# When Instinct Guides and Insight Grounds: Staged RL Training for LLM Agents

**Zijing Zhang**[1*], **Boning Zhang**[2*†]

[1]Peking University
[2]Institute of Automation, Chinese Academy of Sciences
zijingzhang@stu.pku.edu.cn

## Abstract

Large Language Model (LLM) agents have demonstrated strong potential in complex, interactive decision-making tasks. However, when training LLM agents end-to-end with reinforcement learning (RL), efficiently optimizing agent policies in dynamic environments remains a significant challenge. Existing RL-based LLM agent paradigms commonly organize interactions in a cycle where reasoning is followed by action. In our work, we observe a phenomenon we call *Exploration Contraction*, where the explicit introduction of a reasoning stage reduces the diversity of actions—quantified by lower action entropy—which in turn limits exploration and leads to premature policy convergence. To address this limitation, we propose Act-before-Reasoning (ActRe), a two-stage RL training framework. In the first stage, we reverse the typical rollout order, prompting the agent to generate actions prior to reasoning, which encourages exploration driven by model intuition. In the second stage, we restore the standard reasoning-then-action order for training and evaluation, ensuring robust and interpretable decision-making. Experiments on the ALFWorld and WebShop benchmarks show that ActRe effectively mitigates exploration contraction, yielding consistently higher task success rates and improved training robustness compared to strong RL baselines. Our analysis underscores the importance of action entropy in the exploration-exploitation trade-off during LLM agent training and provides a practical approach to maintain the benefits of explicit reasoning while promoting sufficient exploration.

## Introduction

Large language model (LLM) agents have significantly extended their capabilities for complex decision-making and practical applications through multi-turn interactions with external environments (Huang et al. 2023; Abuelsaad et al. 2024; Agashe et al. 2024; Bai et al. 2024; Zeng et al. 2024). Unlike static tasks such as mathematical reasoning or programming (Setlur et al. 2024; Liu et al. 2025), interactive tasks require agents to adapt and respond flexibly to continuous feedback while maintaining robustness under uncertainty and changing conditions (Yao et al. 2023; Shinn et al. 2023).

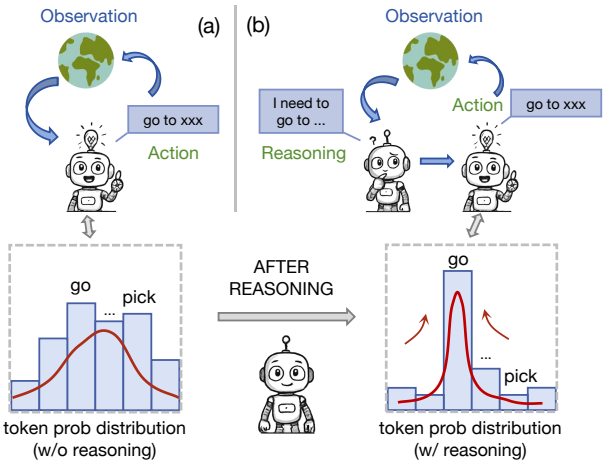



Figure 1: We observe that during RL rollout sampling, the action token entropy of the LLM decreases after deliberate thinking(b), compared to directly generating actions without reasoning(a). This reduction in entropy limits exploration, a phenomenon we term ***Exploration Contraction***.

One of the most representative paradigms for LLM agents is ReAct (Yao et al. 2023), which leverages the model's context understanding and reasoning abilities. In each interaction round, the agent reasons first, then acts. Recent studies have explored the potential of end-to-end reinforcement learning (RL) systems in multi-turn interaction tasks (Wang et al. 2025c,a; Feng et al. 2025a). Similar to ReAct, RL-based generation also follows a cycle of "observation – thinking – action", enabling better alignment with the environment and achieving better performance than base models in certain settings. However, current RL-based methods still suffer from the challenge of Early Policy Convergence, particularly in exploration-intensive environments.

Our work builds on an interesting observation: during RL training in environments that require extensive trial-and-error, LLMs can sometimes achieve better performance by taking actions directly rather than relying on an explicit reasoning process. RAGEN (Wang et al. 2025c) simply attributes this phenomenon to "reasoning degradation". We

*These authors contributed equally.
†Corresponding author.

conduct a deeper analysis of this issue by examining the dynamics of entropy in large models during RL training. While recent studies have shown that premature entropy reduction in RL can suppress exploration in LLMs (Yu et al. 2025; Wang et al. 2025b; Cui et al. 2025), we observe that, unlike tasks such as mathematical reasoning where the diversity of the complete output (reasoning plus action) is crucial, the key factor for agent-environment interaction is the diversity in action generation at each step. Thus, previous approaches and analyses cannot be directly applied to RL agent tasks. We measure action entropy, which is the average entropy of action tokens at each step, to assess exploration diversity. Our results show that adding the reasoning stage reduces action entropy, making the agent's decisions more conservative and limiting exploration. We refer to this phenomenon as *Exploration Contraction*, as illustrated in Figure 1.

