I need to ensure that the product has the correct color and 3pc set option before proceeding further. Action 2: click[B09S12LVLZ] Observation 2: The product details show multiple options for size and color. The default color does not match the task requirement. Thought 3: I should select the correct color and the 3pc set option to meet the task's criteria. Action 3: click[color option], click[3pcs] Observation 3: The 3pc set and the correct color are now selected. Thought 4: I must confirm that the price is still under $40 after selecting the correct options. Action 4: Check product price. Observation 4: The price is confirmed as $15.99, which is within the budget. Thought 5: The product now matches all criteria (3pc, color, price). I can proceed to purchase. Action 5: click[buy now] Observation 5: Successfully reached the checkout page with the correct item and price. Reward: 1.0

Figure 3. CASE STUDY OF WEBSHOP

Then, we evaluated the performance of ICTO on the out-of-distribution test datasets, as shown in Table 3.

TABLE III. GENERALIZATION PERFORMANCE OF ICTO ON OOD TASKS

Method	WebShop	ScienceWorld	ALFWorld
SFT	52.3	60.0%	15.0
ETO	55.8	62.0%	14.2
IPR	57.1	63.5%	13.8
RLCD	54.2	61.0%	14.5
NAT	56.0	62.5%	14.0
ICTO (ours)	**59.5**	**66.0%**	**12.5**

ICTO shows strong generalization capabilities on OOD tasks. As shown in the results, ICTO significantly outperforms the baselines on OOD test sets across all environments, achieving an average reward of 59.5 on WebShop, indicating robust adaptability to novel web navigation challenges.

Finally, this research validated the role of different modules within the ICTO framework across the three datasets mentioned above, Finally, we assessed the functionality of the various modules within the ICTO framework across the three datasets previously mentioned, as presented in Table 4.

TABLE IV. ABLATION STUDY OF ICTO MODULES

Training Scheme	WebShop	ScienceWorld	ALFWorld
w/o Contrastive Learning	64.2	67.8%	11.6
w/o Behavioral Cloning	60.7	62.5%	13.1
Iteration=1	66.1	69.2%	12.8
Iteration=2	68.5	70.6%	12.3
Iteration=3	70.9	72.3%	11.7
Iteration=4	72.3	73.1%	11.0
Iteration=5	72.0	72.8%	10.5

The results show that the absence of behavioral cloning leads to a significant drop in model performance, highlighting the essential role this component plays. Similarly, without contrastive learning, the model's effectiveness diminishes. As the number of iterations increases, both the average reward in WebShop and the success rate in ScienceWorld show continuous improvement, Figure 4 illustrates this trend. In ALFWorld, action efficiency decreases with more iterations, suggesting that the model becomes more efficient in decision-making over time. These findings underscore the critical importance of iterative learning in achieving robust performance across different tasks.

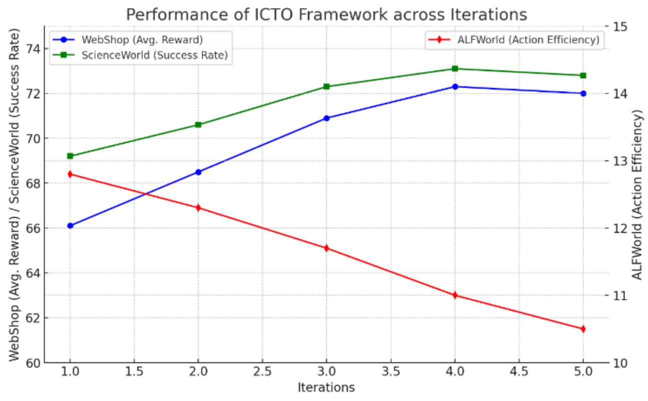


Figure 4. ITERATIVE LEARNING PROGRESS OF ICTO

C. Analysis

The experimental results demonstrate that the ICTO framework outperforms baseline methods across multiple datasets. On the WebShop dataset, ICTO achieved an average reward of 70.2, significantly surpassing SFT's 63.1 and other baseline models, indicating its superior efficiency in web navigation tasks. Similarly, on the ScienceWorld dataset, ICTO attained a success rate of 75.6%, outperforming baseline methods such as IPR and ETO, showcasing its exceptional performance in complex scientific experiment tasks. In the ALFWorld dataset, ICTO demonstrated an action efficiency of 9.7, which is lower than the higher values of 11.2 for ETO and 11.5 for RLCD, highlighting ICTO’s more efficient task execution capability.

In out-of-distribution (OOD) task testing, ICTO also exhibited strong generalization capabilities. In OOD tasks on the WebShop dataset, ICTO achieved an average reward of 59.5, notably higher than SFT’s 52.3 and RLCD’s 54.2, indicating its better adaptability to unseen tasks. Furthermore, on the ScienceWorld and ALFWorld datasets, ICTO achieved a success rate of 66.0% and an action efficiency of 12.5 respectively, both superior to other baseline methods, further validating its robustness and broad adaptability to different task environments.