Since previous studies have shown that the reasoning process is crucial for improving the robustness of LLM agents (Wang et al. 2025c; Fu et al. 2025), it is not appropriate to simply remove the reasoning stage. To address exploration contraction, we propose Act-before-Reasoning (ActRe). ActRe uses a two-stage RL training strategy. In the first stage, we reverse the standard reasoning-then-action rollout order to first generate the action, then generate the corresponding reasoning. This encourages the LLM to rely on intuition at decision time, promoting exploration and increasing the probability of both correct actions and reasoning chains. In the second stage, we restore the standard reasoning-then-action order, aligning with ReAct, to ensure agents perform well under real-world evaluation.

Experiments on ALFWorld and WebShop demonstrate that ActRe effectively alleviates Exploration Contraction and consistently outperforms baselines across various tasks and environments. Retaining the reasoning process also improves generalization to unseen tasks.

In summary, our main contributions are as follows:

1. To the best of our knowledge, we are the first to systematically identify and analyze the phenomenon of Exploration Contraction during RL agent training, providing new insights from the perspective of action entropy and the exploration-exploitation trade-off.

2. We propose the ActRe training strategy, which encourages exploration during RL training and effectively improves LLM agents' performance in interactive environments.

3. We conduct extensive experiments across multiple tasks and domains, showing that ActRe consistently improves task success rates and training stability.

## Preliminary

### Long Horizon Tasks

In tasks requiring long-range reasoning, the agent's multi-step interaction with the environment can be formalized as a Markov decision process (MDP). Let $\mathcal{S}$ denote the state space, $\mathcal{A}$ the action space, and $\mathcal{O}$ the observation space. At each step $t$, the state evolves according to the transition function $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, and rewards are given by $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

In our setting, all states, actions, and observations are expressed as finite token sequences in natural language. On each decision step, the LLM agent first generates a reasoning segment $t_t$ based on the historical trajectory. This think phase serves as a semantic intermediary, guiding subsequent action selection. Next, conditioned on the current observation $o_t$ and the generated reasoning $t_t$, the agent produces the action $a_t$ in an autoregressive manner, composing the action as a sequence of tokens. Specifically, the action $a_t$ is decoded token by token as:

$$a_t = (a_t^{(1)}, ..., a_t^{(m)}) \sim \prod_{i=1}^{m} \pi_\theta(a_t^{(i)} \mid o_t, t_t, a_t^{(<i)}, h_{t-1})$$

where $a_t^{(i)}$ is the $i$-th token of the action $a_t$, and $h_{t-1}$ indicates the complete interaction history up to step $t-1$. The full episode trajectory is then:

$$\tau = (o_1, t_1, a_1, o_2, t_2, a_2, \ldots, o_n, t_n, a_n)$$

In addition to the token level, the sequence of actions at the action level determines the direction of the task, making action-level diversity directly relevant to the exploration-exploitation trade-off.

### Group Relative Policy Optimization

Group Relative Policy Optimization (GRPO) (Feng et al. 2025b) rethinks advantage estimation by discarding the explicit value function and instead adopting a group-based relative perspective. For a given input $(q, a)$, the behavior policy $\pi_{\theta_{\text{old}}}$ generates $G$ independent candidate responses $\{o_i\}_{i=1}^{G}$. The reward for each response $R_i$ is collected, and its group-relative advantage is calculated by z-score normalization within the batch:

$$\tilde{A}_{i,t} = \frac{R_{i,t} - \mu_R}{\sigma_R}$$

where $\mu_R$ and $\sigma_R$ are the mean and standard deviation of all $R_{i,t}$ in the current group, ensuring that each response's advantage is interpreted in the context of its peers.

The optimization objective is defined as follows:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\Big[\,\text{clip}(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\,\tilde{A}_{i,t} - \beta\, D_{\text{KL}}\big(\pi_\theta(\cdot|h_{i,t}) \,\|\, \pi_{\theta_{\text{old}}}(\cdot|h_{i,t})\big)\Big]$$

where

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})}$$

and $h_{i,t}$ denotes the current history prefix. The KL penalty, weighted by $\beta$, explicitly controls how far the updated policy can move from its previous iteration.

### Policy Entropy

Policy entropy serves as a fundamental metric to quantify the uncertainty or randomness in the actions chosen by a policy model. Specifically, for a parameterized policy $\pi_\theta$, and a training dataset $D$, policy entropy measures how unpredictably the model assigns probabilities to possible outputs during training. Formally, we compute the average token-level entropy of the policy as follows:

$$H(\pi_\theta, D) = -\mathbb{E}_{D,\pi_\theta}\left[\log \pi_\theta(y_t \mid y_{<t}, x)\right] = -\frac{1}{|D|}\sum_{x\in D}\frac{1}{|y|}\sum_{t=1}^{|y|}\mathbb{E}_{y_t\sim\pi_\theta}\left[\log \pi_\theta(y_t \mid y_{<t}, x)\right]$$

Here, $|D|$ denotes the number of samples in the training set, and $|y|$ represents the length of the target output sequence for each sample $x$. This metric evaluates, at each generation step $t$, how distributed or concentrated the model's predicted probability $\pi_\theta(y_t \mid y_{<t}, x)$ is across the vocabulary. Higher policy entropy corresponds to greater randomness and diversity in action selection, which can encourage exploration but may also indicate less certainty in predictions. Conversely, lower entropy implies more deterministic behavior, reflecting higher model confidence. Quantifying policy entropy is thus crucial for analyzing the trade-off between exploration and exploitation, and for diagnosing whether a policy is overconfident or overly uncertain during optimization (Wang et al. 2025b; Cui et al. 2025).

## Exploration Contraction

### Settings

We conduct experiments on two highly representative long-horizon benchmark datasets: ALFWorld (Shridhar et al. 2020) and WebShop (Yao et al. 2022). ALFWorld is a simulated household environment benchmark encompassing six diverse task types, such as object pickup, placement, cleaning, and heating. Agents are required to complete these tasks via multi-step reasoning and exploration within a virtual environment, enabling assessment of generalization capabilities in domestic scenarios. WebShop, on the other hand, is a large-scale online shopping environment comprising approximately 12,000 human-authored instructions. It includes tasks such as product search, comparison, and purchase, serving to evaluate agent interaction and decision-making skills in e-commerce settings. Together, these two datasets provide comprehensive coverage of the domains of household automation and online shopping, facilitating robust evaluation of agent performance across distinct, real-world-inspired environments.

### RL without Explicit Reasoning

The LLM agent is typically designed following the ReAct (Yao et al. 2023) paradigm. At each decision step $t$, the agent first generates a reasoning segment $t_t$, then autoregressively generates the action $a_t$. The agent samples the action as

| | Model | ALFWorld | | Webshop | |
|---|---|---|---|---|---|
| | | origin | w/o think | origin | w/o think |
| | gpt-4o | 45.0 | 22.7 | 31.8 | 20.3 |
| 1.5B | sft | 39.7 | 19.9 | 26.4 | 12.8 |
| | grpo | 72.8 | 83.6 | 60.1 | 62.8 |
| | ppo | 64.6 | 61.4 | 51.5 | 49.6 |
| | dapo | 70.5 | 78.9 | 62.3 | 64.0 |
| 7B | sft | 63.8 | 46.4 | 60.3 | 50.6 |
| | grpo | 77.6 | 86.8 | 66.1 | 72.9 |
| | ppo | 80.4 | 82.3 | 68.7 | 67.4 |
| | dapo | 78.7 | 83.2 | 69.2 | 71.6 |

Table 1: Comparison of pass rates using different methods on ALFWorld and Webshop, respectively employing the original optimization approach (origin) and the approach without explicit reasoning process (w/o think).

$$a_t = (a_t^{(1)}, \ldots, a_t^{(m)}) \sim \prod_{i=1}^{m}\pi_\theta(a_t^{(i)} \mid o_t, t_t, a_t^{(<i)}, h_{t-1}),$$

where $a_t^{(i)}$ denotes the $i$-th token in the action. The reasoning segment $t_t$ is produced by $\pi_\theta$ given the interaction history $h_{t-1}$ and current observation $o_t$.

For the no-think variant, we remove the agent's explicit reasoning process. In this case, the agent generates actions directly without producing a "think" segment:

$$a_t \sim \prod_{i=1}^{m}\pi_\theta(a_t^{(i)} \mid o_t, a_t^{(<i)}, h_{t-1}),$$

We use Qwen2.5-1.5B and Qwen2.5-7B models, and train them with various optimization methods, including SFT, GRPO, PPO, and DAPO, across different scales. We find that, for non-RL methods, removing the reasoning process leads to a drop in performance, suggesting that explicit reasoning helps the agent make more rational decisions. However, for most RL methods, removing reasoning does not hurt performance, and in some cases, it even improves it, as shown in Table 1.

Figure 2 (left) provides further evidence. On ALFWorld, GRPO-trained agents without explicit "think" tokens achieve learning curves that match or surpass those with reasoning. This suggests that the reasoning phase, although generally helpful, may introduce unnecessary constraints or negative effects under RL optimization.

### Action Entropy During RL Process

Recent works (Wang et al. 2025b; Cui et al. 2025) analyze entropy dynamics in RL training but predominantly focus on single-turn, isolated tasks such as math, where the LLM-generated sequence is treated as flat and policy entropy is measured by the mean token-level entropy. In agentic RL tasks, however, LLMs interact with environments