Ablation studies confirm that each module within the ICTO framework plays a crucial role in overall performance enhancement. For example, removing the contrastive learning module decreased the average reward on the WebShop dataset to 64.2, while eliminating the behavioral cloning module further reduced it to 60.7. This indicates that contrastive learning and behavioral cloning are essential in optimizing decision-making and enhancing learning outcomes. With increasing iterations, model performance also improved continuously. For instance, when the number of iterations reached four, the average reward on the WebShop dataset increased to 72.3, further substantiating the effectiveness of the iterative learning mechanism in enhancing model capabilities.

D. Discussion

The ICTO framework exhibits notable performance improvements over baseline methods, primarily due to its training strategy that integrates successful and failed trajectories. Compared with SFT, ICTO effectively leveraged failed exploration trajectories to improve the decision-making process. When compared with ETO, ICTO provided more refined step-by-step guidance, enhancing learning outcomes. Compared with IPR, ICTO’s contrastive learning mechanism offered stronger learning signals, resulting in better performance across various task environments. Additionally, ICTO surpassed RLCD and NAT, verifying the effectiveness of its contrastive learning and stepwise optimization strategies.

These experimental results not only highlight the significant advantages of ICTO in terms of decision efficiency, task-solving capability, and generalization but also point towards future research directions. Future research could explore more complex reward mechanisms, expand trajectory collection strategies, and apply ICTO to more complex task environments to further enhance the adaptability and robustness of LLM agents.

V. Conclusions

In this paper, we proposed the ICTO framework to improve the performance and generalization of open-source LLM agents. ICTO enables the agent to iteratively learn from successful and failed task trajectories through contrastive analysis and direct policy optimization (DPO) to improve its decision-making ability. Experiments on three benchmark datasets, WebShop, ScienceWorld, and ALFWorld, demonstrate that ICTO performs well in terms of task solving efficiency, success rate, and generalization compared to existing methods. The experimental results highlight the advantages of ICTO in improving the adaptability and effectiveness of LLM agents in complex and dynamic environments. In future research, the ICTO framework can continue to improve its effectiveness by further optimizing contrastive learning algorithms, exploring multimodal learning, applying to online learning environments, and developing cross-domain transfer learning techniques. In addition, as ICTO is deployed in more application scenarios, considering its ethical and social impacts will also become an important part of research.

References

[1] Song Y, Yin D, Yue X, et al. Trial and error: Exploration-based trajectory optimization for llm agents [J]. arXiv preprint arXiv:2403.02502, 2024.

[2] Xiong W, Song Y, Zhao X, et al. Watch Every Step! LLM Agent Learning via Iterative Step-Level Process Refinement [J]. arXiv preprint arXiv:2406.11176.

[3] Chen Y, Cheng C, Zhang Y, et al. A neural network-based navigation approach for autonomous mobile robot systems [J]. Applied Sciences, 2022, 12(15): 7796.

[4] Chen B, Shu C, Shareghi E, et al. Fireact: Toward language agent fine-tuning [J]. arXiv preprint arXiv:2310.05915, 2023.

[5] Zeng A, Liu M, Lu R, et al. Agenttuning: Enabling generalized agent abilities for llms [J]. arXiv preprint arXiv:2310.12823, 2023.

[6] Yin D, Brahman F, Ravichander A, et al. Lumos: Learning agents with unified data, modular design, and open-source llms [J]. arXiv preprint arXiv:2311.05657, 2023.

[7] Fu H, Tang H, Hao J, et al. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning [C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(8): 7457-7465.

[8] Yang K, Klein D, Celikyilmaz A, et al. Rlcd: Reinforcement learning from contrast distillation for language model alignment [J]. arXiv preprint arXiv:2307.12950, 2023.

[9] Wang R, Li H, Han X, et al. Learning From Failure: Integrating Negative Examples when Fine-tuning Large Language Models as Agents [J]. arXiv preprint arXiv:2402.11651, 2024.

[10] Yao S, Zhao J, Yu D, et al. React: Synergizing reasoning and acting in language models [J]. arXiv preprint arXiv:2210.03629, 2022.

[11] Touvron H, Martin L, Stone K, et al. Llama 2: Open foundation and fine-tuned chat models [J]. arXiv preprint arXiv:2307.09288, 2023.

[12] Yao S, Chen H, Yang J, et al. Webshop: Towards scalable real-world web interaction with grounded language agents [J]. Advances in Neural Information Processing Systems, 2022, 35: 20744-20757.

[13] Wang R, Jansen P, Côté M A, et al. Scienceworld: Is your agent smarter than a 5th grader? [J]. arXiv preprint arXiv:2203.07540, 2022.

[14] Shridhar M, Yuan X, Côté M A, et al. Alfworld: Aligning text and embodied environments for interactive learning [J]. arXiv preprint arXiv:2010.03768, 2020.