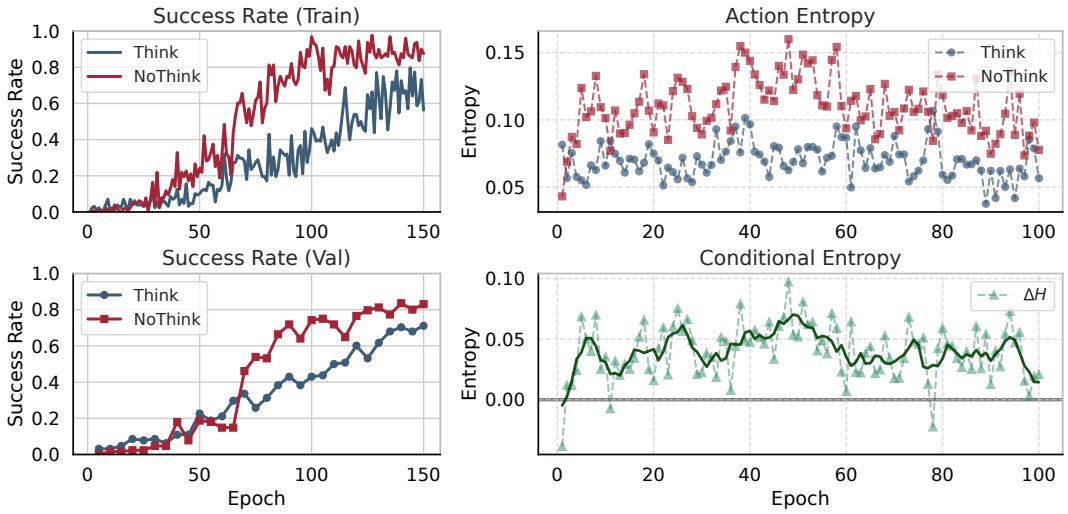



Figure 2: Left: Training and evaluation curves of the Qwen1.5B model on ALFWorld under both think and no-think settings. Right: Comparison of action entropy and conditional action entropy dynamics between the two settings during training.

by generating structured outputs that comprise both reasoning and a final action designed to be parsed by the environment. Crucially, it is the diversity of these environment-interpretable actions——rather than the diversity of the reasoning process——that determines policy diversity. In this context, the reasoning process does not directly determine task success; rather, it influences outcomes only indirectly by shaping the conditional autoregressive probability over the final action tokens. Furthermore, in each step's response, the action segment typically constitutes only a small fraction of the overall output.

Therefore, prior conclusions such as the effect of high-entropy tokens on LLM-generated sequence diversity do not directly reflect policy diversity (Wang et al. 2025b), but instead correspond to the diversity of reasoning paths. Figure 3 (ii) illustrates this: while the agent may generate diverse reasoning processes, these ultimately yield the same distribution over actions. Consequently, previous token-level entropy metrics do not accurately characterize the diversity of agent behavior in these settings.

To better measure policy diversity and the exploration-exploitation trade-off in agent RL, we use action entropy. Specifically, we prompt the LLM to output actions within special tags. Let $a_t = (a_t^{(1)}, a_t^{(2)}, ..., a_t^{(k)})$ denote the action token sequence generated at step $t$, where each token is generated autoregressively. The action entropy at step $t$ is computed as:

$$H_{\text{action}}^{(t)} = \frac{1}{k} \sum_{j=1}^{k} \left( - \sum_{a} \pi(a \mid s_t, a_t^{(<j)}) \log \pi(a \mid s_t, a_t^{(<j)}) \right)$$

Here, $a_t^{(<j)}$ denotes the action prefix up to token $j-1$, and the entropy at each position is calculated over the action vocabulary given the current state $s_t$ and previous action tokens. This formulation captures the average uncertainty in the action token selection process at each decision step, directly reflecting policy diversity.

We track $H_{\text{action}}^{(t)}$ across training epochs and compare its trend under both think and no-think settings (see Figure 2).

For each decision, we report:

- $H(\text{action} \mid \text{obs}, \text{think})$: action entropy conditioned on both observation and reasoning,
- $H(\text{action} \mid \text{obs})$: action entropy conditioned only on the same observation.

The key difference between the two metrics is whether the LLM explicitly generates a reasoning process that directly modulates the distribution over action tokens via autoregressive conditioning.

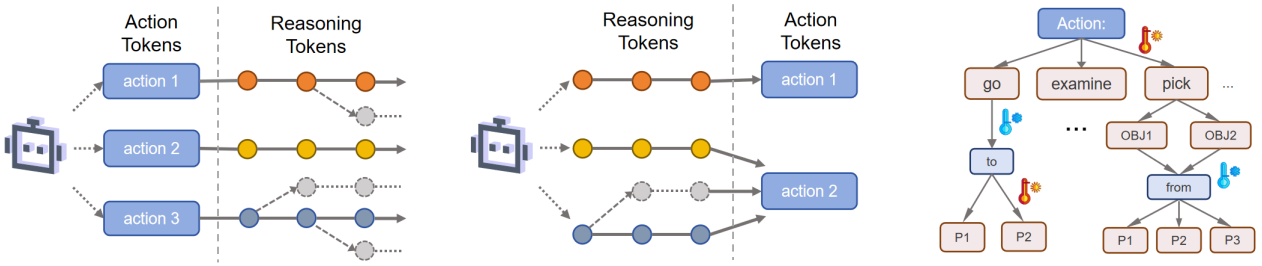
## Conditional Action Entropy

To better understand the influence of thinking on the action distribution, we first introduce the theoretical foundation of conditional entropy. In information theory, conditional entropy $H(Y|X)$ quantifies the average uncertainty remaining in a random variable $Y$ after knowing another variable $X$. It measures how much additional information is needed to specify $Y$ when $X$ is given.

Building on this, we define conditional action entropy in our context. Specifically, $H_{\text{no-think}} = H(\text{action}|\text{obs})$ denotes the conditional action entropy given only the observation, while $H_{\text{think}} = H(\text{action}|\text{obs}, \text{think})$ conditions further on the explicit reasoning process. The reduction in action entropy due to reasoning is thus defined as:

$$\Delta H = H(\text{action}|\text{obs}) - H(\text{action}|\text{obs}, \text{think})$$

This quantity $\Delta H$ directly reflects how much the agent's reasoning process constrains the diversity of subsequent action tokens.




(i) Stage 1: Action-Prioritized (ii) Stage 2: Reasoning-Prioritized (iii) Sampling Action Tokens

Figure 3: Our two-stage agent RL framework: (i) Action-prioritized sampling for exploration; (ii) Reasoning-prioritized sampling for exploitation and inference alignment; (iii) High-temperature sampling for key branching action tokens. Notably, words in the figure represent tokens for simplicity, with actual units depending on the tokenizer.

Figure 2 (bottom right) shows the evolution of this conditional entropy difference throughout training in the ALFWorld environment. Empirically, we observe that $\Delta H$ generally remains positive and statistically significant, indicating that reasoning tends to reduce action entropy. In other words, after engaging in explicit reasoning, the agent's action tokens are typically less diverse than when reasoning is omitted. This supports our Exploration Contraction hypothesis.

This phenomenon is largely due to the autoregressive nature of large language models: once the agent produces a sequence of coherent reasoning tokens, these tokens act as a strong prior for the following action tokens, resulting in more deterministic and self-consistent actions. In some cases, the reasoning step almost fully determines the subsequent action, and as the agent's reasoning ability improves, the action distribution becomes even sharper.

From an algorithmic perspective, the reduction in action entropy via reasoning may be beneficial for ReAct-style agents, as it makes decisions more rational and context-aware. However, in RL scenarios, excessive reduction in action entropy can limit policy exploration, leading to overly conservative strategies that may hinder learning efficiency. Therefore, maintaining a proper balance between exploiting high-quality reasoning and preserving sufficient policy diversity for exploration is an important open challenge in LLM-driven RL.

## Instinct-to-Insight RL Training

Our methodology bridges instinctive exploration and insightful reasoning through a two-stage reinforcement learning framework for LLM agents. In the first stage, we front-load action token generation ahead of reasoning, allowing the agent to rely on instinct to select actions and subsequently guide the reasoning process. This instinct-driven approach encourages broader exploration and supports the discovery of more diverse, potentially correct reasoning trajectories. In the second stage, we reintroduce insight by realigning the agent with the conventional reasoning-then-action paradigm, promoting grounded decision-making. We detail the technical formulation and training procedure in the following sections.

### End-to-End RL Training with GRPO

We utilize an end-to-end reinforcement learning framework to train LLM agents using Generalized Reinforcement Policy Optimization (GRPO). The agent engages in multi-turn episodes with the environment. At each timestep $t$, the agent receives an observation $o_t$, selects an action $a_t$ based on the conditional probability $\pi_\theta(a_t|o_t)$, and observes the subsequent state $o_{t+1}$ and potential reward $r_t$. Episodes conclude upon task completion or reaching a maximum number of steps. For multiple rollouts in the same environment, we compute a task-level reward $R(\tau)$ for each trajectory, typically sparse and provided only at episode's end to indicate task success. The relative advantage $A_t$ for each action $a_t$ is determined by the task outcome, calculated as:

$$A_t = \frac{R(\tau) - \mu}{\sigma}$$

where $\mu$ and $\sigma$ are the mean and standard deviation of the rewards in the same group.

The GRPO objective function, considering multiple rollouts, is:

$$\mathcal{L} = \sum_t [\min\left(r_t A_t,\ \text{clip}(r_t, 1-\epsilon, 1+\epsilon) A_t\right)] - \lambda_{\text{KL}} D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$$

where $r_t = \frac{\pi_\theta(a_t|o_t)}{\pi_{\text{old}}(a_t|o_t)}$, and $\text{clip}(r_t, 1-\epsilon, 1+\epsilon)$ constrains the policy update. The KL term ensures stability. Parameters $\theta$ are updated via gradient ascent on $\mathcal{L}$.

### Two-Stage RL Training

We propose a two-stage RL training strategy to mitigate the exploration contraction effect, as illustrated in Figure 3.

**Stage 1: Action-Prioritized Exploration**: In this stage, we encourage the LLM to rely on intuition for decision making, thereby enhancing its exploration ability. At the same time, we optimize for both correct action and reasoning chain generation. In the original training strategy, the policy autoregressively generates reasoning and action in a sequential manner:

| Model/Method | | ALFWorld | | | | | | | WebShop | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Pick | Look | Clean | Heat | Cool | Pick2 | All | Succ. | Score |
| **Tuning-Free** | GPT-4o | 0.81 | 0.63 | 0.23 | 0.56 | 0.20 | 0.41 | 0.46 | 0.27 | 0.33 |
| | Gemini-2.5Pro | 0.85 | 0.53 | 0.67 | 0.68 | 0.29 | 0.59 | 0.59 | 0.38 | 0.52 |
| **Supervised** | SFT | 0.79 | 0.77 | 0.69 | 0.80 | 0.62 | 0.67 | 0.71 | 0.56 | 0.73 |
| | RFT | 0.78 | 0.82 | 0.70 | 0.83 | 0.68 | 0.59 | 0.73 | 0.60 | 0.78 |
| **RL (1.5B)** | GRPO | 0.81 | 0.58 | 0.86 | 0.76 | 0.63 | 0.71 | 0.73 | 0.57 | 0.76 |
| | Higher-T | 0.86 | 0.62 | 0.79 | 0.73 | 0.66 | **0.77** | 0.75 | 0.56 | 0.73 |
| | EntReg | 0.90 | 0.65 | 0.74 | 0.69 | **0.72** | 0.70 | 0.78 | 0.60 | 0.79 |
| | ActRe | **0.94** | **0.75** | **0.92** | **0.87** | 0.71 | 0.75 | **0.83** | **0.64** | **0.81** |
| **RL (7B)** | GRPO | **0.93** | 0.68 | 0.87 | 0.82 | 0.74 | 0.60 | 0.78 | 0.65 | 0.81 |
| | Higher-T | 0.91 | 0.77 | 0.82 | 0.80 | 0.77 | 0.66 | 0.80 | 0.63 | 0.77 |
| | EntReg | 0.90 | 0.75 | 0.89 | 0.80 | 0.75 | 0.69 | 0.82 | 0.66 | 0.80 |
| | ActRe | 0.91 | **0.82** | **0.93** | **0.88** | **0.78** | **0.76** | **0.86** | **0.69** | **0.83** |

Table 2: Comparison of task-specific and overall success rates of different models and methods on ALFWorld, as well as their success rates (Succ.) and scores (Score) on WebShop.

$$\pi_\theta(t_t, a_t|o_t) = \pi_\theta(t_t|o_t) \cdot \pi_\theta(a_t|o_t, t_t)$$

where $o_t$ is the observation, $t_t$ is the reasoning, and $a_t$ is the action. However, we observe that in many cases, the LLM already commits to a specific action during the reasoning process. Owing to the autoregressive nature of LLMs, this limits the diversity of possible actions.

Therefore, we reverse the generation order:

$$\pi_\theta(a_t, t_t|o_t) = \pi_\theta(a_t|o_t) \cdot \pi_\theta(t_t|o_t, a_t)$$

That is, for a given observation, the LLM first generates the action and then produces the reasoning to rationalize the selected action. To further promote exploration under the action-first paradigm, we raise the sampling temperature specifically for key branching tokens in the action sequence (see Figure 3 iii). For each action, we increase the sampling temperature specifically for key branching tokens:

$$a_t^{(i)} \sim \pi_\theta(a_t^{(i)} \mid o_t, a_t^{(<i)};\ \text{temperature} = T_i)$$

where $T_i > 1$ for key decision tokens. Action-first enables the LLM to explore a wider range of possible actions.

**Stage 2: Reasoning-Prioritized Alignment**: In the second stage, we restore the conventional reasoning-then-action generation order, which is aligned with the ReAct paradigm and real-world agent deployment. Specifically, the agent receives an observation $o_t$, first generates reasoning $t_t$, and then generates action $a_t$ conditioned on both:

$$\pi_\theta(t_t, a_t|o_t) = \pi_\theta(t_t|o_t) \cdot \pi_\theta(a_t|o_t, t_t)$$

During this phase, both reasoning and action sequences are jointly optimized using the RL objective, as in typical downstream usage. This process improves the agent's ability to solve tasks and generalize to unseen scenarios.

# Experiments and Results

## Experimental Setup

We mainly compare three categories of approaches in this study. (1) *Closed-source* large models, including **GPT-4o** and **Gemini-2.5Pro**, following the ReAct paradigm, which represent the performance of general LLMs with massive parameters on agent tasks. (2) *Tuning-based* strategies, such as **SFT** and **RFT**, adapt language models for agent tasks by behavior cloning from expert demonstrations and further leveraging successful trajectories through rejection sampling. (3) *RL-based* approaches. To mitigate the exploration contraction phenomenon during RL training, we introduce several strategies: (i) **GRPO-Vanilla**, which optimizes policy based on feedback from multiple trajectories under the same environment; (ii) **Higher Temperature** (Higher T) (Wang et al. 2025b), where a higher sampling temperature is applied to the LLM during rollout to encourage output diversity; and (iii) **Entropy Regularization** (EntReg) (Ziebart et al. 2008; Hu et al. 2025), which adds an entropy term to the loss function to counteract the decline in policy entropy and prevent policy collapse.

Experiments are conducted on the ALFWorld and WebShop benchmarks. Detailed descriptions of these environments are provided in the Experiments section.

For SFT, we set the learning rate to $1 \times 10^{-5}$. For GRPO, each batch consists of 128 samples, divided into 16 groups with 8 trajectories per group. The default sampling temperature is set to 0.6; for experiments with higher exploration, we increase the sampling temperature to 1.2.

## Main Result

We compare several methods in Table 2. ActRe achieves the best performance, reaching 0.86 on ALFWorld, a 13.7% improvement over original GRPO. Meanwhile, we observe that high-temperature sampling can improve certain tasks but

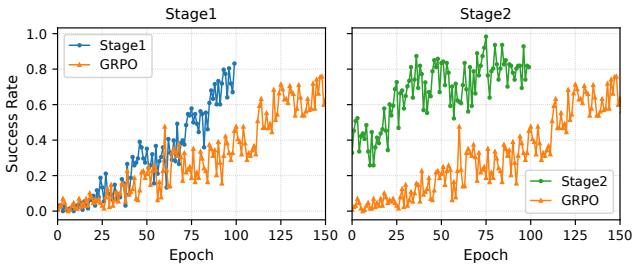



Figure 4: Success rate dynamics of our two-stage RL training scheme and GRPO on ALFWorld, trained with the Qwen-2.5-1.5B model.

shows limited benefit here, likely because it increases reasoning path diversity without substantially improving action diversity, thus restricting exploration. These findings suggest that, in multi-turn agent tasks, it is meaningful to separately consider the effects of reasoning and action on the overall outcome.

### Stage Effectiveness Analysis

Figure 4 illustrates the training progress of our two-stage RL scheme, depicting the change in success rate across epochs and comparing it with GRPO. We observe that during certain phases, ActRe demonstrates a faster improvement in success rate compared to GRPO. This may be attributed to our approach encouraging exploration, which enables the agent to attempt a wider range of potential solutions and develop more diverse trajectories. In the second training stage, the agent achieves a higher initial success rate and subsequently attains a better upper bound than GRPO, likely due to the discovery of additional promising solutions through enhanced exploration. These results indicate that our two-stage exploration-exploitation balanced scheme is effective, as it partially mitigates the exploration contraction problem and leads to broader and more robust policy improvement.

### Ablation Study

We conduct ablation studies on ALFWorld to evaluate the effects of Exploration, Alignment, and the action sampling temperature $T$, with results summarized in Table 3. We observe that the absence of any single component leads to suboptimal performance. Further analysis reveals that Exploration and higher action sampling temperature $T$ encourage the LLM to explore a wider range of possible action branches, promoting policy diversity during the exploration phase. Alignment helps LLMs conform to inference-time reasoning styles and enhances the robustness of reasoning.

| Setting | ALFWorld | Webshop(Succ.) |
|---|---|---|
| Full | **0.83** | **0.64** |
| w/o stage1 | 0.73 | 0.57 |
| w/o stage2 | 0.78 | 0.59 |
| w/o High Action T | 0.81 | 0.61 |

Table 3: Ablation results on ALFWorld. Each setting removes one component from the full model.

## Related Work

### LLM based Agents

LLMs have found broad application in the agent domain, spanning code generation (Huang et al. 2023; Zhang et al. 2024), web interaction (Bai et al. 2024; Agashe et al. 2024; Abuelsaad et al. 2024), embodied tasks (Zeng et al. 2024; Qiao et al. 2024; Fu et al. 2025), gaming, and more. Early efforts typically relied on pretrained models (Yao et al. 2023; Shinn et al. 2023), using prompt engineering and tool integration to handle complex tasks (Yao et al. 2023). Yet, models with limited parameter sizes often exhibit insufficient capability. To address this, some work introduces supervised fine-tuning to enhance decision-making (Zhang and Zhang 2024; Xi et al. 2024; Qin et al. 2024), while others employ single-step or offline reinforcement learning (Yu et al. 2024; Xiong et al. 2024; Zhou and Zanette 2024). Recently, end-to-end RL agents (Wang et al. 2025c; Feng et al. 2025a) have attracted attention for their streamlined training pipelines and online adaptation, without relying on curated step-level rewards. However, these methods still encounter limited effectiveness in leveraging exploration and exploitation. In this work, we further optimize the balance between exploration and exploitation in end-to-end RL agents from an entropy-driven perspective.

### Reinforcement Learning and Entropy

In recent years, reinforcement learning (RL) has been increasingly applied to LLMs, with research spanning both preference alignment and the enhancement of reasoning capabilities (Ouyang et al. 2022; Rafailov et al. 2023; Feng et al. 2025b; Yu et al. 2025). Maximum entropy RL (Ziebart et al. 2008) provides a theoretical framework for balancing exploration and exploitation, and has been widely adopted in RL (Mnih et al. 2016; Haarnoja et al. 2018). Several studies suggest that controlling policy entropy can enhance reasoning quality and exploration, and have investigated the impact of token-level entropy during RL optimization (Wang et al. 2025b; Yu et al. 2025; Cui et al. 2025). However, most of these works are limited to single-turn tasks, and their findings do not directly generalize to agent settings that require multi-turn interaction with complex environments.

## Conclusion

In this work, we analyze the exploration contraction problem in LLM agents during RL training from the perspective of action entropy, and emphasize the importance of balancing action-level exploration and exploitation for agent learning. We propose a two-stage RL training framework that encourages this balance. Experiments show our method mitigates the exploration contraction issue and outperforms GRPO. Future work may extend our approach to multimodal tasks and explore reasoning and more cognitive behaviors.

## References


Abuelsaad, T.; Akkil, D.; Dey, P.; Jagmohan, A.; Vempaty, A.; and Kokku, R. 2024. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. *arXiv preprint arXiv:2407.13032*.

Agashe, S.; Han, J.; Gan, S.; Yang, J.; Li, A.; and Wang, X. E. 2024. Agent s: An open agentic framework that uses computers like a human. *arXiv preprint arXiv:2410.08164*.

Bai, H.; Zhou, Y.; Pan, J.; Cemri, M.; Suhr, A.; Levine, S.; and Kumar, A. 2024. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. *Advances in Neural Information Processing Systems*, 37: 12461–12495.

Cui, G.; Zhang, Y.; Chen, J.; Yuan, L.; Wang, Z.; Zuo, Y.; Li, H.; Fan, Y.; Chen, H.; Chen, W.; et al. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Feng, L.; Xue, Z.; Liu, T.; and An, B. 2025a. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.

Feng, L.; Xue, Z.; Liu, T.; and An, B. 2025b. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.

Fu, D.; He, K.; Wang, Y.; Hong, W.; Gongque, Z.; Zeng, W.; Wang, W.; Wang, J.; Cai, X.; and Xu, W. 2025. Agentrefine: Enhancing agent generalization through refinement tuning. *arXiv preprint arXiv:2501.01702*.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. Pmlr.

Hu, J.; Zhang, Y.; Han, Q.; Jiang, D.; Zhang, X.; and Shum, H.-Y. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Huang, D.; Zhang, J. M.; Luck, M.; Bu, Q.; Qing, Y.; and Cui, H. 2023. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation. *arXiv preprint arXiv:2312.13010*.

Liu, Z.; Yang, Z.; Chen, Y.; Lee, C.; Shoeybi, M.; Catanzaro, B.; and Ping, W. 2025. AceReason-Nemotron 1.1: Advancing Math and Code Reasoning through SFT and RL Synergy. *arXiv preprint arXiv:2506.13284*.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 1928–1937. PmLR.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Qiao, S.; Fang, R.; Zhang, N.; Zhu, Y.; Chen, X.; Deng, S.; Jiang, Y.; Xie, P.; Huang, F.; and Chen, H. 2024. Agent planning with world knowledge model. *Advances in Neural Information Processing Systems*, 37: 114843–114871.

Qin, Y.; Liang, S.; Ye, Y.; Zhu, K.; Yan, L.; Lu, Y.; Lin, Y.; Cong, X.; Tang, X.; Qian, B.; et al. 2024. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In *ICLR*.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36: 53728–53741.

Setlur, A.; Garg, S.; Geng, X.; Garg, N.; Smith, V.; and Kumar, A. 2024. Rl on incorrect synthetic data scales the efficiency of llm math reasoning by eight-fold. *Advances in Neural Information Processing Systems*, 37: 43000–43031.

Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 8634–8652.

Shridhar, M.; Yuan, X.; Côté, M.-A.; Bisk, Y.; Trischler, A.; and Hausknecht, M. 2020. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Wang, H.; Leong, C. T.; Wang, J.; Wang, J.; and Li, W. 2025a. SPA-RL: Reinforcing LLM Agents via Stepwise Progress Attribution. *arXiv preprint arXiv:2505.20732*.

Wang, S.; Yu, L.; Gao, C.; Zheng, C.; Liu, S.; Lu, R.; Dang, K.; Chen, X.; Yang, J.; Zhang, Z.; et al. 2025b. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Wang, Z.; Wang, K.; Wang, Q.; Zhang, P.; Li, L.; Yang, Z.; Jin, X.; Yu, K.; Nguyen, M. N.; Liu, L.; et al. 2025c. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.

Xi, Z.; Ding, Y.; Chen, W.; Hong, B.; Guo, H.; Wang, J.; Yang, D.; Liao, C.; Guo, X.; He, W.; et al. 2024. Agentgym: Evolving large language model-based agents across diverse environments. *arXiv preprint arXiv:2406.04151*.

Xiong, W.; Song, Y.; Zhao, X.; Wu, W.; Wang, X.; Wang, K.; Li, C.; Peng, W.; and Li, S. 2024. Watch Every Step! LLM Agent Learning via Iterative Step-level Process Refinement. In *EMNLP*.

Yao, S.; Chen, H.; Yang, J.; and Narasimhan, K. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35: 20744–20757.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yu, Y.; Wang, Z.; Ma, W.; Guo, Z.; Zhan, J.; Wang, S.; Wu, C.; Guo, Z.; and Zhang, M. 2024. Steptool: A step-grained reinforcement learning framework for tool learning in llms.

Zeng, A.; Liu, M.; Lu, R.; Wang, B.; Liu, X.; Dong, Y.; and Tang, J. 2024. AgentTuning: Enabling Generalized Agent Abilities for LLMs. In *Findings of the Association for Computational Linguistics ACL 2024*, 3053–3077.

Zhang, K.; Li, J.; Li, G.; Shi, X.; and Jin, Z. 2024. CodeAgent: Enhancing Code Generation with Tool-Integrated Agent Systems for Real-World Repo-level Coding Challenges. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, 13643–13658. Association for Computational Linguistics.

Zhang, Z.; and Zhang, A. 2024. You Only Look at Screens: Multimodal Chain-of-Action Agents. In *Findings of the Association for Computational Linguistics ACL 2024*, 3132–3149.

Zhou, Y.; and Zanette, A. 2024. ArCHer: training language model agents via hierarchical multi-turn RL. In *Proceedings of the 41st International Conference on Machine Learning*, 62178–62209.

Ziebart, B. D.; Maas, A. L.; Bagnell, J. A.; Dey, A. K.; et al. 2008. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, 1433–1438. Chicago, IL, USA.